

GiNaC

1.8.10

Generated by Doxygen 1.9.8

1 Namespace Index	1
1.1 Namespace List	1
2 Hierarchical Index	3
2.1 Class Hierarchy	3
3 Class Index	7
3.1 Class List	7
4 File Index	15
4.1 File List	15
5 Namespace Documentation	19
5.1 GiNaC Namespace Reference	19
5.1.1 Typedef Documentation	58
5.1.1.1 archive_node_id	58
5.1.1.2 archive_atom	58
5.1.1.3 synthesize_func	58
5.1.1.4 unarchive_map_t	58
5.1.1.5 exvector	58
5.1.1.6 exset	59
5.1.1.7 exmap	59
5.1.1.8 evalffunctype	59
5.1.1.9 FUNCP_1P	59
5.1.1.10 FUNCP_2P	59
5.1.1.11 FUNCP_CUBA	59
5.1.1.12 epvector	59
5.1.1.13 epp	59
5.1.1.14 exprseq	60
5.1.1.15 paramset	60
5.1.1.16 eval_funcp	60
5.1.1.17 evalf_funcp	60
5.1.1.18 conjugate_funcp	60
5.1.1.19 real_part_funcp	60
5.1.1.20 imag_part_funcp	60
5.1.1.21 expand_funcp	60
5.1.1.22 derivative_funcp	60
5.1.1.23 expl_derivative_funcp	60
5.1.1.24 power_funcp	61
5.1.1.25 series_funcp	61
5.1.1.26 print_funcp	61
5.1.1.27 info_funcp	61
5.1.1.28 eval_funcp_1	61
5.1.1.29 evalf_funcp_1	61

5.1.1.30 conjugate_funcp_1	61
5.1.1.31 real_part_funcp_1	61
5.1.1.32 imag_part_funcp_1	61
5.1.1.33 expand_funcp_1	61
5.1.1.34 derivative_funcp_1	62
5.1.1.35 expl_derivative_funcp_1	62
5.1.1.36 power_funcp_1	62
5.1.1.37 series_funcp_1	62
5.1.1.38 print_funcp_1	62
5.1.1.39 info_funcp_1	62
5.1.1.40 eval_funcp_2	62
5.1.1.41 evalf_funcp_2	62
5.1.1.42 conjugate_funcp_2	62
5.1.1.43 real_part_funcp_2	62
5.1.1.44 imag_part_funcp_2	63
5.1.1.45 expand_funcp_2	63
5.1.1.46 derivative_funcp_2	63
5.1.1.47 expl_derivative_funcp_2	63
5.1.1.48 power_funcp_2	63
5.1.1.49 series_funcp_2	63
5.1.1.50 print_funcp_2	63
5.1.1.51 info_funcp_2	63
5.1.1.52 eval_funcp_3	63
5.1.1.53 evalf_funcp_3	63
5.1.1.54 conjugate_funcp_3	64
5.1.1.55 real_part_funcp_3	64
5.1.1.56 imag_part_funcp_3	64
5.1.1.57 expand_funcp_3	64
5.1.1.58 derivative_funcp_3	64
5.1.1.59 expl_derivative_funcp_3	64
5.1.1.60 power_funcp_3	64
5.1.1.61 series_funcp_3	64
5.1.1.62 print_funcp_3	64
5.1.1.63 info_funcp_3	64
5.1.1.64 eval_funcp_4	65
5.1.1.65 evalf_funcp_4	65
5.1.1.66 conjugate_funcp_4	65
5.1.1.67 real_part_funcp_4	65
5.1.1.68 imag_part_funcp_4	65
5.1.1.69 expand_funcp_4	65
5.1.1.70 derivative_funcp_4	65
5.1.1.71 expl_derivative_funcp_4	65

5.1.1.72 power_funcp_4	65
5.1.1.73 series_funcp_4	65
5.1.1.74 print_funcp_4	66
5.1.1.75 info_funcp_4	66
5.1.1.76 eval_funcp_5	66
5.1.1.77 evalf_funcp_5	66
5.1.1.78 conjugate_funcp_5	66
5.1.1.79 real_part_funcp_5	66
5.1.1.80 imag_part_funcp_5	66
5.1.1.81 expand_funcp_5	66
5.1.1.82 derivative_funcp_5	66
5.1.1.83 expl_derivative_funcp_5	67
5.1.1.84 power_funcp_5	67
5.1.1.85 series_funcp_5	67
5.1.1.86 print_funcp_5	67
5.1.1.87 info_funcp_5	67
5.1.1.88 eval_funcp_6	67
5.1.1.89 evalf_funcp_6	67
5.1.1.90 conjugate_funcp_6	67
5.1.1.91 real_part_funcp_6	67
5.1.1.92 imag_part_funcp_6	68
5.1.1.93 expand_funcp_6	68
5.1.1.94 derivative_funcp_6	68
5.1.1.95 expl_derivative_funcp_6	68
5.1.1.96 power_funcp_6	68
5.1.1.97 series_funcp_6	68
5.1.1.98 print_funcp_6	68
5.1.1.99 info_funcp_6	68
5.1.1.100 eval_funcp_7	68
5.1.1.101 evalf_funcp_7	69
5.1.1.102 conjugate_funcp_7	69
5.1.1.103 real_part_funcp_7	69
5.1.1.104 imag_part_funcp_7	69
5.1.1.105 expand_funcp_7	69
5.1.1.106 derivative_funcp_7	69
5.1.1.107 expl_derivative_funcp_7	69
5.1.1.108 power_funcp_7	69
5.1.1.109 series_funcp_7	69
5.1.1.110 print_funcp_7	70
5.1.1.111 info_funcp_7	70
5.1.1.112 eval_funcp_8	70
5.1.1.113 evalf_funcp_8	70

5.1.1.114 conjugate_funcp_8	70
5.1.1.115 real_part_funcp_8	70
5.1.1.116 imag_part_funcp_8	70
5.1.1.117 expand_funcp_8	70
5.1.1.118 derivative_funcp_8	70
5.1.1.119 expl_derivative_funcp_8	71
5.1.1.120 power_funcp_8	71
5.1.1.121 series_funcp_8	71
5.1.1.122 print_funcp_8	71
5.1.1.123 info_funcp_8	71
5.1.1.124 eval_funcp_9	71
5.1.1.125 evalf_funcp_9	71
5.1.1.126 conjugate_funcp_9	71
5.1.1.127 real_part_funcp_9	71
5.1.1.128 imag_part_funcp_9	72
5.1.1.129 expand_funcp_9	72
5.1.1.130 derivative_funcp_9	72
5.1.1.131 expl_derivative_funcp_9	72
5.1.1.132 power_funcp_9	72
5.1.1.133 series_funcp_9	72
5.1.1.134 print_funcp_9	72
5.1.1.135 info_funcp_9	72
5.1.1.136 eval_funcp_10	72
5.1.1.137 evalf_funcp_10	73
5.1.1.138 conjugate_funcp_10	73
5.1.1.139 real_part_funcp_10	73
5.1.1.140 imag_part_funcp_10	73
5.1.1.141 expand_funcp_10	73
5.1.1.142 derivative_funcp_10	73
5.1.1.143 expl_derivative_funcp_10	73
5.1.1.144 power_funcp_10	73
5.1.1.145 series_funcp_10	73
5.1.1.146 print_funcp_10	74
5.1.1.147 info_funcp_10	74
5.1.1.148 eval_funcp_11	74
5.1.1.149 evalf_funcp_11	74
5.1.1.150 conjugate_funcp_11	74
5.1.1.151 real_part_funcp_11	74
5.1.1.152 imag_part_funcp_11	74
5.1.1.153 expand_funcp_11	74
5.1.1.154 derivative_funcp_11	74
5.1.1.155 expl_derivative_funcp_11	75

5.1.1.156 power_funcp_11	75
5.1.1.157 series_funcp_11	75
5.1.1.158 print_funcp_11	75
5.1.1.159 info_funcp_11	75
5.1.1.160 eval_funcp_12	75
5.1.1.161 evalf_funcp_12	75
5.1.1.162 conjugate_funcp_12	75
5.1.1.163 real_part_funcp_12	76
5.1.1.164 imag_part_funcp_12	76
5.1.1.165 expand_funcp_12	76
5.1.1.166 derivative_funcp_12	76
5.1.1.167 expl_derivative_funcp_12	76
5.1.1.168 power_funcp_12	76
5.1.1.169 series_funcp_12	76
5.1.1.170 print_funcp_12	76
5.1.1.171 info_funcp_12	77
5.1.1.172 eval_funcp_13	77
5.1.1.173 evalf_funcp_13	77
5.1.1.174 conjugate_funcp_13	77
5.1.1.175 real_part_funcp_13	77
5.1.1.176 imag_part_funcp_13	77
5.1.1.177 expand_funcp_13	77
5.1.1.178 derivative_funcp_13	77
5.1.1.179 expl_derivative_funcp_13	78
5.1.1.180 power_funcp_13	78
5.1.1.181 series_funcp_13	78
5.1.1.182 print_funcp_13	78
5.1.1.183 info_funcp_13	78
5.1.1.184 eval_funcp_14	78
5.1.1.185 evalf_funcp_14	78
5.1.1.186 conjugate_funcp_14	78
5.1.1.187 real_part_funcp_14	79
5.1.1.188 imag_part_funcp_14	79
5.1.1.189 expand_funcp_14	79
5.1.1.190 derivative_funcp_14	79
5.1.1.191 expl_derivative_funcp_14	79
5.1.1.192 power_funcp_14	79
5.1.1.193 series_funcp_14	79
5.1.1.194 print_funcp_14	79
5.1.1.195 info_funcp_14	80
5.1.1.196 eval_funcp_exvector	80
5.1.1.197 evalf_funcp_exvector	80

5.1.1.198 conjugate_funcp_exvector	80
5.1.1.199 real_part_funcp_exvector	80
5.1.1.200 imag_part_funcp_exvector	80
5.1.1.201 expand_funcp_exvector	80
5.1.1.202 derivative_funcp_exvector	80
5.1.1.203 expl_derivative_funcp_exvector	80
5.1.1.204 power_funcp_exvector	80
5.1.1.205 series_funcp_exvector	81
5.1.1.206 print_funcp_exvector	81
5.1.1.207 info_funcp_exvector	81
5.1.1.208 exhashmap	81
5.1.1.209 spmap	81
5.1.1.210 lookup_map	81
5.1.1.211 lst	81
5.1.1.212 uintvector	81
5.1.1.213 unsignedvector	81
5.1.1.214 exvectorvector	81
5.1.1.215 sym_desc_vec	82
5.1.1.216 digits_changed_callback	82
5.1.1.217 print_context_class_info	82
5.1.1.218 registered_class_info	82
5.1.2 Enumeration Type Documentation	82
5.1.2.1 anonymous enum	82
5.1.3 Function Documentation	82
5.1.3.1 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [1/34]	82
5.1.3.2 GINAC_BIND_UNARCHIVER() [1/50]	82
5.1.3.3 GINAC_DECLARE_UNARCHIVER() [1/52]	83
5.1.3.4 write_unsigned()	83
5.1.3.5 read_unsigned()	83
5.1.3.6 operator<<() [1/16]	83
5.1.3.7 operator<<() [2/16]	83
5.1.3.8 operator>>() [1/3]	83
5.1.3.9 operator>>() [2/3]	84
5.1.3.10 find_factory_fcn()	84
5.1.3.11 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [2/34]	84
5.1.3.12 is_a() [1/3]	84
5.1.3.13 is_exactly_a() [1/2]	84
5.1.3.14 dynallocate() [1/2]	85
5.1.3.15 dynallocate() [2/2]	85
5.1.3.16 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [3/34]	85
5.1.3.17 print_func< print_dflt >() [1/3]	85
5.1.3.18 GINAC_BIND_UNARCHIVER() [2/50]	85

5.1.3.19 GINAC_BIND_UNARCHIVER() [3/50]	85
5.1.3.20 GINAC_BIND_UNARCHIVER() [4/50]	86
5.1.3.21 GINAC_BIND_UNARCHIVER() [5/50]	86
5.1.3.22 GINAC_BIND_UNARCHIVER() [6/50]	86
5.1.3.23 GINAC_BIND_UNARCHIVER() [7/50]	86
5.1.3.24 GINAC_BIND_UNARCHIVER() [8/50]	86
5.1.3.25 is_dirac_slash()	86
5.1.3.26 base_and_index()	86
5.1.3.27 dirac_ONE()	86
5.1.3.28 get_dim_uint()	87
5.1.3.29 clifford_unit()	87
5.1.3.30 dirac_gamma()	87
5.1.3.31 dirac_gamma5()	88
5.1.3.32 dirac_gammaL()	88
5.1.3.33 dirac_gammaR()	88
5.1.3.34 dirac_slash()	89
5.1.3.35 get_representation_label() [1/2]	89
5.1.3.36 trace_string()	89
5.1.3.37 dirac_trace() [1/3]	89
5.1.3.38 dirac_trace() [2/3]	90
5.1.3.39 dirac_trace() [3/3]	90
5.1.3.40 canonicalize_clifford()	90
5.1.3.41 clifford_star_bar()	91
5.1.3.42 clifford_prime()	91
5.1.3.43 remove_dirac_ONE()	91
5.1.3.44 clifford_max_label()	91
5.1.3.45 clifford_norm()	92
5.1.3.46 clifford_inverse()	92
5.1.3.47 lst_to_clifford() [1/2]	92
5.1.3.48 lst_to_clifford() [2/2]	92
5.1.3.49 get_clifford_comp()	93
5.1.3.50 clifford_to_lst()	93
5.1.3.51 clifford_moebius_map() [1/2]	93
5.1.3.52 clifford_moebius_map() [2/2]	94
5.1.3.53 GINAC_DECLARE_UNARCHIVER() [2/52]	94
5.1.3.54 GINAC_DECLARE_UNARCHIVER() [3/52]	94
5.1.3.55 GINAC_DECLARE_UNARCHIVER() [4/52]	95
5.1.3.56 GINAC_DECLARE_UNARCHIVER() [5/52]	95
5.1.3.57 GINAC_DECLARE_UNARCHIVER() [6/52]	95
5.1.3.58 GINAC_DECLARE_UNARCHIVER() [7/52]	95
5.1.3.59 GINAC_DECLARE_UNARCHIVER() [8/52]	95
5.1.3.60 is_clifford_tinfo()	95

5.1.3.61 clifford_bar()	95
5.1.3.62 clifford_star()	96
5.1.3.63 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [4/34]	96
5.1.3.64 print_func< print_dflt >() [2/3]	96
5.1.3.65 GINAC_BIND_UNARCHIVER() [9/50]	96
5.1.3.66 GINAC_BIND_UNARCHIVER() [10/50]	96
5.1.3.67 GINAC_BIND_UNARCHIVER() [11/50]	96
5.1.3.68 GINAC_BIND_UNARCHIVER() [12/50]	96
5.1.3.69 GINAC_BIND_UNARCHIVER() [13/50]	97
5.1.3.70 permute_free_index_to_front()	97
5.1.3.71 color_ONE()	97
5.1.3.72 color_T()	98
5.1.3.73 color_f()	98
5.1.3.74 color_d()	98
5.1.3.75 color_h()	99
5.1.3.76 is_color_tinfo()	99
5.1.3.77 get_representation_label() [2/2]	99
5.1.3.78 color_trace() [1/3]	99
5.1.3.79 color_trace() [2/3]	100
5.1.3.80 color_trace() [3/3]	100
5.1.3.81 GINAC_DECLARE_UNARCHIVER() [9/52]	100
5.1.3.82 GINAC_DECLARE_UNARCHIVER() [10/52]	100
5.1.3.83 GINAC_DECLARE_UNARCHIVER() [11/52]	101
5.1.3.84 GINAC_DECLARE_UNARCHIVER() [12/52]	101
5.1.3.85 GINAC_DECLARE_UNARCHIVER() [13/52]	101
5.1.3.86 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [5/34]	101
5.1.3.87 GINAC_BIND_UNARCHIVER() [14/50]	101
5.1.3.88 GINAC_DECLARE_UNARCHIVER() [14/52]	101
5.1.3.89 crc32()	101
5.1.3.90 are_ex_trivially_equal()	102
5.1.3.91 operator<<() [3/16]	102
5.1.3.92 operator<<() [4/16]	102
5.1.3.93 operator<<() [5/16]	102
5.1.3.94 nops() [1/2]	103
5.1.3.95 expand() [1/2]	103
5.1.3.96 conjugate()	103
5.1.3.97 real_part()	103
5.1.3.98 imag_part()	103
5.1.3.99 has()	104
5.1.3.100 find()	104
5.1.3.101 is_polynomial()	104
5.1.3.102 degree()	104

5.1.3.103 ldegree()	104
5.1.3.104 coeff()	105
5.1.3.105 numer() [1/2]	105
5.1.3.106 denom() [1/2]	105
5.1.3.107 numer_denom()	105
5.1.3.108 normal()	105
5.1.3.109 to_rational()	106
5.1.3.110 to_polynomial()	106
5.1.3.111 collect()	106
5.1.3.112 eval()	106
5.1.3.113 evalf() [1/2]	106
5.1.3.114 evalm()	106
5.1.3.115 eval_integ()	107
5.1.3.116 diff()	107
5.1.3.117 series()	107
5.1.3.118 match()	107
5.1.3.119 simplify_indexed() [1/3]	107
5.1.3.120 simplify_indexed() [2/3]	108
5.1.3.121 symmetrize() [1/4]	108
5.1.3.122 symmetrize() [2/4]	108
5.1.3.123 antisymmetrize() [1/4]	108
5.1.3.124 antisymmetrize() [2/4]	108
5.1.3.125 symmetrize_cyclic() [1/4]	108
5.1.3.126 symmetrize_cyclic() [2/4]	109
5.1.3.127 op()	109
5.1.3.128 lhs()	109
5.1.3.129 rhs()	109
5.1.3.130 is_zero() [1/2]	109
5.1.3.131 swap() [1/2]	110
5.1.3.132 subs() [1/3]	110
5.1.3.133 subs() [2/3]	110
5.1.3.134 subs() [3/3]	110
5.1.3.135 is_a() [2/3]	110
5.1.3.136 is_exactly_a() [2/2]	111
5.1.3.137 ex_to()	111
5.1.3.138 compile_ex() [1/3]	111
5.1.3.139 compile_ex() [2/3]	112
5.1.3.140 compile_ex() [3/3]	112
5.1.3.141 link_ex() [1/3]	112
5.1.3.142 link_ex() [2/3]	113
5.1.3.143 link_ex() [3/3]	113
5.1.3.144 unlink_ex()	113

5.1.3.145 swap() [2/2]	114
5.1.3.146 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [6/34]	114
5.1.3.147 conjugateepvector()	114
5.1.3.148 factor()	114
5.1.3.149 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [7/34]	115
5.1.3.150 GINAC_DECLARE_UNARCHIVER() [15/52]	115
5.1.3.151 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [8/34]	115
5.1.3.152 GINAC_BIND_UNARCHIVER() [15/50]	115
5.1.3.153 GINAC_DECLARE_UNARCHIVER() [16/52]	115
5.1.3.154 GINAC_BIND_UNARCHIVER() [16/50]	116
5.1.3.155 GINAC_DECLARE_UNARCHIVER() [17/52]	116
5.1.3.156 is_the_function()	116
5.1.3.157 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [9/34]	116
5.1.3.158 GINAC_BIND_UNARCHIVER() [17/50]	116
5.1.3.159 GINAC_DECLARE_UNARCHIVER() [18/52]	116
5.1.3.160 make_hash_seed()	116
5.1.3.161 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [10/34]	117
5.1.3.162 print_func< print_context >()	117
5.1.3.163 GINAC_BIND_UNARCHIVER() [18/50]	117
5.1.3.164 GINAC_BIND_UNARCHIVER() [19/50]	117
5.1.3.165 GINAC_BIND_UNARCHIVER() [20/50]	117
5.1.3.166 is_dummy_pair() [1/2]	117
5.1.3.167 is_dummy_pair() [2/2]	118
5.1.3.168 find_free_and_dummy() [1/2]	118
5.1.3.169 minimal_dim()	118
5.1.3.170 GINAC_DECLARE_UNARCHIVER() [19/52]	118
5.1.3.171 GINAC_DECLARE_UNARCHIVER() [20/52]	119
5.1.3.172 GINAC_DECLARE_UNARCHIVER() [21/52]	119
5.1.3.173 find_free_and_dummy() [2/2]	119
5.1.3.174 find_dummy_indices()	119
5.1.3.175 count_dummy_indices()	119
5.1.3.176 count_free_indices()	120
5.1.3.177 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [11/34]	120
5.1.3.178 GINAC_BIND_UNARCHIVER() [21/50]	120
5.1.3.179 indices_consistent()	120
5.1.3.180 number_of_type()	120
5.1.3.181 rename_dummy_indices()	120
5.1.3.182 find_variant_indices()	121
5.1.3.183 reposition_dummy_indices()	121
5.1.3.184 product_to_exvector()	121
5.1.3.185 idx_symmetrization()	122
5.1.3.186 simplify_indexed() [3/3]	122

5.1.3.187 <code>simplify_indexed_product()</code>	122
5.1.3.188 <code>hasindex()</code>	122
5.1.3.189 <code>get_all_dummy_indices_safely()</code>	122
5.1.3.190 <code>get_all_dummy_indices()</code>	123
5.1.3.191 <code>rename_dummy_indices_uniquely()</code> [1/4]	123
5.1.3.192 <code>rename_dummy_indices_uniquely()</code> [2/4]	123
5.1.3.193 <code>rename_dummy_indices_uniquely()</code> [3/4]	123
5.1.3.194 <code>rename_dummy_indices_uniquely()</code> [4/4]	124
5.1.3.195 <code>expand_dummy_sum()</code>	124
5.1.3.196 <code>GINAC_DECLARE_UNARCHIVER()</code> [22/52]	124
5.1.3.197 <code>conjugate_evalf()</code>	124
5.1.3.198 <code>conjugate_eval()</code>	125
5.1.3.199 <code>conjugate_print_latex()</code>	125
5.1.3.200 <code>conjugate_conjugate()</code>	125
5.1.3.201 <code>conjugate_expl_derivative()</code>	125
5.1.3.202 <code>conjugate_real_part()</code>	125
5.1.3.203 <code>conjugate_imag_part()</code>	125
5.1.3.204 <code>func_arg_info()</code>	126
5.1.3.205 <code>conjugate_info()</code>	126
5.1.3.206 <code>REGISTER_FUNCTION()</code> [1/36]	126
5.1.3.207 <code>real_part_evalf()</code>	126
5.1.3.208 <code>real_part_eval()</code>	126
5.1.3.209 <code>real_part_print_latex()</code>	126
5.1.3.210 <code>real_part_conjugate()</code>	127
5.1.3.211 <code>real_part_real_part()</code>	127
5.1.3.212 <code>real_part_imag_part()</code>	127
5.1.3.213 <code>real_part_expl_derivative()</code>	127
5.1.3.214 <code>REGISTER_FUNCTION()</code> [2/36]	127
5.1.3.215 <code>imag_part_evalf()</code>	127
5.1.3.216 <code>imag_part_eval()</code>	127
5.1.3.217 <code>imag_part_print_latex()</code>	128
5.1.3.218 <code>imag_part_conjugate()</code>	128
5.1.3.219 <code>imag_part_real_part()</code>	128
5.1.3.220 <code>imag_part_imag_part()</code>	128
5.1.3.221 <code>imag_part_expl_derivative()</code>	128
5.1.3.222 <code>REGISTER_FUNCTION()</code> [3/36]	128
5.1.3.223 <code>abs_evalf()</code>	128
5.1.3.224 <code>abs_eval()</code>	129
5.1.3.225 <code>abs_expand()</code>	129
5.1.3.226 <code>abs_expl_derivative()</code>	129
5.1.3.227 <code>abs_print_latex()</code>	129
5.1.3.228 <code>abs_print_csrc_float()</code>	129

5.1.3.229 <code>abs_conjugate()</code>	129
5.1.3.230 <code>abs_real_part()</code>	130
5.1.3.231 <code>abs_imag_part()</code>	130
5.1.3.232 <code>abs_power()</code>	130
5.1.3.233 <code>abs_info()</code>	130
5.1.3.234 <code>REGISTER_FUNCTION()</code> [4/36]	130
5.1.3.235 <code>step_evalf()</code>	130
5.1.3.236 <code>step_eval()</code>	131
5.1.3.237 <code>step_series()</code>	131
5.1.3.238 <code>step_conjugate()</code>	131
5.1.3.239 <code>step_real_part()</code>	131
5.1.3.240 <code>step_imag_part()</code>	131
5.1.3.241 <code>REGISTER_FUNCTION()</code> [5/36]	131
5.1.3.242 <code>csgn_evalf()</code>	132
5.1.3.243 <code>csgn_eval()</code>	132
5.1.3.244 <code>csgn_series()</code>	132
5.1.3.245 <code>csgn_conjugate()</code>	132
5.1.3.246 <code>csgn_real_part()</code>	132
5.1.3.247 <code>csgn_imag_part()</code>	132
5.1.3.248 <code>csgn_power()</code>	133
5.1.3.249 <code>REGISTER_FUNCTION()</code> [6/36]	133
5.1.3.250 <code>eta_evalf()</code>	133
5.1.3.251 <code>eta_eval()</code>	133
5.1.3.252 <code>eta_series()</code>	133
5.1.3.253 <code>eta_conjugate()</code>	134
5.1.3.254 <code>eta_real_part()</code>	134
5.1.3.255 <code>eta_imag_part()</code>	134
5.1.3.256 <code>REGISTER_FUNCTION()</code> [7/36]	134
5.1.3.257 <code>Li2_evalf()</code>	134
5.1.3.258 <code>Li2_eval()</code>	134
5.1.3.259 <code>Li2_deriv()</code>	135
5.1.3.260 <code>Li2_series()</code> [1/2]	135
5.1.3.261 <code>Li2_conjugate()</code>	135
5.1.3.262 <code>REGISTER_FUNCTION()</code> [8/36]	135
5.1.3.263 <code>Li3_eval()</code>	135
5.1.3.264 <code>REGISTER_FUNCTION()</code> [9/36]	135
5.1.3.265 <code>zetaderiv_eval()</code>	136
5.1.3.266 <code>zetaderiv_deriv()</code>	136
5.1.3.267 <code>REGISTER_FUNCTION()</code> [10/36]	136
5.1.3.268 <code>factorial_evalf()</code>	136
5.1.3.269 <code>factorial_eval()</code>	136
5.1.3.270 <code>factorial_print_dflt_latex()</code>	136

5.1.3.271 factorial_conjugate()	137
5.1.3.272 factorial_real_part()	137
5.1.3.273 factorial_imag_part()	137
5.1.3.274 REGISTER_FUNCTION() [11/36]	137
5.1.3.275 binomial_evalf()	137
5.1.3.276 binomial_sym()	137
5.1.3.277 binomial_eval()	138
5.1.3.278 binomial_conjugate()	138
5.1.3.279 binomial_real_part()	138
5.1.3.280 binomial_imag_part()	138
5.1.3.281 REGISTER_FUNCTION() [12/36]	138
5.1.3.282 Order_eval()	138
5.1.3.283 Order_series()	139
5.1.3.284 Order_conjugate()	139
5.1.3.285 Order_real_part()	139
5.1.3.286 Order_imag_part()	139
5.1.3.287 Order_power()	139
5.1.3.288 Order_expl_derivative()	139
5.1.3.289 REGISTER_FUNCTION() [13/36]	140
5.1.3.290 Isolve()	140
5.1.3.291 fsolve()	140
5.1.3.292 zeta() [1/3]	141
5.1.3.293 zeta() [2/3]	141
5.1.3.294 is_the_function< zeta_SERIAL >()	141
5.1.3.295 G() [1/2]	141
5.1.3.296 G() [2/2]	141
5.1.3.297 is_the_function< G_SERIAL >()	142
5.1.3.298 psi() [1/4]	142
5.1.3.299 psi() [2/4]	142
5.1.3.300 is_the_function< psi_SERIAL >()	142
5.1.3.301 iterated_integral() [1/2]	142
5.1.3.302 iterated_integral() [2/2]	143
5.1.3.303 is_the_function< iterated_integral_SERIAL >()	143
5.1.3.304 is_order_function()	143
5.1.3.305 convert_H_to_Li()	143
5.1.3.306 EllipticK_evalf()	143
5.1.3.307 EllipticK_eval()	144
5.1.3.308 EllipticK_deriv()	144
5.1.3.309 EllipticK_series()	144
5.1.3.310 EllipticK_print_latex()	144
5.1.3.311 REGISTER_FUNCTION() [14/36]	144
5.1.3.312 EllipticE_evalf()	144

5.1.3.313 EllipticE_eval()	145
5.1.3.314 EllipticE_deriv()	145
5.1.3.315 EllipticE_series()	145
5.1.3.316 EllipticE_print_latex()	145
5.1.3.317 REGISTER_FUNCTION() [15/36]	145
5.1.3.318 iterated_integral_evalf_impl()	145
5.1.3.319 iterated_integral2_evalf()	146
5.1.3.320 iterated_integral3_evalf()	146
5.1.3.321 iterated_integral2_eval()	146
5.1.3.322 iterated_integral3_eval()	146
5.1.3.323 lgamma_evalf()	146
5.1.3.324 lgamma_eval()	146
5.1.3.325 lgamma_deriv()	147
5.1.3.326 lgamma_series()	147
5.1.3.327 lgamma_conjugate()	147
5.1.3.328 REGISTER_FUNCTION() [16/36]	147
5.1.3.329 tgamma_evalf()	147
5.1.3.330 tgamma_eval()	147
5.1.3.331 tgamma_deriv()	148
5.1.3.332 tgamma_series()	148
5.1.3.333 tgamma_conjugate()	148
5.1.3.334 REGISTER_FUNCTION() [17/36]	148
5.1.3.335 beta_evalf()	148
5.1.3.336 beta_eval()	149
5.1.3.337 beta_deriv()	149
5.1.3.338 beta_series()	149
5.1.3.339 REGISTER_FUNCTION() [18/36]	149
5.1.3.340 psi1_evalf()	149
5.1.3.341 psi1_eval()	150
5.1.3.342 psi1_deriv()	150
5.1.3.343 psi1_series()	150
5.1.3.344 psi2_evalf()	150
5.1.3.345 psi2_eval()	150
5.1.3.346 psi2_deriv()	151
5.1.3.347 psi2_series()	151
5.1.3.348 G2_evalf()	151
5.1.3.349 G2_eval()	151
5.1.3.350 G3_evalf()	151
5.1.3.351 G3_eval()	152
5.1.3.352 Li_evalf()	152
5.1.3.353 Li_eval()	152
5.1.3.354 Li_series()	152

5.1.3.355 Li_deriv()	152
5.1.3.356 Li_print_latex()	153
5.1.3.357 REGISTER_FUNCTION() [19/36]	153
5.1.3.358 S_evalf()	153
5.1.3.359 S_eval()	153
5.1.3.360 S_series()	153
5.1.3.361 S_deriv()	154
5.1.3.362 S_print_latex()	154
5.1.3.363 REGISTER_FUNCTION() [20/36]	154
5.1.3.364 H_evalf()	154
5.1.3.365 H_eval()	154
5.1.3.366 H_series()	155
5.1.3.367 H_deriv()	155
5.1.3.368 H_print_latex()	155
5.1.3.369 REGISTER_FUNCTION() [21/36]	155
5.1.3.370 zeta1_evalf()	155
5.1.3.371 zeta1_eval()	156
5.1.3.372 zeta1_deriv()	156
5.1.3.373 zeta1_print_latex()	156
5.1.3.374 zeta2_evalf()	156
5.1.3.375 zeta2_eval()	156
5.1.3.376 zeta2_deriv()	156
5.1.3.377 zeta2_print_latex()	157
5.1.3.378 exp_evalf()	157
5.1.3.379 exp_eval()	157
5.1.3.380 exp_expand()	157
5.1.3.381 exp_deriv()	157
5.1.3.382 exp_real_part()	157
5.1.3.383 exp_imag_part()	158
5.1.3.384 exp_conjugate()	158
5.1.3.385 exp_power()	158
5.1.3.386 exp_info()	158
5.1.3.387 REGISTER_FUNCTION() [22/36]	158
5.1.3.388 log_evalf()	158
5.1.3.389 log_eval()	159
5.1.3.390 log_deriv()	159
5.1.3.391 log_series()	159
5.1.3.392 log_real_part()	159
5.1.3.393 log_imag_part()	159
5.1.3.394 log_expand()	160
5.1.3.395 log_conjugate()	160
5.1.3.396 log_info()	160

5.1.3.397 REGISTER_FUNCTION() [23/36]	160
5.1.3.398 sin_evalf()	160
5.1.3.399 sin_eval()	160
5.1.3.400 sin_deriv()	161
5.1.3.401 sin_real_part()	161
5.1.3.402 sin_imag_part()	161
5.1.3.403 sin_conjugate()	161
5.1.3.404 trig_info()	161
5.1.3.405 REGISTER_FUNCTION() [24/36]	161
5.1.3.406 cos_evalf()	162
5.1.3.407 cos_eval()	162
5.1.3.408 cos_deriv()	162
5.1.3.409 cos_real_part()	162
5.1.3.410 cos_imag_part()	162
5.1.3.411 cos_conjugate()	162
5.1.3.412 REGISTER_FUNCTION() [25/36]	163
5.1.3.413 tan_evalf()	163
5.1.3.414 tan_eval()	163
5.1.3.415 tan_deriv()	163
5.1.3.416 tan_real_part()	163
5.1.3.417 tan_imag_part()	163
5.1.3.418 tan_series()	164
5.1.3.419 tan_conjugate()	164
5.1.3.420 REGISTER_FUNCTION() [26/36]	164
5.1.3.421 asin_evalf()	164
5.1.3.422 asin_eval()	164
5.1.3.423 asin_deriv()	164
5.1.3.424 asin_conjugate()	165
5.1.3.425 asin_info()	165
5.1.3.426 REGISTER_FUNCTION() [27/36]	165
5.1.3.427 acos_evalf()	165
5.1.3.428 acos_eval()	165
5.1.3.429 acos_deriv()	165
5.1.3.430 acos_conjugate()	166
5.1.3.431 REGISTER_FUNCTION() [28/36]	166
5.1.3.432 atan_evalf()	166
5.1.3.433 atan_eval()	166
5.1.3.434 atan_deriv()	166
5.1.3.435 atan_series()	166
5.1.3.436 atan_conjugate()	167
5.1.3.437 atan_info()	167
5.1.3.438 REGISTER_FUNCTION() [29/36]	167

5.1.3.439 atan2_evalf()	167
5.1.3.440 atan2_eval()	167
5.1.3.441 atan2_deriv()	167
5.1.3.442 atan2_info()	168
5.1.3.443 REGISTER_FUNCTION() [30/36]	168
5.1.3.444 sinh_evalf()	168
5.1.3.445 sinh_eval()	168
5.1.3.446 sinh_deriv()	168
5.1.3.447 sinh_real_part()	168
5.1.3.448 sinh_imag_part()	169
5.1.3.449 sinh_conjugate()	169
5.1.3.450 REGISTER_FUNCTION() [31/36]	169
5.1.3.451 cosh_evalf()	169
5.1.3.452 cosh_eval()	169
5.1.3.453 cosh_deriv()	169
5.1.3.454 cosh_real_part()	170
5.1.3.455 cosh_imag_part()	170
5.1.3.456 cosh_conjugate()	170
5.1.3.457 REGISTER_FUNCTION() [32/36]	170
5.1.3.458 tanh_evalf()	170
5.1.3.459 tanh_eval()	170
5.1.3.460 tanh_deriv()	171
5.1.3.461 tanh_series()	171
5.1.3.462 tanh_real_part()	171
5.1.3.463 tanh_imag_part()	171
5.1.3.464 tanh_conjugate()	171
5.1.3.465 REGISTER_FUNCTION() [33/36]	171
5.1.3.466 asinh_evalf()	172
5.1.3.467 asinh_eval()	172
5.1.3.468 asinh_deriv()	172
5.1.3.469 asinh_conjugate()	172
5.1.3.470 REGISTER_FUNCTION() [34/36]	172
5.1.3.471 acosh_evalf()	172
5.1.3.472 acosh_eval()	173
5.1.3.473 acosh_deriv()	173
5.1.3.474 acosh_conjugate()	173
5.1.3.475 REGISTER_FUNCTION() [35/36]	173
5.1.3.476 atanh_evalf()	173
5.1.3.477 atanh_eval()	173
5.1.3.478 atanh_deriv()	174
5.1.3.479 atanh_series()	174
5.1.3.480 atanh_conjugate()	174

5.1.3.481 REGISTER_FUNCTION() [36/36]	174
5.1.3.482 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [12/34]	174
5.1.3.483 subsvalue()	174
5.1.3.484 adaptivesimpson()	175
5.1.3.485 GINAC_BIND_UNARCHIVER() [22/50]	175
5.1.3.486 GINAC_DECLARE_UNARCHIVER() [23/52]	175
5.1.3.487 ifactor()	175
5.1.3.488 is_discriminant_of_quadratic_number_field()	175
5.1.3.489 kronecker_symbol()	176
5.1.3.490 primitive_dirichlet_character()	176
5.1.3.491 dirichlet_character()	176
5.1.3.492 generalised_Bernoulli_number()	177
5.1.3.493 Bernoulli_polynomial()	177
5.1.3.494 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [13/34]	177
5.1.3.495 GINAC_BIND_UNARCHIVER() [23/50]	177
5.1.3.496 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [14/34]	177
5.1.3.497 GINAC_BIND_UNARCHIVER() [24/50]	178
5.1.3.498 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [15/34]	178
5.1.3.499 GINAC_BIND_UNARCHIVER() [25/50]	178
5.1.3.500 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [16/34]	178
5.1.3.501 GINAC_BIND_UNARCHIVER() [26/50]	178
5.1.3.502 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [17/34]	178
5.1.3.503 GINAC_BIND_UNARCHIVER() [27/50]	178
5.1.3.504 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [18/34]	179
5.1.3.505 GINAC_BIND_UNARCHIVER() [28/50]	179
5.1.3.506 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [19/34]	179
5.1.3.507 GINAC_BIND_UNARCHIVER() [29/50]	179
5.1.3.508 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [20/34]	179
5.1.3.509 GINAC_BIND_UNARCHIVER() [30/50]	179
5.1.3.510 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [21/34]	179
5.1.3.511 GINAC_BIND_UNARCHIVER() [31/50]	179
5.1.3.512 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [22/34]	180
5.1.3.513 GINAC_BIND_UNARCHIVER() [32/50]	180
5.1.3.514 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [23/34]	180
5.1.3.515 GINAC_BIND_UNARCHIVER() [33/50]	180
5.1.3.516 GINAC_DECLARE_UNARCHIVER() [24/52]	180
5.1.3.517 GINAC_DECLARE_UNARCHIVER() [25/52]	180
5.1.3.518 GINAC_DECLARE_UNARCHIVER() [26/52]	180
5.1.3.519 GINAC_DECLARE_UNARCHIVER() [27/52]	180
5.1.3.520 GINAC_DECLARE_UNARCHIVER() [28/52]	181
5.1.3.521 GINAC_DECLARE_UNARCHIVER() [29/52]	181
5.1.3.522 GINAC_DECLARE_UNARCHIVER() [30/52]	181

5.1.3.523 GINAC_DECLARE_UNARCHIVER() [31/52]	181
5.1.3.524 GINAC_DECLARE_UNARCHIVER() [32/52]	181
5.1.3.525 GINAC_DECLARE_UNARCHIVER() [33/52]	181
5.1.3.526 GINAC_DECLARE_UNARCHIVER() [34/52]	181
5.1.3.527 GINAC_DECLARE_UNARCHIVER() [35/52]	181
5.1.3.528 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [24/34]	182
5.1.3.529 GINAC_BIND_UNARCHIVER() [34/50]	182
5.1.3.530 lst_to_matrix()	182
5.1.3.531 diag_matrix() [1/2]	182
5.1.3.532 diag_matrix() [2/2]	182
5.1.3.533 unit_matrix() [1/2]	182
5.1.3.534 symbolic_matrix() [1/2]	183
5.1.3.535 reduced_matrix()	183
5.1.3.536 sub_matrix()	183
5.1.3.537 GINAC_DECLARE_UNARCHIVER() [36/52]	183
5.1.3.538 nops() [2/2]	183
5.1.3.539 expand() [2/2]	184
5.1.3.540 evalf() [2/2]	184
5.1.3.541 rows()	184
5.1.3.542 cols()	184
5.1.3.543 transpose()	184
5.1.3.544 determinant()	184
5.1.3.545 trace()	185
5.1.3.546 charpoly()	185
5.1.3.547 inverse() [1/3]	185
5.1.3.548 inverse() [2/3]	185
5.1.3.549 rank() [1/2]	185
5.1.3.550 rank() [2/2]	185
5.1.3.551 unit_matrix() [2/2]	186
5.1.3.552 symbolic_matrix() [2/2]	186
5.1.3.553 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [25/34]	186
5.1.3.554 tryfactsubs()	186
5.1.3.555 algebraic_match_mul_with_mul()	187
5.1.3.556 GINAC_BIND_UNARCHIVER() [35/50]	187
5.1.3.557 GINAC_DECLARE_UNARCHIVER() [37/52]	187
5.1.3.558 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [26/34]	187
5.1.3.559 reeval_ncmul()	187
5.1.3.560 hold_ncmul()	188
5.1.3.561 GINAC_BIND_UNARCHIVER() [36/50]	188
5.1.3.562 GINAC_DECLARE_UNARCHIVER() [38/52]	188
5.1.3.563 get_first_symbol()	188
5.1.3.564 add_symbol()	188

5.1.3.565 collect_symbols()	189
5.1.3.566 get_symbol_stats()	189
5.1.3.567 lcmcoeff()	189
5.1.3.568 lcm_of_coefficients_denominators()	189
5.1.3.569 multiply_lcm()	190
5.1.3.570 quo()	190
5.1.3.571 rem()	191
5.1.3.572 decomp_rational()	191
5.1.3.573 prem()	192
5.1.3.574 sprem()	192
5.1.3.575 divide()	193
5.1.3.576 divide_in_z()	193
5.1.3.577 sr_gcd()	194
5.1.3.578 interpolate()	195
5.1.3.579 heur_gcd_z()	195
5.1.3.580 heur_gcd()	196
5.1.3.581 gcd_pf_pow()	196
5.1.3.582 gcd_pf_mul()	197
5.1.3.583 gcd() ^[1/2]	197
5.1.3.584 gcd_pf_pow_pow()	198
5.1.3.585 lcm() ^[1/2]	198
5.1.3.586 sqrfree_yun()	198
5.1.3.587 sqrfree()	199
5.1.3.588 sqrfree_parfrac()	199
5.1.3.589 replace_with_symbol() ^[1/2]	200
5.1.3.590 replace_with_symbol() ^[2/2]	201
5.1.3.591 frac_cancel()	201
5.1.3.592 find_common_factor()	201
5.1.3.593 collect_common_factors()	202
5.1.3.594 resultant()	202
5.1.3.595 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() ^[27/34]	202
5.1.3.596 make_real_float()	202
5.1.3.597 read_real_float()	203
5.1.3.598 GINAC_BIND_UNARCHIVER() ^[37/50]	203
5.1.3.599 write_real_float()	203
5.1.3.600 print_real_number()	203
5.1.3.601 print_integer_csrc()	204
5.1.3.602 print_real_csrc()	204
5.1.3.603 coerce()	204
5.1.3.604 coerce< int, cln::cl_I >()	204
5.1.3.605 coerce< unsigned int, cln::cl_I >()	205
5.1.3.606 print_real_cl_N()	205

5.1.3.607 exp()	205
5.1.3.608 log()	205
5.1.3.609 sin()	206
5.1.3.610 cos()	206
5.1.3.611 tan()	207
5.1.3.612 asin()	207
5.1.3.613 acos()	207
5.1.3.614 atan() ^[1/2]	207
5.1.3.615 atan() ^[2/2]	208
5.1.3.616 sinh()	208
5.1.3.617 cosh()	209
5.1.3.618 tanh()	209
5.1.3.619 asinh()	209
5.1.3.620 acosh()	210
5.1.3.621 atanh()	210
5.1.3.622 Li2_series() ^[2/2]	210
5.1.3.623 Li2_projection()	210
5.1.3.624 Li2_()	211
5.1.3.625 Li2()	211
5.1.3.626 zeta() ^[3/3]	211
5.1.3.627 guess_precision()	211
5.1.3.628 lgamma() ^[1/2]	212
5.1.3.629 lgamma() ^[2/2]	212
5.1.3.630 tgamma() ^[1/2]	212
5.1.3.631 tgamma() ^[2/2]	212
5.1.3.632 psi() ^[3/4]	212
5.1.3.633 psi() ^[4/4]	213
5.1.3.634 factorial()	213
5.1.3.635 doublefactorial()	213
5.1.3.636 binomial()	214
5.1.3.637 bernoulli()	214
5.1.3.638 fibonacci()	215
5.1.3.639 abs()	215
5.1.3.640 mod()	215
5.1.3.641 smod()	216
5.1.3.642 irem() ^[1/2]	216
5.1.3.643 irem() ^[2/2]	217
5.1.3.644 iquo() ^[1/2]	217
5.1.3.645 iquo() ^[2/2]	217
5.1.3.646 gcd() ^[2/2]	218
5.1.3.647 lcm() ^[2/2]	218
5.1.3.648 sqrt() ^[1/2]	218

5.1.3.649 isqrt()	219
5.1.3.650 PiEvalf()	219
5.1.3.651 EulerEvalf()	219
5.1.3.652 CatalanEvalf()	219
5.1.3.653 operator<<() [6/16]	219
5.1.3.654 GINAC_DECLARE_UNARCHIVER() [39/52]	220
5.1.3.655 pow() [1/3]	220
5.1.3.656 inverse() [3/3]	220
5.1.3.657 step()	220
5.1.3.658 csgn()	220
5.1.3.659 is_zero() [2/2]	221
5.1.3.660 is_positive()	221
5.1.3.661 is_negative()	221
5.1.3.662 is_integer()	221
5.1.3.663 is_pos_integer()	221
5.1.3.664 is_nonneg_integer()	221
5.1.3.665 is_even()	222
5.1.3.666 is_odd()	222
5.1.3.667 is_prime()	222
5.1.3.668 is_rational()	222
5.1.3.669 is_real()	222
5.1.3.670 is_cinteger()	222
5.1.3.671 is_crational()	223
5.1.3.672 to_int()	223
5.1.3.673 to_long()	223
5.1.3.674 to_double()	223
5.1.3.675 real()	223
5.1.3.676 imag()	223
5.1.3.677 numer() [2/2]	224
5.1.3.678 denom() [2/2]	224
5.1.3.679 exadd()	224
5.1.3.680 exmul()	224
5.1.3.681 exminus()	224
5.1.3.682 operator+() [1/4]	225
5.1.3.683 operator-() [1/4]	225
5.1.3.684 operator*() [1/2]	225
5.1.3.685 operator/() [1/2]	225
5.1.3.686 operator+() [2/4]	225
5.1.3.687 operator-() [2/4]	225
5.1.3.688 operator*() [2/2]	226
5.1.3.689 operator/() [2/2]	226
5.1.3.690 operator+=() [1/2]	226

5.1.3.691 operator==()	[1/2]	226
5.1.3.692 operator*==()	[1/2]	226
5.1.3.693 operator/=()	[1/2]	226
5.1.3.694 operator+=()	[2/2]	227
5.1.3.695 operator-=()	[2/2]	227
5.1.3.696 operator*=()	[2/2]	227
5.1.3.697 operator/=(	[2/2]	227
5.1.3.698 operator+()	[3/4]	227
5.1.3.699 operator-()	[3/4]	227
5.1.3.700 operator+()	[4/4]	227
5.1.3.701 operator-()	[4/4]	228
5.1.3.702 operator++()	[1/4]	228
5.1.3.703 operator--()	[1/4]	228
5.1.3.704 operator++()	[2/4]	228
5.1.3.705 operator--()	[2/4]	228
5.1.3.706 operator++()	[3/4]	229
5.1.3.707 operator--()	[3/4]	229
5.1.3.708 operator++()	[4/4]	229
5.1.3.709 operator--()	[4/4]	229
5.1.3.710 operator==(		229
5.1.3.711 operator"!=()		230
5.1.3.712 operator<()		230
5.1.3.713 operator<=()		230
5.1.3.714 operator>()		230
5.1.3.715 operator>=()		230
5.1.3.716 my_ios_index()		230
5.1.3.717 my_ios_callback()		231
5.1.3.718 get_print_context()		231
5.1.3.719 set_print_context()		231
5.1.3.720 get_print_options()		231
5.1.3.721 set_print_options()		231
5.1.3.722 operator<<()	[7/16]	232
5.1.3.723 operator>>()	[3/3]	232
5.1.3.724 dflt()		232
5.1.3.725 latex()		232
5.1.3.726 python()		232
5.1.3.727 python_repr()		232
5.1.3.728 tree()		232
5.1.3.729 csrc()		233
5.1.3.730 csrc_float()		233
5.1.3.731 csrc_double()		233
5.1.3.732 csrc_cl_N()		233

5.1.3.733 <code>index_dimensions()</code>	233
5.1.3.734 <code>no_index_dimensions()</code>	233
5.1.3.735 <code>GINAC_IMPLEMENT_REGISTERED_CLASS_OPT()</code> [28/34]	233
5.1.3.736 <code>print_sym_pow()</code>	234
5.1.3.737 <code>GINAC_BIND_UNARCHIVER()</code> [38/50]	234
5.1.3.738 <code>GINAC_DECLARE_UNARCHIVER()</code> [40/52]	234
5.1.3.739 <code>pow()</code> [2/3]	234
5.1.3.740 <code>pow()</code> [3/3]	234
5.1.3.741 <code>sqrt()</code> [2/2]	234
5.1.3.742 <code>is_a()</code> [3/3]	235
5.1.3.743 <code>GINAC_IMPLEMENT_REGISTERED_CLASS_OPT()</code> [29/34]	235
5.1.3.744 <code>GINAC_BIND_UNARCHIVER()</code> [39/50]	235
5.1.3.745 <code>GINAC_DECLARE_UNARCHIVER()</code> [41/52]	235
5.1.3.746 <code>series_to_poly()</code>	235
5.1.3.747 <code>is_terminating()</code>	236
5.1.3.748 <code>make_return_type_t()</code>	236
5.1.3.749 <code>set_print_func()</code> [1/2]	236
5.1.3.750 <code>set_print_func()</code> [2/2]	236
5.1.3.751 <code>GINAC_IMPLEMENT_REGISTERED_CLASS_OPT()</code> [30/34]	237
5.1.3.752 <code>GINAC_BIND_UNARCHIVER()</code> [40/50]	237
5.1.3.753 <code>print_operator()</code>	237
5.1.3.754 <code>GINAC_DECLARE_UNARCHIVER()</code> [42/52]	237
5.1.3.755 <code>GINAC_IMPLEMENT_REGISTERED_CLASS_OPT()</code> [31/34]	237
5.1.3.756 <code>get_default_TeX_name()</code>	237
5.1.3.757 <code>GINAC_BIND_UNARCHIVER()</code> [41/50]	238
5.1.3.758 <code>GINAC_BIND_UNARCHIVER()</code> [42/50]	238
5.1.3.759 <code>GINAC_BIND_UNARCHIVER()</code> [43/50]	238
5.1.3.760 <code>GINAC_DECLARE_UNARCHIVER()</code> [43/52]	238
5.1.3.761 <code>GINAC_DECLARE_UNARCHIVER()</code> [44/52]	238
5.1.3.762 <code>GINAC_DECLARE_UNARCHIVER()</code> [45/52]	238
5.1.3.763 <code>GINAC_IMPLEMENT_REGISTERED_CLASS_OPT()</code> [32/34]	238
5.1.3.764 <code>GINAC_BIND_UNARCHIVER()</code> [44/50]	238
5.1.3.765 <code>index0()</code>	239
5.1.3.766 <code>index1()</code>	239
5.1.3.767 <code>index2()</code>	239
5.1.3.768 <code>index3()</code>	239
5.1.3.769 <code>not_symmetric()</code>	239
5.1.3.770 <code>symmetric2()</code>	239
5.1.3.771 <code>symmetric3()</code>	239
5.1.3.772 <code>symmetric4()</code>	240
5.1.3.773 <code>antisymmetric2()</code>	240
5.1.3.774 <code>antisymmetric3()</code>	240

5.1.3.775 antisymmetric4()	240
5.1.3.776 canonicalize()	240
5.1.3.777 symm()	241
5.1.3.778 symmetrize() [3/4]	241
5.1.3.779 antisymmetrize() [3/4]	241
5.1.3.780 symmetrize_cyclic() [3/4]	241
5.1.3.781 GINAC_DECLARE_UNARCHIVER() [46/52]	241
5.1.3.782 sy_none() [1/4]	242
5.1.3.783 sy_none() [2/4]	242
5.1.3.784 sy_none() [3/4]	242
5.1.3.785 sy_none() [4/4]	242
5.1.3.786 sy_symm() [1/4]	242
5.1.3.787 sy_symm() [2/4]	242
5.1.3.788 sy_symm() [3/4]	243
5.1.3.789 sy_symm() [4/4]	243
5.1.3.790 sy_anti() [1/4]	243
5.1.3.791 sy_anti() [2/4]	243
5.1.3.792 sy_anti() [3/4]	243
5.1.3.793 sy_anti() [4/4]	243
5.1.3.794 sy_cycl() [1/4]	244
5.1.3.795 sy_cycl() [2/4]	244
5.1.3.796 sy_cycl() [3/4]	244
5.1.3.797 sy_cycl() [4/4]	244
5.1.3.798 symmetrize() [4/4]	244
5.1.3.799 antisymmetrize() [4/4]	244
5.1.3.800 symmetrize_cyclic() [4/4]	245
5.1.3.801 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [33/34]	245
5.1.3.802 print_func< print_dflt >() [3/3]	245
5.1.3.803 GINAC_BIND_UNARCHIVER() [45/50]	245
5.1.3.804 GINAC_BIND_UNARCHIVER() [46/50]	245
5.1.3.805 GINAC_BIND_UNARCHIVER() [47/50]	245
5.1.3.806 GINAC_BIND_UNARCHIVER() [48/50]	245
5.1.3.807 GINAC_BIND_UNARCHIVER() [49/50]	246
5.1.3.808 delta_tensor()	246
5.1.3.809 metric_tensor()	246
5.1.3.810 lorentz_g()	247
5.1.3.811 spinor_metric()	247
5.1.3.812 epsilon_tensor() [1/2]	248
5.1.3.813 epsilon_tensor() [2/2]	248
5.1.3.814 lorentz_eps()	248
5.1.3.815 GINAC_DECLARE_UNARCHIVER() [47/52]	249
5.1.3.816 GINAC_DECLARE_UNARCHIVER() [48/52]	249

5.1.3.817 GINAC_DECLARE_UNARCHIVER() [49/52]	249
5.1.3.818 GINAC_DECLARE_UNARCHIVER() [50/52]	249
5.1.3.819 GINAC_DECLARE_UNARCHIVER() [51/52]	249
5.1.3.820 log2()	250
5.1.3.821 multinomial_coefficient()	250
5.1.3.822 rotate_left()	250
5.1.3.823 compare_pointers()	250
5.1.3.824 golden_ratio_hash()	251
5.1.3.825 permutation_sign() [1/2]	251
5.1.3.826 permutation_sign() [2/2]	251
5.1.3.827 shaker_sort()	251
5.1.3.828 cyclic_permutation()	252
5.1.3.829 format_index_value() [1/2]	252
5.1.3.830 format_index_value() [2/2]	252
5.1.3.831 operator<<() [8/16]	252
5.1.3.832 operator<<() [9/16]	253
5.1.3.833 operator<<() [10/16]	253
5.1.3.834 operator<<() [11/16]	253
5.1.3.835 operator<<() [12/16]	253
5.1.3.836 operator<<() [13/16]	254
5.1.3.837 operator<<() [14/16]	254
5.1.3.838 operator<<() [15/16]	254
5.1.3.839 operator<<() [16/16]	254
5.1.3.840 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [34/34]	255
5.1.3.841 GINAC_BIND_UNARCHIVER() [50/50]	255
5.1.3.842 haswild()	255
5.1.3.843 GINAC_DECLARE_UNARCHIVER() [52/52]	255
5.1.3.844 wild()	255
5.1.4 Variable Documentation	255
5.1.4.1 unarch_table_instance	255
5.1.4.2 map_evalm	255
5.1.4.3 map_eval_integ	256
5.1.4.4 tensor	256
5.1.4.5 Pi	256
5.1.4.6 Euler	256
5.1.4.7 Catalan	256
5.1.4.8 crctab	257
5.1.4.9 library_initializer	257
5.1.4.10 _num0_bp	257
5.1.4.11 idx	257
5.1.4.12 force_include_tgamma	257
5.1.4.13 force_include_zeta1	257

5.1.4.14 GINAC_BIND_UNARCHIVER	257
5.1.4.15 l	258
5.1.4.16 Digits	258
5.1.4.17 next_print_context_id	258
5.1.4.18 version_major	258
5.1.4.19 version_minor	258
5.1.4.20 version_micro	258
5.1.4.21 _num_120_p	259
5.1.4.22 _ex_120	259
5.1.4.23 _num_60_p	259
5.1.4.24 _ex_60	259
5.1.4.25 _num_48_p	259
5.1.4.26 _ex_48	259
5.1.4.27 _num_30_p	259
5.1.4.28 _ex_30	259
5.1.4.29 _num_25_p	260
5.1.4.30 _ex_25	260
5.1.4.31 _num_24_p	260
5.1.4.32 _ex_24	260
5.1.4.33 _num_20_p	260
5.1.4.34 _ex_20	260
5.1.4.35 _num_18_p	260
5.1.4.36 _ex_18	260
5.1.4.37 _num_15_p	261
5.1.4.38 _ex_15	261
5.1.4.39 _num_12_p	261
5.1.4.40 _ex_12	261
5.1.4.41 _num_11_p	261
5.1.4.42 _ex_11	261
5.1.4.43 _num_10_p	261
5.1.4.44 _ex_10	261
5.1.4.45 _num_9_p	262
5.1.4.46 _ex_9	262
5.1.4.47 _num_8_p	262
5.1.4.48 _ex_8	262
5.1.4.49 _num_7_p	262
5.1.4.50 _ex_7	262
5.1.4.51 _num_6_p	262
5.1.4.52 _ex_6	262
5.1.4.53 _num_5_p	263
5.1.4.54 _ex_5	263
5.1.4.55 _num_4_p	263

5.1.4.56 _ex_4	263
5.1.4.57 _num_3_p	263
5.1.4.58 _ex_3	263
5.1.4.59 _num_2_p	263
5.1.4.60 _ex_2	264
5.1.4.61 _num_1_p	264
5.1.4.62 _ex_1	264
5.1.4.63 _num_1_2_p	264
5.1.4.64 _ex_1_2	264
5.1.4.65 _num_1_3_p	264
5.1.4.66 _ex_1_3	265
5.1.4.67 _num_1_4_p	265
5.1.4.68 _ex_1_4	265
5.1.4.69 _num0_p	265
5.1.4.70 _ex0	265
5.1.4.71 _num1_4_p	266
5.1.4.72 _ex1_4	266
5.1.4.73 _num1_3_p	266
5.1.4.74 _ex1_3	266
5.1.4.75 _num1_2_p	266
5.1.4.76 _ex1_2	266
5.1.4.77 _num1_p	266
5.1.4.78 _ex1	267
5.1.4.79 _num2_p	267
5.1.4.80 _ex2	267
5.1.4.81 _num3_p	267
5.1.4.82 _ex3	268
5.1.4.83 _num4_p	268
5.1.4.84 _ex4	268
5.1.4.85 _num5_p	268
5.1.4.86 _ex5	268
5.1.4.87 _num6_p	268
5.1.4.88 _ex6	268
5.1.4.89 _num7_p	269
5.1.4.90 _ex7	269
5.1.4.91 _num8_p	269
5.1.4.92 _ex8	269
5.1.4.93 _num9_p	269
5.1.4.94 _ex9	269
5.1.4.95 _num10_p	269
5.1.4.96 _ex10	270
5.1.4.97 _num11_p	270

5.1.4.98 _ex11	270
5.1.4.99 _num12_p	270
5.1.4.100 _ex12	270
5.1.4.101 _num15_p	270
5.1.4.102 _ex15	270
5.1.4.103 _num18_p	271
5.1.4.104 _ex18	271
5.1.4.105 _num20_p	271
5.1.4.106 _ex20	271
5.1.4.107 _num24_p	271
5.1.4.108 _ex24	271
5.1.4.109 _num25_p	271
5.1.4.110 _ex25	271
5.1.4.111 _num30_p	272
5.1.4.112 _ex30	272
5.1.4.113 _num48_p	272
5.1.4.114 _ex48	272
5.1.4.115 _num60_p	272
5.1.4.116 _ex60	272
5.1.4.117 _num120_p	272
5.1.4.118 _ex120	272
5.2 GiNaC::Gt_detail Namespace Reference	273
5.2.1 Function Documentation	273
5.2.1.1 integer_partition()	273
5.2.1.2 deconcatenate_path()	273
5.2.1.3 deconcatenate_path< ex >()	273
5.2.1.4 deconcatenate_path< Gt::kernel >()	274
5.2.1.5 operator<<()	274
5.2.1.6 operator<()	274
5.3 GiNaC::internal Namespace Reference	274
5.4 std Namespace Reference	274
5.4.1 Function Documentation	275
5.4.1.1 swap()	275
6 Class Documentation	277
6.1 GiNaC::internal::_iter_rep Struct Reference	277
6.1.1 Constructor & Destructor Documentation	277
6.1.1.1 _iter_rep()	277
6.1.2 Member Function Documentation	277
6.1.2.1 operator==()	277
6.1.2.2 operator"!=()	278
6.1.3 Member Data Documentation	278

6.1.3.1 e	278
6.1.3.2 i	278
6.1.3.3 i_end	278
6.2 GiNaC::_numeric_digits Class Reference	278
6.2.1 Detailed Description	279
6.2.2 Constructor & Destructor Documentation	279
6.2.2.1 _numeric_digits()	279
6.2.3 Member Function Documentation	279
6.2.3.1 operator=()	279
6.2.3.2 operator long()	279
6.2.3.3 print()	280
6.2.3.4 add_callback()	280
6.2.4 Member Data Documentation	280
6.2.4.1 digits	280
6.2.4.2 too_late	280
6.2.4.3 callbacklist	280
6.3 GiNaC::add Class Reference	281
6.3.1 Detailed Description	286
6.3.2 Constructor & Destructor Documentation	286
6.3.2.1 add() [1/6]	286
6.3.2.2 add() [2/6]	286
6.3.2.3 add() [3/6]	286
6.3.2.4 add() [4/6]	286
6.3.2.5 add() [5/6]	287
6.3.2.6 add() [6/6]	287
6.3.3 Member Function Documentation	287
6.3.3.1 precedence()	287
6.3.3.2 info()	287
6.3.3.3 is_polynomial()	288
6.3.3.4 degree()	288
6.3.3.5 ldegree()	288
6.3.3.6 coeff()	288
6.3.3.7 eval()	289
6.3.3.8 evalm()	289
6.3.3.9 series()	289
6.3.3.10 normal()	290
6.3.3.11 integer_content()	290
6.3.3.12 smod()	290
6.3.3.13 max_coefficient()	291
6.3.3.14 conjugate()	291
6.3.3.15 real_part()	291
6.3.3.16 imag_part()	291

6.3.3.17	get_free_indices()	292
6.3.3.18	eval_ncmul()	292
6.3.3.19	derivative()	292
6.3.3.20	return_type()	292
6.3.3.21	return_type_tinfo()	292
6.3.3.22	thisexpairseq() [1/2]	293
6.3.3.23	thisexpairseq() [2/2]	293
6.3.3.24	split_ex_to_pair()	293
6.3.3.25	combine_ex_with_coeff_to_pair()	293
6.3.3.26	combine_pair_with_coeff_to_pair()	294
6.3.3.27	recombine_pair_to_ex()	294
6.3.3.28	expand()	294
6.3.3.29	print_add()	294
6.3.3.30	do_print()	295
6.3.3.31	do_print_latex()	295
6.3.3.32	do_print_csrc()	295
6.3.3.33	do_print_python_repr()	295
6.3.4	Friends And Related Symbol Documentation	295
6.3.4.1	mul	295
6.3.4.2	power	295
6.4	GiNaC::archive Class Reference	296
6.4.1	Detailed Description	297
6.4.2	Member Typedef Documentation	297
6.4.2.1	inv_at_cit	297
6.4.3	Constructor & Destructor Documentation	297
6.4.3.1	archive() [1/3]	297
6.4.3.2	~archive()	297
6.4.3.3	archive() [2/3]	298
6.4.3.4	archive() [3/3]	298
6.4.4	Member Function Documentation	298
6.4.4.1	archive_ex()	298
6.4.4.2	unarchive_ex() [1/3]	298
6.4.4.3	unarchive_ex() [2/3]	299
6.4.4.4	unarchive_ex() [3/3]	299
6.4.4.5	num_expressions()	299
6.4.4.6	get_top_node()	300
6.4.4.7	clear()	300
6.4.4.8	add_node()	300
6.4.4.9	get_node()	300
6.4.4.10	forget()	300
6.4.4.11	printraw()	301
6.4.4.12	atomize()	301

6.4.4.13 unatomize()	301
6.4.5 Friends And Related Symbol Documentation	301
6.4.5.1 operator<<	301
6.4.5.2 operator>>	302
6.4.6 Member Data Documentation	302
6.4.6.1 nodes	302
6.4.6.2 exprs	302
6.4.6.3 atoms	302
6.4.6.4 inverse_atoms	302
6.4.6.5 exprtable	302
6.5 GiNaC::archive_node Class Reference	303
6.5.1 Detailed Description	304
6.5.2 Member Typedef Documentation	304
6.5.2.1 propinfovector	304
6.5.2.2 archive_node_cit	305
6.5.3 Member Enumeration Documentation	305
6.5.3.1 property_type	305
6.5.4 Constructor & Destructor Documentation	305
6.5.4.1 archive_node() [1/2]	305
6.5.4.2 archive_node() [2/2]	305
6.5.5 Member Function Documentation	305
6.5.5.1 operator=()	305
6.5.5.2 add_bool()	306
6.5.5.3 add_unsigned()	306
6.5.5.4 add_string()	306
6.5.5.5 add_ex()	306
6.5.5.6 find_bool()	306
6.5.5.7 find_unsigned()	307
6.5.5.8 find_string()	307
6.5.5.9 find_first()	307
6.5.5.10 find_last()	307
6.5.5.11 find_property_range()	308
6.5.5.12 find_ex()	308
6.5.5.13 find_ex_by_loc()	308
6.5.5.14 find_ex_node()	308
6.5.5.15 get_properties()	309
6.5.5.16 unarchive()	309
6.5.5.17 has_same_ex_as()	309
6.5.5.18 has_ex()	309
6.5.5.19 get_ex()	309
6.5.5.20 forget()	310
6.5.5.21 printraw()	310

6.5.6 Friends And Related Symbol Documentation	310
6.5.6.1 operator<<	310
6.5.6.2 operator>>	310
6.5.7 Member Data Documentation	310
6.5.7.1 a	310
6.5.7.2 props	311
6.5.7.3 has_expression	311
6.5.7.4 e	311
6.6 GiNaC::archive_node::archive_node_cit_range Struct Reference	311
6.6.1 Member Data Documentation	311
6.6.1.1 begin	311
6.6.1.2 end	312
6.7 GiNaC::archive::archived_ex Struct Reference	312
6.7.1 Detailed Description	312
6.7.2 Constructor & Destructor Documentation	312
6.7.2.1 archived_ex() [1/2]	312
6.7.2.2 archived_ex() [2/2]	312
6.7.3 Member Data Documentation	313
6.7.3.1 name	313
6.7.3.2 root	313
6.8 GiNaC::basic Class Reference	313
6.8.1 Detailed Description	317
6.8.2 Constructor & Destructor Documentation	317
6.8.2.1 basic() [1/2]	317
6.8.2.2 ~basic()	317
6.8.2.3 basic() [2/2]	317
6.8.3 Member Function Documentation	317
6.8.3.1 operator=()	317
6.8.3.2 duplicate()	317
6.8.3.3 eval()	318
6.8.3.4 evalf()	318
6.8.3.5 evalm()	318
6.8.3.6 eval_integ()	318
6.8.3.7 eval_ncmul()	319
6.8.3.8 eval_indexed()	319
6.8.3.9 print()	319
6.8.3.10 dbgprint()	319
6.8.3.11 dbgprinttree()	320
6.8.3.12 precedence()	320
6.8.3.13 info()	320
6.8.3.14 nops()	320
6.8.3.15 op()	321

6.8.3.16 operator[]() [1/4]	321
6.8.3.17 operator[]() [2/4]	321
6.8.3.18 let_op()	321
6.8.3.19 operator[]() [3/4]	322
6.8.3.20 operator[]() [4/4]	322
6.8.3.21 has()	322
6.8.3.22 match()	322
6.8.3.23 match_same_type()	323
6.8.3.24 subs()	323
6.8.3.25 map()	323
6.8.3.26 accept()	324
6.8.3.27 is_polynomial()	324
6.8.3.28 degree()	324
6.8.3.29 ldegree()	324
6.8.3.30 coeff()	324
6.8.3.31 expand()	325
6.8.3.32 collect()	325
6.8.3.33 derivative()	325
6.8.3.34 series()	326
6.8.3.35 normal()	326
6.8.3.36 to_rational()	327
6.8.3.37 to_polynomial()	327
6.8.3.38 integer_content()	327
6.8.3.39 smod()	327
6.8.3.40 max_coefficient()	328
6.8.3.41 get_free_indices()	328
6.8.3.42 add_indexed()	328
6.8.3.43 scalar_mul_indexed()	329
6.8.3.44 contract_with()	329
6.8.3.45 return_type()	330
6.8.3.46 return_type_tinfo()	330
6.8.3.47 conjugate()	330
6.8.3.48 real_part()	330
6.8.3.49 imag_part()	331
6.8.3.50 compare_same_type()	331
6.8.3.51 is_equal_same_type()	331
6.8.3.52 calchash()	331
6.8.3.53 print_dispatch() [1/2]	332
6.8.3.54 print_dispatch() [2/2]	332
6.8.3.55 archive()	332
6.8.3.56 read_archive()	333
6.8.3.57 subs_one_level()	333

6.8.3.58 diff()	333
6.8.3.59 compare()	334
6.8.3.60 is_equal()	334
6.8.3.61 hold()	334
6.8.3.62 gethash()	335
6.8.3.63 setflag()	335
6.8.3.64 clearflag()	336
6.8.3.65 ensure_if_modifiable()	336
6.8.3.66 do_print()	336
6.8.3.67 do_print_tree()	336
6.8.3.68 do_print_python_repr()	336
6.8.4 Friends And Related Symbol Documentation	337
6.8.4.1 ex	337
6.8.5 Member Data Documentation	337
6.8.5.1 flags	337
6.8.5.2 hashvalue	337
6.9 GiNaC::basic_log_kernel Class Reference	338
6.9.1 Detailed Description	342
6.9.2 Member Function Documentation	342
6.9.2.1 series_coeff_impl()	342
6.9.2.2 do_print()	342
6.10 GiNaC::basic_multi_iterator< T > Class Template Reference	342
6.10.1 Detailed Description	344
6.10.2 Constructor & Destructor Documentation	344
6.10.2.1 basic_multi_iterator() [1/3]	344
6.10.2.2 basic_multi_iterator() [2/3]	344
6.10.2.3 basic_multi_iterator() [3/3]	344
6.10.2.4 ~basic_multi_iterator()	345
6.10.3 Member Function Documentation	345
6.10.3.1 size()	345
6.10.3.2 overflow()	345
6.10.3.3 get_vector()	345
6.10.3.4 operator[]() [1/2]	345
6.10.3.5 operator[]() [2/2]	346
6.10.3.6 operator()() [1/2]	346
6.10.3.7 operator()() [2/2]	346
6.10.3.8 init()	346
6.10.3.9 operator++()	346
6.10.4 Friends And Related Symbol Documentation	347
6.10.4.1 operator<<	347
6.10.5 Member Data Documentation	347
6.10.5.1 N	347

6.10.5.2 B	347
6.10.5.3 v	347
6.10.5.4 flag_overflow	347
6.11 GiNaC::basic_partition_generator Class Reference	348
6.11.1 Detailed Description	348
6.11.2 Constructor & Destructor Documentation	348
6.11.2.1 basic_partition_generator()	348
6.11.3 Member Data Documentation	348
6.11.3.1 mngen	348
6.12 GiNaC::class_info< OPT > Class Template Reference	349
6.12.1 Constructor & Destructor Documentation	349
6.12.1.1 class_info()	349
6.12.2 Member Function Documentation	350
6.12.2.1 get_parent()	350
6.12.2.2 find()	350
6.12.2.3 dump_hierarchy()	350
6.12.2.4 dump_tree()	350
6.12.2.5 identify_parents()	350
6.12.3 Member Data Documentation	351
6.12.3.1 options	351
6.12.3.2 first	351
6.12.3.3 next	351
6.12.3.4 parent	351
6.12.3.5 parents_identified	351
6.13 GiNaC::clifford Class Reference	352
6.13.1 Detailed Description	359
6.13.2 Constructor & Destructor Documentation	359
6.13.2.1 clifford() [1/4]	359
6.13.2.2 clifford() [2/4]	359
6.13.2.3 clifford() [3/4]	359
6.13.2.4 clifford() [4/4]	360
6.13.3 Member Function Documentation	360
6.13.3.1 precedence()	360
6.13.3.2 archive()	360
6.13.3.3 read_archive()	360
6.13.3.4 eval_ncmul()	361
6.13.3.5 match_same_type()	361
6.13.3.6 thiscontainer() [1/2]	361
6.13.3.7 thiscontainer() [2/2]	361
6.13.3.8 return_type()	361
6.13.3.9 return_type_tinfo()	362
6.13.3.10 get_representation_label()	362

6.13.3.11 get_metric() [1/2]	362
6.13.3.12 get_metric() [2/2]	362
6.13.3.13 same_metric()	362
6.13.3.14 get_commutator_sign()	362
6.13.3.15 nops()	363
6.13.3.16 op()	363
6.13.3.17 let_op()	363
6.13.3.18 subs()	363
6.13.3.19 do_print_dflt()	364
6.13.3.20 do_print_latex()	364
6.13.3.21 do_print_tree()	364
6.13.4 Member Data Documentation	364
6.13.4.1 representation_label	364
6.13.4.2 metric	364
6.13.4.3 commutator_sign	365
6.14 GiNaC::cliffordunit Class Reference	365
6.14.1 Detailed Description	369
6.14.2 Member Function Documentation	369
6.14.2.1 contract_with()	369
6.14.2.2 do_print()	369
6.14.2.3 do_print_latex()	369
6.15 GiNaC::color Class Reference	370
6.15.1 Detailed Description	376
6.15.2 Constructor & Destructor Documentation	376
6.15.2.1 color() [1/4]	376
6.15.2.2 color() [2/4]	377
6.15.2.3 color() [3/4]	377
6.15.2.4 color() [4/4]	377
6.15.3 Member Function Documentation	377
6.15.3.1 archive()	377
6.15.3.2 read_archive()	378
6.15.3.3 eval_ncmul()	378
6.15.3.4 match_same_type()	378
6.15.3.5 thiscontainer() [1/2]	379
6.15.3.6 thiscontainer() [2/2]	379
6.15.3.7 return_type()	379
6.15.3.8 return_type_tinfo()	379
6.15.3.9 get_representation_label()	379
6.15.4 Member Data Documentation	379
6.15.4.1 representation_label	379
6.16 GiNaC::compare_all_equal< T > Class Template Reference	380
6.16.1 Detailed Description	380

6.16.2 Constructor & Destructor Documentation	380
6.16.2.1 ~compare_all_equal()	380
6.16.3 Member Function Documentation	380
6.16.3.1 struct_is_equal()	380
6.16.3.2 struct_compare()	380
6.17 GiNaC::compare_bitwise< T > Class Template Reference	381
6.17.1 Detailed Description	381
6.17.2 Constructor & Destructor Documentation	381
6.17.2.1 ~compare_bitwise()	381
6.17.3 Member Function Documentation	381
6.17.3.1 struct_is_equal()	381
6.17.3.2 struct_compare()	381
6.18 GiNaC::compare_std_less< T > Class Template Reference	382
6.18.1 Detailed Description	382
6.18.2 Constructor & Destructor Documentation	382
6.18.2.1 ~compare_std_less()	382
6.18.3 Member Function Documentation	382
6.18.3.1 struct_is_equal()	382
6.18.3.2 struct_compare()	382
6.19 GiNaC::composition_generator Class Reference	383
6.19.1 Detailed Description	383
6.19.2 Constructor & Destructor Documentation	383
6.19.2.1 composition_generator()	383
6.19.3 Member Function Documentation	383
6.19.3.1 get()	383
6.19.3.2 next()	384
6.19.4 Member Data Documentation	384
6.19.4.1 cmgen	384
6.19.4.2 atend	384
6.19.4.3 trivial	384
6.19.4.4 composition	384
6.19.4.5 current_updated	384
6.20 GiNaC::const_iterator Class Reference	385
6.20.1 Member Typedef Documentation	386
6.20.1.1 iterator_category	386
6.20.1.2 value_type	386
6.20.1.3 difference_type	386
6.20.1.4 pointer	386
6.20.1.5 reference	386
6.20.2 Constructor & Destructor Documentation	386
6.20.2.1 const_iterator() [1/2]	386
6.20.2.2 const_iterator() [2/2]	386

6.20.3 Member Function Documentation	386
6.20.3.1 operator*()	386
6.20.3.2 operator->()	387
6.20.3.3 operator[]()	387
6.20.3.4 operator++() [1/2]	387
6.20.3.5 operator++() [2/2]	387
6.20.3.6 operator+=()	387
6.20.3.7 operator+()	387
6.20.3.8 operator--() [1/2]	387
6.20.3.9 operator--() [2/2]	388
6.20.3.10 operator-=()	388
6.20.3.11 operator-()	388
6.20.3.12 operator==()	388
6.20.3.13 operator!=()	388
6.20.3.14 operator<()	388
6.20.3.15 operator>()	388
6.20.3.16 operator<=()	389
6.20.3.17 operator>=()	389
6.20.4 Friends And Related Symbol Documentation	389
6.20.4.1 ex	389
6.20.4.2 const_preorder_iterator	389
6.20.4.3 const_postorder_iterator	389
6.20.4.4 operator+	389
6.20.4.5 operator-	389
6.20.5 Member Data Documentation	389
6.20.5.1 e	389
6.20.5.2 i	390
6.21 GiNaC::const_postorder_iterator Class Reference	390
6.21.1 Member Typedef Documentation	390
6.21.1.1 iterator_category	390
6.21.1.2 value_type	391
6.21.1.3 difference_type	391
6.21.1.4 pointer	391
6.21.1.5 reference	391
6.21.2 Constructor & Destructor Documentation	391
6.21.2.1 const_postorder_iterator() [1/2]	391
6.21.2.2 const_postorder_iterator() [2/2]	391
6.21.3 Member Function Documentation	391
6.21.3.1 operator*()	391
6.21.3.2 operator->()	391
6.21.3.3 operator++() [1/2]	392
6.21.3.4 operator++() [2/2]	392

6.21.3.5 operator==()	392
6.21.3.6 operator!=()	392
6.21.3.7 descend()	392
6.21.3.8 increment()	392
6.21.4 Member Data Documentation	393
6.21.4.1 s	393
6.22 GiNaC::const_preorder_iterator Class Reference	393
6.22.1 Member Typedef Documentation	394
6.22.1.1 iterator_category	394
6.22.1.2 value_type	394
6.22.1.3 difference_type	394
6.22.1.4 pointer	394
6.22.1.5 reference	394
6.22.2 Constructor & Destructor Documentation	394
6.22.2.1 const_preorder_iterator() [1/2]	394
6.22.2.2 const_preorder_iterator() [2/2]	394
6.22.3 Member Function Documentation	394
6.22.3.1 operator*()	394
6.22.3.2 operator->()	395
6.22.3.3 operator++() [1/2]	395
6.22.3.4 operator++() [2/2]	395
6.22.3.5 operator==()	395
6.22.3.6 operator!=()	395
6.22.3.7 increment()	395
6.22.4 Member Data Documentation	395
6.22.4.1 s	395
6.23 GiNaC::constant Class Reference	396
6.23.1 Detailed Description	399
6.23.2 Constructor & Destructor Documentation	400
6.23.2.1 constant() [1/2]	400
6.23.2.2 constant() [2/2]	400
6.23.3 Member Function Documentation	400
6.23.3.1 info()	400
6.23.3.2 evalf()	400
6.23.3.3 is_polynomial()	401
6.23.3.4 conjugate()	401
6.23.3.5 real_part()	401
6.23.3.6 imag_part()	401
6.23.3.7 archive()	401
6.23.3.8 read_archive()	402
6.23.3.9 derivative()	402
6.23.3.10 is_equal_same_type()	402

6.23.3.11 calchash()	403
6.23.3.12 get_name()	403
6.23.3.13 get_TeX_name()	403
6.23.3.14 do_print()	403
6.23.3.15 do_print_tree()	403
6.23.3.16 do_print_latex()	403
6.23.3.17 do_print_python_repr()	404
6.23.4 Member Data Documentation	404
6.23.4.1 name	404
6.23.4.2 TeX_name	404
6.23.4.3 ef	404
6.23.4.4 number	404
6.23.4.5 serial	404
6.23.4.6 next_serial	405
6.23.4.7 domain	405
6.24 GiNaC::container< C > Class Template Reference	405
6.24.1 Detailed Description	410
6.24.2 Member Typedef Documentation	410
6.24.2.1 STLT	410
6.24.2.2 const_iterator	411
6.24.2.3 const_reverse_iterator	411
6.24.3 Constructor & Destructor Documentation	411
6.24.3.1 container() [1/4]	411
6.24.3.2 container() [2/4]	411
6.24.3.3 container() [3/4]	411
6.24.3.4 container() [4/4]	411
6.24.4 Member Function Documentation	412
6.24.4.1 get_default_flags()	412
6.24.4.2 get_open_delim()	412
6.24.4.3 get_close_delim()	412
6.24.4.4 info()	412
6.24.4.5 precedence()	412
6.24.4.6 nops()	413
6.24.4.7 op()	413
6.24.4.8 let_op()	413
6.24.4.9 subs()	414
6.24.4.10 read_archive()	414
6.24.4.11 archive()	414
6.24.4.12 conjugate()	415
6.24.4.13 real_part()	415
6.24.4.14 imag_part()	415
6.24.4.15 is_equal_same_type()	415

6.24.4.16 thiscontainer() [1/2]	416
6.24.4.17 thiscontainer() [2/2]	416
6.24.4.18 printseq()	416
6.24.4.19 sort_() [1/2]	416
6.24.4.20 sort_() [2/2]	417
6.24.4.21 unique_() [1/2]	417
6.24.4.22 prepend()	417
6.24.4.23 append()	417
6.24.4.24 remove_first()	417
6.24.4.25 remove_last()	417
6.24.4.26 remove_all()	418
6.24.4.27 sort()	418
6.24.4.28 unique()	418
6.24.4.29 begin()	418
6.24.4.30 end()	418
6.24.4.31 rbegin()	418
6.24.4.32 rend()	419
6.24.4.33 do_print()	419
6.24.4.34 do_print_tree()	419
6.24.4.35 do_print_python()	419
6.24.4.36 do_print_python_repr()	419
6.24.4.37 subschildren()	419
6.24.4.38 unique_() [2/2]	420
6.25 GiNaC::container_storage< C > Class Template Reference	420
6.25.1 Detailed Description	421
6.25.2 Member Typedef Documentation	421
6.25.2.1 STLT	421
6.25.3 Constructor & Destructor Documentation	421
6.25.3.1 container_storage() [1/4]	421
6.25.3.2 container_storage() [2/4]	421
6.25.3.3 container_storage() [3/4]	421
6.25.3.4 container_storage() [4/4]	421
6.25.3.5 ~container_storage()	422
6.25.4 Member Function Documentation	422
6.25.4.1 reserve() [1/4]	422
6.25.4.2 reserve() [2/4]	422
6.25.4.3 reserve() [3/4]	422
6.25.4.4 reserve() [4/4]	422
6.25.5 Member Data Documentation	423
6.25.5.1 seq	423
6.26 GiNaC::composition_generator::coolmulti Struct Reference	423
6.26.1 Constructor & Destructor Documentation	424

6.26.1.1 coolmulti()	424
6.26.1.2 ~coolmulti()	424
6.26.2 Member Function Documentation	424
6.26.2.1 next_permutation()	424
6.26.2.2 finished()	424
6.26.3 Member Data Documentation	424
6.26.3.1 head	424
6.26.3.2 i	424
6.26.3.3 after_i	425
6.27 GiNaC::derivative_map_function Struct Reference	425
6.27.1 Detailed Description	425
6.27.2 Constructor & Destructor Documentation	426
6.27.2.1 derivative_map_function()	426
6.27.3 Member Function Documentation	426
6.27.3.1 operator()()	426
6.27.4 Member Data Documentation	426
6.27.4.1 s	426
6.28 GiNaC::determinant_algo Class Reference	426
6.28.1 Detailed Description	426
6.28.2 Member Enumeration Documentation	426
6.28.2.1 anonymous enum	426
6.29 GiNaC::diracgamma Class Reference	427
6.29.1 Detailed Description	431
6.29.2 Member Function Documentation	431
6.29.2.1 contract_with()	431
6.29.2.2 do_print()	432
6.29.2.3 do_print_latex()	432
6.30 GiNaC::diracgamma5 Class Reference	432
6.30.1 Detailed Description	436
6.30.2 Member Function Documentation	436
6.30.2.1 conjugate()	436
6.30.2.2 do_print()	436
6.30.2.3 do_print_latex()	436
6.31 GiNaC::diracgammaL Class Reference	436
6.31.1 Detailed Description	440
6.31.2 Member Function Documentation	440
6.31.2.1 conjugate()	440
6.31.2.2 do_print()	440
6.31.2.3 do_print_latex()	440
6.32 GiNaC::diracgammaR Class Reference	441
6.32.1 Detailed Description	444
6.32.2 Member Function Documentation	444

6.32.2.1 conjugate()	444
6.32.2.2 do_print()	445
6.32.2.3 do_print_latex()	445
6.33 GiNaC::diracone Class Reference	445
6.33.1 Detailed Description	449
6.33.2 Member Function Documentation	449
6.33.2.1 do_print()	449
6.33.2.2 do_print_latex()	449
6.34 GiNaC::do_taylor Class Reference	449
6.34.1 Detailed Description	449
6.35 GiNaC::domain Class Reference	450
6.35.1 Detailed Description	450
6.35.2 Member Enumeration Documentation	450
6.35.2.1 anonymous enum	450
6.36 GiNaC::dunno Class Reference	450
6.36.1 Detailed Description	450
6.37 GiNaC::Ebar_kernel Class Reference	451
6.37.1 Detailed Description	455
6.37.2 Constructor & Destructor Documentation	455
6.37.2.1 Ebar_kernel()	455
6.37.3 Member Function Documentation	455
6.37.3.1 nops()	455
6.37.3.2 op()	455
6.37.3.3 let_op()	456
6.37.3.4 is_numeric()	456
6.37.3.5 get_numerical_value()	456
6.37.3.6 series_coeff_impl()	456
6.37.3.7 do_print()	457
6.37.4 Member Data Documentation	457
6.37.4.1 n	457
6.37.4.2 m	457
6.37.4.3 x	457
6.37.4.4 y	457
6.38 GiNaC::Eisenstein_h_kernel Class Reference	458
6.38.1 Detailed Description	462
6.38.2 Constructor & Destructor Documentation	462
6.38.2.1 Eisenstein_h_kernel()	462
6.38.3 Member Function Documentation	463
6.38.3.1 series()	463
6.38.3.2 nops()	463
6.38.3.3 op()	463
6.38.3.4 let_op()	463

6.38.3.5 is_numeric()	464
6.38.3.6 Laurent_series()	464
6.38.3.7 get_numerical_value()	464
6.38.3.8 uses_Laurent_series()	464
6.38.3.9 coefficient_a0()	465
6.38.3.10 coefficient_an()	465
6.38.3.11 q_expansion_modular_form()	465
6.38.3.12 do_print()	465
6.38.4 Member Data Documentation	466
6.38.4.1 k	466
6.38.4.2 N	466
6.38.4.3 r	466
6.38.4.4 s	466
6.38.4.5 C_norm	466
6.39 GiNaC::Eisenstein_kernel Class Reference	466
6.39.1 Detailed Description	471
6.39.2 Constructor & Destructor Documentation	471
6.39.2.1 Eisenstein_kernel()	471
6.39.3 Member Function Documentation	471
6.39.3.1 series()	471
6.39.3.2 nops()	472
6.39.3.3 op()	472
6.39.3.4 let_op()	472
6.39.3.5 is_numeric()	472
6.39.3.6 Laurent_series()	472
6.39.3.7 get_numerical_value()	473
6.39.3.8 uses_Laurent_series()	473
6.39.3.9 q_expansion_modular_form()	473
6.39.3.10 do_print()	473
6.39.4 Member Data Documentation	473
6.39.4.1 k	473
6.39.4.2 N	474
6.39.4.3 a	474
6.39.4.4 b	474
6.39.4.5 K	474
6.39.4.6 C_norm	474
6.40 GiNaC::composition_generator::coolmulti::element Struct Reference	474
6.40.1 Constructor & Destructor Documentation	475
6.40.1.1 element()	475
6.40.1.2 ~element()	475
6.40.2 Member Data Documentation	475
6.40.2.1 value	475

6.40.2.2 next	475
6.41 GiNaC::ELi_kernel Class Reference	476
6.41.1 Detailed Description	480
6.41.2 Constructor & Destructor Documentation	480
6.41.2.1 ELi_kernel()	480
6.41.3 Member Function Documentation	480
6.41.3.1 nops()	480
6.41.3.2 op()	480
6.41.3.3 let_op()	481
6.41.3.4 is_numeric()	481
6.41.3.5 get_numerical_value()	481
6.41.3.6 series_coeff_impl()	481
6.41.3.7 do_print()	482
6.41.4 Member Data Documentation	482
6.41.4.1 n	482
6.41.4.2 m	482
6.41.4.3 x	482
6.41.4.4 y	482
6.42 std::equal_to< GiNaC::ex > Struct Reference	482
6.42.1 Detailed Description	483
6.42.2 Member Function Documentation	483
6.42.2.1 operator()	483
6.43 GiNaC::error_and_integral Struct Reference	483
6.43.1 Constructor & Destructor Documentation	483
6.43.1.1 error_and_integral()	483
6.43.2 Member Data Documentation	483
6.43.2.1 error	483
6.43.2.2 integral	484
6.44 GiNaC::error_and_integral_is_less Struct Reference	484
6.44.1 Member Function Documentation	484
6.44.1.1 operator()	484
6.45 GiNaC::eval_integ_map_function Struct Reference	484
6.45.1 Detailed Description	485
6.45.2 Member Function Documentation	485
6.45.2.1 operator()	485
6.46 GiNaC::evalf_map_function Struct Reference	485
6.46.1 Detailed Description	486
6.46.2 Member Function Documentation	486
6.46.2.1 operator()	486
6.47 GiNaC::evalm_map_function Struct Reference	486
6.47.1 Detailed Description	487
6.47.2 Member Function Documentation	487

6.47.2.1 operator>()	487
6.48 GiNaC::ex Class Reference	487
6.48.1 Detailed Description	491
6.48.2 Constructor & Destructor Documentation	491
6.48.2.1 ex() [1/10]	491
6.48.2.2 ex() [2/10]	491
6.48.2.3 ex() [3/10]	491
6.48.2.4 ex() [4/10]	492
6.48.2.5 ex() [5/10]	492
6.48.2.6 ex() [6/10]	492
6.48.2.7 ex() [7/10]	492
6.48.2.8 ex() [8/10]	492
6.48.2.9 ex() [9/10]	492
6.48.2.10 ex() [10/10]	493
6.48.3 Member Function Documentation	493
6.48.3.1 swap()	493
6.48.3.2 begin()	493
6.48.3.3 end()	493
6.48.3.4 preorder_begin()	493
6.48.3.5 preorder_end()	494
6.48.3.6 postorder_begin()	494
6.48.3.7 postorder_end()	494
6.48.3.8 eval()	494
6.48.3.9 evalf()	494
6.48.3.10 evalm()	494
6.48.3.11 eval_ncmul()	495
6.48.3.12 eval_integ()	495
6.48.3.13 print()	495
6.48.3.14 dbgprint()	496
6.48.3.15 dbgprinttree()	496
6.48.3.16 info()	496
6.48.3.17 nops()	497
6.48.3.18 op()	497
6.48.3.19 operator[]() [1/4]	497
6.48.3.20 operator[]() [2/4]	498
6.48.3.21 let_op()	498
6.48.3.22 operator[]() [3/4]	498
6.48.3.23 operator[]() [4/4]	498
6.48.3.24 lhs()	498
6.48.3.25 rhs()	498
6.48.3.26 conjugate()	499
6.48.3.27 real_part()	499

6.48.3.28 <code>imag_part()</code>	499
6.48.3.29 <code>has()</code>	499
6.48.3.30 <code>find()</code>	500
6.48.3.31 <code>match()</code> [1/2]	500
6.48.3.32 <code>match()</code> [2/2]	500
6.48.3.33 <code>subs()</code> [1/3]	500
6.48.3.34 <code>subs()</code> [2/3]	501
6.48.3.35 <code>subs()</code> [3/3]	501
6.48.3.36 <code>map()</code> [1/2]	501
6.48.3.37 <code>map()</code> [2/2]	501
6.48.3.38 <code>accept()</code>	501
6.48.3.39 <code>traverse_preorder()</code>	502
6.48.3.40 <code>traverse_postorder()</code>	502
6.48.3.41 <code>traverse()</code>	502
6.48.3.42 <code>is_polynomial()</code>	502
6.48.3.43 <code>degree()</code>	502
6.48.3.44 <code>ldegree()</code>	503
6.48.3.45 <code>coeff()</code>	503
6.48.3.46 <code>lcoeff()</code>	503
6.48.3.47 <code>tcoeff()</code>	503
6.48.3.48 <code>expand()</code>	503
6.48.3.49 <code>collect()</code>	504
6.48.3.50 <code>diff()</code>	504
6.48.3.51 <code>series()</code>	505
6.48.3.52 <code>normal()</code>	505
6.48.3.53 <code>to_rational()</code>	506
6.48.3.54 <code>to_polynomial()</code>	506
6.48.3.55 <code>numer()</code>	506
6.48.3.56 <code>denom()</code>	507
6.48.3.57 <code>numer_denom()</code>	507
6.48.3.58 <code>unit()</code>	507
6.48.3.59 <code>content()</code>	508
6.48.3.60 <code>integer_content()</code>	508
6.48.3.61 <code>primpart()</code> [1/2]	509
6.48.3.62 <code>primpart()</code> [2/2]	509
6.48.3.63 <code>unitcontprim()</code>	510
6.48.3.64 <code>smod()</code>	510
6.48.3.65 <code>max_coefficient()</code>	510
6.48.3.66 <code>get_free_indices()</code>	511
6.48.3.67 <code>simplify_indexed()</code> [1/2]	511
6.48.3.68 <code>simplify_indexed()</code> [2/2]	511
6.48.3.69 <code>compare()</code>	512

6.48.3.70 is_equal()	512
6.48.3.71 is_zero()	513
6.48.3.72 is_zero_matrix()	513
6.48.3.73 symmetrize() [1/2]	513
6.48.3.74 symmetrize() [2/2]	513
6.48.3.75 antisymmetrize() [1/2]	514
6.48.3.76 antisymmetrize() [2/2]	514
6.48.3.77 symmetrize_cyclic() [1/2]	514
6.48.3.78 symmetrize_cyclic() [2/2]	514
6.48.3.79 return_type()	514
6.48.3.80 return_type_tinfo()	515
6.48.3.81 gethash()	515
6.48.3.82 construct_from_basic()	515
6.48.3.83 construct_from_int()	515
6.48.3.84 construct_from_uint()	516
6.48.3.85 construct_from_long()	516
6.48.3.86 construct_from_ulong()	516
6.48.3.87 construct_from_longlong()	516
6.48.3.88 construct_from_ulonglong()	516
6.48.3.89 construct_from_double()	516
6.48.3.90 construct_from_string_and_lst()	517
6.48.3.91 makewritable()	517
6.48.3.92 share()	517
6.48.4 Friends And Related Symbol Documentation	517
6.48.4.1 archive_node	517
6.48.4.2 are_ex_trivially_equal	517
6.48.4.3 ex_to	517
6.48.4.4 is_a	518
6.48.4.5 is_exactly_a	518
6.48.5 Member Data Documentation	518
6.48.5.1 bp	518
6.49 GiNaC::ex_base_is_less Struct Reference	519
6.49.1 Member Function Documentation	519
6.49.1.1 operator()()	519
6.50 GiNaC::ex_is_equal Struct Reference	519
6.50.1 Member Function Documentation	519
6.50.1.1 operator()()	519
6.51 GiNaC::ex_is_less Struct Reference	519
6.51.1 Member Function Documentation	520
6.51.1.1 operator()()	520
6.52 GiNaC::ex_swap Struct Reference	520
6.52.1 Member Function Documentation	520

6.52.1.1 operator()	520
6.53 GiNaC::expair Class Reference	520
6.53.1 Detailed Description	521
6.53.2 Constructor & Destructor Documentation	521
6.53.2.1 expair() [1/2]	521
6.53.2.2 expair() [2/2]	521
6.53.3 Member Function Documentation	522
6.53.3.1 is_equal()	522
6.53.3.2 is_less()	522
6.53.3.3 compare()	522
6.53.3.4 print()	522
6.53.3.5 is_canonical_numeric()	522
6.53.3.6 swap()	523
6.53.3.7 conjugate()	523
6.53.4 Member Data Documentation	523
6.53.4.1 rest	523
6.53.4.2 coeff	523
6.54 GiNaC::expair_is_less Struct Reference	524
6.54.1 Detailed Description	524
6.54.2 Member Function Documentation	524
6.54.2.1 operator()	524
6.55 GiNaC::expair_rest_is_less Struct Reference	524
6.55.1 Detailed Description	524
6.55.2 Member Function Documentation	525
6.55.2.1 operator()	525
6.56 GiNaC::expair_swap Struct Reference	525
6.56.1 Member Function Documentation	525
6.56.1.1 operator()	525
6.57 GiNaC::expairseq Class Reference	525
6.57.1 Detailed Description	530
6.57.2 Constructor & Destructor Documentation	530
6.57.2.1 expairseq() [1/4]	530
6.57.2.2 expairseq() [2/4]	530
6.57.2.3 expairseq() [3/4]	530
6.57.2.4 expairseq() [4/4]	530
6.57.3 Member Function Documentation	531
6.57.3.1 precedence()	531
6.57.3.2 info()	531
6.57.3.3 nops()	531
6.57.3.4 op()	532
6.57.3.5 map()	532
6.57.3.6 eval()	532

6.57.3.7 to_rational()	532
6.57.3.8 to_polynomial()	533
6.57.3.9 match()	533
6.57.3.10 subs()	533
6.57.3.11 conjugate()	533
6.57.3.12 archive()	534
6.57.3.13 read_archive()	534
6.57.3.14 is_equal_same_type()	534
6.57.3.15 return_type()	535
6.57.3.16 calchash()	535
6.57.3.17 expand()	535
6.57.3.18 thisexpairseq() [1/2]	535
6.57.3.19 thisexpairseq() [2/2]	536
6.57.3.20 printseq()	536
6.57.3.21 printpair()	536
6.57.3.22 split_ex_to_pair()	536
6.57.3.23 combine_ex_with_coeff_to_pair()	537
6.57.3.24 combine_pair_with_coeff_to_pair()	537
6.57.3.25 recombine_pair_to_ex()	537
6.57.3.26 expair_needs_further_processing()	537
6.57.3.27 default_overall_coeff()	538
6.57.3.28 combine_overall_coeff() [1/2]	538
6.57.3.29 combine_overall_coeff() [2/2]	538
6.57.3.30 can_make_flat()	538
6.57.3.31 do_print()	538
6.57.3.32 do_print_tree()	539
6.57.3.33 construct_from_2_ex()	539
6.57.3.34 construct_from_2_expairseq()	539
6.57.3.35 construct_from_expairseq_ex()	539
6.57.3.36 construct_from_exvector()	539
6.57.3.37 construct_from_epvector() [1/2]	540
6.57.3.38 construct_from_epvector() [2/2]	540
6.57.3.39 make_flat() [1/2]	540
6.57.3.40 make_flat() [2/2]	540
6.57.3.41 canonicalize()	541
6.57.3.42 combine_same_terms_sorted_seq()	541
6.57.3.43 is_canonical()	541
6.57.3.44 expandchildren()	541
6.57.3.45 evalchildren()	542
6.57.3.46 subschildren()	542
6.57.4 Member Data Documentation	542
6.57.4.1 seq	542

6.57.4.2 overall_coeff	543
6.58 GiNaC::expand_map_function Struct Reference	543
6.58.1 Detailed Description	544
6.58.2 Constructor & Destructor Documentation	544
6.58.2.1 expand_map_function()	544
6.58.3 Member Function Documentation	544
6.58.3.1 operator()()	544
6.58.4 Member Data Documentation	544
6.58.4.1 options	544
6.59 GiNaC::expand_options Class Reference	544
6.59.1 Detailed Description	545
6.59.2 Member Enumeration Documentation	545
6.59.2.1 anonymous enum	545
6.60 GiNaC::factor_options Class Reference	545
6.60.1 Detailed Description	545
6.60.2 Member Enumeration Documentation	545
6.60.2.1 anonymous enum	545
6.61 GiNaC::fail Class Reference	546
6.61.1 Member Function Documentation	549
6.61.1.1 return_type()	549
6.61.1.2 do_print()	549
6.62 GiNaC::fderivative Class Reference	550
6.62.1 Detailed Description	557
6.62.2 Constructor & Destructor Documentation	557
6.62.2.1 fderivative() [1/3]	557
6.62.2.2 fderivative() [2/3]	557
6.62.2.3 fderivative() [3/3]	558
6.62.3 Member Function Documentation	558
6.62.3.1 print()	558
6.62.3.2 eval()	558
6.62.3.3 series()	559
6.62.3.4 thiscontainer() [1/2]	559
6.62.3.5 thiscontainer() [2/2]	559
6.62.3.6 archive()	559
6.62.3.7 read_archive()	560
6.62.3.8 derivative()	560
6.62.3.9 is_equal_same_type()	560
6.62.3.10 match_same_type()	561
6.62.3.11 derivatives()	561
6.62.3.12 do_print()	561
6.62.3.13 do_print_latex()	561
6.62.3.14 do_print_csrc()	562

6.62.3.15 do_print_tree()	562
6.62.4 Member Data Documentation	562
6.62.4.1 parameter_set	562
6.63 GiNaC::function Class Reference	562
6.63.1 Detailed Description	569
6.63.2 Constructor & Destructor Documentation	569
6.63.2.1 function() [1/18]	569
6.63.2.2 function() [2/18]	569
6.63.2.3 function() [3/18]	569
6.63.2.4 function() [4/18]	569
6.63.2.5 function() [5/18]	569
6.63.2.6 function() [6/18]	570
6.63.2.7 function() [7/18]	570
6.63.2.8 function() [8/18]	570
6.63.2.9 function() [9/18]	570
6.63.2.10 function() [10/18]	571
6.63.2.11 function() [11/18]	571
6.63.2.12 function() [12/18]	571
6.63.2.13 function() [13/18]	571
6.63.2.14 function() [14/18]	572
6.63.2.15 function() [15/18]	572
6.63.2.16 function() [16/18]	572
6.63.2.17 function() [17/18]	573
6.63.2.18 function() [18/18]	573
6.63.3 Member Function Documentation	573
6.63.3.1 print()	573
6.63.3.2 precedence()	573
6.63.3.3 expand()	574
6.63.3.4 eval()	574
6.63.3.5 evalf()	574
6.63.3.6 eval_ncmul()	574
6.63.3.7 calchash()	575
6.63.3.8 series()	575
6.63.3.9 thiscontainer() [1/2]	575
6.63.3.10 thiscontainer() [2/2]	575
6.63.3.11 conjugate()	576
6.63.3.12 real_part()	576
6.63.3.13 imag_part()	576
6.63.3.14 archive()	576
6.63.3.15 read_archive()	577
6.63.3.16 info()	577
6.63.3.17 derivative()	577

6.63.3.18	is_equal_same_type()	577
6.63.3.19	match_same_type()	578
6.63.3.20	return_type()	578
6.63.3.21	return_type_tinfo()	578
6.63.3.22	pderivative()	578
6.63.3.23	expl_derivative()	579
6.63.3.24	registered_functions()	579
6.63.3.25	lookup_remember_table()	579
6.63.3.26	store_remember_table()	579
6.63.3.27	power()	579
6.63.3.28	register_new()	579
6.63.3.29	find_function()	580
6.63.3.30	get_registered_functions()	580
6.63.3.31	get_serial()	580
6.63.3.32	get_name()	580
6.63.4	Friends And Related Symbol Documentation	580
6.63.4.1	remember_table_entry	580
6.63.5	Member Data Documentation	580
6.63.5.1	current_serial	580
6.63.5.2	serial	581
6.64	GiNaC::function_options Class Reference	581
6.64.1	Constructor & Destructor Documentation	586
6.64.1.1	function_options() [1/3]	586
6.64.1.2	function_options() [2/3]	586
6.64.1.3	function_options() [3/3]	586
6.64.1.4	~function_options()	586
6.64.2	Member Function Documentation	587
6.64.2.1	initialize()	587
6.64.2.2	dummy()	587
6.64.2.3	set_name()	587
6.64.2.4	latex_name()	587
6.64.2.5	eval_func() [1/15]	587
6.64.2.6	eval_func() [2/15]	587
6.64.2.7	eval_func() [3/15]	588
6.64.2.8	eval_func() [4/15]	588
6.64.2.9	eval_func() [5/15]	588
6.64.2.10	eval_func() [6/15]	588
6.64.2.11	eval_func() [7/15]	588
6.64.2.12	eval_func() [8/15]	588
6.64.2.13	eval_func() [9/15]	588
6.64.2.14	eval_func() [10/15]	589
6.64.2.15	eval_func() [11/15]	589

6.64.2.16 eval_func() [12/15]	589
6.64.2.17 eval_func() [13/15]	589
6.64.2.18 eval_func() [14/15]	589
6.64.2.19 evalf_func() [1/15]	589
6.64.2.20 evalf_func() [2/15]	589
6.64.2.21 evalf_func() [3/15]	590
6.64.2.22 evalf_func() [4/15]	590
6.64.2.23 evalf_func() [5/15]	590
6.64.2.24 evalf_func() [6/15]	590
6.64.2.25 evalf_func() [7/15]	590
6.64.2.26 evalf_func() [8/15]	590
6.64.2.27 evalf_func() [9/15]	590
6.64.2.28 evalf_func() [10/15]	591
6.64.2.29 evalf_func() [11/15]	591
6.64.2.30 evalf_func() [12/15]	591
6.64.2.31 evalf_func() [13/15]	591
6.64.2.32 evalf_func() [14/15]	591
6.64.2.33 conjugate_func() [1/15]	591
6.64.2.34 conjugate_func() [2/15]	591
6.64.2.35 conjugate_func() [3/15]	592
6.64.2.36 conjugate_func() [4/15]	592
6.64.2.37 conjugate_func() [5/15]	592
6.64.2.38 conjugate_func() [6/15]	592
6.64.2.39 conjugate_func() [7/15]	592
6.64.2.40 conjugate_func() [8/15]	592
6.64.2.41 conjugate_func() [9/15]	592
6.64.2.42 conjugate_func() [10/15]	593
6.64.2.43 conjugate_func() [11/15]	593
6.64.2.44 conjugate_func() [12/15]	593
6.64.2.45 conjugate_func() [13/15]	593
6.64.2.46 conjugate_func() [14/15]	593
6.64.2.47 real_part_func() [1/15]	593
6.64.2.48 real_part_func() [2/15]	593
6.64.2.49 real_part_func() [3/15]	594
6.64.2.50 real_part_func() [4/15]	594
6.64.2.51 real_part_func() [5/15]	594
6.64.2.52 real_part_func() [6/15]	594
6.64.2.53 real_part_func() [7/15]	594
6.64.2.54 real_part_func() [8/15]	594
6.64.2.55 real_part_func() [9/15]	594
6.64.2.56 real_part_func() [10/15]	595
6.64.2.57 real_part_func() [11/15]	595

6.64.2.58 <code>real_part_func()</code> [12/15]	595
6.64.2.59 <code>real_part_func()</code> [13/15]	595
6.64.2.60 <code>real_part_func()</code> [14/15]	595
6.64.2.61 <code>imag_part_func()</code> [1/15]	595
6.64.2.62 <code>imag_part_func()</code> [2/15]	595
6.64.2.63 <code>imag_part_func()</code> [3/15]	596
6.64.2.64 <code>imag_part_func()</code> [4/15]	596
6.64.2.65 <code>imag_part_func()</code> [5/15]	596
6.64.2.66 <code>imag_part_func()</code> [6/15]	596
6.64.2.67 <code>imag_part_func()</code> [7/15]	596
6.64.2.68 <code>imag_part_func()</code> [8/15]	596
6.64.2.69 <code>imag_part_func()</code> [9/15]	596
6.64.2.70 <code>imag_part_func()</code> [10/15]	597
6.64.2.71 <code>imag_part_func()</code> [11/15]	597
6.64.2.72 <code>imag_part_func()</code> [12/15]	597
6.64.2.73 <code>imag_part_func()</code> [13/15]	597
6.64.2.74 <code>imag_part_func()</code> [14/15]	597
6.64.2.75 <code>expand_func()</code> [1/15]	597
6.64.2.76 <code>expand_func()</code> [2/15]	597
6.64.2.77 <code>expand_func()</code> [3/15]	598
6.64.2.78 <code>expand_func()</code> [4/15]	598
6.64.2.79 <code>expand_func()</code> [5/15]	598
6.64.2.80 <code>expand_func()</code> [6/15]	598
6.64.2.81 <code>expand_func()</code> [7/15]	598
6.64.2.82 <code>expand_func()</code> [8/15]	598
6.64.2.83 <code>expand_func()</code> [9/15]	598
6.64.2.84 <code>expand_func()</code> [10/15]	599
6.64.2.85 <code>expand_func()</code> [11/15]	599
6.64.2.86 <code>expand_func()</code> [12/15]	599
6.64.2.87 <code>expand_func()</code> [13/15]	599
6.64.2.88 <code>expand_func()</code> [14/15]	599
6.64.2.89 <code>derivative_func()</code> [1/15]	599
6.64.2.90 <code>derivative_func()</code> [2/15]	599
6.64.2.91 <code>derivative_func()</code> [3/15]	600
6.64.2.92 <code>derivative_func()</code> [4/15]	600
6.64.2.93 <code>derivative_func()</code> [5/15]	600
6.64.2.94 <code>derivative_func()</code> [6/15]	600
6.64.2.95 <code>derivative_func()</code> [7/15]	600
6.64.2.96 <code>derivative_func()</code> [8/15]	600
6.64.2.97 <code>derivative_func()</code> [9/15]	600
6.64.2.98 <code>derivative_func()</code> [10/15]	601
6.64.2.99 <code>derivative_func()</code> [11/15]	601

6.64.2.100 derivative_func() [12/15]	601
6.64.2.101 derivative_func() [13/15]	601
6.64.2.102 derivative_func() [14/15]	601
6.64.2.103 expl_derivative_func() [1/15]	601
6.64.2.104 expl_derivative_func() [2/15]	601
6.64.2.105 expl_derivative_func() [3/15]	602
6.64.2.106 expl_derivative_func() [4/15]	602
6.64.2.107 expl_derivative_func() [5/15]	602
6.64.2.108 expl_derivative_func() [6/15]	602
6.64.2.109 expl_derivative_func() [7/15]	602
6.64.2.110 expl_derivative_func() [8/15]	602
6.64.2.111 expl_derivative_func() [9/15]	602
6.64.2.112 expl_derivative_func() [10/15]	603
6.64.2.113 expl_derivative_func() [11/15]	603
6.64.2.114 expl_derivative_func() [12/15]	603
6.64.2.115 expl_derivative_func() [13/15]	603
6.64.2.116 expl_derivative_func() [14/15]	603
6.64.2.117 power_func() [1/15]	603
6.64.2.118 power_func() [2/15]	603
6.64.2.119 power_func() [3/15]	604
6.64.2.120 power_func() [4/15]	604
6.64.2.121 power_func() [5/15]	604
6.64.2.122 power_func() [6/15]	604
6.64.2.123 power_func() [7/15]	604
6.64.2.124 power_func() [8/15]	604
6.64.2.125 power_func() [9/15]	604
6.64.2.126 power_func() [10/15]	605
6.64.2.127 power_func() [11/15]	605
6.64.2.128 power_func() [12/15]	605
6.64.2.129 power_func() [13/15]	605
6.64.2.130 power_func() [14/15]	605
6.64.2.131 series_func() [1/15]	605
6.64.2.132 series_func() [2/15]	605
6.64.2.133 series_func() [3/15]	606
6.64.2.134 series_func() [4/15]	606
6.64.2.135 series_func() [5/15]	606
6.64.2.136 series_func() [6/15]	606
6.64.2.137 series_func() [7/15]	606
6.64.2.138 series_func() [8/15]	606
6.64.2.139 series_func() [9/15]	606
6.64.2.140 series_func() [10/15]	607
6.64.2.141 series_func() [11/15]	607

6.64.2.142 series_func() [12/15]	607
6.64.2.143 series_func() [13/15]	607
6.64.2.144 series_func() [14/15]	607
6.64.2.145 info_func() [1/15]	607
6.64.2.146 info_func() [2/15]	607
6.64.2.147 info_func() [3/15]	608
6.64.2.148 info_func() [4/15]	608
6.64.2.149 info_func() [5/15]	608
6.64.2.150 info_func() [6/15]	608
6.64.2.151 info_func() [7/15]	608
6.64.2.152 info_func() [8/15]	608
6.64.2.153 info_func() [9/15]	608
6.64.2.154 info_func() [10/15]	609
6.64.2.155 info_func() [11/15]	609
6.64.2.156 info_func() [12/15]	609
6.64.2.157 info_func() [13/15]	609
6.64.2.158 info_func() [14/15]	609
6.64.2.159 eval_func() [15/15]	609
6.64.2.160 evalf_func() [15/15]	609
6.64.2.161 conjugate_func() [15/15]	610
6.64.2.162 real_part_func() [15/15]	610
6.64.2.163 imag_part_func() [15/15]	610
6.64.2.164 expand_func() [15/15]	610
6.64.2.165 derivative_func() [15/15]	610
6.64.2.166 expl_derivative_func() [15/15]	610
6.64.2.167 power_func() [15/15]	610
6.64.2.168 series_func() [15/15]	611
6.64.2.169 info_func() [15/15]	611
6.64.2.170 print_func() [1/15]	611
6.64.2.171 print_func() [2/15]	611
6.64.2.172 print_func() [3/15]	611
6.64.2.173 print_func() [4/15]	611
6.64.2.174 print_func() [5/15]	612
6.64.2.175 print_func() [6/15]	612
6.64.2.176 print_func() [7/15]	612
6.64.2.177 print_func() [8/15]	612
6.64.2.178 print_func() [9/15]	612
6.64.2.179 print_func() [10/15]	612
6.64.2.180 print_func() [11/15]	613
6.64.2.181 print_func() [12/15]	613
6.64.2.182 print_func() [13/15]	613
6.64.2.183 print_func() [14/15]	613

6.64.2.184 print_func() [15/15]	613
6.64.2.185 set_return_type()	613
6.64.2.186 do_not_evalf_params()	614
6.64.2.187 remember()	614
6.64.2.188 overloaded()	614
6.64.2.189 set_symmetry()	614
6.64.2.190 get_name()	614
6.64.2.191 get_nparams()	614
6.64.2.192 has_derivative()	614
6.64.2.193 has_power()	615
6.64.2.194 test_and_set_nparams()	615
6.64.2.195 set_print_func()	615
6.64.3 Friends And Related Symbol Documentation	615
6.64.3.1 function	615
6.64.3.2 fderivative	616
6.64.4 Member Data Documentation	616
6.64.4.1 name	616
6.64.4.2 TeX_name	616
6.64.4.3 nparams	616
6.64.4.4 eval_f	616
6.64.4.5 evalf_f	616
6.64.4.6 conjugate_f	617
6.64.4.7 real_part_f	617
6.64.4.8 imag_part_f	617
6.64.4.9 expand_f	617
6.64.4.10 derivative_f	617
6.64.4.11 expl_derivative_f	617
6.64.4.12 power_f	618
6.64.4.13 series_f	618
6.64.4.14 print_dispatch_table	618
6.64.4.15 info_f	618
6.64.4.16 evalf_params_first	618
6.64.4.17 use_return_type	618
6.64.4.18 return_type	618
6.64.4.19 return_type_tinfo	619
6.64.4.20 use_remember	619
6.64.4.21 remember_size	619
6.64.4.22 remember_assoc_size	619
6.64.4.23 remember_strategy	619
6.64.4.24 eval_use_exvector_args	619
6.64.4.25 evalf_use_exvector_args	619
6.64.4.26 conjugate_use_exvector_args	619

6.64.4.27	real_part_use_exvector_args	620
6.64.4.28	imag_part_use_exvector_args	620
6.64.4.29	expand_use_exvector_args	620
6.64.4.30	derivative_use_exvector_args	620
6.64.4.31	expl_derivative_use_exvector_args	620
6.64.4.32	power_use_exvector_args	620
6.64.4.33	series_use_exvector_args	620
6.64.4.34	print_use_exvector_args	620
6.64.4.35	info_use_exvector_args	621
6.64.4.36	functions_with_same_name	621
6.64.4.37	symtree	621
6.65	GiNaC::G2_SERIAL Class Reference	621
6.65.1	Detailed Description	621
6.65.2	Member Data Documentation	621
6.65.2.1	serial	621
6.66	GiNaC::G3_SERIAL Class Reference	622
6.66.1	Detailed Description	622
6.66.2	Member Data Documentation	622
6.66.2.1	serial	622
6.67	GiNaC::gcd_options Struct Reference	622
6.67.1	Detailed Description	622
6.67.2	Member Enumeration Documentation	622
6.67.2.1	anonymous enum	622
6.68	GiNaC::gcdheu_failed Class Reference	623
6.68.1	Detailed Description	623
6.69	GiNaC::Gt Class Reference	623
6.69.1	Detailed Description	628
6.69.2	Constructor & Destructor Documentation	628
6.69.2.1	Gt() [1/2]	628
6.69.2.2	Gt() [2/2]	628
6.69.3	Member Function Documentation	628
6.69.3.1	setArgs()	628
6.69.3.2	getTau()	629
6.69.3.3	getZ()	629
6.69.3.4	getArgs()	629
6.69.3.5	precedence()	629
6.69.3.6	eval()	629
6.69.3.7	evalf()	629
6.69.3.8	nops()	630
6.69.3.9	subs()	630
6.69.3.10	calchash()	630
6.69.3.11	has()	630

6.69.3.12	match()	631
6.69.3.13	archive()	631
6.69.3.14	read_archive()	631
6.69.3.15	zisToFundamental()	631
6.69.3.16	regularise()	632
6.69.3.17	findMoebiusTransform()	632
6.69.3.18	tauToFundamental()	632
6.69.3.19	decomposeIntegrationPath()	632
6.69.3.20	applyIntegrationPath()	632
6.69.3.21	qExpand()	633
6.69.3.22	evaluate()	633
6.69.3.23	ex_zisToFundamental() [1/2]	633
6.69.3.24	ex_regularise() [1/2]	633
6.69.3.25	ex_tauToFundamental() [1/2]	633
6.69.3.26	ex_cutIntegrationPath() [1/2]	634
6.69.3.27	ex_qExpand() [1/2]	634
6.69.3.28	ex_prepare() [1/2]	634
6.69.3.29	ex_evaluate() [1/2]	634
6.69.3.30	lst_evaluate() [1/2]	634
6.69.3.31	ex_zisToFundamental() [2/2]	635
6.69.3.32	ex_regularise() [2/2]	635
6.69.3.33	ex_tauToFundamental() [2/2]	635
6.69.3.34	ex_cutIntegrationPath() [2/2]	635
6.69.3.35	ex_qExpand() [2/2]	635
6.69.3.36	ex_prepare() [2/2]	635
6.69.3.37	ex_evaluate() [2/2]	636
6.69.3.38	lst_evaluate() [2/2]	636
6.69.3.39	apply_function_recursive()	636
6.69.3.40	clear_polylog_cache()	636
6.69.3.41	do_print()	636
6.69.3.42	do_print_latex()	636
6.69.3.43	to_numeric()	637
6.69.3.44	to_integer()	637
6.69.4	Member Data Documentation	637
6.69.4.1	cutIntegrationPath_threshold_real	637
6.69.4.2	cutIntegrationPath_threshold_imag	637
6.69.4.3	qExpand_minOrder	637
6.69.4.4	qExpand_stepSize	637
6.69.4.5	auto_clear_polylog_cache	638
6.69.4.6	enable_tauToFundamental	638
6.69.4.7	nargs	638
6.69.4.8	args	638

6.69.4.9 z	638
6.69.4.10 tau	638
6.69.4.11 ngexand_transformer	639
6.70 GiNaC::has_distance< T > Class Template Reference	639
6.70.1 Detailed Description	639
6.70.2 Member Typedef Documentation	640
6.70.2.1 yes_type	640
6.70.2.2 no_type	640
6.70.3 Member Enumeration Documentation	640
6.70.3.1 anonymous enum	640
6.70.4 Member Function Documentation	640
6.70.4.1 test() [1/2]	640
6.70.4.2 test() [2/2]	640
6.71 GiNaC::has_options Class Reference	641
6.71.1 Detailed Description	641
6.71.2 Member Enumeration Documentation	641
6.71.2.1 anonymous enum	641
6.72 std::hash< GiNaC::ex > Struct Reference	641
6.72.1 Detailed Description	641
6.72.2 Member Function Documentation	642
6.72.2.1 operator()()	642
6.73 GiNaC::idx Class Reference	642
6.73.1 Detailed Description	646
6.73.2 Constructor & Destructor Documentation	646
6.73.2.1 idx()	646
6.73.3 Member Function Documentation	646
6.73.3.1 info()	646
6.73.3.2 nops()	647
6.73.3.3 op()	647
6.73.3.4 map()	647
6.73.3.5 evalf()	647
6.73.3.6 subs()	648
6.73.3.7 archive()	648
6.73.3.8 read_archive()	648
6.73.3.9 derivative()	649
6.73.3.10 match_same_type()	649
6.73.3.11 calchash()	649
6.73.3.12 is_dummy_pair_same_type()	650
6.73.3.13 get_value()	650
6.73.3.14 is_numeric()	650
6.73.3.15 is_symbolic()	650
6.73.3.16 get_dim()	650

6.73.3.17 is_dim_numeric()	651
6.73.3.18 is_dim_symbolic()	651
6.73.3.19 replace_dim()	651
6.73.3.20 minimal_dim()	651
6.73.3.21 print_index()	651
6.73.3.22 do_print()	652
6.73.3.23 do_print_csrc()	652
6.73.3.24 do_print_latex()	652
6.73.3.25 do_print_tree()	652
6.73.4 Member Data Documentation	652
6.73.4.1 value	652
6.73.4.2 dim	653
6.74 GiNaC::idx_is_equal_ignore_dim Struct Reference	653
6.74.1 Member Function Documentation	653
6.74.1.1 operator()	653
6.75 GiNaC::indexed Class Reference	653
6.75.1 Detailed Description	660
6.75.2 Constructor & Destructor Documentation	660
6.75.2.1 indexed() [1/13]	660
6.75.2.2 indexed() [2/13]	660
6.75.2.3 indexed() [3/13]	661
6.75.2.4 indexed() [4/13]	661
6.75.2.5 indexed() [5/13]	661
6.75.2.6 indexed() [6/13]	662
6.75.2.7 indexed() [7/13]	662
6.75.2.8 indexed() [8/13]	663
6.75.2.9 indexed() [9/13]	663
6.75.2.10 indexed() [10/13]	664
6.75.2.11 indexed() [11/13]	664
6.75.2.12 indexed() [12/13]	664
6.75.2.13 indexed() [13/13]	664
6.75.3 Member Function Documentation	664
6.75.3.1 precedence()	664
6.75.3.2 info()	665
6.75.3.3 eval()	665
6.75.3.4 real_part()	665
6.75.3.5 imag_part()	665
6.75.3.6 get_free_indices()	665
6.75.3.7 archive()	666
6.75.3.8 read_archive()	666
6.75.3.9 derivative()	666
6.75.3.10 thiscontainer() [1/2]	666

6.75.3.11 thiscontainer() [2/2]	667
6.75.3.12 return_type()	667
6.75.3.13 return_type_tinfo()	667
6.75.3.14 expand()	667
6.75.3.15 all_index_values_are()	667
6.75.3.16 get_indices()	668
6.75.3.17 get_dummy_indices() [1/2]	668
6.75.3.18 get_dummy_indices() [2/2]	668
6.75.3.19 has_dummy_index_for()	668
6.75.3.20 get_symmetry()	668
6.75.3.21 printindices()	669
6.75.3.22 print_indexed()	669
6.75.3.23 do_print()	669
6.75.3.24 do_print_latex()	669
6.75.3.25 do_print_tree()	669
6.75.3.26 validate()	670
6.75.4 Friends And Related Symbol Documentation	670
6.75.4.1 simplify_indexed	670
6.75.4.2 simplify_indexed_product	670
6.75.4.3 reposition_dummy_indices	670
6.75.5 Member Data Documentation	671
6.75.5.1 symtree	671
6.76 GiNaC::info_flags Class Reference	671
6.76.1 Detailed Description	671
6.76.2 Member Enumeration Documentation	671
6.76.2.1 anonymous enum	671
6.77 GiNaC::integral Class Reference	672
6.77.1 Detailed Description	676
6.77.2 Constructor & Destructor Documentation	676
6.77.2.1 integral()	676
6.77.3 Member Function Documentation	677
6.77.3.1 precedence()	677
6.77.3.2 eval()	677
6.77.3.3 evalf()	677
6.77.3.4 degree()	677
6.77.3.5 ldegree()	677
6.77.3.6 eval_ncmul()	678
6.77.3.7 nops()	678
6.77.3.8 op()	678
6.77.3.9 let_op()	678
6.77.3.10 expand()	678
6.77.3.11 get_free_indices()	679

6.77.3.12 return_type()	679
6.77.3.13 return_type_tinfo()	679
6.77.3.14 conjugate()	679
6.77.3.15 eval_integ()	679
6.77.3.16 archive()	680
6.77.3.17 read_archive()	680
6.77.3.18 derivative()	680
6.77.3.19 series()	681
6.77.3.20 do_print()	681
6.77.3.21 do_print_latex()	681
6.77.4 Member Data Documentation	681
6.77.4.1 max_integration_level	681
6.77.4.2 relative_integration_error	681
6.77.4.3 x	682
6.77.4.4 a	682
6.77.4.5 b	682
6.77.4.6 f	682
6.78 GiNaC::integration_kernel Class Reference	683
6.78.1 Detailed Description	687
6.78.2 Member Function Documentation	687
6.78.2.1 series()	687
6.78.2.2 has_trailing_zero()	687
6.78.2.3 is_numeric()	688
6.78.2.4 Laurent_series()	688
6.78.2.5 get_numerical_value()	688
6.78.2.6 uses_Laurent_series()	688
6.78.2.7 series_coeff_impl()	689
6.78.2.8 get_cache_size()	689
6.78.2.9 set_cache_step()	689
6.78.2.10 get_series_coeff()	689
6.78.2.11 series_coeff()	689
6.78.2.12 get_numerical_value_impl()	690
6.78.2.13 do_print()	690
6.78.3 Member Data Documentation	690
6.78.3.1 cache_step_size	690
6.78.3.2 series_vec	690
6.79 GiNaC::is_not_a_clifford Struct Reference	690
6.79.1 Detailed Description	691
6.79.2 Member Function Documentation	691
6.79.2.1 operator>()	691
6.80 GiNaC::is_summation_idx Struct Reference	691
6.80.1 Member Function Documentation	691

6.80.1.1 operator()	691
6.81 GiNaC::iterated_integral2_SERIAL Class Reference	691
6.81.1 Detailed Description	692
6.81.2 Member Data Documentation	692
6.81.2.1 serial	692
6.82 GiNaC::iterated_integral3_SERIAL Class Reference	692
6.82.1 Detailed Description	692
6.82.2 Member Data Documentation	693
6.82.2.1 serial	693
6.83 GiNaC::Gt::kernel Struct Reference	693
6.83.1 Constructor & Destructor Documentation	693
6.83.1.1 kernel() [1/3]	693
6.83.1.2 kernel() [2/3]	694
6.83.1.3 kernel() [3/3]	694
6.83.2 Member Function Documentation	694
6.83.2.1 check()	694
6.83.3 Friends And Related Symbol Documentation	694
6.83.3.1 Gt	694
6.83.4 Member Data Documentation	694
6.83.4.1 ni	694
6.83.4.2 zi	694
6.83.4.3 deform	695
6.84 GiNaC::Kronecker_dtau_kernel Class Reference	695
6.84.1 Detailed Description	699
6.84.2 Constructor & Destructor Documentation	700
6.84.2.1 Kronecker_dtau_kernel()	700
6.84.3 Member Function Documentation	700
6.84.3.1 nops()	700
6.84.3.2 op()	700
6.84.3.3 let_op()	700
6.84.3.4 is_numeric()	700
6.84.3.5 get_numerical_value()	701
6.84.3.6 series_coeff_impl()	701
6.84.3.7 do_print()	701
6.84.4 Member Data Documentation	701
6.84.4.1 n	701
6.84.4.2 z	701
6.84.4.3 K	702
6.84.4.4 C_norm	702
6.85 GiNaC::Kronecker_dz_kernel Class Reference	702
6.85.1 Detailed Description	706
6.85.2 Constructor & Destructor Documentation	707

6.85.2.1 Kronecker_dz_kernel()	707
6.85.3 Member Function Documentation	707
6.85.3.1 nops()	707
6.85.3.2 op()	707
6.85.3.3 let_op()	707
6.85.3.4 is_numeric()	707
6.85.3.5 get_numerical_value()	708
6.85.3.6 series_coeff_impl()	708
6.85.3.7 do_print()	708
6.85.4 Member Data Documentation	708
6.85.4.1 n	708
6.85.4.2 z_j	708
6.85.4.3 tau	709
6.85.4.4 K	709
6.85.4.5 C_norm	709
6.86 GiNaC::lanczos_coeffs Class Reference	709
6.86.1 Constructor & Destructor Documentation	709
6.86.1.1 lanczos_coeffs()	709
6.86.2 Member Function Documentation	710
6.86.2.1 sufficiently_accurate()	710
6.86.2.2 get_order()	710
6.86.2.3 calc_lanczos_A()	710
6.86.3 Member Data Documentation	710
6.86.3.1 coeffs	710
6.86.3.2 current_vector	710
6.87 std::less< GiNaC::ptr< T > > Struct Template Reference	711
6.87.1 Detailed Description	711
6.87.2 Member Function Documentation	711
6.87.2.1 operator()()	711
6.88 GiNaC::library_init Class Reference	711
6.88.1 Detailed Description	712
6.88.2 Constructor & Destructor Documentation	712
6.88.2.1 library_init()	712
6.88.2.2 ~library_init()	712
6.88.3 Member Function Documentation	713
6.88.3.1 init_unarchivers()	713
6.88.4 Member Data Documentation	713
6.88.4.1 count	713
6.89 GiNaC::make_flat_inserter Class Reference	713
6.89.1 Detailed Description	713
6.89.2 Constructor & Destructor Documentation	714
6.89.2.1 make_flat_inserter() [1/2]	714

6.89.2.2 make_flat_inserter() [2/2]	714
6.89.3 Member Function Documentation	714
6.89.3.1 handle_factor()	714
6.89.3.2 combine_indices()	714
6.89.4 Member Data Documentation	714
6.89.4.1 do_renaming	714
6.89.4.2 used_indices	715
6.90 GiNaC::map_function Struct Reference	715
6.90.1 Detailed Description	716
6.90.2 Member Typedef Documentation	716
6.90.2.1 argument_type	716
6.90.2.2 result_type	716
6.90.3 Constructor & Destructor Documentation	716
6.90.3.1 ~map_function()	716
6.90.4 Member Function Documentation	716
6.90.4.1 operator()()	716
6.91 GiNaC::matrix Class Reference	716
6.91.1 Detailed Description	721
6.91.2 Constructor & Destructor Documentation	721
6.91.2.1 matrix() [1/5]	721
6.91.2.2 matrix() [2/5]	722
6.91.2.3 matrix() [3/5]	722
6.91.2.4 matrix() [4/5]	722
6.91.2.5 matrix() [5/5]	722
6.91.3 Member Function Documentation	723
6.91.3.1 nops()	723
6.91.3.2 op()	723
6.91.3.3 let_op()	723
6.91.3.4 evalm()	723
6.91.3.5 subs()	724
6.91.3.6 eval_indexed()	724
6.91.3.7 add_indexed()	724
6.91.3.8 scalar_mul_indexed()	724
6.91.3.9 contract_with()	725
6.91.3.10 conjugate()	725
6.91.3.11 real_part()	725
6.91.3.12 imag_part()	725
6.91.3.13 archive()	725
6.91.3.14 read_archive()	726
6.91.3.15 match_same_type()	726
6.91.3.16 return_type()	726
6.91.3.17 rows()	726

6.91.3.18 cols()	727
6.91.3.19 add()	727
6.91.3.20 sub()	727
6.91.3.21 mul() [1/2]	727
6.91.3.22 mul() [2/2]	728
6.91.3.23 mul_scalar()	728
6.91.3.24 pow()	728
6.91.3.25 operator>() [1/2]	728
6.91.3.26 operator>() [2/2]	729
6.91.3.27 set()	729
6.91.3.28 transpose()	729
6.91.3.29 determinant()	730
6.91.3.30 trace()	730
6.91.3.31 charpoly()	731
6.91.3.32 inverse() [1/2]	731
6.91.3.33 inverse() [2/2]	731
6.91.3.34 solve()	732
6.91.3.35 rank() [1/2]	733
6.91.3.36 rank() [2/2]	733
6.91.3.37 is_zero_matrix()	733
6.91.3.38 determinant_minor()	733
6.91.3.39 echelon_form()	734
6.91.3.40 gauss_elimination()	734
6.91.3.41 division_free_elimination()	734
6.91.3.42 fraction_free_elimination()	735
6.91.3.43 markowitz_elimination()	735
6.91.3.44 pivot()	735
6.91.3.45 print_elements()	736
6.91.3.46 do_print()	736
6.91.3.47 do_print_latex()	736
6.91.3.48 do_print_python_repr()	737
6.91.4 Member Data Documentation	737
6.91.4.1 row	737
6.91.4.2 col	737
6.91.4.3 m	737
6.92 GiNaC::minkmetric Class Reference	738
6.92.1 Detailed Description	742
6.92.2 Constructor & Destructor Documentation	742
6.92.2.1 minkmetric()	742
6.92.3 Member Function Documentation	742
6.92.3.1 info()	742
6.92.3.2 eval_indexed()	742

6.92.3.3 archive()	743
6.92.3.4 read_archive()	743
6.92.3.5 return_type()	743
6.92.3.6 do_print()	743
6.92.3.7 do_print_latex()	743
6.92.4 Member Data Documentation	744
6.92.4.1 pos_sig	744
6.93 GiNaC::modular_form_kernel Class Reference	744
6.93.1 Detailed Description	748
6.93.2 Constructor & Destructor Documentation	748
6.93.2.1 modular_form_kernel()	748
6.93.3 Member Function Documentation	749
6.93.3.1 series()	749
6.93.3.2 nops()	749
6.93.3.3 op()	749
6.93.3.4 let_op()	749
6.93.3.5 is_numeric()	750
6.93.3.6 Laurent_series()	750
6.93.3.7 get_numerical_value()	750
6.93.3.8 uses_Laurent_series()	750
6.93.3.9 q_expansion_modular_form()	751
6.93.3.10 do_print()	751
6.93.4 Member Data Documentation	751
6.93.4.1 k	751
6.93.4.2 P	751
6.93.4.3 C_norm	751
6.94 GiNaC::basic_partition_generator::mpartition2 Struct Reference	751
6.94.1 Constructor & Destructor Documentation	752
6.94.1.1 mpartition2()	752
6.94.2 Member Function Documentation	752
6.94.2.1 next_partition()	752
6.94.3 Member Data Documentation	752
6.94.3.1 x	752
6.94.3.2 n	752
6.94.3.3 m	753
6.95 GiNaC::mul Class Reference	753
6.95.1 Detailed Description	758
6.95.2 Constructor & Destructor Documentation	758
6.95.2.1 mul() [1/7]	758
6.95.2.2 mul() [2/7]	758
6.95.2.3 mul() [3/7]	758
6.95.2.4 mul() [4/7]	759

6.95.2.5 mul() [5/7]	759
6.95.2.6 mul() [6/7]	759
6.95.2.7 mul() [7/7]	759
6.95.3 Member Function Documentation	759
6.95.3.1 precedence()	759
6.95.3.2 info()	760
6.95.3.3 is_polynomial()	760
6.95.3.4 degree()	760
6.95.3.5 ldegree()	760
6.95.3.6 coeff()	761
6.95.3.7 has()	761
6.95.3.8 eval()	761
6.95.3.9 evalf()	762
6.95.3.10 real_part()	762
6.95.3.11 imag_part()	762
6.95.3.12 evalm()	762
6.95.3.13 series()	762
6.95.3.14 normal()	763
6.95.3.15 integer_content()	763
6.95.3.16 smod()	763
6.95.3.17 max_coefficient()	764
6.95.3.18 get_free_indices()	764
6.95.3.19 conjugate()	764
6.95.3.20 derivative()	764
6.95.3.21 eval_ncmul()	765
6.95.3.22 return_type()	765
6.95.3.23 return_type_tinfo()	765
6.95.3.24 thisexpairseq() [1/2]	765
6.95.3.25 thisexpairseq() [2/2]	765
6.95.3.26 split_ex_to_pair()	766
6.95.3.27 combine_ex_with_coeff_to_pair()	766
6.95.3.28 combine_pair_with_coeff_to_pair()	766
6.95.3.29 recombine_pair_to_ex()	766
6.95.3.30 expair_needs_further_processing()	767
6.95.3.31 default_overall_coeff()	767
6.95.3.32 combine_overall_coeff() [1/2]	767
6.95.3.33 combine_overall_coeff() [2/2]	767
6.95.3.34 can_make_flat()	767
6.95.3.35 expand()	768
6.95.3.36 algebraic_subs_mul()	768
6.95.3.37 find_real_imag()	768
6.95.3.38 print_overall_coeff()	768

6.95.3.39 do_print()	769
6.95.3.40 do_print_latex()	769
6.95.3.41 do_print_csrc()	769
6.95.3.42 do_print_python_repr()	769
6.95.3.43 can_be_further_expanded()	769
6.95.3.44 expandchildren()	770
6.95.4 Friends And Related Symbol Documentation	770
6.95.4.1 add	770
6.95.4.2 ncmul	770
6.95.4.3 power	770
6.96 GiNaC::multi_iterator_counter< T > Class Template Reference	771
6.96.1 Detailed Description	772
6.96.2 Constructor & Destructor Documentation	772
6.96.2.1 multi_iterator_counter() [1/3]	772
6.96.2.2 multi_iterator_counter() [2/3]	772
6.96.2.3 multi_iterator_counter() [3/3]	773
6.96.3 Member Function Documentation	773
6.96.3.1 init()	773
6.96.3.2 operator++()	773
6.96.4 Friends And Related Symbol Documentation	773
6.96.4.1 operator<<	773
6.97 GiNaC::multi_iterator_counter_indv< T > Class Template Reference	774
6.97.1 Detailed Description	775
6.97.2 Constructor & Destructor Documentation	775
6.97.2.1 multi_iterator_counter_indv() [1/3]	775
6.97.2.2 multi_iterator_counter_indv() [2/3]	775
6.97.2.3 multi_iterator_counter_indv() [3/3]	776
6.97.3 Member Function Documentation	776
6.97.3.1 init()	776
6.97.3.2 operator++()	776
6.97.4 Friends And Related Symbol Documentation	776
6.97.4.1 operator<<	776
6.97.5 Member Data Documentation	777
6.97.5.1 Nv	777
6.98 GiNaC::multi_iterator_ordered< T > Class Template Reference	777
6.98.1 Detailed Description	778
6.98.2 Constructor & Destructor Documentation	779
6.98.2.1 multi_iterator_ordered() [1/3]	779
6.98.2.2 multi_iterator_ordered() [2/3]	779
6.98.2.3 multi_iterator_ordered() [3/3]	779
6.98.3 Member Function Documentation	779
6.98.3.1 init()	779

6.98.3.2 operator++()	780
6.98.4 Friends And Related Symbol Documentation	780
6.98.4.1 operator<<	780
6.99 GiNaC::multi_iterator_ordered_eq< T > Class Template Reference	780
6.99.1 Detailed Description	781
6.99.2 Constructor & Destructor Documentation	782
6.99.2.1 multi_iterator_ordered_eq() [1/3]	782
6.99.2.2 multi_iterator_ordered_eq() [2/3]	782
6.99.2.3 multi_iterator_ordered_eq() [3/3]	782
6.99.3 Member Function Documentation	782
6.99.3.1 init()	782
6.99.3.2 operator++()	783
6.99.4 Friends And Related Symbol Documentation	783
6.99.4.1 operator<<	783
6.100 GiNaC::multi_iterator_ordered_eq_indv< T > Class Template Reference	783
6.100.1 Detailed Description	784
6.100.2 Constructor & Destructor Documentation	785
6.100.2.1 multi_iterator_ordered_eq_indv() [1/3]	785
6.100.2.2 multi_iterator_ordered_eq_indv() [2/3]	785
6.100.2.3 multi_iterator_ordered_eq_indv() [3/3]	785
6.100.3 Member Function Documentation	785
6.100.3.1 init()	785
6.100.3.2 operator++()	786
6.100.4 Friends And Related Symbol Documentation	786
6.100.4.1 operator<<	786
6.100.5 Member Data Documentation	786
6.100.5.1 Nv	786
6.101 GiNaC::multi_iterator_permutation< T > Class Template Reference	786
6.101.1 Detailed Description	788
6.101.2 Constructor & Destructor Documentation	788
6.101.2.1 multi_iterator_permutation() [1/3]	788
6.101.2.2 multi_iterator_permutation() [2/3]	788
6.101.2.3 multi_iterator_permutation() [3/3]	788
6.101.3 Member Function Documentation	789
6.101.3.1 init()	789
6.101.3.2 operator++()	789
6.101.3.3 get_sign()	789
6.101.4 Friends And Related Symbol Documentation	790
6.101.4.1 operator<<	790
6.102 GiNaC::multi_iterator_shuffle< T > Class Template Reference	790
6.102.1 Detailed Description	791
6.102.2 Constructor & Destructor Documentation	792

6.102.2.1 multi_iterator_shuffle() [1/2]	792
6.102.2.2 multi_iterator_shuffle() [2/2]	792
6.102.3 Member Function Documentation	792
6.102.3.1 init()	792
6.102.3.2 operator++()	792
6.102.4 Friends And Related Symbol Documentation	793
6.102.4.1 operator<<	793
6.102.5 Member Data Documentation	793
6.102.5.1 N_internal	793
6.102.5.2 v_internal	793
6.102.5.3 v_orig	793
6.103 GiNaC::multi_iterator_shuffle_prime< T > Class Template Reference	793
6.103.1 Detailed Description	795
6.103.2 Constructor & Destructor Documentation	795
6.103.2.1 multi_iterator_shuffle_prime() [1/2]	795
6.103.2.2 multi_iterator_shuffle_prime() [2/2]	795
6.103.3 Member Function Documentation	796
6.103.3.1 init()	796
6.103.4 Friends And Related Symbol Documentation	796
6.103.4.1 operator<<	796
6.104 GiNaC::multiple_polylog_kernel Class Reference	796
6.104.1 Detailed Description	801
6.104.2 Constructor & Destructor Documentation	801
6.104.2.1 multiple_polylog_kernel()	801
6.104.3 Member Function Documentation	801
6.104.3.1 nops()	801
6.104.3.2 op()	801
6.104.3.3 let_op()	801
6.104.3.4 is_numeric()	802
6.104.3.5 series_coeff_impl()	802
6.104.3.6 do_print()	802
6.104.4 Member Data Documentation	802
6.104.4.1 z	802
6.105 GiNaC::ncmul Class Reference	803
6.105.1 Detailed Description	808
6.105.2 Constructor & Destructor Documentation	808
6.105.2.1 ncmul() [1/7]	808
6.105.2.2 ncmul() [2/7]	808
6.105.2.3 ncmul() [3/7]	809
6.105.2.4 ncmul() [4/7]	809
6.105.2.5 ncmul() [5/7]	809
6.105.2.6 ncmul() [6/7]	809

6.105.2.7 ncmul() [7/7]	809
6.105.3 Member Function Documentation	809
6.105.3.1 precedence()	809
6.105.3.2 info()	810
6.105.3.3 degree()	810
6.105.3.4 ldegree()	810
6.105.3.5 expand()	810
6.105.3.6 coeff()	811
6.105.3.7 eval()	811
6.105.3.8 evalm()	811
6.105.3.9 get_free_indices()	812
6.105.3.10 thiscontainer() [1/2]	812
6.105.3.11 thiscontainer() [2/2]	812
6.105.3.12 conjugate()	812
6.105.3.13 real_part()	812
6.105.3.14 imag_part()	812
6.105.3.15 derivative()	813
6.105.3.16 return_type()	813
6.105.3.17 return_type_tinfo()	813
6.105.3.18 do_print()	813
6.105.3.19 do_print_csrc()	813
6.105.3.20 count_factors()	814
6.105.3.21 append_factors()	814
6.105.3.22 expandchildren()	814
6.105.3.23 get_factors()	814
6.105.4 Friends And Related Symbol Documentation	814
6.105.4.1 power	814
6.105.4.2 reeval_ncmul	814
6.105.4.3 hold_ncmul	815
6.106 GiNaC::normal_map_function Struct Reference	815
6.106.1 Detailed Description	815
6.106.2 Member Function Documentation	816
6.106.2.1 operator()()	816
6.107 GiNaC::numeric Class Reference	816
6.107.1 Detailed Description	821
6.107.2 Constructor & Destructor Documentation	822
6.107.2.1 numeric() [1/10]	822
6.107.2.2 numeric() [2/10]	822
6.107.2.3 numeric() [3/10]	822
6.107.2.4 numeric() [4/10]	822
6.107.2.5 numeric() [5/10]	822
6.107.2.6 numeric() [6/10]	822

6.107.2.7 numeric() [7/10]	822
6.107.2.8 numeric() [8/10]	823
6.107.2.9 numeric() [9/10]	823
6.107.2.10 numeric() [10/10]	823
6.107.3 Member Function Documentation	823
6.107.3.1 precedence()	823
6.107.3.2 info()	824
6.107.3.3 is_polynomial()	824
6.107.3.4 degree()	824
6.107.3.5 ldegree()	824
6.107.3.6 coeff()	825
6.107.3.7 has()	825
6.107.3.8 eval()	825
6.107.3.9 evalf()	825
6.107.3.10 subs()	826
6.107.3.11 normal()	826
6.107.3.12 to_rational()	826
6.107.3.13 to_polynomial()	826
6.107.3.14 integer_content()	827
6.107.3.15 smod()	827
6.107.3.16 max_coefficient()	827
6.107.3.17 conjugate()	828
6.107.3.18 real_part()	828
6.107.3.19 imag_part()	828
6.107.3.20 archive()	828
6.107.3.21 read_archive()	828
6.107.3.22 derivative()	829
6.107.3.23 is_equal_same_type()	829
6.107.3.24 calchash()	829
6.107.3.25 add()	829
6.107.3.26 sub()	830
6.107.3.27 mul()	830
6.107.3.28 div()	830
6.107.3.29 power()	830
6.107.3.30 add_dyn()	831
6.107.3.31 sub_dyn()	831
6.107.3.32 mul_dyn()	831
6.107.3.33 div_dyn()	831
6.107.3.34 power_dyn()	832
6.107.3.35 operator=() [1/6]	832
6.107.3.36 operator=() [2/6]	832
6.107.3.37 operator=() [3/6]	832

6.107.3.38 operator=() [4/6]	832
6.107.3.39 operator=() [5/6]	833
6.107.3.40 operator=() [6/6]	833
6.107.3.41 inverse()	833
6.107.3.42 step()	833
6.107.3.43 csgn()	833
6.107.3.44 compare()	834
6.107.3.45 is_equal()	834
6.107.3.46 is_zero()	834
6.107.3.47 is_positive()	834
6.107.3.48 is_negative()	835
6.107.3.49 is_integer()	835
6.107.3.50 is_pos_integer()	835
6.107.3.51 is_nonneg_integer()	835
6.107.3.52 is_even()	835
6.107.3.53 is_odd()	836
6.107.3.54 is_prime()	836
6.107.3.55 is_rational()	836
6.107.3.56 is_real()	836
6.107.3.57 is_cinteger()	837
6.107.3.58 is_crational()	837
6.107.3.59 operator==(())	837
6.107.3.60 operator!=(())	837
6.107.3.61 operator<()	837
6.107.3.62 operator<=()	838
6.107.3.63 operator>()	838
6.107.3.64 operator>=()	838
6.107.3.65 to_int()	838
6.107.3.66 to_long()	839
6.107.3.67 to_double()	839
6.107.3.68 to_cl_N()	839
6.107.3.69 real()	839
6.107.3.70 imag()	840
6.107.3.71 numer()	840
6.107.3.72 denom()	840
6.107.3.73 int_length()	840
6.107.3.74 print_numeric()	841
6.107.3.75 do_print()	841
6.107.3.76 do_print_latex()	841
6.107.3.77 do_print_csrc()	841
6.107.3.78 do_print_csrc_cl_N()	841
6.107.3.79 do_print_tree()	842

6.107.3.80 do_print_python_repr()	842
6.107.4 Member Data Documentation	842
6.107.4.1 value	842
6.108 GiNaC::op0_is_equal Struct Reference	842
6.108.1 Member Function Documentation	843
6.108.1.1 operator()	843
6.109 GiNaC::partition_generator Class Reference	843
6.109.1 Detailed Description	844
6.109.2 Constructor & Destructor Documentation	844
6.109.2.1 partition_generator()	844
6.109.3 Member Function Documentation	844
6.109.3.1 get()	844
6.109.3.2 next()	844
6.109.4 Member Data Documentation	844
6.109.4.1 partition	844
6.109.4.2 current_updated	844
6.110 GiNaC::partition_with_zero_parts_generator Class Reference	845
6.110.1 Detailed Description	845
6.110.2 Constructor & Destructor Documentation	845
6.110.2.1 partition_with_zero_parts_generator()	845
6.110.3 Member Function Documentation	846
6.110.3.1 get()	846
6.110.3.2 next()	846
6.110.4 Member Data Documentation	846
6.110.4.1 m	846
6.110.4.2 partition	846
6.110.4.3 current_updated	846
6.111 GiNaC::Gt_detail::pathintegral_term Class Reference	846
6.111.1 Constructor & Destructor Documentation	847
6.111.1.1 pathintegral_term() [1/2]	847
6.111.1.2 pathintegral_term() [2/2]	847
6.111.2 Member Function Documentation	848
6.111.2.1 integrate()	848
6.111.2.2 evaluate()	848
6.111.2.3 G_path() [1/2]	848
6.111.2.4 G_path() [2/2]	848
6.111.3 Friends And Related Symbol Documentation	848
6.111.3.1 pathintegral_term_list	848
6.111.3.2 operator<<	849
6.111.3.3 operator<	849
6.111.4 Member Data Documentation	849
6.111.4.1 prefactor	849

6.111.4.2 power	849
6.111.4.3 polylog	849
6.111.4.4 denom	849
6.112 GiNaC::Gt_detail::pathintegral_term_list Class Reference	849
6.112.1 Member Function Documentation	850
6.112.1.1 begin()	850
6.112.1.2 end()	850
6.112.1.3 clear()	850
6.112.1.4 empty()	850
6.112.1.5 size()	851
6.112.1.6 add() [1/3]	851
6.112.1.7 add() [2/3]	851
6.112.1.8 add() [3/3]	851
6.112.1.9 pop()	851
6.112.2 Member Data Documentation	851
6.112.2.1 terms	851
6.113 GiNaC::pointer_to_map_function Class Reference	852
6.113.1 Constructor & Destructor Documentation	852
6.113.1.1 pointer_to_map_function()	852
6.113.2 Member Function Documentation	852
6.113.2.1 operator>()	852
6.113.3 Member Data Documentation	853
6.113.3.1 ptr	853
6.114 GiNaC::pointer_to_map_function_1arg< T1 > Class Template Reference	853
6.114.1 Constructor & Destructor Documentation	854
6.114.1.1 pointer_to_map_function_1arg()	854
6.114.2 Member Function Documentation	854
6.114.2.1 operator>()	854
6.114.3 Member Data Documentation	854
6.114.3.1 ptr	854
6.114.3.2 arg1	854
6.115 GiNaC::pointer_to_map_function_2args< T1, T2 > Class Template Reference	854
6.115.1 Constructor & Destructor Documentation	855
6.115.1.1 pointer_to_map_function_2args()	855
6.115.2 Member Function Documentation	855
6.115.2.1 operator>()	855
6.115.3 Member Data Documentation	856
6.115.3.1 ptr	856
6.115.3.2 arg1	856
6.115.3.3 arg2	856
6.116 GiNaC::pointer_to_map_function_3args< T1, T2, T3 > Class Template Reference	856
6.116.1 Constructor & Destructor Documentation	857

6.116.1.1 pointer_to_map_function_3args()	857
6.116.2 Member Function Documentation	857
6.116.2.1 operator()()	857
6.116.3 Member Data Documentation	857
6.116.3.1 ptr	857
6.116.3.2 arg1	858
6.116.3.3 arg2	858
6.116.3.4 arg3	858
6.117 GiNaC::pointer_to_member_to_map_function< C > Class Template Reference	858
6.117.1 Constructor & Destructor Documentation	859
6.117.1.1 pointer_to_member_to_map_function()	859
6.117.2 Member Function Documentation	859
6.117.2.1 operator()()	859
6.117.3 Member Data Documentation	859
6.117.3.1 ptr	859
6.117.3.2 c	859
6.118 GiNaC::pointer_to_member_to_map_function_1arg< C, T1 > Class Template Reference	860
6.118.1 Constructor & Destructor Documentation	860
6.118.1.1 pointer_to_member_to_map_function_1arg()	860
6.118.2 Member Function Documentation	861
6.118.2.1 operator()()	861
6.118.3 Member Data Documentation	861
6.118.3.1 ptr	861
6.118.3.2 c	861
6.118.3.3 arg1	861
6.119 GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 > Class Template Reference	861
6.119.1 Constructor & Destructor Documentation	862
6.119.1.1 pointer_to_member_to_map_function_2args()	862
6.119.2 Member Function Documentation	862
6.119.2.1 operator()()	862
6.119.3 Member Data Documentation	863
6.119.3.1 ptr	863
6.119.3.2 c	863
6.119.3.3 arg1	863
6.119.3.4 arg2	863
6.120 GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 > Class Template Reference	863
6.120.1 Constructor & Destructor Documentation	864
6.120.1.1 pointer_to_member_to_map_function_3args()	864
6.120.2 Member Function Documentation	864
6.120.2.1 operator()()	864
6.120.3 Member Data Documentation	865
6.120.3.1 ptr	865

6.120.3.2 c	865
6.120.3.3 arg1	865
6.120.3.4 arg2	865
6.120.3.5 arg3	865
6.121 GiNaC::pole_error Class Reference	865
6.121.1 Detailed Description	866
6.121.2 Constructor & Destructor Documentation	866
6.121.2.1 pole_error()	866
6.121.3 Member Function Documentation	866
6.121.3.1 degree()	866
6.121.4 Member Data Documentation	866
6.121.4.1 deg	866
6.122 GiNaC::possymbol Class Reference	867
6.122.1 Detailed Description	871
6.122.2 Constructor & Destructor Documentation	871
6.122.2.1 possymbol() [1/3]	871
6.122.2.2 possymbol() [2/3]	871
6.122.2.3 possymbol() [3/3]	871
6.122.3 Member Function Documentation	871
6.122.3.1 get_domain()	871
6.122.3.2 duplicate()	872
6.123 GiNaC::power Class Reference	872
6.123.1 Detailed Description	876
6.123.2 Constructor & Destructor Documentation	876
6.123.2.1 power() [1/2]	876
6.123.2.2 power() [2/2]	876
6.123.3 Member Function Documentation	876
6.123.3.1 precedence()	876
6.123.3.2 info()	877
6.123.3.3 nops()	877
6.123.3.4 op()	877
6.123.3.5 map()	877
6.123.3.6 is_polynomial()	878
6.123.3.7 degree()	878
6.123.3.8 ldegree()	878
6.123.3.9 coeff()	878
6.123.3.10 eval()	879
6.123.3.11 evalf()	879
6.123.3.12 evalm()	880
6.123.3.13 series()	880
6.123.3.14 subs()	880
6.123.3.15 has()	881

6.123.3.16	normal()	881
6.123.3.17	to_rational()	881
6.123.3.18	to_polynomial()	882
6.123.3.19	conjugate()	882
6.123.3.20	real_part()	882
6.123.3.21	imag_part()	882
6.123.3.22	archive()	882
6.123.3.23	read_archive()	883
6.123.3.24	derivative()	883
6.123.3.25	eval_ncmul()	883
6.123.3.26	return_type()	883
6.123.3.27	return_type_tinfo()	883
6.123.3.28	expand()	884
6.123.3.29	print_power()	884
6.123.3.30	do_print_dfft()	884
6.123.3.31	do_print_latex()	884
6.123.3.32	do_print_csrc()	885
6.123.3.33	do_print_python()	885
6.123.3.34	do_print_python_repr()	885
6.123.3.35	do_print_csrc_cl_N()	885
6.123.3.36	expand_add()	885
6.123.3.37	expand_add_2()	886
6.123.3.38	expand_mul()	886
6.123.4	Friends And Related Symbol Documentation	886
6.123.4.1	mul	886
6.123.5	Member Data Documentation	887
6.123.5.1	basis	887
6.123.5.2	exponent	887
6.124	GiNaC::print_context Class Reference	887
6.124.1	Detailed Description	888
6.124.2	Constructor & Destructor Documentation	888
6.124.2.1	print_context()	888
6.124.2.2	~print_context()	888
6.124.3	Member Data Documentation	888
6.124.3.1	s	888
6.124.3.2	options	888
6.125	GiNaC::print_context_options Class Reference	889
6.125.1	Detailed Description	889
6.125.2	Constructor & Destructor Documentation	889
6.125.2.1	print_context_options()	889
6.125.3	Member Function Documentation	889
6.125.3.1	get_name()	889

6.125.3.2 <code>get_parent_name()</code>	890
6.125.3.3 <code>get_id()</code>	890
6.125.4 Member Data Documentation	890
6.125.4.1 <code>name</code>	890
6.125.4.2 <code>parent_name</code>	890
6.125.4.3 <code>id</code>	890
6.126 <code>GiNaC::print_csrc</code> Class Reference	891
6.126.1 Detailed Description	891
6.126.2 Constructor & Destructor Documentation	891
6.126.2.1 <code>print_csrc()</code>	891
6.127 <code>GiNaC::print_csrc_cl_N</code> Class Reference	892
6.127.1 Detailed Description	892
6.127.2 Constructor & Destructor Documentation	893
6.127.2.1 <code>print_csrc_cl_N()</code>	893
6.128 <code>GiNaC::print_csrc_double</code> Class Reference	893
6.128.1 Detailed Description	894
6.128.2 Constructor & Destructor Documentation	894
6.128.2.1 <code>print_csrc_double()</code>	894
6.129 <code>GiNaC::print_csrc_float</code> Class Reference	894
6.129.1 Detailed Description	895
6.129.2 Constructor & Destructor Documentation	895
6.129.2.1 <code>print_csrc_float()</code>	895
6.130 <code>GiNaC::print_dflt</code> Class Reference	895
6.130.1 Detailed Description	896
6.130.2 Constructor & Destructor Documentation	896
6.130.2.1 <code>print_dflt()</code>	896
6.131 <code>GiNaC::print_functor</code> Class Reference	896
6.131.1 Detailed Description	897
6.131.2 Constructor & Destructor Documentation	897
6.131.2.1 <code>print_functor()</code> [1/5]	897
6.131.2.2 <code>print_functor()</code> [2/5]	897
6.131.2.3 <code>print_functor()</code> [3/5]	897
6.131.2.4 <code>print_functor()</code> [4/5]	897
6.131.2.5 <code>print_functor()</code> [5/5]	897
6.131.3 Member Function Documentation	897
6.131.3.1 <code>operator=()</code>	897
6.131.3.2 <code>operator>()</code>	898
6.131.3.3 <code>is_valid()</code>	898
6.131.4 Member Data Documentation	898
6.131.4.1 <code>impl</code>	898
6.132 <code>GiNaC::print_functor_impl</code> Class Reference	898
6.132.1 Detailed Description	899

6.132.2 Constructor & Destructor Documentation	899
6.132.2.1 ~print_funcutor_impl()	899
6.132.3 Member Function Documentation	899
6.132.3.1 duplicate()	899
6.132.3.2 operator()()	899
6.133 GiNaC::print_latex Class Reference	899
6.133.1 Detailed Description	900
6.133.2 Constructor & Destructor Documentation	900
6.133.2.1 print_latex()	900
6.134 GiNaC::print_memfun_handler< T, C > Class Template Reference	900
6.134.1 Detailed Description	901
6.134.2 Member Typedef Documentation	901
6.134.2.1 F	901
6.134.3 Constructor & Destructor Documentation	901
6.134.3.1 print_memfun_handler()	901
6.134.4 Member Function Documentation	901
6.134.4.1 duplicate()	901
6.134.4.2 operator()()	902
6.134.5 Member Data Documentation	902
6.134.5.1 f	902
6.135 GiNaC::print_options Class Reference	902
6.135.1 Detailed Description	902
6.135.2 Member Enumeration Documentation	902
6.135.2.1 anonymous enum	902
6.136 GiNaC::print_ptrfun_handler< T, C > Class Template Reference	903
6.136.1 Detailed Description	903
6.136.2 Member Typedef Documentation	904
6.136.2.1 F	904
6.136.3 Constructor & Destructor Documentation	904
6.136.3.1 print_ptrfun_handler()	904
6.136.4 Member Function Documentation	904
6.136.4.1 duplicate()	904
6.136.4.2 operator()()	904
6.136.5 Member Data Documentation	904
6.136.5.1 f	904
6.137 GiNaC::print_python Class Reference	905
6.137.1 Detailed Description	905
6.137.2 Constructor & Destructor Documentation	905
6.137.2.1 print_python()	905
6.138 GiNaC::print_python_repr Class Reference	906
6.138.1 Detailed Description	906
6.138.2 Constructor & Destructor Documentation	906

6.138.2.1 print_python_repr()	906
6.139 GiNaC::print_tree Class Reference	907
6.139.1 Detailed Description	907
6.139.2 Constructor & Destructor Documentation	907
6.139.2.1 print_tree() [1/2]	907
6.139.2.2 print_tree() [2/2]	908
6.139.3 Member Data Documentation	908
6.139.3.1 delta_indent	908
6.140 GiNaC::archive_node::property Struct Reference	908
6.140.1 Detailed Description	908
6.140.2 Constructor & Destructor Documentation	909
6.140.2.1 property() [1/2]	909
6.140.2.2 property() [2/2]	909
6.140.3 Member Data Documentation	909
6.140.3.1 type	909
6.140.3.2 name	909
6.140.3.3 value	909
6.141 GiNaC::archive_node::property_info Struct Reference	909
6.141.1 Detailed Description	910
6.141.2 Constructor & Destructor Documentation	910
6.141.2.1 property_info() [1/2]	910
6.141.2.2 property_info() [2/2]	910
6.141.3 Member Data Documentation	910
6.141.3.1 type	910
6.141.3.2 name	911
6.141.3.3 count	911
6.142 GiNaC::pseries Class Reference	911
6.142.1 Detailed Description	915
6.142.2 Constructor & Destructor Documentation	915
6.142.2.1 pseries() [1/2]	915
6.142.2.2 pseries() [2/2]	916
6.142.3 Member Function Documentation	916
6.142.3.1 precedence()	916
6.142.3.2 nops()	916
6.142.3.3 op()	916
6.142.3.4 degree()	917
6.142.3.5 ldegree()	917
6.142.3.6 coeff()	917
6.142.3.7 collect()	918
6.142.3.8 eval()	918
6.142.3.9 evalf()	918
6.142.3.10 series()	918

6.142.3.11 subs()	918
6.142.3.12 normal()	919
6.142.3.13 expand()	919
6.142.3.14 conjugate()	919
6.142.3.15 real_part()	919
6.142.3.16 imag_part()	920
6.142.3.17 eval_integ()	920
6.142.3.18 evalm()	920
6.142.3.19 archive()	920
6.142.3.20 read_archive()	920
6.142.3.21 derivative()	921
6.142.3.22 get_var()	921
6.142.3.23 get_point()	921
6.142.3.24 convert_to_poly()	921
6.142.3.25 is_compatible_to()	921
6.142.3.26 is_zero()	922
6.142.3.27 is_terminating()	922
6.142.3.28 coeffop()	922
6.142.3.29 exponop()	922
6.142.3.30 add_series()	922
6.142.3.31 mul_const()	923
6.142.3.32 mul_series()	923
6.142.3.33 power_const()	924
6.142.3.34 shift_exponents()	924
6.142.3.35 print_series()	924
6.142.3.36 do_print()	924
6.142.3.37 do_print_latex()	925
6.142.3.38 do_print_tree()	925
6.142.3.39 do_print_python()	925
6.142.3.40 do_print_python_repr()	925
6.142.4 Member Data Documentation	925
6.142.4.1 seq	925
6.142.4.2 var	926
6.142.4.3 point	926
6.143 GiNaC::psi1_SERIAL Class Reference	926
6.143.1 Detailed Description	926
6.143.2 Member Data Documentation	927
6.143.2.1 serial	927
6.144 GiNaC::psi2_SERIAL Class Reference	927
6.144.1 Detailed Description	927
6.144.2 Member Data Documentation	927
6.144.2.1 serial	927

6.145 GiNaC::ptr< T > Class Template Reference	928
6.145.1 Detailed Description	928
6.145.2 Constructor & Destructor Documentation	929
6.145.2.1 ptr() [1/3]	929
6.145.2.2 ptr() [2/3]	929
6.145.2.3 ptr() [3/3]	929
6.145.2.4 ~ptr()	929
6.145.3 Member Function Documentation	929
6.145.3.1 operator=()	929
6.145.3.2 operator*()	930
6.145.3.3 operator->()	930
6.145.3.4 makewritable()	930
6.145.3.5 swap()	930
6.145.3.6 operator==()	930
6.145.3.7 operator"!=()	930
6.145.4 Friends And Related Symbol Documentation	931
6.145.4.1 std::less< ptr< T > >	931
6.145.4.2 get_pointer	931
6.145.4.3 operator== [1/2]	931
6.145.4.4 operator"!=[1/2]	931
6.145.4.5 operator== [2/2]	931
6.145.4.6 operator"!=[2/2]	931
6.145.4.7 operator<<	932
6.145.5 Member Data Documentation	932
6.145.5.1 p	932
6.146 GiNaC::realsymbol Class Reference	932
6.146.1 Detailed Description	936
6.146.2 Constructor & Destructor Documentation	936
6.146.2.1 realsymbol() [1/3]	936
6.146.2.2 realsymbol() [2/3]	936
6.146.2.3 realsymbol() [3/3]	936
6.146.3 Member Function Documentation	936
6.146.3.1 get_domain()	936
6.146.3.2 conjugate()	937
6.146.3.3 real_part()	937
6.146.3.4 imag_part()	937
6.146.3.5 duplicate()	937
6.147 GiNaC::refcounted Class Reference	938
6.147.1 Detailed Description	938
6.147.2 Constructor & Destructor Documentation	939
6.147.2.1 refcounted()	939
6.147.3 Member Function Documentation	939

6.147.3.1 add_reference()	939
6.147.3.2 remove_reference()	939
6.147.3.3 get_refcount()	939
6.147.3.4 set_refcount()	939
6.147.4 Member Data Documentation	939
6.147.4.1 refcount	939
6.148 GiNaC::registered_class_options Class Reference	940
6.148.1 Detailed Description	940
6.148.2 Constructor & Destructor Documentation	940
6.148.2.1 registered_class_options()	940
6.148.3 Member Function Documentation	941
6.148.3.1 get_name()	941
6.148.3.2 get_parent_name()	941
6.148.3.3 get_id()	941
6.148.3.4 get_print_dispatch_table()	941
6.148.3.5 print_func() [1/3]	941
6.148.3.6 print_func() [2/3]	941
6.148.3.7 print_func() [3/3]	942
6.148.3.8 set_print_func()	942
6.148.4 Member Data Documentation	942
6.148.4.1 name	942
6.148.4.2 parent_name	942
6.148.4.3 tinfo_key	942
6.148.4.4 print_dispatch_table	943
6.149 GiNaC::relational Class Reference	943
6.149.1 Detailed Description	947
6.149.2 Member Typedef Documentation	947
6.149.2.1 safe_bool	947
6.149.3 Member Enumeration Documentation	947
6.149.3.1 operators	947
6.149.4 Constructor & Destructor Documentation	948
6.149.4.1 relational()	948
6.149.5 Member Function Documentation	948
6.149.5.1 precedence()	948
6.149.5.2 info()	948
6.149.5.3 nops()	948
6.149.5.4 op()	949
6.149.5.5 map()	949
6.149.5.6 subs()	949
6.149.5.7 archive()	949
6.149.5.8 read_archive()	950
6.149.5.9 canonical()	950

6.149.5.10 eval_ncmul()	950
6.149.5.11 match_same_type()	950
6.149.5.12 return_type()	951
6.149.5.13 return_type_tinfo()	951
6.149.5.14 calchash()	951
6.149.5.15 do_print()	951
6.149.5.16 do_print_python_repr()	951
6.149.5.17 lhs()	952
6.149.5.18 rhs()	952
6.149.5.19 make_safe_bool()	952
6.149.5.20 operator safe_bool()	952
6.149.5.21 operator"!()	952
6.149.6 Member Data Documentation	953
6.149.6.1 lh	953
6.149.6.2 rh	953
6.149.6.3 o	953
6.150 GiNaC::remember_strategies Class Reference	953
6.150.1 Detailed Description	953
6.150.2 Member Enumeration Documentation	953
6.150.2.1 anonymous enum	953
6.151 GiNaC::remember_table Class Reference	954
6.151.1 Detailed Description	955
6.151.2 Constructor & Destructor Documentation	955
6.151.2.1 remember_table() [1/2]	955
6.151.2.2 remember_table() [2/2]	955
6.151.3 Member Function Documentation	955
6.151.3.1 lookup_entry()	955
6.151.3.2 add_entry()	955
6.151.3.3 clear_all_entries()	956
6.151.3.4 show_statistics()	956
6.151.3.5 remember_tables()	956
6.151.3.6 init_table()	956
6.151.4 Member Data Documentation	956
6.151.4.1 table_size	956
6.151.4.2 max_assoc_size	956
6.151.4.3 remember_strategy	956
6.152 GiNaC::remember_table_entry Class Reference	957
6.152.1 Detailed Description	957
6.152.2 Constructor & Destructor Documentation	957
6.152.2.1 remember_table_entry()	957
6.152.3 Member Function Documentation	958
6.152.3.1 is_equal()	958

6.152.3.2 <code>get_result()</code>	958
6.152.3.3 <code>get_last_access()</code>	958
6.152.3.4 <code>get_successful_hits()</code>	958
6.152.4 Member Data Documentation	958
6.152.4.1 <code>hashvalue</code>	958
6.152.4.2 <code>seq</code>	958
6.152.4.3 <code>result</code>	959
6.152.4.4 <code>last_access</code>	959
6.152.4.5 <code>successful_hits</code>	959
6.152.4.6 <code>access_counter</code>	959
6.153 <code>GiNaC::remember_table_list</code> Class Reference	959
6.153.1 Detailed Description	960
6.153.2 Constructor & Destructor Documentation	960
6.153.2.1 <code>remember_table_list()</code>	960
6.153.3 Member Function Documentation	960
6.153.3.1 <code>add_entry()</code>	960
6.153.3.2 <code>lookup_entry()</code>	960
6.153.4 Member Data Documentation	960
6.153.4.1 <code>max_assoc_size</code>	960
6.153.4.2 <code>remember_strategy</code>	961
6.154 <code>GiNaC::return_type_t</code> Struct Reference	961
6.154.1 Detailed Description	961
6.154.2 Member Function Documentation	961
6.154.2.1 <code>operator<()</code>	961
6.154.2.2 <code>operator==(())</code>	962
6.154.2.3 <code>operator"!=()</code>	962
6.154.3 Member Data Documentation	962
6.154.3.1 <code>tinfo</code>	962
6.154.3.2 <code>rl</code>	962
6.155 <code>GiNaC::return_types</code> Class Reference	962
6.155.1 Member Enumeration Documentation	962
6.155.1.1 anonymous enum	962
6.156 <code>GiNaC::relational::safe_bool_helper</code> Struct Reference	963
6.156.1 Member Function Documentation	963
6.156.1.1 <code>nonnull()</code>	963
6.157 <code>GiNaC::scalar_products</code> Class Reference	963
6.157.1 Detailed Description	964
6.157.2 Member Function Documentation	964
6.157.2.1 <code>add()</code> [1/2]	964
6.157.2.2 <code>add()</code> [2/2]	964
6.157.2.3 <code>add_vectors()</code>	964
6.157.2.4 <code>clear()</code>	965

6.157.2.5 is_defined()	965
6.157.2.6 evaluate()	965
6.157.2.7 debugprint()	965
6.157.3 Member Data Documentation	965
6.157.3.1 spm	965
6.158 GiNaC::series_options Class Reference	966
6.158.1 Detailed Description	966
6.158.2 Member Enumeration Documentation	966
6.158.2.1 anonymous enum	966
6.159 GiNaC::solve_algo Class Reference	966
6.159.1 Detailed Description	966
6.159.2 Member Enumeration Documentation	967
6.159.2.1 anonymous enum	967
6.160 GiNaC::spinidx Class Reference	967
6.160.1 Detailed Description	973
6.160.2 Constructor & Destructor Documentation	973
6.160.2.1 spinidx()	973
6.160.3 Member Function Documentation	973
6.160.3.1 is_dummy_pair_same_type()	973
6.160.3.2 conjugate()	974
6.160.3.3 archive()	974
6.160.3.4 read_archive()	974
6.160.3.5 match_same_type()	975
6.160.3.6 is_dotted()	975
6.160.3.7 is_undotted()	975
6.160.3.8 toggle_dot()	975
6.160.3.9 toggle_variance_dot()	976
6.160.3.10 do_print()	976
6.160.3.11 do_print_latex()	976
6.160.3.12 do_print_tree()	976
6.160.4 Member Data Documentation	976
6.160.4.1 dotted	976
6.161 GiNaC::spinmetric Class Reference	977
6.161.1 Detailed Description	981
6.161.2 Member Function Documentation	981
6.161.2.1 info()	981
6.161.2.2 eval_indexed()	981
6.161.2.3 contract_with()	982
6.161.2.4 do_print()	982
6.161.2.5 do_print_latex()	982
6.162 GiNaC::spmapkey Class Reference	982
6.162.1 Constructor & Destructor Documentation	983

6.162.1.1 spmapkey() [1/2]	983
6.162.1.2 spmapkey() [2/2]	983
6.162.2 Member Function Documentation	983
6.162.2.1 operator==()	983
6.162.2.2 operator<()	983
6.162.2.3 debugprint()	983
6.162.3 Member Data Documentation	983
6.162.3.1 v1	983
6.162.3.2 v2	984
6.162.3.3 dim	984
6.163 GiNaC::status_flags Class Reference	984
6.163.1 Detailed Description	984
6.163.2 Member Enumeration Documentation	984
6.163.2.1 anonymous enum	984
6.164 GiNaC::structure< T, ComparisonPolicy > Class Template Reference	985
6.164.1 Detailed Description	989
6.164.2 Constructor & Destructor Documentation	989
6.164.2.1 structure()	989
6.164.3 Member Function Documentation	989
6.164.3.1 get_class_name()	989
6.164.3.2 eval()	989
6.164.3.3 evalm()	990
6.164.3.4 eval_ncmul()	990
6.164.3.5 eval_indexed()	990
6.164.3.6 print()	990
6.164.3.7 precedence()	991
6.164.3.8 info()	991
6.164.3.9 nops()	991
6.164.3.10 op()	991
6.164.3.11 operator[]() [1/4]	991
6.164.3.12 operator[]() [2/4]	992
6.164.3.13 let_op()	992
6.164.3.14 operator[]() [3/4]	992
6.164.3.15 operator[]() [4/4]	992
6.164.3.16 has()	992
6.164.3.17 match()	993
6.164.3.18 match_same_type()	993
6.164.3.19 subs()	993
6.164.3.20 map()	994
6.164.3.21 degree()	994
6.164.3.22 ldegree()	994
6.164.3.23 coeff()	994

6.164.3.24	expand()	994
6.164.3.25	collect()	995
6.164.3.26	derivative()	996
6.164.3.27	series()	996
6.164.3.28	normal()	997
6.164.3.29	to_rational()	997
6.164.3.30	to_polynomial()	997
6.164.3.31	integer_content()	997
6.164.3.32	smod()	997
6.164.3.33	max_coefficient()	998
6.164.3.34	get_free_indices()	998
6.164.3.35	add_indexed()	998
6.164.3.36	scalar_mul_indexed()	999
6.164.3.37	contract_with()	999
6.164.3.38	return_type()	1000
6.164.3.39	return_type_tinfo()	1000
6.164.3.40	is_equal_same_type()	1001
6.164.3.41	calchash()	1001
6.164.3.42	operator->()	1001
6.164.3.43	get_struct() [1/2]	1001
6.164.3.44	get_struct() [2/2]	1001
6.164.4	Member Data Documentation	1002
6.164.4.1	obj	1002
6.165	GiNaC::su3d Class Reference	1002
6.165.1	Detailed Description	1006
6.165.2	Member Function Documentation	1006
6.165.2.1	eval_indexed()	1006
6.165.2.2	contract_with()	1006
6.165.2.3	return_type()	1006
6.165.2.4	do_print()	1006
6.165.2.5	do_print_latex()	1007
6.166	GiNaC::su3f Class Reference	1007
6.166.1	Detailed Description	1011
6.166.2	Member Function Documentation	1011
6.166.2.1	eval_indexed()	1011
6.166.2.2	contract_with()	1011
6.166.2.3	return_type()	1011
6.166.2.4	do_print()	1011
6.166.2.5	do_print_latex()	1012
6.167	GiNaC::su3one Class Reference	1012
6.167.1	Detailed Description	1016
6.167.2	Member Function Documentation	1016

6.167.2.1 do_print()	1016
6.167.2.2 do_print_latex()	1016
6.168 GiNaC::su3t Class Reference	1016
6.168.1 Detailed Description	1020
6.168.2 Member Function Documentation	1020
6.168.2.1 contract_with()	1020
6.168.2.2 do_print()	1020
6.168.2.3 do_print_latex()	1021
6.169 GiNaC::subs_options Class Reference	1021
6.169.1 Detailed Description	1021
6.169.2 Member Enumeration Documentation	1021
6.169.2.1 anonymous enum	1021
6.170 GiNaC::sy_is_less Class Reference	1022
6.170.1 Constructor & Destructor Documentation	1022
6.170.1.1 sy_is_less()	1022
6.170.2 Member Function Documentation	1022
6.170.2.1 operator()()	1022
6.170.3 Member Data Documentation	1022
6.170.3.1 v	1022
6.171 GiNaC::sy_swap Class Reference	1023
6.171.1 Constructor & Destructor Documentation	1023
6.171.1.1 sy_swap()	1023
6.171.2 Member Function Documentation	1023
6.171.2.1 operator()()	1023
6.171.3 Member Data Documentation	1023
6.171.3.1 v	1023
6.171.3.2 swapped	1023
6.172 GiNaC::sym_desc Struct Reference	1024
6.172.1 Detailed Description	1024
6.172.2 Constructor & Destructor Documentation	1024
6.172.2.1 sym_desc()	1024
6.172.3 Member Function Documentation	1025
6.172.3.1 operator<()	1025
6.172.4 Member Data Documentation	1025
6.172.4.1 sym	1025
6.172.4.2 deg_a	1025
6.172.4.3 deg_b	1025
6.172.4.4 ldeg_a	1025
6.172.4.5 ldeg_b	1025
6.172.4.6 max_deg	1026
6.172.4.7 max_lcnops	1026
6.173 GiNaC::symbol Class Reference	1026

6.173.1 Detailed Description	1030
6.173.2 Constructor & Destructor Documentation	1030
6.173.2.1 symbol() [1/2]	1030
6.173.2.2 symbol() [2/2]	1030
6.173.3 Member Function Documentation	1031
6.173.3.1 info()	1031
6.173.3.2 eval()	1031
6.173.3.3 evalf()	1031
6.173.3.4 series()	1031
6.173.3.5 subs()	1032
6.173.3.6 normal()	1032
6.173.3.7 to_rational()	1032
6.173.3.8 to_polynomial()	1032
6.173.3.9 conjugate()	1033
6.173.3.10 real_part()	1033
6.173.3.11 imag_part()	1033
6.173.3.12 is_polynomial()	1033
6.173.3.13 archive()	1033
6.173.3.14 read_archive()	1034
6.173.3.15 derivative()	1034
6.173.3.16 is_equal_same_type()	1034
6.173.3.17 calchash()	1035
6.173.3.18 get_domain()	1035
6.173.3.19 set_name()	1035
6.173.3.20 set_TeX_name()	1035
6.173.3.21 get_name()	1035
6.173.3.22 get_TeX_name()	1036
6.173.3.23 do_print()	1036
6.173.3.24 do_print_latex()	1036
6.173.3.25 do_print_tree()	1036
6.173.3.26 do_print_python_repr()	1036
6.173.4 Member Data Documentation	1036
6.173.4.1 serial	1036
6.173.4.2 name	1037
6.173.4.3 TeX_name	1037
6.173.4.4 next_serial	1037
6.174 GiNaC::symbolset Class Reference	1037
6.174.1 Constructor & Destructor Documentation	1038
6.174.1.1 symbolset()	1038
6.174.2 Member Function Documentation	1038
6.174.2.1 insert_symbols()	1038
6.174.2.2 has()	1038

6.174.3 Member Data Documentation	1038
6.174.3.1 s	1038
6.175 GiNaC::symmetry Class Reference	1039
6.175.1 Detailed Description	1043
6.175.2 Member Enumeration Documentation	1043
6.175.2.1 symmetry_type	1043
6.175.3 Constructor & Destructor Documentation	1043
6.175.3.1 symmetry() [1/2]	1043
6.175.3.2 symmetry() [2/2]	1043
6.175.4 Member Function Documentation	1044
6.175.4.1 archive()	1044
6.175.4.2 read_archive()	1044
6.175.4.3 calchash()	1044
6.175.4.4 get_type()	1045
6.175.4.5 set_type()	1045
6.175.4.6 add()	1045
6.175.4.7 validate()	1045
6.175.4.8 has_symmetry()	1045
6.175.4.9 has_nonsymmetric()	1046
6.175.4.10 has_cyclic()	1046
6.175.4.11 do_print()	1046
6.175.4.12 do_print_tree()	1046
6.175.5 Friends And Related Symbol Documentation	1046
6.175.5.1 sy_is_less	1046
6.175.5.2 sy_swap	1046
6.175.5.3 canonicalize	1046
6.175.6 Member Data Documentation	1047
6.175.6.1 type	1047
6.175.6.2 indices	1047
6.175.6.3 children	1047
6.176 GiNaC::symminfo Class Reference	1047
6.176.1 Detailed Description	1048
6.176.2 Constructor & Destructor Documentation	1048
6.176.2.1 symminfo() [1/2]	1048
6.176.2.2 symminfo() [2/2]	1048
6.176.3 Member Data Documentation	1048
6.176.3.1 symmterm	1048
6.176.3.2 coeff	1049
6.176.3.3 orig	1049
6.176.3.4 num	1049
6.177 GiNaC::symminfo_is_less_by_orig Class Reference	1049
6.177.1 Member Function Documentation	1049

6.177.1.1 operator()	1049
6.178 GiNaC::symminfo_is_less_by_symmterm Class Reference	1050
6.178.1 Member Function Documentation	1050
6.178.1.1 operator()	1050
6.179 GiNaC::tensdelta Class Reference	1050
6.179.1 Detailed Description	1054
6.179.2 Member Function Documentation	1054
6.179.2.1 info()	1054
6.179.2.2 eval_indexed()	1054
6.179.2.3 contract_with()	1055
6.179.2.4 return_type()	1055
6.179.2.5 do_print()	1055
6.179.2.6 do_print_latex()	1055
6.180 GiNaC::tensepsilon Class Reference	1055
6.180.1 Detailed Description	1059
6.180.2 Constructor & Destructor Documentation	1059
6.180.2.1 tensepsilon()	1059
6.180.3 Member Function Documentation	1060
6.180.3.1 info()	1060
6.180.3.2 eval_indexed()	1060
6.180.3.3 contract_with()	1060
6.180.3.4 archive()	1060
6.180.3.5 read_archive()	1061
6.180.3.6 return_type()	1061
6.180.3.7 do_print()	1061
6.180.3.8 do_print_latex()	1061
6.180.4 Member Data Documentation	1061
6.180.4.1 minkowski	1061
6.180.4.2 pos_sig	1062
6.181 GiNaC::tensmetric Class Reference	1062
6.181.1 Detailed Description	1066
6.181.2 Member Function Documentation	1066
6.181.2.1 info()	1066
6.181.2.2 eval_indexed()	1066
6.181.2.3 contract_with()	1066
6.181.2.4 return_type()	1067
6.181.2.5 do_print()	1067
6.182 GiNaC::tensor Class Reference	1067
6.182.1 Detailed Description	1071
6.182.2 Member Function Documentation	1071
6.182.2.1 return_type()	1071
6.182.2.2 replace_contr_index()	1071

6.183 GiNaC::terminfo Class Reference	1071
6.183.1 Detailed Description	1072
6.183.2 Constructor & Destructor Documentation	1072
6.183.2.1 terminfo()	1072
6.183.3 Member Data Documentation	1072
6.183.3.1 orig	1072
6.183.3.2 symm	1072
6.184 GiNaC::terminfo_is_less Class Reference	1073
6.184.1 Member Function Documentation	1073
6.184.1.1 operator()()	1073
6.185 GiNaC::Gt_detail::TransformExpressionWithCache< Type > Class Template Reference	1073
6.185.1 Constructor & Destructor Documentation	1074
6.185.1.1 TransformExpressionWithCache()	1074
6.185.2 Member Function Documentation	1074
6.185.2.1 operator()()	1074
6.185.2.2 print_cache()	1074
6.185.2.3 get_cache()	1074
6.185.2.4 clear_cache()	1074
6.185.2.5 check_type() [1/2]	1075
6.185.2.6 check_type() [2/2]	1075
6.185.2.7 impl()	1075
6.185.3 Member Data Documentation	1075
6.185.3.1 func_obj	1075
6.185.3.2 func_default	1075
6.185.3.3 cache	1075
6.186 GiNaC::class_info< OPT >::tree_node Struct Reference	1076
6.186.1 Constructor & Destructor Documentation	1076
6.186.1.1 tree_node()	1076
6.186.2 Member Function Documentation	1076
6.186.2.1 add_child()	1076
6.186.3 Member Data Documentation	1076
6.186.3.1 children	1076
6.186.3.2 info	1076
6.187 GiNaC::unarchive_table_t Class Reference	1077
6.187.1 Constructor & Destructor Documentation	1077
6.187.1.1 unarchive_table_t()	1077
6.187.1.2 ~unarchive_table_t()	1077
6.187.2 Member Function Documentation	1077
6.187.2.1 find()	1077
6.187.2.2 insert()	1077
6.187.3 Member Data Documentation	1078
6.187.3.1 usecount	1078

6.187.3.2 unarch_map	1078
6.188 GiNaC::user_defined_kernel Class Reference	1078
6.188.1 Detailed Description	1082
6.188.2 Constructor & Destructor Documentation	1083
6.188.2.1 user_defined_kernel()	1083
6.188.3 Member Function Documentation	1083
6.188.3.1 nops()	1083
6.188.3.2 op()	1083
6.188.3.3 let_op()	1083
6.188.3.4 is_numeric()	1083
6.188.3.5 Laurent_series()	1084
6.188.3.6 uses_Laurent_series()	1084
6.188.3.7 do_print()	1084
6.188.4 Member Data Documentation	1084
6.188.4.1 f	1084
6.188.4.2 x	1084
6.189 GiNaC::varidx Class Reference	1085
6.189.1 Detailed Description	1089
6.189.2 Constructor & Destructor Documentation	1089
6.189.2.1 varidx()	1089
6.189.3 Member Function Documentation	1090
6.189.3.1 is_dummy_pair_same_type()	1090
6.189.3.2 archive()	1090
6.189.3.3 read_archive()	1090
6.189.3.4 match_same_type()	1091
6.189.3.5 is_covariant()	1091
6.189.3.6 is_contravariant()	1091
6.189.3.7 toggle_variance()	1091
6.189.3.8 do_print()	1092
6.189.3.9 do_print_tree()	1092
6.189.4 Member Data Documentation	1092
6.189.4.1 covariant	1092
6.190 GiNaC::visitor Class Reference	1092
6.190.1 Detailed Description	1093
6.190.2 Constructor & Destructor Documentation	1093
6.190.2.1 ~visitor()	1093
6.191 GiNaC::wildcard Class Reference	1093
6.191.1 Detailed Description	1097
6.191.2 Constructor & Destructor Documentation	1097
6.191.2.1 wildcard()	1097
6.191.3 Member Function Documentation	1097
6.191.3.1 match()	1097

6.191.3.2 archive()	1097
6.191.3.3 read_archive()	1098
6.191.3.4 calchash()	1098
6.191.3.5 get_label()	1098
6.191.3.6 do_print()	1098
6.191.3.7 do_print_tree()	1098
6.191.3.8 do_print_python_repr()	1099
6.191.4 Member Data Documentation	1099
6.191.4.1 label	1099
6.192 GiNaC::zeta1_SERIAL Class Reference	1099
6.192.1 Detailed Description	1099
6.192.2 Member Data Documentation	1100
6.192.2.1 serial	1100
6.193 GiNaC::zeta2_SERIAL Class Reference	1100
6.193.1 Detailed Description	1100
6.193.2 Member Data Documentation	1100
6.193.2.1 serial	1100
7 File Documentation	1101
7.1 add.cpp File Reference	1101
7.1.1 Detailed Description	1101
7.2 add.h File Reference	1102
7.2.1 Detailed Description	1102
7.3 archive.cpp File Reference	1102
7.3.1 Detailed Description	1103
7.4 archive.h File Reference	1103
7.4.1 Detailed Description	1104
7.4.2 Macro Definition Documentation	1104
7.4.2.1 GINAC_DECLARE_UNARCHIVER	1104
7.4.2.2 GINAC_BIND_UNARCHIVER	1105
7.5 assertion.h File Reference	1105
7.5.1 Detailed Description	1106
7.5.2 Macro Definition Documentation	1106
7.5.2.1 GINAC_ASSERT	1106
7.6 basic.cpp File Reference	1106
7.6.1 Detailed Description	1107
7.7 basic.h File Reference	1107
7.7.1 Detailed Description	1108
7.8 class_info.h File Reference	1108
7.8.1 Detailed Description	1109
7.9 clifford.cpp File Reference	1109
7.9.1 Detailed Description	1111

7.10 clifford.h File Reference	1111
7.10.1 Detailed Description	1113
7.11 color.cpp File Reference	1113
7.11.1 Detailed Description	1114
7.11.2 Macro Definition Documentation	1114
7.11.2.1 TEST_PERMUTATION	1114
7.11.2.2 CMPINDICES	1115
7.12 color.h File Reference	1115
7.12.1 Detailed Description	1116
7.13 compiler.h File Reference	1116
7.13.1 Detailed Description	1116
7.13.2 Macro Definition Documentation	1116
7.13.2.1 unlikely	1116
7.13.2.2 likely	1116
7.13.2.3 attribute_pure	1116
7.14 constant.cpp File Reference	1117
7.14.1 Detailed Description	1117
7.15 constant.h File Reference	1117
7.15.1 Detailed Description	1118
7.16 container.h File Reference	1118
7.16.1 Detailed Description	1119
7.17 crc32.h File Reference	1119
7.17.1 Detailed Description	1119
7.18 ex.cpp File Reference	1119
7.18.1 Detailed Description	1120
7.19 ex.h File Reference	1120
7.19.1 Detailed Description	1122
7.20 excompiler.cpp File Reference	1122
7.20.1 Detailed Description	1123
7.21 excompiler.h File Reference	1123
7.21.1 Detailed Description	1124
7.22 expair.cpp File Reference	1124
7.22.1 Detailed Description	1124
7.23 expair.h File Reference	1124
7.23.1 Detailed Description	1125
7.24 expairseq.cpp File Reference	1125
7.24.1 Detailed Description	1126
7.25 expairseq.h File Reference	1126
7.25.1 Detailed Description	1126
7.26 exprseq.cpp File Reference	1127
7.26.1 Detailed Description	1127
7.27 exprseq.h File Reference	1127

7.27.1 Detailed Description	1127
7.28 factor.cpp File Reference	1128
7.28.1 Detailed Description	1128
7.28.2 Macro Definition Documentation	1129
7.28.2.1 DCOUT	1129
7.28.2.2 DCOUTVAR	1129
7.28.2.3 DCOUT2	1129
7.28.2.4 USE_SAME_DEGREE_FACTOR	1129
7.28.3 Variable Documentation	1129
7.28.3.1 value	1129
7.28.3.2 r	1129
7.28.3.3 c	1130
7.28.3.4 m	1131
7.28.3.5 lr	1131
7.28.3.6 cache	1131
7.28.3.7 factors	1131
7.28.3.8 one	1131
7.28.3.9 n	1132
7.28.3.10 len	1132
7.28.3.11 last	1132
7.28.3.12 k	1133
7.28.3.13 poly	1133
7.28.3.14 x	1133
7.28.3.15 evalpoint	1134
7.28.3.16 R	1134
7.28.3.17 syms_wox	1134
7.28.3.18 unit	1134
7.28.3.19 cont	1134
7.28.3.20 pp	1135
7.28.3.21 vn	1135
7.28.3.22 vnlst	1135
7.28.3.23 modulus	1135
7.28.3.24 syms	1135
7.28.3.25 options	1135
7.29 factor.h File Reference	1136
7.29.1 Detailed Description	1136
7.30 fail.cpp File Reference	1136
7.30.1 Detailed Description	1136
7.31 fail.h File Reference	1137
7.31.1 Detailed Description	1137
7.32 fderivative.cpp File Reference	1137
7.32.1 Detailed Description	1138

7.33 fderivative.h File Reference	1138
7.33.1 Detailed Description	1138
7.34 flags.h File Reference	1138
7.34.1 Detailed Description	1139
7.35 function.cpp File Reference	1139
7.35.1 Detailed Description	1140
7.36 function.h File Reference	1140
7.36.1 Detailed Description	1147
7.36.2 Macro Definition Documentation	1147
7.36.2.1 DECLARE_FUNCTION_1P	1147
7.36.2.2 DECLARE_FUNCTION_2P	1147
7.36.2.3 DECLARE_FUNCTION_3P	1147
7.36.2.4 DECLARE_FUNCTION_4P	1148
7.36.2.5 DECLARE_FUNCTION_5P	1148
7.36.2.6 DECLARE_FUNCTION_6P	1148
7.36.2.7 DECLARE_FUNCTION_7P	1148
7.36.2.8 DECLARE_FUNCTION_8P	1149
7.36.2.9 DECLARE_FUNCTION_9P	1149
7.36.2.10 DECLARE_FUNCTION_10P	1149
7.36.2.11 DECLARE_FUNCTION_11P	1149
7.36.2.12 DECLARE_FUNCTION_12P	1150
7.36.2.13 DECLARE_FUNCTION_13P	1150
7.36.2.14 DECLARE_FUNCTION_14P	1150
7.36.2.15 REGISTER_FUNCTION	1150
7.36.2.16 is_ex_the_function	1151
7.37 ginac.h File Reference	1151
7.37.1 Detailed Description	1151
7.38 Gt.cpp File Reference	1152
7.38.1 Detailed Description	1152
7.38.2 Macro Definition Documentation	1152
7.38.2.1 DEBUG_GT	1152
7.39 Gt.h File Reference	1153
7.39.1 Detailed Description	1153
7.40 Gt_helpers.cpp File Reference	1153
7.40.1 Detailed Description	1154
7.41 Gt_helpers.h File Reference	1154
7.41.1 Detailed Description	1155
7.42 hash_map.h File Reference	1155
7.42.1 Detailed Description	1155
7.43 hash_seed.h File Reference	1155
7.43.1 Detailed Description	1156
7.44 idx.cpp File Reference	1156

7.44.1 Detailed Description	1157
7.45 idx.h File Reference	1157
7.45.1 Detailed Description	1158
7.46 indexed.cpp File Reference	1158
7.46.1 Detailed Description	1159
7.47 indexed.h File Reference	1160
7.47.1 Detailed Description	1160
7.48 inifcns.cpp File Reference	1161
7.48.1 Detailed Description	1164
7.49 inifcns.h File Reference	1164
7.49.1 Detailed Description	1165
7.50 inifcns_elliptic.cpp File Reference	1165
7.50.1 Detailed Description	1166
7.51 inifcns_gamma.cpp File Reference	1167
7.51.1 Detailed Description	1168
7.52 inifcns_nstdsums.cpp File Reference	1168
7.52.1 Detailed Description	1169
7.53 inifcns_trans.cpp File Reference	1170
7.53.1 Detailed Description	1173
7.54 integral.cpp File Reference	1173
7.54.1 Detailed Description	1174
7.55 integral.h File Reference	1174
7.55.1 Detailed Description	1174
7.56 integration_kernel.cpp File Reference	1174
7.56.1 Detailed Description	1176
7.56.2 Variable Documentation	1176
7.56.2.1 qbar	1176
7.56.2.2 order	1176
7.56.2.3 cache_vec	1176
7.56.2.4 x	1176
7.57 integration_kernel.h File Reference	1176
7.57.1 Detailed Description	1178
7.58 lst.cpp File Reference	1178
7.58.1 Detailed Description	1178
7.59 lst.h File Reference	1178
7.59.1 Detailed Description	1179
7.60 matrix.cpp File Reference	1179
7.60.1 Detailed Description	1180
7.61 matrix.h File Reference	1180
7.61.1 Detailed Description	1181
7.62 mul.cpp File Reference	1181
7.62.1 Detailed Description	1182

7.63 mul.h File Reference	1182
7.63.1 Detailed Description	1182
7.64 ncmul.cpp File Reference	1183
7.64.1 Detailed Description	1183
7.65 ncmul.h File Reference	1183
7.65.1 Detailed Description	1184
7.66 normal.cpp File Reference	1184
7.66.1 Detailed Description	1186
7.66.2 Macro Definition Documentation	1186
7.66.2.1 FAST_COMPARE	1186
7.66.2.2 USE_REMEMBER	1187
7.66.2.3 USE_TRIAL_DIVISION	1187
7.66.2.4 STATISTICS	1187
7.67 normal.h File Reference	1187
7.67.1 Detailed Description	1188
7.68 numeric.cpp File Reference	1188
7.68.1 Detailed Description	1191
7.69 numeric.h File Reference	1191
7.69.1 Detailed Description	1194
7.70 operators.cpp File Reference	1194
7.70.1 Detailed Description	1196
7.71 operators.h File Reference	1196
7.71.1 Detailed Description	1197
7.72 power.cpp File Reference	1198
7.72.1 Detailed Description	1198
7.73 power.h File Reference	1198
7.73.1 Detailed Description	1199
7.74 print.cpp File Reference	1199
7.74.1 Detailed Description	1199
7.75 print.h File Reference	1200
7.75.1 Detailed Description	1201
7.75.2 Macro Definition Documentation	1201
7.75.2.1 GINAC_DECLARE_PRINT_CONTEXT_COMMON	1201
7.75.2.2 GINAC_DECLARE_PRINT_CONTEXT_BASE	1201
7.75.2.3 GINAC_DECLARE_PRINT_CONTEXT	1202
7.75.2.4 GINAC_IMPLEMENT_PRINT_CONTEXT	1202
7.76 pseries.cpp File Reference	1202
7.76.1 Detailed Description	1203
7.77 pseries.h File Reference	1203
7.77.1 Detailed Description	1203
7.78 ptr.h File Reference	1204
7.78.1 Detailed Description	1204

7.79 registrar.cpp File Reference	1204
7.79.1 Detailed Description	1204
7.80 registrar.h File Reference	1205
7.80.1 Detailed Description	1206
7.80.2 Macro Definition Documentation	1206
7.80.2.1 GINAC_DECLARE_REGISTERED_CLASS_COMMON	1206
7.80.2.2 GINAC_DECLARE_REGISTERED_CLASS_NO_CTORS	1206
7.80.2.3 GINAC_DECLARE_REGISTERED_CLASS	1207
7.80.2.4 GINAC_IMPLEMENT_REGISTERED_CLASS	1207
7.80.2.5 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT	1207
7.80.2.6 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT_T	1208
7.81 relational.cpp File Reference	1208
7.81.1 Detailed Description	1208
7.82 relational.h File Reference	1208
7.82.1 Detailed Description	1209
7.83 remember.cpp File Reference	1209
7.83.1 Detailed Description	1209
7.84 remember.h File Reference	1209
7.84.1 Detailed Description	1210
7.85 structure.h File Reference	1210
7.85.1 Detailed Description	1210
7.86 symbol.cpp File Reference	1211
7.86.1 Detailed Description	1211
7.87 symbol.h File Reference	1211
7.87.1 Detailed Description	1212
7.88 symmetry.cpp File Reference	1212
7.88.1 Detailed Description	1213
7.89 symmetry.h File Reference	1213
7.89.1 Detailed Description	1214
7.90 tensor.cpp File Reference	1215
7.90.1 Detailed Description	1215
7.91 tensor.h File Reference	1216
7.91.1 Detailed Description	1217
7.92 utils.cpp File Reference	1217
7.92.1 Detailed Description	1219
7.93 utils.h File Reference	1219
7.93.1 Detailed Description	1220
7.93.2 Macro Definition Documentation	1221
7.93.2.1 DEFAULT_CTOR	1221
7.93.2.2 DEFAULT_COMPARE	1221
7.93.2.3 DEFAULT_PRINT	1221
7.93.2.4 DEFAULT_PRINT_LATEX	1221

7.94 <code>utils_multi_iterator.h</code> File Reference	1221
7.94.1 Detailed Description	1223
7.95 <code>version.h</code> File Reference	1223
7.95.1 Detailed Description	1223
7.95.2 Macro Definition Documentation	1224
7.95.2.1 <code>GINACLIB_MAJOR_VERSION</code>	1224
7.95.2.2 <code>GINACLIB_MINOR_VERSION</code>	1224
7.95.2.3 <code>GINACLIB_MICRO_VERSION</code>	1224
7.95.2.4 <code>GINAC_LT_CURRENT</code>	1224
7.95.2.5 <code>GINAC_LT_REVISION</code>	1224
7.95.2.6 <code>GINAC_LT_AGE</code>	1224
7.95.2.7 <code>GINACLIB_ARCHIVE_VERSION</code>	1224
7.95.2.8 <code>GINACLIB_ARCHIVE_AGE</code>	1224
7.95.2.9 <code>GINACLIB_STR_HELPER</code>	1224
7.95.2.10 <code>GINACLIB_STR</code>	1224
7.95.2.11 <code>GINACLIB_VERSION</code>	1225
7.96 <code>wildcard.cpp</code> File Reference	1225
7.96.1 Detailed Description	1225
7.97 <code>wildcard.h</code> File Reference	1225
7.97.1 Detailed Description	1226

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

GiNaC	19
GiNaC::Gt_detail	273
GiNaC::internal	274
std	274

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

GiNaC::internal::_iter_rep	277
GiNaC::_numeric_digits	278
GiNaC::archive	296
GiNaC::archive_node	303
GiNaC::archive_node::archive_node_cit_range	311
GiNaC::archive::archived_ex	312
GiNaC::basic_multi_iterator< T >	342
GiNaC::multi_iterator_counter< T >	771
GiNaC::multi_iterator_counter_indv< T >	774
GiNaC::multi_iterator_ordered< T >	777
GiNaC::multi_iterator_ordered_eq< T >	780
GiNaC::multi_iterator_ordered_eq_indv< T >	783
GiNaC::multi_iterator_permutation< T >	786
GiNaC::multi_iterator_shuffle< T >	790
GiNaC::multi_iterator_shuffle_prime< T >	793
GiNaC::basic_partition_generator	348
GiNaC::partition_generator	843
GiNaC::partition_with_zero_parts_generator	845
GiNaC::class_info< OPT >	349
GiNaC::compare_all_equal< T >	380
GiNaC::compare_bitwise< T >	381
GiNaC::compare_std_less< T >	382
ComparisonPolicy	
GiNaC::structure< T, ComparisonPolicy >	985
GiNaC::composition_generator	383
GiNaC::const_iterator	385
GiNaC::const_postorder_iterator	390
GiNaC::const_preorder_iterator	393
GiNaC::container_storage< C >	420
GiNaC::container< C >	405
GiNaC::function	562
GiNaC::fderivative	550
GiNaC::indexed	653
GiNaC::clifford	352

GiNaC::color	370
GiNaC::ncmul	803
GiNaC::composition_generator::coolmulti	423
GiNaC::determinant_algo	426
GiNaC::do_taylor	449
GiNaC::domain	450
std::domain_error	
GiNaC::pole_error	865
GiNaC::dunno	450
GiNaC::composition_generator::coolmulti::element	474
std::equal_to< GiNaC::ex >	482
GiNaC::error_and_integral	483
GiNaC::error_and_integral_is_less	484
GiNaC::ex	487
GiNaC::ex_base_is_less	519
GiNaC::ex_is_equal	519
GiNaC::ex_is_less	519
GiNaC::ex_swap	520
GiNaC::expair	520
GiNaC::expair_is_less	524
GiNaC::expair_rest_is_less	524
GiNaC::expair_swap	525
GiNaC::expand_options	544
GiNaC::factor_options	545
GiNaC::function_options	581
GiNaC::G2_SERIAL	621
GiNaC::G3_SERIAL	622
GiNaC::gcd_options	622
GiNaC::gcdheu_failed	623
GiNaC::has_distance< T >	639
GiNaC::has_options	641
std::hash< GiNaC::ex >	641
GiNaC::idx_is_equal_ignore_dim	653
GiNaC::info_flags	671
GiNaC::is_not_a_clifford	690
GiNaC::is_summation_idx	691
GiNaC::iterated_integral2_SERIAL	691
GiNaC::iterated_integral3_SERIAL	692
GiNaC::Gt::kernel	693
GiNaC::lanczos_coeffs	709
std::less< GiNaC::ptr< T > >	711
GiNaC::library_init	711
std::list	
GiNaC::remember_table_list	959
GiNaC::make_flat_inserter	713
GiNaC::map_function	715
GiNaC::derivative_map_function	425
GiNaC::eval_integ_map_function	484
GiNaC::evalf_map_function	485
GiNaC::evalm_map_function	486
GiNaC::expand_map_function	543
GiNaC::normal_map_function	815
GiNaC::pointer_to_map_function	852
GiNaC::pointer_to_map_function_1arg< T1 >	853
GiNaC::pointer_to_map_function_2args< T1, T2 >	854
GiNaC::pointer_to_map_function_3args< T1, T2, T3 >	856
GiNaC::pointer_to_member_to_map_function< C >	858
GiNaC::pointer_to_member_to_map_function_1arg< C, T1 >	860

GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 >	861
GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >	863
GiNaC::basic_partition_generator::mpartition2	751
GiNaC::op0_is_equal	842
GiNaC::Gt_detail::pathintegral_term	846
GiNaC::Gt_detail::pathintegral_term_list	849
GiNaC::print_context	887
GiNaC::print_csrc	891
GiNaC::print_csrc_cl_N	892
GiNaC::print_csrc_double	893
GiNaC::print_csrc_float	894
GiNaC::print_dflt	895
GiNaC::print_latex	899
GiNaC::print_python	905
GiNaC::print_python_repr	906
GiNaC::print_tree	907
GiNaC::print_context_options	889
GiNaC::print_functor	896
GiNaC::print_functor_impl	898
GiNaC::print_memfun_handler< T, C >	900
GiNaC::print_ptrfun_handler< T, C >	903
GiNaC::print_options	902
GiNaC::archive_node::property	908
GiNaC::archive_node::property_info	909
GiNaC::psi1_SERIAL	926
GiNaC::psi2_SERIAL	927
GiNaC::ptr< T >	928
GiNaC::ptr< GiNaC::basic >	928
GiNaC::refcounted	938
GiNaC::basic	313
GiNaC::Gt	623
GiNaC::constant	396
GiNaC::container< C >	405
GiNaC::expairseq	525
GiNaC::add	281
GiNaC::mul	753
GiNaC::fail	546
GiNaC::idx	642
GiNaC::varidx	1085
GiNaC::spinidx	967
GiNaC::integral	672
GiNaC::integration_kernel	683
GiNaC::ELi_kernel	476
GiNaC::Ebar_kernel	451
GiNaC::Eisenstein_h_kernel	458
GiNaC::Eisenstein_kernel	466
GiNaC::Kronecker_dtau_kernel	695
GiNaC::Kronecker_dz_kernel	702
GiNaC::basic_log_kernel	338
GiNaC::modular_form_kernel	744
GiNaC::multiple_polylog_kernel	796
GiNaC::user_defined_kernel	1078
GiNaC::matrix	716
GiNaC::numeric	816
GiNaC::power	872
GiNaC::pseries	911
GiNaC::relational	943

GiNaC::structure< T, ComparisonPolicy >	985
GiNaC::symbol	1026
GiNaC::realsymbol	932
GiNaC::possymbol	867
GiNaC::symmetry	1039
GiNaC::tensor	1067
GiNaC::cliffordunit	365
GiNaC::diracgamma	427
GiNaC::diracgamma5	432
GiNaC::diracgammaL	436
GiNaC::diracgammaR	441
GiNaC::diracone	445
GiNaC::su3d	1002
GiNaC::su3f	1007
GiNaC::su3one	1012
GiNaC::su3t	1016
GiNaC::tensdelta	1050
GiNaC::tensepsilon	1055
GiNaC::tensmetric	1062
GiNaC::minkmetric	738
GiNaC::spinmetric	977
GiNaC::wildcard	1093
GiNaC::registered_class_options	940
GiNaC::remember_strategies	953
GiNaC::remember_table_entry	957
GiNaC::return_type_t	961
GiNaC::return_types	962
GiNaC::relational::safe_bool_helper	963
GiNaC::scalar_products	963
GiNaC::series_options	966
GiNaC::solve_algo	966
GiNaC::spmapkey	982
GiNaC::status_flags	984
GiNaC::subs_options	1021
GiNaC::sy_is_less	1022
GiNaC::sy_swap	1023
GiNaC::sym_desc	1024
GiNaC::symbolset	1037
GiNaC::symminfo	1047
GiNaC::symminfo_is_less_by_orig	1049
GiNaC::symminfo_is_less_by_symmterm	1050
GiNaC::terminfo	1071
GiNaC::terminfo_is_less	1073
GiNaC::Gt_detail::TransformExpressionWithCache< Type >	1073
GiNaC::Gt_detail::TransformExpressionWithCache< GiNaC::G2_SERIAL >	1073
GiNaC::class_info< OPT >::tree_node	1076
GiNaC::unarchive_table_t	1077
std::vector	
GiNaC::remember_table	954
GiNaC::visitor	1092
GiNaC::zeta1_SERIAL	1099
GiNaC::zeta2_SERIAL	1100

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

GiNaC::internal::_iter_rep	277
GiNaC::_numeric_digits	
This class is used to instantiate a global singleton object Digits which behaves just like Maple's	
Digits	278
GiNaC::add	
Sum of expressions	281
GiNaC::archive	
This class holds archived versions of GiNaC expressions (class ex)	296
GiNaC::archive_node	
This class stores all properties needed to record/retrieve the state of one object of class basic	
(or a derived class)	303
GiNaC::archive_node::archive_node_cit_range	311
GiNaC::archive::archived_ex	
Archived expression descriptor	312
GiNaC::basic	
This class is the ABC (abstract base class) of GiNaC 's class hierarchy	313
GiNaC::basic_log_kernel	
The basic integration kernel with a logarithmic singularity at the origin	338
GiNaC::basic_multi_iterator< T >	
Basic_multi_iterator is a base class	342
GiNaC::basic_partition_generator	
Base class for generating all bounded combinatorial partitions of an integer n with exactly m	
parts in non-decreasing order	348
GiNaC::class_info< OPT >	349
GiNaC::clifford	
This class holds an object representing an element of the Clifford algebra (the Dirac gamma	
matrices)	352
GiNaC::cliffordunit	
This class represents the Clifford algebra generators (units)	365
GiNaC::color	
This class holds a generator T_a or the unity element of the Lie algebra of SU(3), as used for	
calculations in quantum chromodynamics	370
GiNaC::compare_all_equal< T >	
Comparison policy: all structures of one type are equal	380
GiNaC::compare_bitwise< T >	
Comparison policy: use bit-wise comparison to compare structures	381

GiNaC::compare_std_less< T >	
Comparison policy: use <code>std::equal_to</code> / <code>std::less</code> (defaults to operators <code>==</code> and <code><</code>) to compare structures	382
GiNaC::composition_generator	
Generate all compositions of a partition of an integer n , starting with the compositions which has non-decreasing order	383
GiNaC::const_iterator	385
GiNaC::const_postorder_iterator	390
GiNaC::const_preorder_iterator	393
GiNaC::constant	
This class holds constants, symbols with specific numerical value	396
GiNaC::container< C >	
Wrapper template for making GiNaC classes out of STL containers	405
GiNaC::container_storage< C >	
Helper template for encapsulating the reserve() mechanics of STL containers	420
GiNaC::composition_generator::coolmulti	423
GiNaC::derivative_map_function	
Function object to be applied by basic::derivative()	425
GiNaC::determinant_algo	
Switch to control algorithm for determinant computation	426
GiNaC::diracgamma	
This class represents the Dirac gamma Lorentz vector	427
GiNaC::diracgamma5	
This class represents the Dirac gamma5 object which anticommutes with all other gammas	432
GiNaC::diracgammaL	
This class represents the Dirac gammaL object which behaves like $1/2 (1-\text{gamma}5)$	436
GiNaC::diracgammaR	
This class represents the Dirac gammaL object which behaves like $1/2 (1+\text{gamma}5)$	441
GiNaC::diracone	
This class represents the Clifford algebra unity element	445
GiNaC::do_taylor	
Exception class thrown by classes which provide their own series expansion to signal that ordinary Taylor expansion is safe	449
GiNaC::domain	
Domain of an object	450
GiNaC::dunno	
Exception class thrown by functions to signal unimplemented functionality so the expression may just be <code>.hold()</code>	450
GiNaC::Ebar_kernel	
The Ebar-kernel	451
GiNaC::Eisenstein_h_kernel	
The kernel corresponding to the Eisenstein series $h_{k,N,r,s}(\tau)$	458
GiNaC::Eisenstein_kernel	
The kernel corresponding to the Eisenstein series $E_{k,N,a,b,K}(\tau)$	466
GiNaC::composition_generator::coolmulti::element	474
GiNaC::ELi_kernel	
The ELi-kernel	476
std::equal_to< GiNaC::ex >	
Specialization of <code>std::equal_to()</code> for <code>ex</code> objects	482
GiNaC::error_and_integral	483
GiNaC::error_and_integral_is_less	484
GiNaC::eval_integ_map_function	
Function object to be applied by basic::eval_integ()	484
GiNaC::evalf_map_function	
Function object to be applied by basic::evalf()	485
GiNaC::evalm_map_function	
Function object to be applied by basic::evalm()	486

GiNaC::ex	
Lightweight wrapper for GiNaC 's symbolic objects	487
GiNaC::ex_base_is_less	519
GiNaC::ex_is_equal	519
GiNaC::ex_is_less	519
GiNaC::ex_swap	520
GiNaC::expair	
A pair of expressions	520
GiNaC::expair_is_less	
Function object for insertion into third argument of STL's sort() etc	524
GiNaC::expair_rest_is_less	
Function object not caring about the numerical coefficients for insertion into third argument of STL's sort()	524
GiNaC::expair_swap	525
GiNaC::expairseq	
A sequence of class expair	525
GiNaC::expand_map_function	
Function object to be applied by basic::expand()	543
GiNaC::expand_options	
Flags to control the behavior of expand()	544
GiNaC::factor_options	
Flags to control the polynomial factorization	545
GiNaC::fail	546
GiNaC::fderivative	
This class represents the (abstract) derivative of a symbolic function	550
GiNaC::function	
The class function is used to implement builtin functions like sin, cos... and user defined functions	562
GiNaC::function_options	581
GiNaC::G2_SERIAL	
Generalized multiple polylogarithm	621
GiNaC::G3_SERIAL	
Generalized multiple polylogarithm with explicit imaginary parts	622
GiNaC::gcd_options	
Flags to control the behavior of gcd() and friends	622
GiNaC::gcdheu_failed	
Exception thrown by heur_gcd() to signal failure	623
GiNaC::Gt	
Elliptic Multiple Polylogarithm	623
GiNaC::has_distance< T >	
SFINAE test for distance	639
GiNaC::has_options	
Flags to control the behavior of has()	641
std::hash< GiNaC::ex >	
Specialization of std::hash() for ex objects	641
GiNaC::idx	
This class holds one index of an indexed object	642
GiNaC::idx_is_equal_ignore_dim	653
GiNaC::indexed	
This class holds an indexed expression	653
GiNaC::info_flags	
Possible attributes an object can have	671
GiNaC::integral	
Symbolic integral	672
GiNaC::integration_kernel	
The base class for integration kernels for iterated integrals	683
GiNaC::is_not_a_clifford	
Predicate for finding non-clifford objects	690
GiNaC::is_summation_idx	691

GiNaC::iterated_integral2_SERIAL	
Complete elliptic integral of the first kind	691
GiNaC::iterated_integral3_SERIAL	
Iterated integral with explicit truncation	692
GiNaC::Gt::kernel	693
GiNaC::Kronecker_dtau_kernel	
The kernel corresponding to integrating the Kronecker coefficient function $g^{(n)}(z_j, K\tau)$ in τ (or equivalently in $\bar{q}$)	695
GiNaC::Kronecker_dz_kernel	
The kernel corresponding to integrating the Kronecker coefficient function $g^{(n-1)}(z - z_j, K\tau)$ in z	702
GiNaC::lanczos_coeffs	709
std::less< GiNaC::ptr< T > >	
Specialization of <code>std::less</code> for <code>ptr<T></code> to enable ordering of <code>ptr<T></code> objects (e.g	711
GiNaC::library_init	
Helper class to initialize the library	711
GiNaC::make_flat_inserter	
Class to handle the renaming of dummy indices	713
GiNaC::map_function	
Function object for <code>map()</code>	715
GiNaC::matrix	
Symbolic matrices	716
GiNaC::minkmetric	
This class represents a Minkowski metric tensor	738
GiNaC::modular_form_kernel	
A kernel corresponding to a polynomial in Eisenstein series	744
GiNaC::basic_partition_generator::mpartition2	751
GiNaC::mul	
Product of expressions	753
GiNaC::multi_iterator_counter< T >	
The class <code>multi_iterator_counter</code> defines a <code>multi_iterator</code> $(i_1, i_2, \dots, i_k)$, such that	771
GiNaC::multi_iterator_counter_indv< T >	
The class <code>multi_iterator_counter_indv</code> defines a <code>multi_iterator</code> $(i_1, i_2, \dots, i_k)$, such that	774
GiNaC::multi_iterator_ordered< T >	
The class <code>multi_iterator_ordered</code> defines a <code>multi_iterator</code> $(i_1, i_2, \dots, i_k)$, such that	777
GiNaC::multi_iterator_ordered_eq< T >	
The class <code>multi_iterator_ordered_eq</code> defines a <code>multi_iterator</code> $(i_1, i_2, \dots, i_k)$, such that	780
GiNaC::multi_iterator_ordered_eq_indv< T >	
The class <code>multi_iterator_ordered_eq_indv</code> defines a <code>multi_iterator</code> $(i_1, i_2, \dots, i_k)$, such that	783
GiNaC::multi_iterator_permutation< T >	
The class <code>multi_iterator_permutation</code> defines a <code>multi_iterator</code> $(i_1, i_2, \dots, i_k)$, for which	786
GiNaC::multi_iterator_shuffle< T >	
The class <code>multi_iterator_shuffle</code> defines a <code>multi_iterator</code> , which runs over all shuffles of a and b	790
GiNaC::multi_iterator_shuffle_prime< T >	
The class <code>multi_iterator_shuffle_prime</code> defines a <code>multi_iterator</code> , which runs over all shuffles of a and b, excluding the first one (a,b)	793
GiNaC::multiple_polylog_kernel	
The integration kernel for multiple polylogarithms	796
GiNaC::ncmul	
Non-commutative product of expressions	803
GiNaC::normal_map_function	
Function object to be applied by <code>basic::normal()</code>	815
GiNaC::numeric	
This class is a wrapper around CLN-numbers within the <code>GiNaC</code> class hierarchy	816
GiNaC::op0_is_equal	842
GiNaC::partition_generator	
Generate all bounded combinatorial partitions of an integer n with exactly m parts (not including zero parts) in non-decreasing order	843

GiNaC::partition_with_zero_parts_generator	
Generate all bounded combinatorial partitions of an integer n with exactly m parts (including zero parts) in non-decreasing order	845
GiNaC::Gt_detail::pathintegral_term	846
GiNaC::Gt_detail::pathintegral_term_list	849
GiNaC::pointer_to_map_function	852
GiNaC::pointer_to_map_function_1arg< T1 >	853
GiNaC::pointer_to_map_function_2args< T1, T2 >	854
GiNaC::pointer_to_map_function_3args< T1, T2, T3 >	856
GiNaC::pointer_to_member_to_map_function< C >	858
GiNaC::pointer_to_member_to_map_function_1arg< C, T1 >	860
GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 >	861
GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >	863
GiNaC::pole_error	
Exception class thrown when a singularity is encountered	865
GiNaC::possymbol	
Specialization of symbol to real positive domain	867
GiNaC::power	
This class holds a two-component object, a basis and and exponent representing exponentiation	872
GiNaC::print_context	
Base class for print_contexts	887
GiNaC::print_context_options	
This class stores information about a registered print_context class	889
GiNaC::print_csrc	
Base context for C source output	891
GiNaC::print_csrc_cl_N	
Context for C source output using CLN numbers	892
GiNaC::print_csrc_double	
Context for C source output using double precision	893
GiNaC::print_csrc_float	
Context for C source output using float precision	894
GiNaC::print_dflt	
Context for default (ginsh-parsable) output	895
GiNaC::print_functor	
This class represents a print method for a certain algebraic class and print_context type	896
GiNaC::print_functor_impl	
Base class for print_functor handlers	898
GiNaC::print_latex	
Context for latex-parsable output	899
GiNaC::print_memfun_handler< T, C >	
Print_functor handler for member functions of class T, context type C	900
GiNaC::print_options	
Flags to control the behavior of a print_context	902
GiNaC::print_ptrfun_handler< T, C >	
Print_functor handler for pointer-to-functions of class T, context type C	903
GiNaC::print_python	
Context for python pretty-print output	905
GiNaC::print_python_repr	
Context for python-parsable output	906
GiNaC::print_tree	
Context for tree-like output for debugging	907
GiNaC::archive_node::property	
Archived property (data type, name and associated data)	908
GiNaC::archive_node::property_info	
Information about a stored property	909
GiNaC::pseries	
This class holds a extended truncated power series (positive and negative integer powers)	911

GiNaC::psi1_SERIAL	
Polylogarithm and multiple polylogarithm	926
GiNaC::psi2_SERIAL	
Derivatives of Psi-function (aka polygamma-functions)	927
GiNaC::ptr< T >	
Class of (intrusively) reference-counted pointers that support copy-on-write semantics	928
GiNaC::realsymbol	
Specialization of symbol to real domain	932
GiNaC::refcounted	
Base class for reference-counted objects	938
GiNaC::registered_class_options	
This class stores information about a registered GiNaC class	940
GiNaC::relational	
This class holds a relation consisting of two expressions and a logical relation between them	943
GiNaC::remember_strategies	
Strategies how to clean up the function remember cache	953
GiNaC::remember_table	
The remember table is organized like an n-fold associative cache in a microprocessor	954
GiNaC::remember_table_entry	
A single entry in the remember table of a function	957
GiNaC::remember_table_list	
A list of entries in the remember table having some least significant bits of the hashvalue in common	959
GiNaC::return_type_t	
To distinguish between different kinds of non-commutative objects	961
GiNaC::return_types	
.	962
GiNaC::relational::safe_bool_helper	
.	963
GiNaC::scalar_products	
Helper class for storing information about known scalar products which are to be automatically replaced by simplify_indexed()	963
GiNaC::series_options	
Flags to control series expansion	966
GiNaC::solve_algo	
Switch to control algorithm for linear system solving	966
GiNaC::spinidx	
This class holds a spinor index that can be dotted or undotted and that also has a variance	967
GiNaC::spinmetric	
This class represents an antisymmetric spinor metric tensor which can be used to raise/lower indices of 2-component Weyl spinors	977
GiNaC::spmapkey	
.	982
GiNaC::status_flags	
Flags to store information about the state of an object	984
GiNaC::structure< T, ComparisonPolicy >	
Wrapper template for making GiNaC classes out of C++ structures	985
GiNaC::su3d	
This class represents the tensor of symmetric su(3) structure constants	1002
GiNaC::su3f	
This class represents the tensor of antisymmetric su(3) structure constants	1007
GiNaC::su3one	
This class represents the su(3) unity element	1012
GiNaC::su3t	
This class represents an su(3) generator	1016
GiNaC::subs_options	
Flags to control the behavior of subs()	1021
GiNaC::sy_is_less	
.	1022
GiNaC::sy_swap	
.	1023

GiNaC::sym_desc	
This structure holds information about the highest and lowest degrees in which a symbol appears in two multivariate polynomials "a" and "b"	1024
GiNaC::symbol	
Basic CAS symbol	1026
GiNaC::symbolset	1037
GiNaC::symmetry	
This class describes the symmetry of a group of indices	1039
GiNaC::symminfo	
This structure stores the individual symmetrized terms obtained during the simplification of sums	1047
GiNaC::symminfo_is_less_by_orig	1049
GiNaC::symminfo_is_less_by_symmterm	1050
GiNaC::tensdelta	
This class represents the delta tensor	1050
GiNaC::tensepsilon	
This class represents the totally antisymmetric epsilon tensor	1055
GiNaC::tensmetric	
This class represents a general metric tensor which can be used to raise/lower indices	1062
GiNaC::tensor	
This class holds one of GiNaC 's predefined special tensors such as the delta and the metric tensors	1067
GiNaC::terminfo	
This structure stores the original and symmetrized versions of terms obtained during the simplification of sums	1071
GiNaC::terminfo_is_less	1073
GiNaC::Gt_detail::TransformExpressionWithCache< Type >	1073
GiNaC::class_info< OPT >::tree_node	1076
GiNaC::unarchive_table_t	1077
GiNaC::user_defined_kernel	
A user-defined integration kernel	1078
GiNaC::varidx	
This class holds an index with a variance (co- or contravariant)	1085
GiNaC::visitor	
Degenerate base class for visitors	1092
GiNaC::wildcard	
This class acts as a wildcard for subs() , match() , has() and find()	1093
GiNaC::zeta1_SERIAL	
Complex conjugate	1099
GiNaC::zeta2_SERIAL	
Alternating Euler sum or colored MZV	1100

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

add.cpp	Implementation of GiNaC 's sums of expressions	1101
add.h	Interface to GiNaC 's sums of expressions	1102
archive.cpp	Archiving of GiNaC expressions	1102
archive.h	Archiving of GiNaC expressions	1103
assertion.h	Assertion macro definition	1105
basic.cpp	Implementation of GiNaC 's ABC	1106
basic.h	Interface to GiNaC 's ABC	1107
class_info.h	Helper templates to provide per-class information for class hierarchies	1108
clifford.cpp	Implementation of GiNaC 's clifford algebra (Dirac gamma) objects	1109
clifford.h	Interface to GiNaC 's clifford algebra (Dirac gamma) objects	1111
color.cpp	Implementation of GiNaC 's color (SU(3) Lie algebra) objects	1113
color.h	Interface to GiNaC 's color (SU(3) Lie algebra) objects	1115
compiler.h	Definition of optimizing macros	1116
constant.cpp	Implementation of GiNaC 's constant types and some special constants	1117
constant.h	Interface to GiNaC 's constant types and some special constants	1117
container.h	Wrapper template for making GiNaC classes out of STL containers	1118
crc32.h	CRC32 hash function	1119
ex.cpp	Implementation of GiNaC 's light-weight expression handles	1119

ex.h	Interface to GiNaC 's light-weight expression handles	1120
excompiler.cpp	Functions to facilitate the conversion of a ex to a function pointer suited for fast numerical integration	1122
excompiler.h	Functions to facilitate the conversion of a ex to a function pointer suited for fast numerical integration	1123
expair.cpp	Implementation of expression pairs (building blocks of expairseq)	1124
expair.h	Definition of expression pairs (building blocks of expairseq)	1124
expairseq.cpp	Implementation of sequences of expression pairs	1125
expairseq.h	Interface to sequences of expression pairs	1126
exprseq.cpp	Implementation of GiNaC 's exprseq	1127
exprseq.h	Definition of GiNaC 's exprseq	1127
factor.cpp	Polynomial factorization (implementation)	1128
factor.h	Polynomial factorization	1136
fail.cpp	Implementation of class signaling failure of operation	1136
fail.h	Interface to class signaling failure of operation	1137
fderivative.cpp	Implementation of abstract derivatives of functions	1137
fderivative.h	Interface to abstract derivatives of functions	1138
flags.h	Collection of all flags used through the GiNaC framework	1138
function.cpp	Implementation of class of symbolic functions	1139
function.h	Interface to class of symbolic functions	1140
ginac.h	This include file includes all other public GiNaC headers	1151
Gt.cpp	Implementation of GiNaC 's elliptic multiple polylogarithms	1152
Gt.h	Interface to GiNaC 's elliptic multiple polylogarithms	1153
Gt_helpers.cpp	Implementation of helper functions for elliptic multiple polylogarithms	1153
Gt_helpers.h	Interface to helper functions for elliptic multiple polylogarithms	1154
hash_map.h	Replacement for <code>map<></code> using hash tables	1155
hash_seed.h	Type-specific hash seed	1155
idx.cpp	Implementation of GiNaC 's indices	1156
idx.h	Interface to GiNaC 's indices	1157
indexed.cpp	Implementation of GiNaC 's indexed expressions	1158

indexed.h	Interface to GiNaC's indexed expressions	1160
inifcns.cpp	Implementation of GiNaC's initially known functions	1161
inifcns.h	Interface to GiNaC's initially known functions	1164
inifcns_elliptic.cpp	Implementation of some special functions related to elliptic curves	1165
inifcns_gamma.cpp	Implementation of Gamma-function, Beta-function, Polygamma-functions, and some related stuff	1167
inifcns_nstdsums.cpp	Implementation of some special functions that have a representation as nested sums	1168
inifcns_trans.cpp	Implementation of transcendental (and trigonometric and hyperbolic) functions	1170
integral.cpp	Implementation of GiNaC's symbolic integral	1173
integral.h	Interface to GiNaC's symbolic integral	1174
integration_kernel.cpp	Implementation of GiNaC's integration kernels for iterated integrals	1174
integration_kernel.h	Interface to GiNaC's integration kernels for iterated integrals	1176
lst.cpp	Implementation of GiNaC's lst	1178
lst.h	Definition of GiNaC's lst	1178
matrix.cpp	Implementation of symbolic matrices	1179
matrix.h	Interface to symbolic matrices	1180
mul.cpp	Implementation of GiNaC's products of expressions	1181
mul.h	Interface to GiNaC's products of expressions	1182
ncmul.cpp	Implementation of GiNaC's non-commutative products of expressions	1183
ncmul.h	Interface to GiNaC's non-commutative products of expressions	1183
normal.cpp	This file implements several functions that work on univariate and multivariate polynomials and rational functions	1184
normal.h	This file defines several functions that work on univariate and multivariate polynomials and rational functions	1187
numeric.cpp	This file contains the interface to the underlying bignum package	1188
numeric.h	Makes the interface to the underlying bignum package available	1191
operators.cpp	Implementation of GiNaC's overloaded operators	1194
operators.h	Interface to GiNaC's overloaded operators	1196
power.cpp	Implementation of GiNaC's symbolic exponentiation (basis^{exponent})	1198
power.h	Interface to GiNaC's symbolic exponentiation (basis^{exponent})	1198
print.cpp	Implementation of helper classes for expression output	1199

print.h	Definition of helper classes for expression output	1200
pseries.cpp	Implementation of class for extended truncated power series and methods for series expansion	1202
pseries.h	Interface to class for extended truncated power series	1203
ptr.h	Reference-counted pointer template	1204
registrar.cpp	GiNaC's class registrar (for class basic and all classes derived from it)	1204
registrar.h	GiNaC's class registrar (for class basic and all classes derived from it)	1205
relational.cpp	Implementation of relations between expressions	1208
relational.h	Interface to relations between expressions	1208
remember.cpp	Implementation of helper classes for using the remember option in GiNaC functions	1209
remember.h	Interface to helper classes for using the remember option in GiNaC functions	1209
structure.h	Wrapper template for making GiNaC classes out of C++ structures	1210
symbol.cpp	Implementation of GiNaC's symbolic objects	1211
symbol.h	Interface to GiNaC's symbolic objects	1211
symmetry.cpp	Implementation of GiNaC's symmetry definitions	1212
symmetry.h	Interface to GiNaC's symmetry definitions	1213
tensor.cpp	Implementation of GiNaC's special tensors	1215
tensor.h	Interface to GiNaC's special tensors	1216
utils.cpp	Implementation of several small and furry utilities needed within GiNaC but not of any interest to the user of the library	1217
utils.h	Interface to several small and furry utilities needed within GiNaC but not of any interest to the user of the library	1219
utils_multi_iterator.h	Utilities for summing over multiple indices	1221
version.h	GiNaC library version information	1223
wildcard.cpp	Implementation of GiNaC's wildcard objects	1225
wildcard.h	Interface to GiNaC's wildcard objects	1225

Chapter 5

Namespace Documentation

5.1 GiNaC Namespace Reference

Namespaces

- namespace [Gt_detail](#)
- namespace [internal](#)

Classes

- class [_numeric_digits](#)
This class is used to instantiate a global singleton object Digits which behaves just like Maple's Digits.
- class [add](#)
Sum of expressions.
- class [archive](#)
*This class holds archived versions of [GiNaC](#) expressions (class *ex*).*
- class [archive_node](#)
*This class stores all properties needed to record/retrieve the state of one object of class *basic* (or a derived class).*
- class [basic](#)
This class is the ABC (abstract base class) of [GiNaC](#)'s class hierarchy.
- class [basic_log_kernel](#)
The basic integration kernel with a logarithmic singularity at the origin.
- class [basic_multi_iterator](#)
[basic_multi_iterator](#) is a base class.
- class [basic_partition_generator](#)
*Base class for generating all bounded combinatorial partitions of an integer *n* with exactly *m* parts in non-decreasing order.*
- class [class_info](#)
- class [clifford](#)
This class holds an object representing an element of the Clifford algebra (the Dirac gamma matrices).
- class [cliffordunit](#)
This class represents the Clifford algebra generators (units).
- class [color](#)
This class holds a generator T_a or the unity element of the Lie algebra of $SU(3)$, as used for calculations in quantum chromodynamics.
- class [compare_all_equal](#)

- Comparison policy: all structures of one type are equal.*

 - class [compare_bitwise](#)

Comparison policy: use bit-wise comparison to compare structures.
- class [compare_std_less](#)

Comparison policy: use `std::equal_to`/`std::less` (defaults to operators `==` and `<`) to compare structures.
- class [composition_generator](#)

Generate all compositions of a partition of an integer n , starting with the compositions which has non-decreasing order.
- class [const_iterator](#)
 - class [const_postorder_iterator](#)
 - class [const_preorder_iterator](#)
 - class [constant](#)

This class holds constants, symbols with specific numerical value.
- class [container](#)

Wrapper template for making [GiNaC](#) classes out of STL containers.
- class [container_storage](#)

Helper template for encapsulating the [reserve\(\)](#) mechanics of STL containers.
- struct [derivative_map_function](#)

Function object to be applied by [basic::derivative\(\)](#).
- class [determinant_algo](#)

Switch to control algorithm for determinant computation.
- class [diracgamma](#)

This class represents the Dirac gamma Lorentz vector.
- class [diracgamma5](#)

This class represents the Dirac gamma5 object which anticommutes with all other gammas.
- class [diracgammaL](#)

This class represents the Dirac gammaL object which behaves like $1/2 (1-\text{gamma}5)$.
- class [diracgammaR](#)

This class represents the Dirac gammaL object which behaves like $1/2 (1+\text{gamma}5)$.
- class [diracone](#)

This class represents the Clifford algebra unity element.
- class [do_taylor](#)

Exception class thrown by classes which provide their own series expansion to signal that ordinary Taylor expansion is safe.
- class [domain](#)

Domain of an object.
- class [dunno](#)

Exception class thrown by functions to signal unimplemented functionality so the expression may just be `.hold()`
- class [Ebar_kernel](#)

The Ebar-kernel.
- class [Eisenstein_h_kernel](#)

The kernel corresponding to the Eisenstein series $h_{k,N,r,s}(\tau)$.
- class [Eisenstein_kernel](#)

The kernel corresponding to the Eisenstein series $E_{k,N,a,b,K}(\tau)$.
- class [ELi_kernel](#)

The ELi-kernel.
- struct [error_and_integral](#)
 - struct [error_and_integral_is_less](#)
 - struct [eval_integ_map_function](#)

Function object to be applied by [basic::eval_integ\(\)](#).
- struct [evalf_map_function](#)

- Function object to be applied by `basic::evalf()`.
- struct `evalm_map_function`
 - Function object to be applied by `basic::evalm()`.
- class `ex`
 - Lightweight wrapper for GiNaC's symbolic objects.
- struct `ex_base_is_less`
- struct `ex_is_equal`
- struct `ex_is_less`
- struct `ex_swap`
- class `expair`
 - A pair of expressions.
- struct `expair_is_less`
 - Function object for insertion into third argument of STL's `sort()` etc.
- struct `expair_rest_is_less`
 - Function object not caring about the numerical coefficients for insertion into third argument of STL's `sort()`.
- struct `expair_swap`
- class `expairseq`
 - A sequence of class `expair`.
- struct `expand_map_function`
 - Function object to be applied by `basic::expand()`.
- class `expand_options`
 - Flags to control the behavior of `expand()`.
- class `factor_options`
 - Flags to control the polynomial factorization.
- class `fail`
- class `fderivative`
 - This class represents the (abstract) derivative of a symbolic function.
- class `function`
 - The class `function` is used to implement builtin functions like `sin`, `cos`... and user defined functions.
- class `function_options`
- class `G2_SERIAL`
 - Generalized multiple polylogarithm.
- class `G3_SERIAL`
 - Generalized multiple polylogarithm with explicit imaginary parts.
- struct `gcd_options`
 - Flags to control the behavior of `gcd()` and friends.
- class `gcdheu_failed`
 - Exception thrown by `heur_gcd()` to signal failure.
- class `Gt`
 - Elliptic Multiple Polylogarithm.
- class `has_distance`
 - SFINAE test for distance.
- class `has_options`
 - Flags to control the behavior of `has()`.
- class `idx`
 - This class holds one index of an indexed object.
- struct `idx_is_equal_ignore_dim`
- class `indexed`
 - This class holds an indexed expression.
- class `info_flags`
 - Possible attributes an object can have.

- class [integral](#)
Symbolic integral.
- class [integration_kernel](#)
The base class for integration kernels for iterated integrals.
- struct [is_not_a_clifford](#)
Predicate for finding non-clifford objects.
- struct [is_summation_idx](#)
- class [iterated_integral2_SERIAL](#)
Complete elliptic integral of the first kind.
- class [iterated_integral3_SERIAL](#)
Iterated integral with explicit truncation.
- class [Kronecker_dtau_kernel](#)
The kernel corresponding to integrating the Kronecker coefficient function $g^{(n)}(z_j, K\tau)$ in τ (or equivalently in $\bar{q}$).
- class [Kronecker_dz_kernel](#)
The kernel corresponding to integrating the Kronecker coefficient function $g^{(n-1)}(z - z_j, K\tau)$ in z .
- class [lanczos_coeffs](#)
- class [library_init](#)
Helper class to initialize the library.
- class [make_flat_inserter](#)
Class to handle the renaming of dummy indices.
- struct [map_function](#)
Function object for map().
- class [matrix](#)
Symbolic matrices.
- class [minkmetric](#)
This class represents a Minkowski metric tensor.
- class [modular_form_kernel](#)
A kernel corresponding to a polynomial in Eisenstein series.
- class [mul](#)
Product of expressions.
- class [multi_iterator_counter](#)
The class [multi_iterator_counter](#) defines a `multi_iterator` $(i_1, i_2, \dots, i_k)$, such that.
- class [multi_iterator_counter_indv](#)
The class [multi_iterator_counter_indv](#) defines a `multi_iterator` $(i_1, i_2, \dots, i_k)$, such that.
- class [multi_iterator_ordered](#)
The class [multi_iterator_ordered](#) defines a `multi_iterator` $(i_1, i_2, \dots, i_k)$, such that.
- class [multi_iterator_ordered_eq](#)
The class [multi_iterator_ordered_eq](#) defines a `multi_iterator` $(i_1, i_2, \dots, i_k)$, such that.
- class [multi_iterator_ordered_eq_indv](#)
The class [multi_iterator_ordered_eq_indv](#) defines a `multi_iterator` $(i_1, i_2, \dots, i_k)$, such that.
- class [multi_iterator_permutation](#)
The class [multi_iterator_permutation](#) defines a `multi_iterator` $(i_1, i_2, \dots, i_k)$, for which.
- class [multi_iterator_shuffle](#)
The class [multi_iterator_shuffle](#) defines a `multi_iterator`, which runs over all shuffles of a and b .
- class [multi_iterator_shuffle_prime](#)
The class [multi_iterator_shuffle_prime](#) defines a `multi_iterator`, which runs over all shuffles of a and b , excluding the first one (a,b) .
- class [multiple_polylog_kernel](#)
The integration kernel for multiple polylogarithms.
- class [ncmul](#)
Non-commutative product of expressions.

- struct [normal_map_function](#)
Function object to be applied by [basic::normal\(\)](#).
- class [numeric](#)
This class is a wrapper around CLN-numbers within the [GiNaC](#) class hierarchy.
- struct [op0_is_equal](#)
- class [partition_generator](#)
Generate all bounded combinatorial partitions of an integer n with exactly m parts (not including zero parts) in non-decreasing order.
- class [partition_with_zero_parts_generator](#)
Generate all bounded combinatorial partitions of an integer n with exactly m parts (including zero parts) in non-decreasing order.
- class [pointer_to_map_function](#)
- class [pointer_to_map_function_1arg](#)
- class [pointer_to_map_function_2args](#)
- class [pointer_to_map_function_3args](#)
- class [pointer_to_member_to_map_function](#)
- class [pointer_to_member_to_map_function_1arg](#)
- class [pointer_to_member_to_map_function_2args](#)
- class [pointer_to_member_to_map_function_3args](#)
- class [pole_error](#)
Exception class thrown when a singularity is encountered.
- class [possymbol](#)
Specialization of symbol to real positive domain.
- class [power](#)
This class holds a two-component object, a basis and an exponent representing exponentiation.
- class [print_context](#)
Base class for `print_contexts`.
- class [print_context_options](#)
This class stores information about a registered [print_context](#) class.
- class [print_csrc](#)
Base context for C source output.
- class [print_csrc_cl_N](#)
Context for C source output using CLN numbers.
- class [print_csrc_double](#)
Context for C source output using double precision.
- class [print_csrc_float](#)
Context for C source output using float precision.
- class [print_dflt](#)
Context for default (ginsh-parsable) output.
- class [print_functor](#)
This class represents a print method for a certain algebraic class and [print_context](#) type.
- class [print_functor_impl](#)
Base class for [print_functor](#) handlers.
- class [print_latex](#)
Context for latex-parsable output.
- class [print_memfun_handler](#)
[print_functor](#) handler for member functions of class T , context type C
- class [print_options](#)
Flags to control the behavior of a [print_context](#).
- class [print_ptrfun_handler](#)
[print_functor](#) handler for pointer-to-functions of class T , context type C
- class [print_python](#)

- Context for python pretty-print output.*

 - class [print_python_repr](#)
- Context for python-parsable output.*

 - class [print_tree](#)
- Context for tree-like output for debugging.*

 - class [pseries](#)
- This class holds a extended truncated power series (positive and negative integer powers).*

 - class [psi1_SERIAL](#)
- Polylogarithm and multiple polylogarithm.*

 - class [psi2_SERIAL](#)
- Derivatives of Psi-function (aka polygamma-functions).*

 - class [ptr](#)
- Class of (intrusively) reference-counted pointers that support copy-on-write semantics.*

 - class [realsymbol](#)
- Specialization of symbol to real domain.*

 - class [refcounted](#)
- Base class for reference-counted objects.*

 - class [registered_class_options](#)
- This class stores information about a registered [GiNaC](#) class.*

 - class [relational](#)
- This class holds a relation consisting of two expressions and a logical relation between them.*

 - class [remember_strategies](#)
- Strategies how to clean up the function remember cache.*

 - class [remember_table](#)
- The remember table is organized like an n-fold associative cache in a microprocessor.*

 - class [remember_table_entry](#)
- A single entry in the remember table of a function.*

 - class [remember_table_list](#)
- A list of entries in the remember table having some least significant bits of the hashvalue in common.*

 - struct [return_type_t](#)
- To distinguish between different kinds of non-commutative objects.*

 - class [return_types](#)
- class [scalar_products](#)

Helper class for storing information about known scalar products which are to be automatically replaced by [simplify_indexed\(\)](#).
- class [series_options](#)

Flags to control series expansion.
- class [solve_algo](#)

Switch to control algorithm for linear system solving.
- class [spinidx](#)

This class holds a spinor index that can be dotted or undotted and that also has a variance.
- class [spinmetric](#)

This class represents an antisymmetric spinor metric tensor which can be used to raise/lower indices of 2-component Weyl spinors.
- class [smapkey](#)
- class [status_flags](#)

Flags to store information about the state of an object.
- class [structure](#)

Wrapper template for making [GiNaC](#) classes out of C++ structures.
- class [su3d](#)

This class represents the tensor of symmetric su(3) structure constants.

- class [su3f](#)
This class represents the tensor of antisymmetric $su(3)$ structure constants.
- class [su3one](#)
This class represents the $su(3)$ unity element.
- class [su3t](#)
This class represents an $su(3)$ generator.
- class [subs_options](#)
Flags to control the behavior of [subs\(\)](#).
- class [sy_is_less](#)
- class [sy_swap](#)
- struct [sym_desc](#)
This structure holds information about the highest and lowest degrees in which a symbol appears in two multivariate polynomials "a" and "b".
- class [symbol](#)
Basic CAS symbol.
- class [symbolset](#)
- class [symmetry](#)
This class describes the symmetry of a group of indices.
- class [symminfo](#)
This structure stores the individual symmetrized terms obtained during the simplification of sums.
- class [symminfo_is_less_by_orig](#)
- class [symminfo_is_less_by_symmterm](#)
- class [tensdelta](#)
This class represents the delta tensor.
- class [tensepsilon](#)
This class represents the totally antisymmetric epsilon tensor.
- class [tensmetric](#)
This class represents a general metric tensor which can be used to raise/lower indices.
- class [tensor](#)
This class holds one of [GiNaC](#)'s predefined special tensors such as the delta and the metric tensors.
- class [terminfo](#)
This structure stores the original and symmetrized versions of terms obtained during the simplification of sums.
- class [terminfo_is_less](#)
- class [unarchive_table_t](#)
- class [user_defined_kernel](#)
A user-defined integration kernel.
- class [varidx](#)
This class holds an index with a variance (co- or contravariant).
- class [visitor](#)
Degenerate base class for visitors.
- class [wildcard](#)
This class acts as a wildcard for [subs\(\)](#), [match\(\)](#), [has\(\)](#) and [find\(\)](#).
- class [zeta1_SERIAL](#)
Complex conjugate.
- class [zeta2_SERIAL](#)
Alternating Euler sum or colored MZV.

Typedefs

- typedef unsigned [archive_node_id](#)
Numerical ID value to refer to an [archive_node](#).
- typedef unsigned [archive_atom](#)
Numerical ID value to refer to a string.
- typedef [basic](#) [*\(* synthesize_func\)](#) ()
- typedef std::map< std::string, [synthesize_func](#) > [unarchive_map_t](#)
- typedef std::vector< [ex](#) > [exvector](#)
- typedef std::set< [ex](#), [ex_is_less](#) > [exset](#)
- typedef std::map< [ex](#), [ex](#), [ex_is_less](#) > [exmap](#)
- typedef [ex](#)[* evalfunctype](#) ()
- typedef double([* FUNCP_1P](#)) (double)
Function pointer with one function parameter.
- typedef double([* FUNCP_2P](#)) (double, double)
Function pointer with two function parameters.
- typedef void([* FUNCP_CUBA](#)) (const int *, const double[], const int *, double[])
Function pointer for use with the CUBA library (<http://www.feynarts.de/cuba>).
- typedef std::vector< [expair](#) > [epvector](#)
expair-vector
- typedef [epvector](#)::iterator [epp](#)
expair-vector pointer
- typedef [container](#)< std::vector > [exprseq](#)
- typedef std::multiset< unsigned > [paramset](#)
- typedef [ex](#)[* eval_funcp](#) ()
- typedef [ex](#)[* evalf_funcp](#) ()
- typedef [ex](#)[* conjugate_funcp](#) ()
- typedef [ex](#)[* real_part_funcp](#) ()
- typedef [ex](#)[* imag_part_funcp](#) ()
- typedef [ex](#)[* expand_funcp](#) ()
- typedef [ex](#)[* derivative_funcp](#) ()
- typedef [ex](#)[* expl_derivative_funcp](#) ()
- typedef [ex](#)[* power_funcp](#) ()
- typedef [ex](#)[* series_funcp](#) ()
- typedef void([* print_funcp](#)) ()
- typedef bool([* info_funcp](#)) ()
- typedef [ex](#)[* eval_funcp_1](#) (const [ex](#) &)
- typedef [ex](#)[* evalf_funcp_1](#) (const [ex](#) &)
- typedef [ex](#)[* conjugate_funcp_1](#) (const [ex](#) &)
- typedef [ex](#)[* real_part_funcp_1](#) (const [ex](#) &)
- typedef [ex](#)[* imag_part_funcp_1](#) (const [ex](#) &)
- typedef [ex](#)[* expand_funcp_1](#) (const [ex](#) &, unsigned)
- typedef [ex](#)[* derivative_funcp_1](#) (const [ex](#) &, unsigned)
- typedef [ex](#)[* expl_derivative_funcp_1](#) (const [ex](#) &, const [symbol](#) &)
- typedef [ex](#)[* power_funcp_1](#) (const [ex](#) &, const [ex](#) &)
- typedef [ex](#)[* series_funcp_1](#) (const [ex](#) &, const [relational](#) &, int, unsigned)
- typedef void([* print_funcp_1](#)) (const [ex](#) &, const [print_context](#) &)
- typedef bool([* info_funcp_1](#)) (const [ex](#) &, unsigned)
- typedef [ex](#)[* eval_funcp_2](#) (const [ex](#) &, const [ex](#) &)
- typedef [ex](#)[* evalf_funcp_2](#) (const [ex](#) &, const [ex](#) &)
- typedef [ex](#)[* conjugate_funcp_2](#) (const [ex](#) &, const [ex](#) &)
- typedef [ex](#)[* real_part_funcp_2](#) (const [ex](#) &, const [ex](#) &)
- typedef [ex](#)[* imag_part_funcp_2](#) (const [ex](#) &, const [ex](#) &)
- typedef [ex](#)[* expand_funcp_2](#) (const [ex](#) &, const [ex](#) &, unsigned)

- [illegible]

- [illegible]

- typedef [ex](#)(* [power_funcp_13](#)) (const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &)
- typedef [ex](#)(* [series_funcp_13](#)) (const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [relational](#) &, int, unsigned)
- typedef void(* [print_funcp_13](#)) (const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [print_context](#) &)
- typedef bool(* [info_funcp_13](#)) (const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, unsigned)
- typedef [ex](#)(* [eval_funcp_14](#)) (const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &)
- typedef [ex](#)(* [evalf_funcp_14](#)) (const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &)
- typedef [ex](#)(* [conjugate_funcp_14](#)) (const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &)
- typedef [ex](#)(* [real_part_funcp_14](#)) (const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &)
- typedef [ex](#)(* [imag_part_funcp_14](#)) (const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &)
- typedef [ex](#)(* [expand_funcp_14](#)) (const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, unsigned)
- typedef [ex](#)(* [derivative_funcp_14](#)) (const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, unsigned)
- typedef [ex](#)(* [expl_derivative_funcp_14](#)) (const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [symbol](#) &)
- typedef [ex](#)(* [power_funcp_14](#)) (const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &)
- typedef [ex](#)(* [series_funcp_14](#)) (const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [relational](#) &, int, unsigned)
- typedef void(* [print_funcp_14](#)) (const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [print_context](#) &)
- typedef bool(* [info_funcp_14](#)) (const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, const [ex](#) &, unsigned)
- typedef [ex](#)(* [eval_funcp_exvector](#)) (const [exvector](#) &)
- typedef [ex](#)(* [evalf_funcp_exvector](#)) (const [exvector](#) &)
- typedef [ex](#)(* [conjugate_funcp_exvector](#)) (const [exvector](#) &)
- typedef [ex](#)(* [real_part_funcp_exvector](#)) (const [exvector](#) &)
- typedef [ex](#)(* [imag_part_funcp_exvector](#)) (const [exvector](#) &)
- typedef [ex](#)(* [expand_funcp_exvector](#)) (const [exvector](#) &, unsigned)
- typedef [ex](#)(* [derivative_funcp_exvector](#)) (const [exvector](#) &, unsigned)
- typedef [ex](#)(* [expl_derivative_funcp_exvector](#)) (const [exvector](#) &, const [symbol](#) &)
- typedef [ex](#)(* [power_funcp_exvector](#)) (const [exvector](#) &, const [ex](#) &)
- typedef [ex](#)(* [series_funcp_exvector](#)) (const [exvector](#) &, const [relational](#) &, int, unsigned)
- typedef void(* [print_funcp_exvector](#)) (const [exvector](#) &, const [print_context](#) &)
- typedef bool(* [info_funcp_exvector](#)) (const [exvector](#) &, unsigned)
- template<typename T, class Hash = std::hash<[ex](#)>, class KeyEqual = std::equal_to<[ex](#)>, class Allocator = std::allocator<std::pair<const [ex](#), T>>>
 using [exhashmap](#) = std::unordered_map< [ex](#), T, Hash, KeyEqual, Allocator >
- typedef std::map< [spmapkey](#), [ex](#) > [spmap](#)
- typedef map< [error_and_integral](#), [ex](#), [error_and_integral_is_less](#) > [lookup_map](#)
- typedef [container](#)< std::list > [lst](#)
- typedef std::vector< std::size_t > [uintvector](#)
- typedef std::vector< unsigned > [unsignedvector](#)
- typedef std::vector< [exvector](#) > [exvectorvector](#)

- typedef std::vector< [sym_desc](#) > [sym_desc_vec](#)
- typedef void(* [digits_changed_callback](#)) (long)
Function pointer to implement callbacks in the case 'Digits' gets changed.
- typedef [class_info](#)< [print_context_options](#) > [print_context_class_info](#)
- typedef [class_info](#)< [registered_class_options](#) > [registered_class_info](#)

Enumerations

- enum { [callback_registered](#) = 1 }

Functions

- [GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([add](#), [expairseq](#), [print_func](#)< [print_context](#) >(&[add::do_print](#)), [print_func](#)< [print_latex](#) >(&[add::do_print_latex](#)), [print_func](#)< [print_csrc](#) >(&[add::do_print_csrc](#)), [print_func](#)< [print_tree](#) >(&[add::do_print_tree](#)), [print_func](#)< [print_python_repr](#) >(&[add::do_print_python_repr](#)))
[add](#)
- [GINAC_BIND_UNARCHIVER](#) ([add](#))
- [GINAC_DECLARE_UNARCHIVER](#) ([add](#))
- static void [write_unsigned](#) (std::ostream &os, unsigned val)
Write unsigned integer quantity to stream.
- static unsigned [read_unsigned](#) (std::istream &is)
Read unsigned integer quantity from stream.
- std::ostream & [operator<<](#) (std::ostream &os, const [archive_node](#) &n)
Write [archive_node](#) to binary data stream.
- std::ostream & [operator<<](#) (std::ostream &os, const [archive](#) &r)
Write archive to binary data stream.
- std::istream & [operator>>](#) (std::istream &is, [archive_node](#) &n)
Read [archive_node](#) from binary data stream.
- std::istream & [operator>>](#) (std::istream &is, [archive](#) &r)
Read archive from binary data stream.
- static [synthesize_func](#) [find_factory_fcn](#) (const std::string &name)
- [GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([basic](#), void, [print_func](#)< [print_context](#) >(&[basic::do_print](#)), [print_func](#)< [print_tree](#) >(&[basic::do_print_tree](#)), [print_func](#)< [print_python_repr](#) >(&[basic::do_print_python_repr](#)))
[basic](#)
basic copy constructor: implicitly assumes that the other class is of the exact same type (as it's used by duplicate()), so it can copy the tinfo_key and the hash value.
- template<class T >
bool [is_a](#) (const [basic](#) &obj)
Check if obj is a T, including base classes.
- template<class T >
bool [is_exactly_a](#) (const [basic](#) &obj)
Check if obj is a T, not including base classes.
- template<class B , typename... Args>
B & [dynallocate](#) (Args &&... args)
Constructs a new (class basic or derived) B object on the heap.
- template<class B >
B & [dynallocate](#) (std::initializer_list< [ex](#) > il)
Constructs a new (class basic or derived) B object on the heap.
- [GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([clifford](#), [indexed](#), [print_func](#)< [print_dflt](#) >(&[clifford::do_print_dflt](#)), [print_func](#)< [print_latex](#) >(&[clifford::do_print_latex](#)), [print_func](#)< [print_tree](#) >(&[clifford::do_print_tree](#)))
[GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#)([diracone](#)
- [print_func](#)< [print_dflt](#) > (&[diracone::do_print](#)), [print_func](#)< [print_latex](#) >(&[diracone](#)

- [GINAC_BIND_UNARCHIVER](#) ([clifford](#))
- [GINAC_BIND_UNARCHIVER](#) ([cliffordunit](#))
- [GINAC_BIND_UNARCHIVER](#) ([diracone](#))
- [GINAC_BIND_UNARCHIVER](#) ([diracgamma](#))
- [GINAC_BIND_UNARCHIVER](#) ([diracgamma5](#))
- [GINAC_BIND_UNARCHIVER](#) ([diracgammaL](#))
- [GINAC_BIND_UNARCHIVER](#) ([diracgammaR](#))
- static bool [is_dirac_slash](#) (const [ex](#) &seq0)
- static void [base_and_index](#) (const [ex](#) &c, [ex](#) &b, [ex](#) &i)
This function decomposes $\gamma \sim \mu \rightarrow (1, \mu)$ and $a \rightarrow (a.ix, ix)$
- [ex](#) [dirac_ONE](#) (unsigned char rl=0)
Create a Clifford unity object.
- static unsigned [get_dim_uint](#) (const [ex](#) &e)
- [ex](#) [clifford_unit](#) (const [ex](#) &mu, const [ex](#) &metr, unsigned char rl=0)
Create a Clifford unit object.
- [ex](#) [dirac_gamma](#) (const [ex](#) &mu, unsigned char rl=0)
Create a Dirac gamma object.
- [ex](#) [dirac_gamma5](#) (unsigned char rl=0)
Create a Dirac gamma5 object.
- [ex](#) [dirac_gammaL](#) (unsigned char rl=0)
Create a Dirac gammaL object.
- [ex](#) [dirac_gammaR](#) (unsigned char rl=0)
Create a Dirac gammaR object.
- [ex](#) [dirac_slash](#) (const [ex](#) &e, const [ex](#) &dim, unsigned char rl=0)
Create a term of the form $e_\mu * \gamma \sim \mu$ with a unique index μ .
- static unsigned char [get_representation_label](#) (const [return_type_t](#) &ti)
Extract representation label from tinfo key (as returned by [return_type_tinfo\(\)](#)).
- static [ex](#) [trace_string](#) (exvector::const_iterator ix, size_t num)
Take trace of a string of an even number of Dirac gammas given a vector of indices.
- [ex](#) [dirac_trace](#) (const [ex](#) &e, const std::set< unsigned char > &rls, const [ex](#) &trONE=4)
Calculate dirac traces over the specified set of representation labels.
- [ex](#) [dirac_trace](#) (const [ex](#) &e, const [lst](#) &rls, const [ex](#) &trONE=4)
Calculate dirac traces over the specified list of representation labels.
- [ex](#) [dirac_trace](#) (const [ex](#) &e, unsigned char rl=0, const [ex](#) &trONE=4)
Calculate the trace of an expression containing gamma objects with a specified representation label.
- [ex](#) [canonicalize_clifford](#) (const [ex](#) &e)
Bring all products of clifford objects in an expression into a canonical order.
- [ex](#) [clifford_star_bar](#) (const [ex](#) &e, bool do_bar, unsigned [options](#))
An auxillary function performing [clifford_star\(\)](#) and [clifford_bar\(\)](#).
- [ex](#) [clifford_prime](#) (const [ex](#) &e)
Automorphism of the Clifford algebra, simply changes signs of all clifford units.
- [ex](#) [remove_dirac_ONE](#) (const [ex](#) &e, unsigned char rl=0, unsigned [options](#)=0)
Replaces [dirac_ONE](#)'s (with a [representation_label](#) no less than rl) in e with 1.
- int [clifford_max_label](#) (const [ex](#) &e, bool ignore_ONE=false)
Returns the maximal representation label of a clifford object if e contains at least one, otherwise returns -1.
- [ex](#) [clifford_norm](#) (const [ex](#) &e)
Calculation of the norm in the Clifford algebra.
- [ex](#) [clifford_inverse](#) (const [ex](#) &e)
Calculation of the inverse in the Clifford algebra.
- [ex](#) [lst_to_clifford](#) (const [ex](#) &v, const [ex](#) &mu, const [ex](#) &metr, unsigned char rl=0)
List or vector conversion into the Clifford vector.

- `ex lst_to_clifford` (const `ex` &v, const `ex` &e)
List or vector conversion into the Clifford vector.
- static `ex get_clifford_comp` (const `ex` &e, const `ex` &c, bool root=true)
Auxiliary structure to define a function for stripping one Clifford unit from vectors.
- `lst clifford_to_lst` (const `ex` &e, const `ex` &c, bool algebraic=true)
An inverse function to `lst_to_clifford()`.
- `ex clifford_moebius_map` (const `ex` &a, const `ex` &b, const `ex` &c, const `ex` &d, const `ex` &v, const `ex` &G, unsigned char rl=0)
Calculations of Moebius transformations (conformal map) defined by a 2x2 Clifford matrix (a b|c d) in linear spaces with arbitrary signature.
- `ex clifford_moebius_map` (const `ex` &M, const `ex` &v, const `ex` &G, unsigned char rl=0)
The second form of Moebius transformations defined by a 2x2 Clifford matrix M This function takes the transformation matrix M as a single entity.
- `GINAC_DECLARE_UNARCHIVER` (clifford)
- `GINAC_DECLARE_UNARCHIVER` (diracone)
- `GINAC_DECLARE_UNARCHIVER` (cliffordunit)
- `GINAC_DECLARE_UNARCHIVER` (diracgamma)
- `GINAC_DECLARE_UNARCHIVER` (diracgamma5)
- `GINAC_DECLARE_UNARCHIVER` (diracgammaL)
- `GINAC_DECLARE_UNARCHIVER` (diracgammaR)
- bool `is_clifford_tinfo` (const `return_type_t` &ti)
Check whether a given `return_type_t` object (as returned by `return_type_tinfo()`) is that of a clifford object (with an arbitrary representation label).
- `ex clifford_bar` (const `ex` &e)
Main anti-automorphism of the Clifford algebra: makes reversion and changes signs of all clifford units.
- `ex clifford_star` (const `ex` &e)
Reversion of the Clifford algebra, reverse the order of all clifford units in ncmul.
- `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (su3one, tensor, print_func< `print_dflt` >(&su3one::do_print). print_func< `print_latex` >(&su3one::do_print_latex)) `GINAC_IMPLEMENT_REGISTERED_CLASS_↵`
`OPT(su3t`
- `print_func< print_dflt > (&su3t::do_print). print_func< print_latex >(&su3t`
- `GINAC_BIND_UNARCHIVER` (color)
- `GINAC_BIND_UNARCHIVER` (su3one)
- `GINAC_BIND_UNARCHIVER` (su3t)
- `GINAC_BIND_UNARCHIVER` (su3f)
- `GINAC_BIND_UNARCHIVER` (su3d)
- static `ex permute_free_index_to_front` (const `exvector` &iv3, const `exvector` &iv2, int &sig)
Given a vector iv3 of three indices and a vector iv2 of two indices that is a subset of iv3, return the (free) index that is in iv3 but not in iv2 and the sign introduced by permuting that index to the front.
- `ex color_ONE` (unsigned char rl=0)
Create the su(3) unity element.
- `ex color_T` (const `ex` &a, unsigned char rl=0)
Create an su(3) generator.
- `ex color_f` (const `ex` &a, const `ex` &b, const `ex` &c)
Create an su(3) antisymmetric structure constant.
- `ex color_d` (const `ex` &a, const `ex` &b, const `ex` &c)
Create an su(3) symmetric structure constant.
- `ex color_h` (const `ex` &a, const `ex` &b, const `ex` &c)
*This returns the linear combination d.a.b.c+l*f.a.b.c.*
- static bool `is_color_tinfo` (const `return_type_t` &ti)
Check whether a given tinfo key (as returned by `return_type_tinfo()`) is that of a color object (with an arbitrary representation label).
- static unsigned char `get_representation_label` (const `return_type_t` &ti)

- Extract representation label from tinfo key (as returned by return_type_tinfo()).*
- `ex color_trace` (const `ex` &`e`, const `std::set< unsigned char >` &`rls`)
- Calculate color traces over the specified set of representation labels.*
- `ex color_trace` (const `ex` &`e`, const `lst` &`rl`)
- Calculate color traces over the specified list of representation labels.*
- `ex color_trace` (const `ex` &`e`, unsigned char `rl=0`)
- Calculate the trace of an expression containing color objects with a specified representation label.*
- `GINAC_DECLARE_UNARCHIVER` (`color`)
- `GINAC_DECLARE_UNARCHIVER` (`su3one`)
- `GINAC_DECLARE_UNARCHIVER` (`su3t`)
- `GINAC_DECLARE_UNARCHIVER` (`su3f`)
- `GINAC_DECLARE_UNARCHIVER` (`su3d`)
- `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`constant`, `basic`, `print_func< print_context >` (&`constant::do_print`), `print_func< print_latex >` (&`constant::do_print_latex`), `print_func< print_tree >` (&`constant::do_print_tree`), `print_func< print_python_repr >` (&`constant::do_print_python_repr`)) const ant
- `GINAC_BIND_UNARCHIVER` (`constant`)
- `GINAC_DECLARE_UNARCHIVER` (`constant`)
- static unsigned `crc32` (const char *`c`, const unsigned `len`, const unsigned `crcinit`)
- bool `are_ex_trivially_equal` (const `ex` &`e1`, const `ex` &`e2`)
- Compare two objects of class quickly without doing a deep tree traversal.*
- `std::ostream & operator<<` (`std::ostream` &`os`, const `exvector` &`e`)
- `std::ostream & operator<<` (`std::ostream` &`os`, const `exset` &`e`)
- `std::ostream & operator<<` (`std::ostream` &`os`, const `exmap` &`e`)
- `size_t nops` (const `ex` &`thisex`)
- `ex expand` (const `ex` &`thisex`, unsigned `options=0`)
- `ex conjugate` (const `ex` &`thisex`)
- `ex real_part` (const `ex` &`thisex`)
- `ex imag_part` (const `ex` &`thisex`)
- bool `has` (const `ex` &`thisex`, const `ex` &`pattern`, unsigned `options=0`)
- bool `find` (const `ex` &`thisex`, const `ex` &`pattern`, `exset` &`found`)
- bool `is_polynomial` (const `ex` &`thisex`, const `ex` &`vars`)
- int `degree` (const `ex` &`thisex`, const `ex` &`s`)
- int `ldegree` (const `ex` &`thisex`, const `ex` &`s`)
- `ex coeff` (const `ex` &`thisex`, const `ex` &`s`, int `n=1`)
- `ex numer` (const `ex` &`thisex`)
- `ex denom` (const `ex` &`thisex`)
- `ex numer_denom` (const `ex` &`thisex`)
- `ex normal` (const `ex` &`thisex`)
- `ex to_rational` (const `ex` &`thisex`, `exmap` &`repl`)
- `ex to_polynomial` (const `ex` &`thisex`, `exmap` &`repl`)
- `ex collect` (const `ex` &`thisex`, const `ex` &`s`, bool `distributed=false`)
- `ex eval` (const `ex` &`thisex`)
- `ex evalf` (const `ex` &`thisex`)
- `ex evalm` (const `ex` &`thisex`)
- `ex eval_integ` (const `ex` &`thisex`)
- `ex diff` (const `ex` &`thisex`, const `symbol` &`s`, unsigned `nth=1`)
- `ex series` (const `ex` &`thisex`, const `ex` &`r`, int `order`, unsigned `options=0`)
- bool `match` (const `ex` &`thisex`, const `ex` &`pattern`, `exmap` &`repl_lst`)
- `ex simplify_indexed` (const `ex` &`thisex`, unsigned `options=0`)
- `ex simplify_indexed` (const `ex` &`thisex`, const `scalar_products` &`sp`, unsigned `options=0`)
- `ex symmetrize` (const `ex` &`thisex`)
- `ex symmetrize` (const `ex` &`thisex`, const `lst` &`l`)
- `ex antisymmetrize` (const `ex` &`thisex`)
- `ex antisymmetrize` (const `ex` &`thisex`, const `lst` &`l`)

- [ex symmetrize_cyclic](#) (const [ex](#) &thisex)
- [ex symmetrize_cyclic](#) (const [ex](#) &thisex, const [lst](#) &l)
- [ex op](#) (const [ex](#) &thisex, size_t i)
- [ex lhs](#) (const [ex](#) &thisex)
- [ex rhs](#) (const [ex](#) &thisex)
- bool [is_zero](#) (const [ex](#) &thisex)
- void [swap](#) ([ex](#) &e1, [ex](#) &e2)
- [ex subs](#) (const [ex](#) &thisex, const [exmap](#) &m, unsigned [options](#)=0)
- [ex subs](#) (const [ex](#) &thisex, const [lst](#) &ls, const [lst](#) &lr, unsigned [options](#)=0)
- [ex subs](#) (const [ex](#) &thisex, const [ex](#) &e, unsigned [options](#)=0)
- template<class T >
bool [is_a](#) (const [ex](#) &obj)
Check if ex is a handle to a T, including base classes.
- template<class T >
bool [is_exactly_a](#) (const [ex](#) &obj)
Check if ex is a handle to a T, not including base classes.
- template<class T >
[attribute_pure](#) const T & [ex_to](#) (const [ex](#) &e)
Return a reference to the basic-derived class T object embedded in an expression.
- void [compile_ex](#) (const [ex](#) &expr, const [symbol](#) &sym, [FUNCP_1P](#) &fp, const std::string filename="")
Takes an expression and produces a function pointer to the compiled and linked C code equivalent in double precision.
- void [compile_ex](#) (const [ex](#) &expr, const [symbol](#) &sym1, const [symbol](#) &sym2, [FUNCP_2P](#) &fp, const std::string filename="")
Takes an expression and produces a function pointer to the compiled and linked C code equivalent in double precision.
- void [compile_ex](#) (const [lst](#) &exprs, const [lst](#) &syms, [FUNCP_CUBA](#) &fp, const std::string filename="")
Takes an expression and produces a function pointer to the compiled and linked C code equivalent in double precision.
- void [link_ex](#) (const std::string filename, [FUNCP_1P](#) &fp)
Opens an existing so-file and returns a function pointer of type FUNCP_1P to the contained function.
- void [link_ex](#) (const std::string filename, [FUNCP_2P](#) &fp)
Opens an existing so-file and returns a function pointer of type FUNCP_2P to the contained function.
- void [link_ex](#) (const std::string filename, [FUNCP_CUBA](#) &fp)
Opens an existing so-file and returns a function pointer of type FUNCP_CUBA to the contained function.
- void [unlink_ex](#) (const std::string filename)
Closes all linked .so files that have the supplied filename.
- void [swap](#) ([expair](#) &e1, [expair](#) &e2)
- [GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([expairseq](#), [basic](#), print_func< [print_context](#) >(&[expairseq](#)::do_print). print_func< [print_tree](#) >(&[expairseq](#)::do_print_tree)) class [exp_is_less](#)
- [epvector](#) * [conjugateepvector](#) (const [epvector](#) &)
Complex conjugate every element of an epvector.
- [ex factor](#) (const [ex](#) &poly, unsigned [options](#))
Interface function to the outside world.
- [GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([fail](#), [basic](#), print_func< [print_context](#) >(&[fail](#)::do_print). print_func< [print_tree](#) >(&[fail](#)::do_print_tree)) [GINAC_BIND_UNARCHIVER](#)([fail](#))
- [GINAC_DECLARE_UNARCHIVER](#) ([fail](#))
- [GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([fderivative](#), [function](#), print_func< [print_context](#) >(&[fderivative](#)::do_print). print_func< [print_latex](#) >(&[fderivative](#)::do_print_latex). print_func< [print_csrc](#) >(&[fderivative](#)::do_print_csrc). print_func< [print_tree](#) >(&[fderivative](#)::do_print_tree)) [fderivative](#)
- [GINAC_BIND_UNARCHIVER](#) ([fderivative](#))
- [GINAC_DECLARE_UNARCHIVER](#) ([fderivative](#))
- [GINAC_BIND_UNARCHIVER](#) ([function](#))
- [GINAC_DECLARE_UNARCHIVER](#) ([function](#))
- template<typename T >
bool [is_the_function](#) (const [ex](#) &x)

- `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`Gt`, `basic`, `print_func`< `print_dflt` >(&`Gt::do_print`), `print_func`< `print_python` >(&`Gt::do_print`), `print_func`< `print_latex` >(&`Gt::do_print_latex`)) `Gt`
- `GINAC_BIND_UNARCHIVER` (`Gt`)
- `GINAC_DECLARE_UNARCHIVER` (`Gt`)
- static unsigned `make_hash_seed` (const std::type_info &info)

We need a hash function which gives different values for objects of different types.
- `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`idx`, `basic`, `print_func`< `print_context` >(&`idx::do_print`), `print_func`< `print_latex` >(&`idx::do_print_latex`), `print_func`< `print_csrc` >(&`idx::do_print_csrc`), `print_func`< `print_tree` >(&`idx::do_print_tree`)) `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT`(`varidx`, `print_func`< `print_context` > (&`varidx::do_print`), `print_func`< `print_latex` >(&`varidx`
- `GINAC_BIND_UNARCHIVER` (`idx`)
- `GINAC_BIND_UNARCHIVER` (`varidx`)
- `GINAC_BIND_UNARCHIVER` (`spinidx`)
- bool `is_dummy_pair` (const `idx` &i1, const `idx` &i2)

Check whether two indices form a dummy pair.
- bool `is_dummy_pair` (const `ex` &e1, const `ex` &e2)

Check whether two expressions form a dummy index pair.
- void `find_free_and_dummy` (exvector::const_iterator it, exvector::const_iterator itend, `exvector` &out_free, `exvector` &out_dummy)

Given a vector of indices, split them into two vectors, one containing the free indices, the other containing the dummy indices (numeric indices are neither free nor dummy ones).
- `ex` `minimal_dim` (const `ex` &dim1, const `ex` &dim2)

Return the minimum of two index dimensions.
- `GINAC_DECLARE_UNARCHIVER` (`idx`)
- `GINAC_DECLARE_UNARCHIVER` (`varidx`)
- `GINAC_DECLARE_UNARCHIVER` (`spinidx`)
- void `find_free_and_dummy` (const `exvector` &v, `exvector` &out_free, `exvector` &out_dummy)

Given a vector of indices, split them into two vectors, one containing the free indices, the other containing the dummy indices (numeric indices are neither free nor dummy ones).
- void `find_dummy_indices` (const `exvector` &v, `exvector` &out_dummy)

Given a vector of indices, find the dummy indices.
- size_t `count_dummy_indices` (const `exvector` &v)

Count the number of dummy index pairs in an index vector.
- size_t `count_free_indices` (const `exvector` &v)

Count the number of dummy index pairs in an index vector.
- `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`indexed`, `exprseq`, `print_func`< `print_context` >(&`indexed::do_print`), `print_func`< `print_latex` >(&`indexed::do_print_latex`), `print_func`< `print_tree` >(&`indexed::do_print_tree`)) `indexed`
- `GINAC_BIND_UNARCHIVER` (`indexed`)
- static bool `indices_consistent` (const `exvector` &v1, const `exvector` &v2)

Check whether two sorted index vectors are consistent (i.e.
- template<class T>

size_t `number_of_type` (const `exvector` &v)
- template<class T>

static `ex` `rename_dummy_indices` (const `ex` &e, `exvector` &global_dummy_indices, `exvector` &local_dummy_↵
_indices)

Rename dummy indices in an expression.
- static void `find_variant_indices` (const `exvector` &v, `exvector` &variant_indices)

Given a set of indices, extract those of class varidx.
- bool `reposition_dummy_indices` (`ex` &e, `exvector` &variant_dummy_indices, `exvector` &moved_indices)

Raise/lower dummy indices in a single indexed objects to canonicalize their variance.
- static void `product_to_exvector` (const `ex` &e, `exvector` &v, bool &non_commutative)
- template<class T>

`ex` `idx_symmetrization` (const `ex` &r, const `exvector` &local_dummy_indices)

- [ex simplify_indexed](#) (const [ex](#) &e, [exvector](#) &free_indices, [exvector](#) &dummy_indices, const [scalar_products](#) &sp)
Simplify indexed expression, return list of free indices.
- [ex simplify_indexed_product](#) (const [ex](#) &e, [exvector](#) &free_indices, [exvector](#) &dummy_indices, const [scalar_products](#) &sp)
Simplify product of indexed expressions (commutative, noncommutative and simple squares), return list of free indices.
- bool [hasindex](#) (const [ex](#) &x, const [ex](#) &sym)
- [exvector get_all_dummy_indices_safely](#) (const [ex](#) &e)
More reliable version of the form.
- [exvector get_all_dummy_indices](#) (const [ex](#) &e)
Returns all dummy indices from the exvector.
- [lst rename_dummy_indices_uniquely](#) (const [exvector](#) &va, const [exvector](#) &vb)
Similar to above, where va and vb are the same and the return value is a list of two lists for substitution in b.
- [ex rename_dummy_indices_uniquely](#) (const [exvector](#) &va, const [exvector](#) &vb, const [ex](#) &b)
Same as above, where va and vb contain the indices of a and b and are sorted.
- [ex rename_dummy_indices_uniquely](#) (const [ex](#) &a, const [ex](#) &b)
Returns b with all dummy indices, which are common with a, renamed.
- [ex rename_dummy_indices_uniquely](#) ([exvector](#) &va, const [ex](#) &b, bool modify_va=false)
Returns b with all dummy indices, which are listed in va, renamed if modify_va is set to TRUE all dummy indices of b will be appended to va.
- [ex expand_dummy_sum](#) (const [ex](#) &e, bool subs_idx=false)
This function returns the given expression with expanded sums for all dummy index summations, where the dimensionality of the dummy index is a nonnegative integer.
- [GINAC_DECLARE_UNARCHIVER](#) ([indexed](#))
- static [ex conjugate_evalf](#) (const [ex](#) &arg)
- static [ex conjugate_eval](#) (const [ex](#) &arg)
- static void [conjugate_print_latex](#) (const [ex](#) &arg, const [print_context](#) &c)
- static [ex conjugate_conjugate](#) (const [ex](#) &arg)
- static [ex conjugate_expl_derivative](#) (const [ex](#) &arg, const [symbol](#) &s)
- static [ex conjugate_real_part](#) (const [ex](#) &arg)
- static [ex conjugate_imag_part](#) (const [ex](#) &arg)
- static bool [func_arg_info](#) (const [ex](#) &arg, unsigned inf)
- static bool [conjugate_info](#) (const [ex](#) &arg, unsigned inf)
- [REGISTER_FUNCTION](#) (conjugate_function, eval_func([conjugate_eval](#)). evalf_func([conjugate_evalf](#)).
expl_derivative_func([conjugate_expl_derivative](#)). info_func([conjugate_info](#)). print_func<[print_latex](#)
>([conjugate_print_latex](#)). conjugate_func([conjugate_conjugate](#)). real_part_func([conjugate_real_part](#)).
imag_part_func([conjugate_imag_part](#)). set_name("conjugate","conjugate"))
- static [ex real_part_evalf](#) (const [ex](#) &arg)
- static [ex real_part_eval](#) (const [ex](#) &arg)
- static void [real_part_print_latex](#) (const [ex](#) &arg, const [print_context](#) &c)
- static [ex real_part_conjugate](#) (const [ex](#) &arg)
- static [ex real_part_real_part](#) (const [ex](#) &arg)
- static [ex real_part_imag_part](#) (const [ex](#) &arg)
- static [ex real_part_expl_derivative](#) (const [ex](#) &arg, const [symbol](#) &s)
- [REGISTER_FUNCTION](#) (real_part_function, eval_func([real_part_eval](#)). evalf_func([real_part_evalf](#)). expl_↵
_derivative_func([real_part_expl_derivative](#)). print_func<[print_latex](#) >([real_part_print_latex](#)). conjugate_↵
_func([real_part_conjugate](#)). real_part_func([real_part_real_part](#)). imag_part_func([real_part_imag_part](#)).
set_name("real_part","real_part"))
- static [ex imag_part_evalf](#) (const [ex](#) &arg)
- static [ex imag_part_eval](#) (const [ex](#) &arg)
- static void [imag_part_print_latex](#) (const [ex](#) &arg, const [print_context](#) &c)
- static [ex imag_part_conjugate](#) (const [ex](#) &arg)
- static [ex imag_part_real_part](#) (const [ex](#) &arg)

- static [ex imag_part_imag_part](#) (const [ex](#) &arg)
- static [ex imag_part_expl_derivative](#) (const [ex](#) &arg, const [symbol](#) &s)
- [REGISTER_FUNCTION](#) ([imag_part_function](#), [eval_func\(imag_part_eval\)](#). [evalf_func\(imag_part_evalf\)](#). [expl_derivative_func\(imag_part_expl_derivative\)](#). [print_func< print_latex >\(imag_part_print_latex\)](#). [conjugate_func\(imag_part_conjugate\)](#). [real_part_func\(imag_part_real_part\)](#). [imag_part_func\(imag_part_imag_part\)](#). [set_name\("imag_part", "imag_part"\)](#))
- static [ex abs_evalf](#) (const [ex](#) &arg)
- static [ex abs_eval](#) (const [ex](#) &arg)
- static [ex abs_expand](#) (const [ex](#) &arg, unsigned [options](#))
- static [ex abs_expl_derivative](#) (const [ex](#) &arg, const [symbol](#) &s)
- static void [abs_print_latex](#) (const [ex](#) &arg, const [print_context](#) &c)
- static void [abs_print_csrc_float](#) (const [ex](#) &arg, const [print_context](#) &c)
- static [ex abs_conjugate](#) (const [ex](#) &arg)
- static [ex abs_real_part](#) (const [ex](#) &arg)
- static [ex abs_imag_part](#) (const [ex](#) &arg)
- static [ex abs_power](#) (const [ex](#) &arg, const [ex](#) &exp)
- bool [abs_info](#) (const [ex](#) &arg, unsigned inf)
- [REGISTER_FUNCTION](#) ([abs](#), [eval_func\(abs_eval\)](#). [evalf_func\(abs_evalf\)](#). [expand_func\(abs_expand\)](#). [expl_derivative_func\(abs_expl_derivative\)](#). [info_func\(abs_info\)](#). [print_func< print_latex >\(abs_print_latex\)](#). [print_func< print_csrc_float >\(abs_print_csrc_float\)](#). [print_func< print_csrc_double >\(abs_print_csrc_double\)](#). [conjugate_func\(abs_conjugate\)](#). [real_part_func\(abs_real_part\)](#). [imag_part_func\(abs_imag_part\)](#). [power_←_func\(abs_power\)](#))
- static [ex step_evalf](#) (const [ex](#) &arg)
- static [ex step_eval](#) (const [ex](#) &arg)
- static [ex step_series](#) (const [ex](#) &arg, const [relational](#) &rel, int [order](#), unsigned [options](#))
- static [ex step_conjugate](#) (const [ex](#) &arg)
- static [ex step_real_part](#) (const [ex](#) &arg)
- static [ex step_imag_part](#) (const [ex](#) &arg)
- [REGISTER_FUNCTION](#) ([step](#), [eval_func\(step_eval\)](#). [evalf_func\(step_evalf\)](#). [series_func\(step_series\)](#). [conjugate_func\(step_conjugate\)](#). [real_part_func\(step_real_part\)](#). [imag_part_func\(step_imag_part\)](#))
- static [ex csgn_evalf](#) (const [ex](#) &arg)
- static [ex csgn_eval](#) (const [ex](#) &arg)
- static [ex csgn_series](#) (const [ex](#) &arg, const [relational](#) &rel, int [order](#), unsigned [options](#))
- static [ex csgn_conjugate](#) (const [ex](#) &arg)
- static [ex csgn_real_part](#) (const [ex](#) &arg)
- static [ex csgn_imag_part](#) (const [ex](#) &arg)
- static [ex csgn_power](#) (const [ex](#) &arg, const [ex](#) &exp)
- [REGISTER_FUNCTION](#) ([csgn](#), [eval_func\(csgn_eval\)](#). [evalf_func\(csgn_evalf\)](#). [series_func\(csgn_series\)](#). [conjugate_func\(csgn_conjugate\)](#). [real_part_func\(csgn_real_part\)](#). [imag_part_func\(csgn_imag_part\)](#). [power_func\(csgn_power\)](#))
- static [ex eta_evalf](#) (const [ex](#) &x, const [ex](#) &y)
- static [ex eta_eval](#) (const [ex](#) &x, const [ex](#) &y)
- static [ex eta_series](#) (const [ex](#) &x, const [ex](#) &y, const [relational](#) &rel, int [order](#), unsigned [options](#))
- static [ex eta_conjugate](#) (const [ex](#) &x, const [ex](#) &y)
- static [ex eta_real_part](#) (const [ex](#) &x, const [ex](#) &y)
- static [ex eta_imag_part](#) (const [ex](#) &x, const [ex](#) &y)
- [REGISTER_FUNCTION](#) ([eta](#), [eval_func\(eta_eval\)](#). [evalf_func\(eta_evalf\)](#). [series_func\(eta_series\)](#). [latex_name\("\\eta"\)](#). [set_symmetry\(sy_symm\(0, 1\)\)](#). [conjugate_func\(eta_conjugate\)](#). [real_part_←_func\(eta_real_part\)](#). [imag_part_func\(eta_imag_part\)](#))
- static [ex Li2_evalf](#) (const [ex](#) &x)
- static [ex Li2_eval](#) (const [ex](#) &x)
- static [ex Li2_deriv](#) (const [ex](#) &x, unsigned [deriv_param](#))
- static [ex Li2_series](#) (const [ex](#) &x, const [relational](#) &rel, int [order](#), unsigned [options](#))
- static [ex Li2_conjugate](#) (const [ex](#) &x)
- [REGISTER_FUNCTION](#) ([Li2](#), [eval_func\(Li2_eval\)](#). [evalf_func\(Li2_evalf\)](#). [derivative_func\(Li2_deriv\)](#). [series_←_func\(Li2_series\)](#). [conjugate_func\(Li2_conjugate\)](#). [latex_name\("\\mathrm{Li}_2"\)](#))

- static `ex Li3_eval` (const `ex &x`)
 - `REGISTER_FUNCTION` (Li3, eval_func(Li3_eval). latex_name("\\mathrm{Li}_3"))
 - static `ex zetaderiv_eval` (const `ex &n`, const `ex &x`)
 - static `ex zetaderiv_deriv` (const `ex &n`, const `ex &x`, unsigned deriv_param)
 - `REGISTER_FUNCTION` (zetaderiv, eval_func(zetaderiv_eval). derivative_func(zetaderiv_deriv). latex_name("\\zeta^{\\prime}"))
 - static `ex factorial_evalf` (const `ex &x`)
 - static `ex factorial_eval` (const `ex &x`)
 - static void `factorial_print_dflt_latex` (const `ex &x`, const `print_context &c`)
 - static `ex factorial_conjugate` (const `ex &x`)
 - static `ex factorial_real_part` (const `ex &x`)
 - static `ex factorial_imag_part` (const `ex &x`)
 - `REGISTER_FUNCTION` (factorial, eval_func(factorial_eval). evalf_func(factorial_evalf). print_func<print_dflt>(factorial_print_dflt_latex). print_func<print_latex>(factorial_print_dflt_latex). conjugate_func(factorial_conjugate). real_part_func(factorial_real_part). imag_part_func(factorial_imag_part))
 - static `ex binomial_evalf` (const `ex &x`, const `ex &y`)
 - static `ex binomial_sym` (const `ex &x`, const `numeric &y`)
 - static `ex binomial_eval` (const `ex &x`, const `ex &y`)
 - static `ex binomial_conjugate` (const `ex &x`, const `ex &y`)
 - static `ex binomial_real_part` (const `ex &x`, const `ex &y`)
 - static `ex binomial_imag_part` (const `ex &x`, const `ex &y`)
 - `REGISTER_FUNCTION` (binomial, eval_func(binomial_eval). evalf_func(binomial_evalf). conjugate_func(binomial_conjugate). real_part_func(binomial_real_part). imag_part_func(binomial_imag_part))
 - static `ex Order_eval` (const `ex &x`)
 - static `ex Order_series` (const `ex &x`, const `relational &r`, int `order`, unsigned `options`)
 - static `ex Order_conjugate` (const `ex &x`)
 - static `ex Order_real_part` (const `ex &x`)
 - static `ex Order_imag_part` (const `ex &x`)
 - static `ex Order_power` (const `ex &x`, const `ex &e`)
 - static `ex Order_expl_derivative` (const `ex &arg`, const `symbol &s`)
 - `REGISTER_FUNCTION` (Order, eval_func(Order_eval). series_func(Order_series). latex_name("\\mathcal{O}"). expl_derivative_func(Order_expl_derivative). conjugate_func(Order_conjugate). real_part_func(Order_real_part). imag_part_func(Order_imag_part). power_func(Order_power))
 - `ex Isolve` (const `ex &eqns`, const `ex &symbols`, unsigned `options=solve_algo::automatic`)
- Factorial function.*
- const `numeric fsolve` (const `ex &f`, const `symbol &x`, const `numeric &x1`, const `numeric &x2`)
- Find a real root of real-valued function f(x) numerically within a given interval.*
- template<typename T1 >
 `function zeta` (const T1 &p1)
 - template<typename T1 , typename T2 >
 `function zeta` (const T1 &p1, const T2 &p2)
 - template<> bool `is_the_function< zeta_SERIAL >` (const `ex &x`)
 - template<typename T1 , typename T2 >
 `function G` (const T1 &x, const T2 &y)
 - template<typename T1 , typename T2 , typename T3 >
 `function G` (const T1 &x, const T2 &s, const T3 &y)
 - template<> bool `is_the_function< G_SERIAL >` (const `ex &x`)
 - template<typename T1 >
 `function psi` (const T1 &p1)
 - template<typename T1 , typename T2 >
 `function psi` (const T1 &p1, const T2 &p2)
 - template<> bool `is_the_function< psi_SERIAL >` (const `ex &x`)
 - template<typename T1 , typename T2 >
 `function iterated_integral` (const T1 &kernel_lst, const T2 &lambda)

- `template<typename T1, typename T2, typename T3 >`
`function iterated_integral` (const T1 &kernel_lst, const T2 &lambda, const T3 &N_trunc)
- `template<> bool is_the_function< iterated_integral_SERIAL >` (const `ex` &x)
- `bool is_order_function` (const `ex` &e)
Check whether a function is the Order ($O(n)$) function.
- `ex convert_H_to_Li` (const `ex` &parameterlst, const `ex` &arg)
Converts a given list containing parameters for H in Remiddi/Vermaseren notation into the corresponding GiNaC functions.
- `static ex EllipticK_evalf` (const `ex` &k)
- `static ex EllipticK_eval` (const `ex` &k)
- `static ex EllipticK_deriv` (const `ex` &k, unsigned deriv_param)
- `static ex EllipticK_series` (const `ex` &k, const `relational` &rel, int `order`, unsigned `options`)
- `static void EllipticK_print_latex` (const `ex` &k, const `print_context` &c)
- `REGISTER_FUNCTION` (EllipticK, evalf_func(EllipticK_evalf). eval_func(EllipticK_eval). derivative_↔
func(EllipticK_deriv). series_func(EllipticK_series). print_func< `print_latex` >(EllipticK_print_latex). do_↔
not_evalf_params())
- `static ex EllipticE_evalf` (const `ex` &k)
- `static ex EllipticE_eval` (const `ex` &k)
- `static ex EllipticE_deriv` (const `ex` &k, unsigned deriv_param)
- `static ex EllipticE_series` (const `ex` &k, const `relational` &rel, int `order`, unsigned `options`)
- `static void EllipticE_print_latex` (const `ex` &k, const `print_context` &c)
- `REGISTER_FUNCTION` (EllipticE, evalf_func(EllipticE_evalf). eval_func(EllipticE_eval). derivative_↔
func(EllipticE_deriv). series_func(EllipticE_series). print_func< `print_latex` >(EllipticE_print_latex). do_↔
not_evalf_params())
- `static ex iterated_integral_evalf_impl` (const `ex` &kernel_lst, const `ex` &lambda, const `ex` &N_trunc)
- `static ex iterated_integral2_evalf` (const `ex` &kernel_lst, const `ex` &lambda)
- `static ex iterated_integral3_evalf` (const `ex` &kernel_lst, const `ex` &lambda, const `ex` &N_trunc)
- `static ex iterated_integral2_eval` (const `ex` &kernel_lst, const `ex` &lambda)
- `static ex iterated_integral3_eval` (const `ex` &kernel_lst, const `ex` &lambda, const `ex` &N_trunc)
- `static ex lgamma_evalf` (const `ex` &x)
- `static ex lgamma_eval` (const `ex` &x)
Evaluation of $\lgamma(x)$, the natural logarithm of the Gamma function.
- `static ex lgamma_deriv` (const `ex` &x, unsigned deriv_param)
- `static ex lgamma_series` (const `ex` &arg, const `relational` &rel, int `order`, unsigned `options`)
- `static ex lgamma_conjugate` (const `ex` &x)
- `REGISTER_FUNCTION` (lgamma, eval_func(lgamma_eval). evalf_func(lgamma_evalf). derivative_↔
func(lgamma_deriv). series_func(lgamma_series). conjugate_func(lgamma_conjugate). latex_name("\\log
\\Gamma"))
- `static ex tgamma_evalf` (const `ex` &x)
- `static ex tgamma_eval` (const `ex` &x)
Evaluation of $tgamma(x)$, the true Gamma function.
- `static ex tgamma_deriv` (const `ex` &x, unsigned deriv_param)
- `static ex tgamma_series` (const `ex` &arg, const `relational` &rel, int `order`, unsigned `options`)
- `static ex tgamma_conjugate` (const `ex` &x)
- `REGISTER_FUNCTION` (tgamma, eval_func(tgamma_eval). evalf_func(tgamma_evalf). derivative_↔
func(tgamma_deriv). series_func(tgamma_series). conjugate_func(tgamma_conjugate). latex_↔
name("\\Gamma"))
- `static ex beta_evalf` (const `ex` &x, const `ex` &y)
- `static ex beta_eval` (const `ex` &x, const `ex` &y)
- `static ex beta_deriv` (const `ex` &x, const `ex` &y, unsigned deriv_param)
- `static ex beta_series` (const `ex` &arg1, const `ex` &arg2, const `relational` &rel, int `order`, unsigned `options`)
- `REGISTER_FUNCTION` (beta, eval_func(beta_eval). evalf_func(beta_evalf). derivative_func(beta_deriv).
series_func(beta_series). latex_name("\\mathrm{B}"). set_symmetry(sy_symm(0, 1)))
- `static ex psi1_evalf` (const `ex` &x)
- `static ex psi1_eval` (const `ex` &x)

Evaluation of digamma-function $\psi(x)$.

- static `ex psi1_deriv` (const `ex` &`x`, unsigned `deriv_param`)
- static `ex psi1_series` (const `ex` &`arg`, const `relational` &`rel`, int `order`, unsigned `options`)
- static `ex psi2_evalf` (const `ex` &`n`, const `ex` &`x`)
- static `ex psi2_eval` (const `ex` &`n`, const `ex` &`x`)

Evaluation of polygamma-function $\psi(n,x)$.

- static `ex psi2_deriv` (const `ex` &`n`, const `ex` &`x`, unsigned `deriv_param`)
- static `ex psi2_series` (const `ex` &`n`, const `ex` &`arg`, const `relational` &`rel`, int `order`, unsigned `options`)
- static `ex G2_evalf` (const `ex` &`x_`, const `ex` &`y`)
- static `ex G2_eval` (const `ex` &`x_`, const `ex` &`y`)
- static `ex G3_evalf` (const `ex` &`x_`, const `ex` &`s_`, const `ex` &`y`)
- static `ex G3_eval` (const `ex` &`x_`, const `ex` &`s_`, const `ex` &`y`)
- static `ex Li_evalf` (const `ex` &`m_`, const `ex` &`x_`)
- static `ex Li_eval` (const `ex` &`m_`, const `ex` &`x_`)
- static `ex Li_series` (const `ex` &`m`, const `ex` &`x`, const `relational` &`rel`, int `order`, unsigned `options`)
- static `ex Li_deriv` (const `ex` &`m_`, const `ex` &`x_`, unsigned `deriv_param`)
- static void `Li_print_latex` (const `ex` &`m_`, const `ex` &`x_`, const `print_context` &`c`)
- `REGISTER_FUNCTION` (`Li`, `evalf_func`(`Li_evalf`). `eval_func`(`Li_eval`). `series_func`(`Li_series`). `derivative_`↵
`func`(`Li_deriv`). `print_func`< `print_latex` >(`Li_print_latex`). `do_not_evalf_params`()
- static `ex S_evalf` (const `ex` &`n`, const `ex` &`p`, const `ex` &`x`)
- static `ex S_eval` (const `ex` &`n`, const `ex` &`p`, const `ex` &`x`)
- static `ex S_series` (const `ex` &`n`, const `ex` &`p`, const `ex` &`x`, const `relational` &`rel`, int `order`, unsigned `options`)
- static `ex S_deriv` (const `ex` &`n`, const `ex` &`p`, const `ex` &`x`, unsigned `deriv_param`)
- static void `S_print_latex` (const `ex` &`n`, const `ex` &`p`, const `ex` &`x`, const `print_context` &`c`)
- `REGISTER_FUNCTION` (`S`, `evalf_func`(`S_evalf`). `eval_func`(`S_eval`). `series_func`(`S_series`). `derivative_`↵
`func`(`S_deriv`). `print_func`< `print_latex` >(`S_print_latex`). `do_not_evalf_params`()
- static `ex H_evalf` (const `ex` &`x1`, const `ex` &`x2`)
- static `ex H_eval` (const `ex` &`m_`, const `ex` &`x`)
- static `ex H_series` (const `ex` &`m`, const `ex` &`x`, const `relational` &`rel`, int `order`, unsigned `options`)
- static `ex H_deriv` (const `ex` &`m_`, const `ex` &`x`, unsigned `deriv_param`)
- static void `H_print_latex` (const `ex` &`m_`, const `ex` &`x`, const `print_context` &`c`)
- `REGISTER_FUNCTION` (`H`, `evalf_func`(`H_evalf`). `eval_func`(`H_eval`). `series_func`(`H_series`). `derivative_`↵
`func`(`H_deriv`). `print_func`< `print_latex` >(`H_print_latex`). `do_not_evalf_params`()
- static `ex zeta1_evalf` (const `ex` &`x`)
- static `ex zeta1_eval` (const `ex` &`m`)
- static `ex zeta1_deriv` (const `ex` &`m`, unsigned `deriv_param`)
- static void `zeta1_print_latex` (const `ex` &`m_`, const `print_context` &`c`)
- static `ex zeta2_evalf` (const `ex` &`x`, const `ex` &`s`)
- static `ex zeta2_eval` (const `ex` &`m`, const `ex` &`s_`)
- static `ex zeta2_deriv` (const `ex` &`m`, const `ex` &`s`, unsigned `deriv_param`)
- static void `zeta2_print_latex` (const `ex` &`m_`, const `ex` &`s_`, const `print_context` &`c`)
- static `ex exp_evalf` (const `ex` &`x`)
- static `ex exp_eval` (const `ex` &`x`)
- static `ex exp_expand` (const `ex` &`arg`, unsigned `options`)
- static `ex exp_deriv` (const `ex` &`x`, unsigned `deriv_param`)
- static `ex exp_real_part` (const `ex` &`x`)
- static `ex exp_imag_part` (const `ex` &`x`)
- static `ex exp_conjugate` (const `ex` &`x`)
- static `ex exp_power` (const `ex` &`x`, const `ex` &`a`)
- static bool `exp_info` (const `ex` &`x`, unsigned `inf`)
- `REGISTER_FUNCTION` (`exp`, `eval_func`(`exp_eval`). `evalf_func`(`exp_evalf`). `info_func`(`exp_info`). `expand_`↵
`func`(`exp_expand`). `derivative_func`(`exp_deriv`). `real_part_func`(`exp_real_part`). `imag_part_func`(`exp_imag_part`).
`conjugate_func`(`exp_conjugate`). `power_func`(`exp_power`). `latex_name`("\\exp")
- static `ex log_evalf` (const `ex` &`x`)
- static `ex log_eval` (const `ex` &`x`)

- static `ex log_deriv` (const `ex &x`, unsigned `deriv_param`)
- static `ex log_series` (const `ex &arg`, const `relational &rel`, int `order`, unsigned `options`)
- static `ex log_real_part` (const `ex &x`)
- static `ex log_imag_part` (const `ex &x`)
- static `ex log_expand` (const `ex &arg`, unsigned `options`)
- static `ex log_conjugate` (const `ex &x`)
- static bool `log_info` (const `ex &x`, unsigned `inf`)
- `REGISTER_FUNCTION` (`log`, `eval_func(log_eval)`. `evalf_func(log_evalf)`. `info_func(log_info)`. `expand_func(log_expand)`. `derivative_func(log_deriv)`. `series_func(log_series)`. `real_part_func(log_real_part)`. `imag_part_func(log_imag_part)`. `conjugate_func(log_conjugate)`. `latex_name("\\ln")`)
- static `ex sin_evalf` (const `ex &x`)
- static `ex sin_eval` (const `ex &x`)
- static `ex sin_deriv` (const `ex &x`, unsigned `deriv_param`)
- static `ex sin_real_part` (const `ex &x`)
- static `ex sin_imag_part` (const `ex &x`)
- static `ex sin_conjugate` (const `ex &x`)
- static bool `trig_info` (const `ex &x`, unsigned `inf`)
- `REGISTER_FUNCTION` (`sin`, `eval_func(sin_eval)`. `evalf_func(sin_evalf)`. `info_func(trig_info)`. `derivative_func(sin_deriv)`. `real_part_func(sin_real_part)`. `imag_part_func(sin_imag_part)`. `conjugate_func(sin_conjugate)`. `latex_name("\\sin")`)
- static `ex cos_evalf` (const `ex &x`)
- static `ex cos_eval` (const `ex &x`)
- static `ex cos_deriv` (const `ex &x`, unsigned `deriv_param`)
- static `ex cos_real_part` (const `ex &x`)
- static `ex cos_imag_part` (const `ex &x`)
- static `ex cos_conjugate` (const `ex &x`)
- `REGISTER_FUNCTION` (`cos`, `eval_func(cos_eval)`. `info_func(trig_info)`. `evalf_func(cos_evalf)`. `derivative_func(cos_deriv)`. `real_part_func(cos_real_part)`. `imag_part_func(cos_imag_part)`. `conjugate_func(cos_conjugate)`. `latex_name("\\cos")`)
- static `ex tan_evalf` (const `ex &x`)
- static `ex tan_eval` (const `ex &x`)
- static `ex tan_deriv` (const `ex &x`, unsigned `deriv_param`)
- static `ex tan_real_part` (const `ex &x`)
- static `ex tan_imag_part` (const `ex &x`)
- static `ex tan_series` (const `ex &x`, const `relational &rel`, int `order`, unsigned `options`)
- static `ex tan_conjugate` (const `ex &x`)
- `REGISTER_FUNCTION` (`tan`, `eval_func(tan_eval)`. `evalf_func(tan_evalf)`. `info_func(trig_info)`. `derivative_func(tan_deriv)`. `series_func(tan_series)`. `real_part_func(tan_real_part)`. `imag_part_func(tan_imag_part)`. `conjugate_func(tan_conjugate)`. `latex_name("\\tan")`)
- static `ex asin_evalf` (const `ex &x`)
- static `ex asin_eval` (const `ex &x`)
- static `ex asin_deriv` (const `ex &x`, unsigned `deriv_param`)
- static `ex asin_conjugate` (const `ex &x`)
- static bool `asin_info` (const `ex &x`, unsigned `inf`)
- `REGISTER_FUNCTION` (`asin`, `eval_func(asin_eval)`. `evalf_func(asin_evalf)`. `info_func(asin_info)`. `derivative_func(asin_deriv)`. `conjugate_func(asin_conjugate)`. `latex_name("\\arcsin")`)
- static `ex acos_evalf` (const `ex &x`)
- static `ex acos_eval` (const `ex &x`)
- static `ex acos_deriv` (const `ex &x`, unsigned `deriv_param`)
- static `ex acos_conjugate` (const `ex &x`)
- `REGISTER_FUNCTION` (`acos`, `eval_func(acos_eval)`. `evalf_func(acos_evalf)`. `info_func(asin_info)`. `derivative_func(acos_deriv)`. `conjugate_func(acos_conjugate)`. `latex_name("\\arccos")`)
- static `ex atan_evalf` (const `ex &x`)
- static `ex atan_eval` (const `ex &x`)
- static `ex atan_deriv` (const `ex &x`, unsigned `deriv_param`)

- static `ex atan_series` (const `ex` &arg, const `relational` &rel, int `order`, unsigned `options`)
- static `ex atan_conjugate` (const `ex` &x)
- static bool `atan_info` (const `ex` &x, unsigned inf)
- `REGISTER_FUNCTION` (`atan`, `eval_func(atan_eval)`. `evalf_func(atan_evalf)`. `info_func(atan_info)`. `derivative_func(atan_deriv)`. `series_func(atan_series)`. `conjugate_func(atan_conjugate)`. `latex_name("\\arctan")`)
- static `ex atan2_evalf` (const `ex` &y, const `ex` &x)
- static `ex atan2_eval` (const `ex` &y, const `ex` &x)
- static `ex atan2_deriv` (const `ex` &y, const `ex` &x, unsigned `deriv_param`)
- static bool `atan2_info` (const `ex` &y, const `ex` &x, unsigned inf)
- `REGISTER_FUNCTION` (`atan2`, `eval_func(atan2_eval)`. `evalf_func(atan2_evalf)`. `info_func(atan2_info)`. `evalf_func(atan2_evalf)`. `derivative_func(atan2_deriv)`)
- static `ex sinh_evalf` (const `ex` &x)
- static `ex sinh_eval` (const `ex` &x)
- static `ex sinh_deriv` (const `ex` &x, unsigned `deriv_param`)
- static `ex sinh_real_part` (const `ex` &x)
- static `ex sinh_imag_part` (const `ex` &x)
- static `ex sinh_conjugate` (const `ex` &x)
- `REGISTER_FUNCTION` (`sinh`, `eval_func(sinh_eval)`. `evalf_func(sinh_evalf)`. `info_func(atan_info)`. `derivative_func(sinh_deriv)`. `real_part_func(sinh_real_part)`. `imag_part_func(sinh_imag_part)`. `conjugate_func(sinh_conjugate)`. `latex_name("\\sinh")`)
- static `ex cosh_evalf` (const `ex` &x)
- static `ex cosh_eval` (const `ex` &x)
- static `ex cosh_deriv` (const `ex` &x, unsigned `deriv_param`)
- static `ex cosh_real_part` (const `ex` &x)
- static `ex cosh_imag_part` (const `ex` &x)
- static `ex cosh_conjugate` (const `ex` &x)
- `REGISTER_FUNCTION` (`cosh`, `eval_func(cosh_eval)`. `evalf_func(cosh_evalf)`. `info_func(exp_info)`. `derivative_func(cosh_deriv)`. `real_part_func(cosh_real_part)`. `imag_part_func(cosh_imag_part)`. `conjugate_func(cosh_conjugate)`. `latex_name("\\cosh")`)
- static `ex tanh_evalf` (const `ex` &x)
- static `ex tanh_eval` (const `ex` &x)
- static `ex tanh_deriv` (const `ex` &x, unsigned `deriv_param`)
- static `ex tanh_series` (const `ex` &x, const `relational` &rel, int `order`, unsigned `options`)
- static `ex tanh_real_part` (const `ex` &x)
- static `ex tanh_imag_part` (const `ex` &x)
- static `ex tanh_conjugate` (const `ex` &x)
- `REGISTER_FUNCTION` (`tanh`, `eval_func(tanh_eval)`. `evalf_func(tanh_evalf)`. `info_func(atan_info)`. `derivative_func(tanh_deriv)`. `series_func(tanh_series)`. `real_part_func(tanh_real_part)`. `imag_part_func(tanh_imag_part)`. `conjugate_func(tanh_conjugate)`. `latex_name("\\tanh")`)
- static `ex asinh_evalf` (const `ex` &x)
- static `ex asinh_eval` (const `ex` &x)
- static `ex asinh_deriv` (const `ex` &x, unsigned `deriv_param`)
- static `ex asinh_conjugate` (const `ex` &x)
- `REGISTER_FUNCTION` (`asinh`, `eval_func(asinh_eval)`. `evalf_func(asinh_evalf)`. `info_func(atan_info)`. `derivative_func(asinh_deriv)`. `conjugate_func(asinh_conjugate)`)
- static `ex acosh_evalf` (const `ex` &x)
- static `ex acosh_eval` (const `ex` &x)
- static `ex acosh_deriv` (const `ex` &x, unsigned `deriv_param`)
- static `ex acosh_conjugate` (const `ex` &x)
- `REGISTER_FUNCTION` (`acosh`, `eval_func(acosh_eval)`. `evalf_func(acosh_evalf)`. `info_func(asin_info)`. `derivative_func(acosh_deriv)`. `conjugate_func(acosh_conjugate)`)
- static `ex atanh_evalf` (const `ex` &x)
- static `ex atanh_eval` (const `ex` &x)
- static `ex atanh_deriv` (const `ex` &x, unsigned `deriv_param`)

- static `ex atanh_series` (const `ex` &arg, const `relational` &rel, int `order`, unsigned `options`)
- static `ex atanh_conjugate` (const `ex` &x)
- `REGISTER_FUNCTION` (`atanh`, `eval_func(atanh_eval)`. `evalf_func(atanh_evalf)`. `info_func(asin_info)`. `derivative_func(atanh_deriv)`. `series_func(atanh_series)`. `conjugate_func(atanh_conjugate)`)
- `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`integral`, `basic`, `print_func< print_dflt >(&integral::do_print)`. `print_func< print_python >(&integral::do_print)`. `print_func< print_latex >(&integral::do_print_latex)`) `integral`
- `ex subvalue` (const `ex` &var, const `ex` &value, const `ex` &fun)
- `ex adaptivesimpson` (const `ex` &x, const `ex` &a_in, const `ex` &b_in, const `ex` &f, const `ex` &error)

Numeric integration routine based upon the "Adaptive Quadrature" one in "Numerical Analysis" by Burden and Faires.
- `GINAC_BIND_UNARCHIVER` (`integral`)
- `GINAC_DECLARE_UNARCHIVER` (`integral`)
- `ex ifactor` (const `numeric` &n)

Returns the decomposition of the positive integer n into prime numbers in the form $l_1(p_1, \dots, p_r)$, $l_2(a_1, \dots, a_r)$ such that $n = p_1^{a_1} \dots p_r^{a_r}$.
- bool `is_discriminant_of_quadratic_number_field` (const `numeric` &n)

Returns true if the integer n is either one or the discriminant of a quadratic number field.
- `numeric kronecker_symbol` (const `numeric` &a, const `numeric` &n)

Returns the Kronecker symbol a: integer n: integer.
- `numeric primitive_dirichlet_character` (const `numeric` &n, const `numeric` &a)

Defines a primitive Dirichlet character through the Kronecker symbol.
- `numeric dirichlet_character` (const `numeric` &n, const `numeric` &a, const `numeric` &N)

Defines a Dirichlet character through the Kronecker symbol.
- `numeric generalised_Bernoulli_number` (const `numeric` &k, const `numeric` &b)

The generalised Bernoulli number.
- `ex Bernoulli_polynomial` (const `numeric` &k, const `ex` &x)

The Bernoulli polynomials.
- `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`integration_kernel`, `basic`, `print_func< print_context >(&integration_kernel::do_print)`) `integration_kernel`
- `GINAC_BIND_UNARCHIVER` (`integration_kernel`)
- `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`basic_log_kernel`, `integration_kernel`, `print_func< print_context >(&basic_log_kernel::do_print)`) `basic_log_kernel`
- `GINAC_BIND_UNARCHIVER` (`basic_log_kernel`)
- `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`multiple_polylog_kernel`, `integration_kernel`, `print_func< print_context >(&multiple_polylog_kernel::do_print)`) `multiple_polylog_kernel`
- `GINAC_BIND_UNARCHIVER` (`multiple_polylog_kernel`)
- `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`ELi_kernel`, `integration_kernel`, `print_func< print_context >(&ELi_kernel::do_print)`) `ELi_kernel`
- `GINAC_BIND_UNARCHIVER` (`ELi_kernel`)
- `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`Ebar_kernel`, `integration_kernel`, `print_func< print_context >(&Ebar_kernel::do_print)`) `Ebar_kernel`
- `GINAC_BIND_UNARCHIVER` (`Ebar_kernel`)
- `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`Kronecker_dtau_kernel`, `integration_kernel`, `print_func< print_context >(&Kronecker_dtau_kernel::do_print)`) `Kronecker_dtau_kernel`
- `GINAC_BIND_UNARCHIVER` (`Kronecker_dtau_kernel`)
- `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`Kronecker_dz_kernel`, `integration_kernel`, `print_func< print_context >(&Kronecker_dz_kernel::do_print)`) `Kronecker_dz_kernel`
- `GINAC_BIND_UNARCHIVER` (`Kronecker_dz_kernel`)
- `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`Eisenstein_kernel`, `integration_kernel`, `print_func< print_context >(&Eisenstein_kernel::do_print)`) `Eisenstein_kernel`
- `GINAC_BIND_UNARCHIVER` (`Eisenstein_kernel`)
- `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`Eisenstein_h_kernel`, `integration_kernel`, `print_func< print_context >(&Eisenstein_h_kernel::do_print)`) `Eisenstein_h_kernel`
- `GINAC_BIND_UNARCHIVER` (`Eisenstein_h_kernel`)

- [GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([modular_form_kernel](#), [integration_kernel](#), [print_func](#)<[print_context](#) >(&[modular_form_kernel](#)::do_print)) [modular_form_kernel](#)
- [GINAC_BIND_UNARCHIVER](#) ([modular_form_kernel](#))
- [GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([user_defined_kernel](#), [integration_kernel](#), [print_func](#)<[print_context](#) >(&[user_defined_kernel](#)::do_print)) [user_defined_kernel](#)
- [GINAC_BIND_UNARCHIVER](#) ([user_defined_kernel](#))
- [GINAC_DECLARE_UNARCHIVER](#) ([integration_kernel](#))
- [GINAC_DECLARE_UNARCHIVER](#) ([basic_log_kernel](#))
- [GINAC_DECLARE_UNARCHIVER](#) ([multiple_polylog_kernel](#))
- [GINAC_DECLARE_UNARCHIVER](#) ([ELi_kernel](#))
- [GINAC_DECLARE_UNARCHIVER](#) ([Ebar_kernel](#))
- [GINAC_DECLARE_UNARCHIVER](#) ([Kronecker_dtau_kernel](#))
- [GINAC_DECLARE_UNARCHIVER](#) ([Kronecker_dz_kernel](#))
- [GINAC_DECLARE_UNARCHIVER](#) ([Eisenstein_kernel](#))
- [GINAC_DECLARE_UNARCHIVER](#) ([Eisenstein_h_kernel](#))
- [GINAC_DECLARE_UNARCHIVER](#) ([modular_form_kernel](#))
- [GINAC_DECLARE_UNARCHIVER](#) ([user_defined_kernel](#))
- [GINAC_DECLARE_UNARCHIVER](#) ([lst](#))
- [GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([matrix](#), [basic](#), [print_func](#)< [print_context](#) >(&[matrix](#)::do_print). [print_func](#)< [print_latex](#) >(&[matrix](#)::do_print_latex). [print_func](#)< [print_tree](#) >(&[matrix](#)::do_print_tree). [print_func](#)< [print_python_repr](#) >(&[matrix](#)::do_print_python_repr)) [matrix](#)
- Default ctor.*
- [GINAC_BIND_UNARCHIVER](#) ([matrix](#))
- [ex lst_to_matrix](#) (const [lst](#) &l)
- Convert list of lists to matrix.*
- [ex diag_matrix](#) (const [lst](#) &l)
- Convert list of diagonal elements to matrix.*
- [ex diag_matrix](#) (std::initializer_list< [ex](#) > l)
- [ex unit_matrix](#) (unsigned [r](#), unsigned [c](#))
- Create an r times c unit matrix.*
- [ex symbolic_matrix](#) (unsigned [r](#), unsigned [c](#), const std::string &base_name, const std::string &tex_base_name)
- Create an r times c matrix of newly generated symbols consisting of the given base name plus the numeric row/column position of each element.*
- [ex reduced_matrix](#) (const [matrix](#) &m, unsigned [r](#), unsigned [c](#))
- Return the reduced matrix that is formed by deleting the rth row and cth column of matrix m.*
- [ex sub_matrix](#) (const [matrix](#) &m, unsigned [r](#), unsigned [nr](#), unsigned [c](#), unsigned [nc](#))
- Return the nr times nc submatrix starting at position r, c of matrix m.*
- [GINAC_DECLARE_UNARCHIVER](#) ([matrix](#))
- [size_t nops](#) (const [matrix](#) &m)
- [ex expand](#) (const [matrix](#) &m, unsigned [options](#)=0)
- [ex evalf](#) (const [matrix](#) &m)
- unsigned [rows](#) (const [matrix](#) &m)
- unsigned [cols](#) (const [matrix](#) &m)
- [matrix transpose](#) (const [matrix](#) &m)
- [ex determinant](#) (const [matrix](#) &m, unsigned [options](#)=[determinant_algo::automatic](#))
- [ex trace](#) (const [matrix](#) &m)
- [ex charpoly](#) (const [matrix](#) &m, const [ex](#) &lambda)
- [matrix inverse](#) (const [matrix](#) &m)
- [matrix inverse](#) (const [matrix](#) &m, unsigned [algo](#))
- unsigned [rank](#) (const [matrix](#) &m)
- unsigned [rank](#) (const [matrix](#) &m, unsigned [solve_algo](#))
- [ex unit_matrix](#) (unsigned [x](#))
- Create a x times x unit matrix.*

- `ex symbolic_matrix` (unsigned `r`, unsigned `c`, const std::string &base_name)
Create an r times c matrix of newly generated symbols consisting of the given base name plus the numeric row/column position of each element.
- `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`mul`, `expairseq`, `print_func`< `print_context` >(&`mul::do_print`).
`print_func`< `print_latex` >(&`mul::do_print_latex`). `print_func`< `print_csrc` >(&`mul::do_print_csrc`). `print_`↵
`func`< `print_tree` >(&`mul::do_print_tree`). `print_func`< `print_python_repr` >(&`mul::do_print_python_repr`))
`mul`
- bool `tryfactsubs` (const `ex` &origfactor, const `ex` &patternfactor, int &nummatches, `exmap` &repls)
- bool `algebraic_match_mul_with_mul` (const `mul` &`e`, const `ex` &pat, `exmap` &repls, int `factor`, int &nummatches, const std::vector< bool > &subsed, std::vector< bool > &matched)
Checks whether e matches to the pattern pat and the (possibly to be updated) list of replacements $repls$.
- `GINAC_BIND_UNARCHIVER` (`mul`)
- `GINAC_DECLARE_UNARCHIVER` (`mul`)
- `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`ncmul`, `exprseq`, `print_func`< `print_context` >(&`ncmul::do_print`).
`print_func`< `print_tree` >(&`ncmul::do_print_tree`). `print_func`< `print_csrc` >(&`ncmul::do_print_csrc`). `print_`↵
`func`< `print_python_repr` >(&`ncmul::do_print_csrc`)) `ncmul`
- `ex reeval_ncmul` (const `exvector` &`v`)
- `ex hold_ncmul` (const `exvector` &`v`)
- `GINAC_BIND_UNARCHIVER` (`ncmul`)
- `GINAC_DECLARE_UNARCHIVER` (`ncmul`)
- static bool `get_first_symbol` (const `ex` &`e`, `ex` &`x`)
Return pointer to first symbol found in expression.
- static void `add_symbol` (const `ex` &`s`, `sym_desc_vec` &`v`)
- static void `collect_symbols` (const `ex` &`e`, `sym_desc_vec` &`v`)
- static void `get_symbol_stats` (const `ex` &`a`, const `ex` &`b`, `sym_desc_vec` &`v`)
Collect statistical information about symbols in polynomials.
- static `numeric lcmcoeff` (const `ex` &`e`, const `numeric` &`l`)
- static `numeric lcm_of_coefficients_denominators` (const `ex` &`e`)
Compute LCM of denominators of coefficients of a polynomial.
- static `ex multiply_lcm` (const `ex` &`e`, const `numeric` &`lcm`)
Bring polynomial from $Q[X]$ to $Z[X]$ by multiplying in the previously determined LCM of the coefficient's denominators.
- `ex quo` (const `ex` &`a`, const `ex` &`b`, const `ex` &`x`, bool check_args)
Quotient $q(x)$ of polynomials $a(x)$ and $b(x)$ in $Q[x]$.
- `ex rem` (const `ex` &`a`, const `ex` &`b`, const `ex` &`x`, bool check_args)
Remainder $r(x)$ of polynomials $a(x)$ and $b(x)$ in $Q[x]$.
- `ex decomp_rational` (const `ex` &`a`, const `ex` &`x`)
Decompose rational function $a(x)=N(x)/D(x)$ into $P(x)+n(x)/D(x)$ with $\text{degree}(n, x) < \text{degree}(D, x)$.
- `ex prem` (const `ex` &`a`, const `ex` &`b`, const `ex` &`x`, bool check_args)
Pseudo-remainder of polynomials $a(x)$ and $b(x)$ in $Q[x]$.
- `ex sprem` (const `ex` &`a`, const `ex` &`b`, const `ex` &`x`, bool check_args)
Sparse pseudo-remainder of polynomials $a(x)$ and $b(x)$ in $Q[x]$.
- bool `divide` (const `ex` &`a`, const `ex` &`b`, `ex` &`q`, bool check_args)
Exact polynomial division of $a(X)$ by $b(X)$ in $Q[X]$.
- static bool `divide_in_z` (const `ex` &`a`, const `ex` &`b`, `ex` &`q`, `sym_desc_vec::const_iterator` var)
Exact polynomial division of $a(X)$ by $b(X)$ in $Z[X]$.
- static `ex sr_gcd` (const `ex` &`a`, const `ex` &`b`, `sym_desc_vec::const_iterator` var)
Compute GCD of multivariate polynomials using the subresultant PRS algorithm.
- static `ex interpolate` (const `ex` &gamma, const `numeric` &xi, const `ex` &`x`, int degree_hint=1)
 ξ -adic polynomial interpolation
- static bool `heur_gcd_z` (`ex` &res, const `ex` &`a`, const `ex` &`b`, `ex` *ca, `ex` *cb, `sym_desc_vec::const_iterator` var)
Compute GCD of multivariate polynomials using the heuristic GCD algorithm.
- static bool `heur_gcd` (`ex` &res, const `ex` &`a`, const `ex` &`b`, `ex` *ca, `ex` *cb, `sym_desc_vec::const_iterator` var)

- Compute GCD of multivariate polynomials using the heuristic GCD algorithm.*
 - static `ex gcd_pf_pow` (const `ex` &a, const `ex` &b, `ex` *ca, `ex` *cb)
 - static `ex gcd_pf_mul` (const `ex` &a, const `ex` &b, `ex` *ca, `ex` *cb)
 - `ex gcd` (const `ex` &a, const `ex` &b, `ex` *ca, `ex` *cb, bool check_args, unsigned options)
 - Compute GCD (Greatest Common Divisor) of multivariate polynomials $a(X)$ and $b(X)$ in $\mathbb{Z}[X]$.*
 - static `ex gcd_pf_pow_pow` (const `ex` &a, const `ex` &b, `ex` *ca, `ex` *cb)
 - `ex lcm` (const `ex` &a, const `ex` &b, bool check_args)
 - Compute LCM (Least Common Multiple) of multivariate polynomials in $\mathbb{Z}[X]$.*
 - static `epvector sqrfree_yun` (const `ex` &a, const `symbol` &x)
 - Compute square-free factorization of multivariate polynomial $a(x)$ using Yun's algorithm.*
 - `ex sqrfree` (const `ex` &a, const `lst` &l)
 - Compute a square-free factorization of a multivariate polynomial in $\mathbb{Q}[X]$.*
 - `ex sqrfree_parfrac` (const `ex` &a, const `symbol` &x)
 - Compute square-free partial fraction decomposition of rational function $a(x)$.*
 - static `ex replace_with_symbol` (const `ex` &e, `exmap` &repl, `exmap` &rev_lookup, `lst` &modifier)
 - Create a symbol for replacing the expression "e" (or return a previously assigned symbol).*
 - static `ex replace_with_symbol` (const `ex` &e, `exmap` &repl)
 - Create a symbol for replacing the expression "e" (or return a previously assigned symbol).*
 - static `ex frac_cancel` (const `ex` &n, const `ex` &d)
 - Fraction cancellation.*
 - static `ex find_common_factor` (const `ex` &e, `ex` &factor, `exmap` &repl)
 - Remove the common factor in the terms of a sum 'e' by calculating the GCD, and multiply it into the expression 'factor' (which needs to be initialized to 1, unless you're accumulating factors).*
 - `ex collect_common_factors` (const `ex` &e)
 - Collect common factors in sums.*
 - `ex resultant` (const `ex` &e1, const `ex` &e2, const `ex` &s)
 - Resultant of two expressions e_1, e_2 with respect to symbol s .*
 - `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (numeric, basic, print_func< `print_context` >(&numeric::do_print). print_func< `print_latex` >(&numeric::do_print_latex). print_func< `print_csrc` >(&numeric::do_print_csrc). print_func< `print_csrc_cl_N` >(&numeric::do_print_csrc_cl_N). print_func< `print_tree` >(&numeric::do_print_tree). print_func< `print_python_repr` >(&numeric::do_print_python_repr)) numeric
 - default ctor.*
 - static const `cln::cl_F make_real_float` (const `cln::cl_idecoded_float` &dec)
 - Construct a floating point number from sign, mantissa, and exponent.*
 - static const `cln::cl_F read_real_float` (std::istream &s)
 - Read serialized floating point number.*
 - `GINAC_BIND_UNARCHIVER` (numeric)
 - static void `write_real_float` (std::ostream &s, const `cln::cl_R` &n)
 - static void `print_real_number` (const `print_context` &c, const `cln::cl_R` &x)
 - Helper function to print a real number in a nicer way than is CLN's default.*
 - static void `print_integer_csrc` (const `print_context` &c, const `cln::cl_I` &x)
 - Helper function to print integer number in C++ source format.*
 - static void `print_real_csrc` (const `print_context` &c, const `cln::cl_R` &x)
 - Helper function to print real number in C++ source format.*
 - template<typename T1, typename T2>
static bool `coerce` (T1 &dst, const T2 &arg)
 - template<> bool `coerce`< `int`, `cln::cl_I` > (int &dst, const `cln::cl_I` &arg)
 - Check if CLN integer can be converted into int.*
 - template<> bool `coerce`< `unsigned int`, `cln::cl_I` > (unsigned int &dst, const `cln::cl_I` &arg)
 - static void `print_real_cl_N` (const `print_context` &c, const `cln::cl_R` &x)
 - Helper function to print real number in C++ source format using `cl_N` types.*
 - const `numeric exp` (const `numeric` &x)

- Exponential function.*
 - const [numeric log](#) (const [numeric](#) &x)
 - Natural logarithm.*
 - const [numeric sin](#) (const [numeric](#) &x)
 - Numeric sine (trigonometric function).*
 - const [numeric cos](#) (const [numeric](#) &x)
 - Numeric cosine (trigonometric function).*
 - const [numeric tan](#) (const [numeric](#) &x)
 - Numeric tangent (trigonometric function).*
 - const [numeric asin](#) (const [numeric](#) &x)
 - Numeric inverse sine (trigonometric function).*
 - const [numeric acos](#) (const [numeric](#) &x)
 - Numeric inverse cosine (trigonometric function).*
 - const [numeric atan](#) (const [numeric](#) &x)
 - Numeric arcustangent.*
 - const [numeric atan](#) (const [numeric](#) &y, const [numeric](#) &x)
 - Numeric arcustangent of two arguments, analytically continued in a suitable way.*
 - const [numeric sinh](#) (const [numeric](#) &x)
 - Numeric hyperbolic sine (trigonometric function).*
 - const [numeric cosh](#) (const [numeric](#) &x)
 - Numeric hyperbolic cosine (trigonometric function).*
 - const [numeric tanh](#) (const [numeric](#) &x)
 - Numeric hyperbolic tangent (trigonometric function).*
 - const [numeric asinh](#) (const [numeric](#) &x)
 - Numeric inverse hyperbolic sine (trigonometric function).*
 - const [numeric acosh](#) (const [numeric](#) &x)
 - Numeric inverse hyperbolic cosine (trigonometric function).*
 - const [numeric atanh](#) (const [numeric](#) &x)
 - Numeric inverse hyperbolic tangent (trigonometric function).*
 - static [cln::cl_N Li2_series](#) (const [cln::cl_N](#) &x, const [cln::float_format_t](#) &prec)
 - Numeric evaluation of Dilogarithm within circle of convergence (unit circle) using a power series.*
 - static [cln::cl_N Li2_projection](#) (const [cln::cl_N](#) &x, const [cln::float_format_t](#) &prec)
 - Folds Li2's argument inside a small rectangle to enhance convergence.*
 - const [cln::cl_N Li2_](#) (const [cln::cl_N](#) &value)
 - Numeric evaluation of Dilogarithm.*
 - const [numeric Li2](#) (const [numeric](#) &x)
 - const [numeric zeta](#) (const [numeric](#) &x)
 - Numeric evaluation of Riemann's Zeta function.*
 - static [cln::float_format_t guess_precision](#) (const [cln::cl_N](#) &x)
 - const [cln::cl_N lgamma](#) (const [cln::cl_N](#) &x)
 - The Gamma function.*
 - const [numeric lgamma](#) (const [numeric](#) &x)
 - const [cln::cl_N tgamma](#) (const [cln::cl_N](#) &x)
 - const [numeric tgamma](#) (const [numeric](#) &x)
 - const [numeric psi](#) (const [numeric](#) &x)
 - The psi function (aka polygamma function).*
 - const [numeric psi](#) (const [numeric](#) &n, const [numeric](#) &x)
 - The psi functions (aka polygamma functions).*
 - const [numeric factorial](#) (const [numeric](#) &n)
 - Factorial combinatorial function.*
 - const [numeric doublefactorial](#) (const [numeric](#) &n)

The double factorial combinatorial function.

- `const numeric binomial` (`const numeric &n`, `const numeric &k`)

The Binomial coefficients.

- `const numeric bernoulli` (`const numeric &nn`)

Bernoulli number.

- `const numeric fibonacci` (`const numeric &n`)

Fibonacci number.

- `const numeric abs` (`const numeric &x`)

Absolute value.

- `const numeric mod` (`const numeric &a`, `const numeric &b`)

Modulus (in positive representation).

- `const numeric smod` (`const numeric &a_`, `const numeric &b_`)

Modulus (in symmetric representation).

- `const numeric irem` (`const numeric &a`, `const numeric &b`)

Numeric integer remainder.

- `const numeric irem` (`const numeric &a`, `const numeric &b`, `numeric &q`)

Numeric integer remainder.

- `const numeric iquo` (`const numeric &a`, `const numeric &b`)

Numeric integer quotient.

- `const numeric iquo` (`const numeric &a`, `const numeric &b`, `numeric &r`)

Numeric integer quotient.

- `const numeric gcd` (`const numeric &a`, `const numeric &b`)

Greatest Common Divisor.

- `const numeric lcm` (`const numeric &a`, `const numeric &b`)

Least Common Multiple.

- `const numeric sqrt` (`const numeric &x`)

Numeric square root.

- `const numeric isqrt` (`const numeric &x`)

Integer numeric square root.

- `ex PiEvalf` ()

Floating point evaluation of Archimedes' constant Pi.

- `ex EulerEvalf` ()

Floating point evaluation of Euler's constant gamma.

- `ex CatalanEvalf` ()

Floating point evaluation of Catalan's constant.

- `std::ostream & operator<<` (`std::ostream &os`, `const _numeric_digits &e`)

- `GINAC_DECLARE_UNARCHIVER` (`numeric`)

- `const numeric pow` (`const numeric &x`, `const numeric &y`)

- `const numeric inverse` (`const numeric &x`)

- `numeric step` (`const numeric &x`)

- `int csgn` (`const numeric &x`)

- `bool is_zero` (`const numeric &x`)

- `bool is_positive` (`const numeric &x`)

- `bool is_negative` (`const numeric &x`)

- `bool is_integer` (`const numeric &x`)

- `bool is_pos_integer` (`const numeric &x`)

- `bool is_nonneg_integer` (`const numeric &x`)

- `bool is_even` (`const numeric &x`)

- `bool is_odd` (`const numeric &x`)

- `bool is_prime` (`const numeric &x`)

- `bool is_rational` (`const numeric &x`)

- `bool is_real` (`const numeric &x`)

- bool `is_cinteger` (const `numeric` &`x`)
- bool `is_crational` (const `numeric` &`x`)
- int `to_int` (const `numeric` &`x`)
- long `to_long` (const `numeric` &`x`)
- double `to_double` (const `numeric` &`x`)
- const `numeric` `real` (const `numeric` &`x`)
- const `numeric` `imag` (const `numeric` &`x`)
- const `numeric` `number` (const `numeric` &`x`)
- const `numeric` `denom` (const `numeric` &`x`)
- static const `ex` `exadd` (const `ex` &`lh`, const `ex` &`rh`)
 - Used internally by `operator+()` to add two `ex` objects.*
- static const `ex` `exmul` (const `ex` &`lh`, const `ex` &`rh`)
 - Used internally by `operator*()` to multiply two `ex` objects.*
- static const `ex` `exminus` (const `ex` &`lh`)
 - Used internally by `operator-()` and friends to change the sign of an argument.*
- const `ex` `operator+` (const `ex` &`lh`, const `ex` &`rh`)
- const `ex` `operator-` (const `ex` &`lh`, const `ex` &`rh`)
- const `ex` `operator*` (const `ex` &`lh`, const `ex` &`rh`)
- const `ex` `operator/` (const `ex` &`lh`, const `ex` &`rh`)
- const `numeric` `operator+` (const `numeric` &`lh`, const `numeric` &`rh`)
- const `numeric` `operator-` (const `numeric` &`lh`, const `numeric` &`rh`)
- const `numeric` `operator*` (const `numeric` &`lh`, const `numeric` &`rh`)
- const `numeric` `operator/` (const `numeric` &`lh`, const `numeric` &`rh`)
- `ex` & `operator+=` (`ex` &`lh`, const `ex` &`rh`)
- `ex` & `operator-=` (`ex` &`lh`, const `ex` &`rh`)
- `ex` & `operator*=` (`ex` &`lh`, const `ex` &`rh`)
- `ex` & `operator/=` (`ex` &`lh`, const `ex` &`rh`)
- `numeric` & `operator+=` (`numeric` &`lh`, const `numeric` &`rh`)
- `numeric` & `operator-=` (`numeric` &`lh`, const `numeric` &`rh`)
- `numeric` & `operator*=` (`numeric` &`lh`, const `numeric` &`rh`)
- `numeric` & `operator/=` (`numeric` &`lh`, const `numeric` &`rh`)
- const `ex` `operator+` (const `ex` &`lh`)
- const `ex` `operator-` (const `ex` &`lh`)
- const `numeric` `operator+` (const `numeric` &`lh`)
- const `numeric` `operator-` (const `numeric` &`lh`)
- `ex` & `operator++` (`ex` &`rh`)
 - Expression prefix increment.*
- `ex` & `operator--` (`ex` &`rh`)
 - Expression prefix decrement.*
- const `ex` `operator++` (`ex` &`lh`, int)
 - Expression postfix increment.*
- const `ex` `operator--` (`ex` &`lh`, int)
 - Expression postfix decrement.*
- `numeric` & `operator++` (`numeric` &`rh`)
 - Numeric prefix increment.*
- `numeric` & `operator--` (`numeric` &`rh`)
 - Numeric prefix decrement.*
- const `numeric` `operator++` (`numeric` &`lh`, int)
 - Numeric postfix increment.*
- const `numeric` `operator--` (`numeric` &`lh`, int)
 - Numeric postfix decrement.*
- const `relational` `operator==` (const `ex` &`lh`, const `ex` &`rh`)
- const `relational` `operator!=` (const `ex` &`lh`, const `ex` &`rh`)

- const [relational operator<](#) (const [ex](#) &lh, const [ex](#) &rh)
- const [relational operator<=](#) (const [ex](#) &lh, const [ex](#) &rh)
- const [relational operator>](#) (const [ex](#) &lh, const [ex](#) &rh)
- const [relational operator>=](#) (const [ex](#) &lh, const [ex](#) &rh)
- static int [my_ios_index](#) ()
- static void [my_ios_callback](#) (std::ios_base::event ev, std::ios_base &s, int i)
- static [print_context](#) * [get_print_context](#) (std::ios_base &s)
- static void [set_print_context](#) (std::ios_base &s, const [print_context](#) &c)
- static unsigned [get_print_options](#) (std::ios_base &s)
- static void [set_print_options](#) (std::ostream &s, unsigned [options](#))
- std::ostream & [operator<<](#) (std::ostream &os, const [ex](#) &e)
- std::istream & [operator>>](#) (std::istream &is, [ex](#) &e)
- std::ostream & [dflt](#) (std::ostream &os)
- std::ostream & [latex](#) (std::ostream &os)
- std::ostream & [python](#) (std::ostream &os)
- std::ostream & [python_repr](#) (std::ostream &os)
- std::ostream & [tree](#) (std::ostream &os)
- std::ostream & [csrc](#) (std::ostream &os)
- std::ostream & [csrc_float](#) (std::ostream &os)
- std::ostream & [csrc_double](#) (std::ostream &os)
- std::ostream & [csrc_cl_N](#) (std::ostream &os)
- std::ostream & [index_dimensions](#) (std::ostream &os)
- std::ostream & [no_index_dimensions](#) (std::ostream &os)
- [GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([power](#), [basic](#), print_func< [print_dflt](#) >(&[power::do_print_dflt](#)), print_func< [print_latex](#) >(&[power::do_print_latex](#)), print_func< [print_csrc](#) >(&[power::do_print_csrc](#)), print_func< [print_python](#) >(&[power::do_print_python](#)), print_func< [print_python_repr](#) >(&[power::do_print_python_repr](#)), print_func< [print_csrc_cl_N](#) >(&[power::do_print_csrc_cl_N](#))) [power](#)
- static void [print_sym_pow](#) (const [print_context](#) &c, const [symbol](#) &x, int exp)
- [GINAC_BIND_UNARCHIVER](#) ([power](#))
- [GINAC_DECLARE_UNARCHIVER](#) ([power](#))
- [ex pow](#) (const [ex](#) &b, const [ex](#) &e)
Symbolic exponentiation.
- template<typename T1 , typename T2 >
[ex pow](#) (const T1 &b, const T2 &e)
- [ex sqrt](#) (const [ex](#) &a)
Square root expression.
- template<class T >
bool [is_a](#) (const [print_context](#) &obj)
Check if obj is a T, including base classes.
- [GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([pseries](#), [basic](#), print_func< [print_context](#) >(&[pseries::do_print](#)), print_func< [print_latex](#) >(&[pseries::do_print_latex](#)), print_func< [print_tree](#) >(&[pseries::do_print_tree](#)), print_func< [print_python](#) >(&[pseries::do_print_python](#)), print_func< [print_python_repr](#) >(&[pseries::do_print_python_repr](#))) [pseries](#)
- [GINAC_BIND_UNARCHIVER](#) ([pseries](#))
- [GINAC_DECLARE_UNARCHIVER](#) ([pseries](#))
- [ex series_to_poly](#) (const [ex](#) &e)
Convert the pseries object embedded in an expression to an ordinary polynomial in the expansion variable.
- bool [is_terminating](#) (const [pseries](#) &s)
- template<typename T >
[return_type_t](#) [make_return_type_t](#) (const unsigned rl=0)
- template<class Alg , class Ctx , class T , class C >
void [set_print_func](#) (void f(const T &, const C &c, unsigned))
Add or replace a print method.
- template<class Alg , class Ctx , class T , class C >
void [set_print_func](#) (void(T::*f)(const C &, unsigned))

Add or replace a print method.

- `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`relational`, `basic`, `print_func` < `print_context` > (&`relational::do_print`). `print_func` < `print_tree` > (&`relational::do_print_tree`). `print_func` < `print_python_repr` > (&`relational::do_print_python_repr`) `relational`
- `GINAC_BIND_UNARCHIVER` (`relational`)
- static void `print_operator` (const `print_context` &c, `relational::operators` o)
- `GINAC_DECLARE_UNARCHIVER` (`relational`)
- `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`symbol`, `basic`, `print_func` < `print_context` > (&`symbol::do_print`). `print_func` < `print_latex` > (&`symbol::do_print_latex`). `print_func` < `print_tree` > (&`symbol::do_print_tree`). `print_func` < `print_python_repr` > (&`symbol::do_print_python_repr`) `symbol`
- static const std::string & `get_default_TeX_name` (const std::string &name)

Return default TeX name for symbol.

- `GINAC_BIND_UNARCHIVER` (`symbol`)
- `GINAC_BIND_UNARCHIVER` (`realsymbol`)
- `GINAC_BIND_UNARCHIVER` (`possymbol`)
- `GINAC_DECLARE_UNARCHIVER` (`symbol`)
- `GINAC_DECLARE_UNARCHIVER` (`realsymbol`)
- `GINAC_DECLARE_UNARCHIVER` (`possymbol`)
- `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`symmetry`, `basic`, `print_func` < `print_context` > (&`symmetry::do_print`). `print_func` < `print_tree` > (&`symmetry::do_print_tree`) `symmetry`
- `GINAC_BIND_UNARCHIVER` (`symmetry`)
- static const `symmetry` & `index0` ()
- static const `symmetry` & `index1` ()
- static const `symmetry` & `index2` ()
- static const `symmetry` & `index3` ()
- const `symmetry` & `not_symmetric` ()
- const `symmetry` & `symmetric2` ()
- const `symmetry` & `symmetric3` ()
- const `symmetry` & `symmetric4` ()
- const `symmetry` & `antisymmetric2` ()
- const `symmetry` & `antisymmetric3` ()
- const `symmetry` & `antisymmetric4` ()
- int `canonicalize` (exvector::iterator v, const `symmetry` &symm)

Canonicalize the order of elements of an expression vector, according to the symmetry properties defined in a symmetry tree.

- static `ex symm` (const `ex` &e, exvector::const_iterator first, exvector::const_iterator last, bool asymmetric)
- `ex symmetrize` (const `ex` &e, exvector::const_iterator first, exvector::const_iterator last)

Symmetrize expression over a set of objects (symbols, indices).

- `ex antisymmetrize` (const `ex` &e, exvector::const_iterator first, exvector::const_iterator last)

Antisymmetrize expression over a set of objects (symbols, indices).

- `ex symmetrize_cyclic` (const `ex` &e, exvector::const_iterator first, exvector::const_iterator last)

Symmetrize expression by cyclic permutation over a set of objects (symbols, indices).

- `GINAC_DECLARE_UNARCHIVER` (`symmetry`)
- `symmetry sy_none` ()
- `symmetry sy_none` (const `symmetry` &c1, const `symmetry` &c2)
- `symmetry sy_none` (const `symmetry` &c1, const `symmetry` &c2, const `symmetry` &c3)
- `symmetry sy_none` (const `symmetry` &c1, const `symmetry` &c2, const `symmetry` &c3, const `symmetry` &c4)
- `symmetry sy_symm` ()
- `symmetry sy_symm` (const `symmetry` &c1, const `symmetry` &c2)
- `symmetry sy_symm` (const `symmetry` &c1, const `symmetry` &c2, const `symmetry` &c3)
- `symmetry sy_symm` (const `symmetry` &c1, const `symmetry` &c2, const `symmetry` &c3, const `symmetry` &c4)
- `symmetry sy_anti` ()
- `symmetry sy_anti` (const `symmetry` &c1, const `symmetry` &c2)
- `symmetry sy_anti` (const `symmetry` &c1, const `symmetry` &c2, const `symmetry` &c3)

- [symmetry sy_anti](#) (const [symmetry](#) &c1, const [symmetry](#) &c2, const [symmetry](#) &c3, const [symmetry](#) &c4)
- [symmetry sy_cycl](#) ()
- [symmetry sy_cycl](#) (const [symmetry](#) &c1, const [symmetry](#) &c2)
- [symmetry sy_cycl](#) (const [symmetry](#) &c1, const [symmetry](#) &c2, const [symmetry](#) &c3)
- [symmetry sy_cycl](#) (const [symmetry](#) &c1, const [symmetry](#) &c2, const [symmetry](#) &c3, const [symmetry](#) &c4)
- [ex symmetrize](#) (const [ex](#) &e, const [exvector](#) &v)
Symmetrize expression over a set of objects (symbols, indices).
- [ex antisymmetrize](#) (const [ex](#) &e, const [exvector](#) &v)
Antisymmetrize expression over a set of objects (symbols, indices).
- [ex symmetrize_cyclic](#) (const [ex](#) &e, const [exvector](#) &v)
Symmetrize expression by cyclic permutation over a set of objects (symbols, indices).
- [GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([tensdelta](#), [tensor](#), print_func< [print_dflt](#) >(&[tensdelta::do_print](#)),
print_func< [print_latex](#) >(&[tensdelta::do_print_latex](#))) [GINAC_IMPLEMENT_REGISTERED_CLASS_↵](#)
OPT([tensmetric](#)
- [print_func< print_dflt >](#) (&[tensmetric::do_print](#)). [print_func< print_latex >](#) (&[tensmetric](#)
- [GINAC_BIND_UNARCHIVER](#) ([minkmetric](#))
- [GINAC_BIND_UNARCHIVER](#) ([tensepsilon](#))
- [GINAC_BIND_UNARCHIVER](#) ([tensdelta](#))
- [GINAC_BIND_UNARCHIVER](#) ([tensmetric](#))
- [GINAC_BIND_UNARCHIVER](#) ([spinmetric](#))
- [ex delta_tensor](#) (const [ex](#) &i1, const [ex](#) &i2)
Create a delta tensor with specified indices.
- [ex metric_tensor](#) (const [ex](#) &i1, const [ex](#) &i2)
Create a symmetric metric tensor with specified indices.
- [ex lorentz_g](#) (const [ex](#) &i1, const [ex](#) &i2, bool pos_sig=false)
Create a Minkowski metric tensor with specified indices.
- [ex spinor_metric](#) (const [ex](#) &i1, const [ex](#) &i2)
Create a spinor metric tensor with specified indices.
- [ex epsilon_tensor](#) (const [ex](#) &i1, const [ex](#) &i2)
Create an epsilon tensor in a Euclidean space with two indices.
- [ex epsilon_tensor](#) (const [ex](#) &i1, const [ex](#) &i2, const [ex](#) &i3)
Create an epsilon tensor in a Euclidean space with three indices.
- [ex lorentz_eps](#) (const [ex](#) &i1, const [ex](#) &i2, const [ex](#) &i3, const [ex](#) &i4, bool pos_sig=false)
Create an epsilon tensor in a Minkowski space with four indices.
- [GINAC_DECLARE_UNARCHIVER](#) ([tensdelta](#))
- [GINAC_DECLARE_UNARCHIVER](#) ([tensmetric](#))
- [GINAC_DECLARE_UNARCHIVER](#) ([minkmetric](#))
- [GINAC_DECLARE_UNARCHIVER](#) ([spinmetric](#))
- [GINAC_DECLARE_UNARCHIVER](#) ([tensepsilon](#))
- unsigned [log2](#) (unsigned [n](#))
Integer binary logarithm.
- const [numeric multinomial_coefficient](#) (const std::vector< unsigned > &p)
Compute the multinomial coefficient $n!/(p1!*p2!*...*pk!)$ where $n = p1+p2+...+pk$, i.e.
- unsigned [rotate_left](#) (unsigned [n](#))
Rotate bits of unsigned value by one bit to the left.
- template<class T >
int [compare_pointers](#) (const T *a, const T *b)
Compare two pointers (just to establish some sort of canonical order).
- unsigned [golden_ratio_hash](#) (uintptr_t [n](#))
Truncated multiplication with golden ratio, for computing hash values.
- template<class It >
int [permutation_sign](#) (It first, It last)

- `template<class It , class Cmp , class Swap >`
`int permutation_sign (It first, It last, Cmp comp, Swap swapit)`
- `template<class It , class Cmp , class Swap >`
`void shaker_sort (It first, It last, Cmp comp, Swap swapit)`
- `template<class It , class Swap >`
`void cyclic_permutation (It first, It last, It new_first, Swap swapit)`
- `template<typename T >`
`std::enable_if< has_distance< T >::value, typename std::iterator_traits< T >::difference_type >::type`
`format_index_value (const T &a, const T &b)`
For printing a multi-index: If the templates are used, where T is an iterator, printing the address where the iterator points to is not meaningful.
- `template<typename T >`
`std::enable_if<!has_distance< T >::value, T >::type format_index_value (const T &a, const T &b)`
For all other cases we simply print the value.
- `template<class T >`
`std::ostream & operator<< (std::ostream &os, const basic_multi_iterator< T > &v)`
Output operator.
- `template<class T >`
`std::ostream & operator<< (std::ostream &os, const multi_iterator_ordered< T > &v)`
Output operator.
- `template<class T >`
`std::ostream & operator<< (std::ostream &os, const multi_iterator_ordered_eq< T > &v)`
Output operator.
- `template<class T >`
`std::ostream & operator<< (std::ostream &os, const multi_iterator_ordered_eq_indv< T > &v)`
Output operator.
- `template<class T >`
`std::ostream & operator<< (std::ostream &os, const multi_iterator_counter< T > &v)`
Output operator.
- `template<class T >`
`std::ostream & operator<< (std::ostream &os, const multi_iterator_counter_indv< T > &v)`
Output operator.
- `template<class T >`
`std::ostream & operator<< (std::ostream &os, const multi_iterator_permutation< T > &v)`
Output operator.
- `template<class T >`
`std::ostream & operator<< (std::ostream &os, const multi_iterator_shuffle< T > &v)`
Output operator.
- `template<class T >`
`std::ostream & operator<< (std::ostream &os, const multi_iterator_shuffle_prime< T > &v)`
Output operator.
- `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (wildcard, basic, print_func< print_context >(&wildcard::do_print).`
`print_func< print_tree >(&wildcard::do_print_tree). print_func< print_python_repr >(&wildcard::do_print_python_repr))`
`wildcard`
- `GINAC_BIND_UNARCHIVER (wildcard)`
- `bool haswild (const ex &x)`
Check whether x has a wildcard anywhere as a subexpression.
- `GINAC_DECLARE_UNARCHIVER (wildcard)`
- `ex wild (unsigned label=0)`
Create a wildcard object with the specified label.

Variables

- static `unarchive_table_t` `unarch_table_instance`
- `GiNaC::evalm_map_function` `map_evalm`
- `GiNaC::eval_integ_map_function` `map_eval_integ`
- `tensor`
- const `constant` `Pi` ("Pi", `PiEvalf`, "\\pi", `domain::positive`)
Pi.
- const `constant` `Euler` ("Euler", `EulerEvalf`, "\\gamma_E", `domain::positive`)
Euler's constant.
- const `constant` `Catalan` ("Catalan", `CatalanEvalf`, "G", `domain::positive`)
Catalan's constant.
- static unsigned const `crctab` [256]
- static `library_init` `library_initializer`
For construction of flyweights, etc.
- const `basic` * `_num0_bp`
- `idx`
- unsigned `force_include_tgamma` = `tgamma_SERIAL::serial`
- unsigned `force_include_zeta1` = `zeta1_SERIAL::serial`
- template<> `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT_T`(`lst`, `basic`, `print_func`< `print_context`>(&`lst::do_print`). `print_func`< `print_tree`>(&`lst::do_print_tree`)) template<> bool `lst` `GINAC_BIND_UNARCHIVER`(`lst`)
Specialization of `container::info()` for `lst`.
- const `numeric` `l` = `numeric`(`cln::complex`(`cln::cl_l`(0),`cln::cl_l`(1)))
Imaginary unit.
- `_numeric_digits` `Digits`
Accuracy in decimal digits.
- unsigned `next_print_context_id` = 0
Next free ID for `print_context` types.
- const int `version_major` = `GINACLIB_MAJOR_VERSION`
- const int `version_minor` = `GINACLIB_MINOR_VERSION`
- const int `version_micro` = `GINACLIB_MICRO_VERSION`
- const `numeric` * `_num_120_p`
- const `ex_ex_120` = `ex`(* `_num_120_p`)
- const `numeric` * `_num_60_p`
- const `ex_ex_60` = `ex`(* `_num_60_p`)
- const `numeric` * `_num_48_p`
- const `ex_ex_48` = `ex`(* `_num_48_p`)
- const `numeric` * `_num_30_p`
- const `ex_ex_30` = `ex`(* `_num_30_p`)
- const `numeric` * `_num_25_p`
- const `ex_ex_25` = `ex`(* `_num_25_p`)
- const `numeric` * `_num_24_p`
- const `ex_ex_24` = `ex`(* `_num_24_p`)
- const `numeric` * `_num_20_p`
- const `ex_ex_20` = `ex`(* `_num_20_p`)
- const `numeric` * `_num_18_p`
- const `ex_ex_18` = `ex`(* `_num_18_p`)
- const `numeric` * `_num_15_p`
- const `ex_ex_15` = `ex`(* `_num_15_p`)
- const `numeric` * `_num_12_p`
- const `ex_ex_12` = `ex`(* `_num_12_p`)
- const `numeric` * `_num_11_p`
- const `ex_ex_11` = `ex`(* `_num_11_p`)

- const [numeric](#) * [_num_10_p](#)
- const [ex](#) [_ex_10](#) = [ex](#)(*[_num_10_p](#))
- const [numeric](#) * [_num_9_p](#)
- const [ex](#) [_ex_9](#) = [ex](#)(*[_num_9_p](#))
- const [numeric](#) * [_num_8_p](#)
- const [ex](#) [_ex_8](#) = [ex](#)(*[_num_8_p](#))
- const [numeric](#) * [_num_7_p](#)
- const [ex](#) [_ex_7](#) = [ex](#)(*[_num_7_p](#))
- const [numeric](#) * [_num_6_p](#)
- const [ex](#) [_ex_6](#) = [ex](#)(*[_num_6_p](#))
- const [numeric](#) * [_num_5_p](#)
- const [ex](#) [_ex_5](#) = [ex](#)(*[_num_5_p](#))
- const [numeric](#) * [_num_4_p](#)
- const [ex](#) [_ex_4](#) = [ex](#)(*[_num_4_p](#))
- const [numeric](#) * [_num_3_p](#)
- const [ex](#) [_ex_3](#) = [ex](#)(*[_num_3_p](#))
- const [numeric](#) * [_num_2_p](#)
- const [ex](#) [_ex_2](#) = [ex](#)(*[_num_2_p](#))
- const [numeric](#) * [_num_1_p](#)
- const [ex](#) [_ex_1](#) = [ex](#)(*[_num_1_p](#))
- const [numeric](#) * [_num_1_2_p](#)
- const [ex](#) [_ex_1_2](#) = [ex](#)(*[_num_1_2_p](#))
- const [numeric](#) * [_num_1_3_p](#)
- const [ex](#) [_ex_1_3](#) = [ex](#)(*[_num_1_3_p](#))
- const [numeric](#) * [_num_1_4_p](#)
- const [ex](#) [_ex_1_4](#) = [ex](#)(*[_num_1_4_p](#))
- const [numeric](#) * [_num0_p](#)
- const [ex](#) [_ex0](#) = [ex](#)(*[_num0_p](#))
- const [numeric](#) * [_num1_4_p](#)
- const [ex](#) [_ex1_4](#) = [ex](#)(*[_num1_4_p](#))
- const [numeric](#) * [_num1_3_p](#)
- const [ex](#) [_ex1_3](#) = [ex](#)(*[_num1_3_p](#))
- const [numeric](#) * [_num1_2_p](#)
- const [ex](#) [_ex1_2](#) = [ex](#)(*[_num1_2_p](#))
- const [numeric](#) * [_num1_p](#)
- const [ex](#) [_ex1](#) = [ex](#)(*[_num1_p](#))
- const [numeric](#) * [_num2_p](#)
- const [ex](#) [_ex2](#) = [ex](#)(*[_num2_p](#))
- const [numeric](#) * [_num3_p](#)
- const [ex](#) [_ex3](#) = [ex](#)(*[_num3_p](#))
- const [numeric](#) * [_num4_p](#)
- const [ex](#) [_ex4](#) = [ex](#)(*[_num4_p](#))
- const [numeric](#) * [_num5_p](#)
- const [ex](#) [_ex5](#) = [ex](#)(*[_num5_p](#))
- const [numeric](#) * [_num6_p](#)
- const [ex](#) [_ex6](#) = [ex](#)(*[_num6_p](#))
- const [numeric](#) * [_num7_p](#)
- const [ex](#) [_ex7](#) = [ex](#)(*[_num7_p](#))
- const [numeric](#) * [_num8_p](#)
- const [ex](#) [_ex8](#) = [ex](#)(*[_num8_p](#))
- const [numeric](#) * [_num9_p](#)
- const [ex](#) [_ex9](#) = [ex](#)(*[_num9_p](#))
- const [numeric](#) * [_num10_p](#)
- const [ex](#) [_ex10](#) = [ex](#)(*[_num10_p](#))
- const [numeric](#) * [_num11_p](#)

- const `ex_ex11` = `ex(*_num11_p)`
- const `numeric *_num12_p`
- const `ex_ex12` = `ex(*_num12_p)`
- const `numeric *_num15_p`
- const `ex_ex15` = `ex(*_num15_p)`
- const `numeric *_num18_p`
- const `ex_ex18` = `ex(*_num18_p)`
- const `numeric *_num20_p`
- const `ex_ex20` = `ex(*_num20_p)`
- const `numeric *_num24_p`
- const `ex_ex24` = `ex(*_num24_p)`
- const `numeric *_num25_p`
- const `ex_ex25` = `ex(*_num25_p)`
- const `numeric *_num30_p`
- const `ex_ex30` = `ex(*_num30_p)`
- const `numeric *_num48_p`
- const `ex_ex48` = `ex(*_num48_p)`
- const `numeric *_num60_p`
- const `ex_ex60` = `ex(*_num60_p)`
- const `numeric *_num120_p`
- const `ex_ex120` = `ex(*_num120_p)`

5.1.1 Typedef Documentation

5.1.1.1 `archive_node_id`

```
typedef unsigned GiNaC::archive_node_id
```

Numerical ID value to refer to an [archive_node](#).

5.1.1.2 `archive_atom`

```
typedef unsigned GiNaC::archive_atom
```

Numerical ID value to refer to a string.

5.1.1.3 `synthesize_func`

```
typedef basic *(* GiNaC::synthesize_func) ()
```

5.1.1.4 `unarchive_map_t`

```
typedef std::map<std::string, synthesize_func> GiNaC::unarchive_map_t
```

5.1.1.5 `exvector`

```
typedef std::vector<ex> GiNaC::exvector
```


5.1.1.6 exset

```
typedef std::set<ex, ex_is_less> GiNaC::exset
```

5.1.1.7 exmap

```
typedef std::map<ex, ex, ex_is_less> GiNaC::exmap
```

5.1.1.8 evalffunctype

```
typedef ex(* GiNaC::evalffunctype) ()
```

5.1.1.9 FUNCP_1P

```
typedef double(* GiNaC::FUNCP_1P) (double)
```

Function pointer with one function parameter.

5.1.1.10 FUNCP_2P

```
typedef double(* GiNaC::FUNCP_2P) (double, double)
```

Function pointer with two function parameters.

5.1.1.11 FUNCP_CUBA

```
typedef void(* GiNaC::FUNCP_CUBA) (const int *, const double[], const int *, double[])
```

Function pointer for use with the CUBA library (<http://www.feynarts.de/cuba>).

5.1.1.12 epvector

```
typedef std::vector<expair> GiNaC::epvector
```

expair-vector

5.1.1.13 epp

```
typedef epvector::iterator GiNaC::epp
```

expair-vector pointer

5.1.1.14 `exprseq`

```
typedef container<std::vector> GiNaC::exprseq
```

5.1.1.15 `paramset`

```
typedef std::multiset<unsigned> GiNaC::paramset
```

5.1.1.16 `eval_funcp`

```
typedef ex(* GiNaC::eval\_funcp) ()
```

5.1.1.17 `evalf_funcp`

```
typedef ex(* GiNaC::evalf\_funcp) ()
```

5.1.1.18 `conjugate_funcp`

```
typedef ex(* GiNaC::conjugate\_funcp) ()
```

5.1.1.19 `real_part_funcp`

```
typedef ex(* GiNaC::real\_part\_funcp) ()
```

5.1.1.20 `imag_part_funcp`

```
typedef ex(* GiNaC::imag\_part\_funcp) ()
```

5.1.1.21 `expand_funcp`

```
typedef ex(* GiNaC::expand\_funcp) ()
```

5.1.1.22 `derivative_funcp`

```
typedef ex(* GiNaC::derivative\_funcp) ()
```

5.1.1.23 `expl_derivative_funcp`

```
typedef ex(* GiNaC::expl\_derivative\_funcp) ()
```

5.1.1.24 power_funcp

```
typedef ex(* GiNaC::power_funcp) ()
```

5.1.1.25 series_funcp

```
typedef ex(* GiNaC::series_funcp) ()
```

5.1.1.26 print_funcp

```
typedef void(* GiNaC::print_funcp) ()
```

5.1.1.27 info_funcp

```
typedef bool(* GiNaC::info_funcp) ()
```

5.1.1.28 eval_funcp_1

```
typedef ex(* GiNaC::eval_funcp_1) (const ex &)
```

5.1.1.29 evalf_funcp_1

```
typedef ex(* GiNaC::evalf_funcp_1) (const ex &)
```

5.1.1.30 conjugate_funcp_1

```
typedef ex(* GiNaC::conjugate_funcp_1) (const ex &)
```

5.1.1.31 real_part_funcp_1

```
typedef ex(* GiNaC::real_part_funcp_1) (const ex &)
```

5.1.1.32 imag_part_funcp_1

```
typedef ex(* GiNaC::imag_part_funcp_1) (const ex &)
```

5.1.1.33 expand_funcp_1

```
typedef ex(* GiNaC::expand_funcp_1) (const ex &, unsigned)
```

5.1.1.34 derivative_funcp_1

```
typedef ex(* GiNaC::derivative_funcp_1) (const ex &, unsigned)
```

5.1.1.35 expl_derivative_funcp_1

```
typedef ex(* GiNaC::expl_derivative_funcp_1) (const ex &, const symbol &)
```

5.1.1.36 power_funcp_1

```
typedef ex(* GiNaC::power_funcp_1) (const ex &, const ex &)
```

5.1.1.37 series_funcp_1

```
typedef ex(* GiNaC::series_funcp_1) (const ex &, const relational &, int, unsigned)
```

5.1.1.38 print_funcp_1

```
typedef void(* GiNaC::print_funcp_1) (const ex &, const print\_context &)
```

5.1.1.39 info_funcp_1

```
typedef bool(* GiNaC::info_funcp_1) (const ex &, unsigned)
```

5.1.1.40 eval_funcp_2

```
typedef ex(* GiNaC::eval_funcp_2) (const ex &, const ex &)
```

5.1.1.41 evalf_funcp_2

```
typedef ex(* GiNaC::evalf_funcp_2) (const ex &, const ex &)
```

5.1.1.42 conjugate_funcp_2

```
typedef ex(* GiNaC::conjugate_funcp_2) (const ex &, const ex &)
```

5.1.1.43 real_part_funcp_2

```
typedef ex(* GiNaC::real_part_funcp_2) (const ex &, const ex &)
```

5.1.1.44 imag_part_funcp_2

```
typedef ex(* GiNaC::imag_part_funcp_2) (const ex &, const ex &)
```

5.1.1.45 expand_funcp_2

```
typedef ex(* GiNaC::expand_funcp_2) (const ex &, const ex &, unsigned)
```

5.1.1.46 derivative_funcp_2

```
typedef ex(* GiNaC::derivative_funcp_2) (const ex &, const ex &, unsigned)
```

5.1.1.47 expl_derivative_funcp_2

```
typedef ex(* GiNaC::expl_derivative_funcp_2) (const ex &, const ex &, const symbol &)
```

5.1.1.48 power_funcp_2

```
typedef ex(* GiNaC::power_funcp_2) (const ex &, const ex &, const ex &)
```

5.1.1.49 series_funcp_2

```
typedef ex(* GiNaC::series_funcp_2) (const ex &, const ex &, const relational &, int, unsigned)
```

5.1.1.50 print_funcp_2

```
typedef void(* GiNaC::print_funcp_2) (const ex &, const ex &, const print_context &)
```

5.1.1.51 info_funcp_2

```
typedef bool(* GiNaC::info_funcp_2) (const ex &, const ex &, unsigned)
```

5.1.1.52 eval_funcp_3

```
typedef ex(* GiNaC::eval_funcp_3) (const ex &, const ex &, const ex &)
```

5.1.1.53 evalf_funcp_3

```
typedef ex(* GiNaC::evalf_funcp_3) (const ex &, const ex &, const ex &)
```

5.1.1.54 conjugate_funcp_3

```
typedef ex(* GiNaC::conjugate_funcp_3) (const ex &, const ex &, const ex &)
```

5.1.1.55 real_part_funcp_3

```
typedef ex(* GiNaC::real_part_funcp_3) (const ex &, const ex &, const ex &)
```

5.1.1.56 imag_part_funcp_3

```
typedef ex(* GiNaC::imag_part_funcp_3) (const ex &, const ex &, const ex &)
```

5.1.1.57 expand_funcp_3

```
typedef ex(* GiNaC::expand_funcp_3) (const ex &, const ex &, const ex &, unsigned)
```

5.1.1.58 derivative_funcp_3

```
typedef ex(* GiNaC::derivative_funcp_3) (const ex &, const ex &, const ex &, unsigned)
```

5.1.1.59 expl_derivative_funcp_3

```
typedef ex(* GiNaC::expl_derivative_funcp_3) (const ex &, const ex &, const ex &, const symbol  
&)
```

5.1.1.60 power_funcp_3

```
typedef ex(* GiNaC::power_funcp_3) (const ex &, const ex &, const ex &, const ex &)
```

5.1.1.61 series_funcp_3

```
typedef ex(* GiNaC::series_funcp_3) (const ex &, const ex &, const ex &, const relational &, int, unsigned)
```

5.1.1.62 print_funcp_3

```
typedef void(* GiNaC::print_funcp_3) (const ex &, const ex &, const ex &, const print_context  
&)
```

5.1.1.63 info_funcp_3

```
typedef bool(* GiNaC::info_funcp_3) (const ex &, const ex &, const ex &, unsigned)
```

5.1.1.64 eval_funcp_4

```
typedef ex(* GiNaC::eval_funcp_4) (const ex &, const ex &, const ex &, const ex &)
```

5.1.1.65 evalf_funcp_4

```
typedef ex(* GiNaC::evalf_funcp_4) (const ex &, const ex &, const ex &, const ex &)
```

5.1.1.66 conjugate_funcp_4

```
typedef ex(* GiNaC::conjugate_funcp_4) (const ex &, const ex &, const ex &, const ex &)
```

5.1.1.67 real_part_funcp_4

```
typedef ex(* GiNaC::real_part_funcp_4) (const ex &, const ex &, const ex &, const ex &)
```

5.1.1.68 imag_part_funcp_4

```
typedef ex(* GiNaC::imag_part_funcp_4) (const ex &, const ex &, const ex &, const ex &)
```

5.1.1.69 expand_funcp_4

```
typedef ex(* GiNaC::expand_funcp_4) (const ex &, const ex &, const ex &, const ex &, unsigned)
```

5.1.1.70 derivative_funcp_4

```
typedef ex(* GiNaC::derivative_funcp_4) (const ex &, const ex &, const ex &, const ex &, unsigned)
```

5.1.1.71 expl_derivative_funcp_4

```
typedef ex(* GiNaC::expl_derivative_funcp_4) (const ex &, const ex &, const ex &, const ex &,  
const symbol &)
```

5.1.1.72 power_funcp_4

```
typedef ex(* GiNaC::power_funcp_4) (const ex &, const ex &, const ex &, const ex &, const ex  
&)
```

5.1.1.73 series_funcp_4

```
typedef ex(* GiNaC::series_funcp_4) (const ex &, const ex &, const ex &, const ex &, const  
relational &, int, unsigned)
```

5.1.1.74 print_funcp_4

```
typedef void(* GiNaC::print_funcp_4) (const ex &, const ex &, const ex &, const ex &, const  
print_context &)
```

5.1.1.75 info_funcp_4

```
typedef bool(* GiNaC::info_funcp_4) (const ex &, const ex &, const ex &, const ex &, unsigned)
```

5.1.1.76 eval_funcp_5

```
typedef ex(* GiNaC::eval_funcp_5) (const ex &, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.77 evalf_funcp_5

```
typedef ex(* GiNaC::evalf_funcp_5) (const ex &, const ex &, const ex &, const ex &, const ex  
&)
```

5.1.1.78 conjugate_funcp_5

```
typedef ex(* GiNaC::conjugate_funcp_5) (const ex &, const ex &, const ex &, const ex &, const  
ex &)
```

5.1.1.79 real_part_funcp_5

```
typedef ex(* GiNaC::real_part_funcp_5) (const ex &, const ex &, const ex &, const ex &, const  
ex &)
```

5.1.1.80 imag_part_funcp_5

```
typedef ex(* GiNaC::imag_part_funcp_5) (const ex &, const ex &, const ex &, const ex &, const  
ex &)
```

5.1.1.81 expand_funcp_5

```
typedef ex(* GiNaC::expand_funcp_5) (const ex &, const ex &, const ex &, const ex &, const ex  
&, unsigned)
```

5.1.1.82 derivative_funcp_5

```
typedef ex(* GiNaC::derivative_funcp_5) (const ex &, const ex &, const ex &, const ex &, const  
ex &, unsigned)
```


5.1.1.83 expl_derivative_funcp_5

```
typedef ex(* GiNaC::expl_derivative_funcp_5) (const ex &, const ex &, const ex &, const ex &,
const ex &, const symbol &)
```

5.1.1.84 power_funcp_5

```
typedef ex(* GiNaC::power_funcp_5) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &)
```

5.1.1.85 series_funcp_5

```
typedef ex(* GiNaC::series_funcp_5) (const ex &, const ex &, const ex &, const ex &, const ex
&, const relational &, int, unsigned)
```

5.1.1.86 print_funcp_5

```
typedef void(* GiNaC::print_funcp_5) (const ex &, const ex &, const ex &, const ex &, const ex
&, const print_context &)
```

5.1.1.87 info_funcp_5

```
typedef bool(* GiNaC::info_funcp_5) (const ex &, const ex &, const ex &, const ex &, const ex
&, unsigned)
```

5.1.1.88 eval_funcp_6

```
typedef ex(* GiNaC::eval_funcp_6) (const ex &, const ex &, const ex &, const ex &, const ex &,
const ex &)
```

5.1.1.89 evalf_funcp_6

```
typedef ex(* GiNaC::evalf_funcp_6) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &)
```

5.1.1.90 conjugate_funcp_6

```
typedef ex(* GiNaC::conjugate_funcp_6) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &)
```

5.1.1.91 real_part_funcp_6

```
typedef ex(* GiNaC::real_part_funcp_6) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &)
```

5.1.1.92 imag_funcp_6

```
typedef ex(* GiNaC::imag_funcp_6) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &)
```

5.1.1.93 expand_funcp_6

```
typedef ex(* GiNaC::expand_funcp_6) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, unsigned)
```

5.1.1.94 derivative_funcp_6

```
typedef ex(* GiNaC::derivative_funcp_6) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, unsigned)
```

5.1.1.95 expl_derivative_funcp_6

```
typedef ex(* GiNaC::expl_derivative_funcp_6) (const ex &, const ex &, const ex &, const ex &,
const ex &, const ex &, const symbol &)
```

5.1.1.96 power_funcp_6

```
typedef ex(* GiNaC::power_funcp_6) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &)
```

5.1.1.97 series_funcp_6

```
typedef ex(* GiNaC::series_funcp_6) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const relational &, int, unsigned)
```

5.1.1.98 print_funcp_6

```
typedef void(* GiNaC::print_funcp_6) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const print_context &)
```

5.1.1.99 info_funcp_6

```
typedef bool(* GiNaC::info_funcp_6) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, unsigned)
```

5.1.1.100 eval_funcp_7

```
typedef ex(* GiNaC::eval_funcp_7) (const ex &, const ex &, const ex &, const ex &, const ex &,
const ex &, const ex &)
```

5.1.1.101 evalf_funcp_7

```
typedef ex(* GiNaC::evalf_funcp_7) (const ex &, const ex &, const ex &, const ex &, const ex  
&, const ex &, const ex &)
```

5.1.1.102 conjugate_funcp_7

```
typedef ex(* GiNaC::conjugate_funcp_7) (const ex &, const ex &, const ex &, const ex &, const  
ex &, const ex &, const ex &)
```

5.1.1.103 real_part_funcp_7

```
typedef ex(* GiNaC::real_part_funcp_7) (const ex &, const ex &, const ex &, const ex &, const  
ex &, const ex &, const ex &)
```

5.1.1.104 imag_part_funcp_7

```
typedef ex(* GiNaC::imag_part_funcp_7) (const ex &, const ex &, const ex &, const ex &, const  
ex &, const ex &, const ex &)
```

5.1.1.105 expand_funcp_7

```
typedef ex(* GiNaC::expand_funcp_7) (const ex &, const ex &, const ex &, const ex &, const ex  
&, const ex &, const ex &, unsigned)
```

5.1.1.106 derivative_funcp_7

```
typedef ex(* GiNaC::derivative_funcp_7) (const ex &, const ex &, const ex &, const ex &, const  
ex &, const ex &, const ex &, unsigned)
```

5.1.1.107 expl_derivative_funcp_7

```
typedef ex(* GiNaC::expl_derivative_funcp_7) (const ex &, const ex &, const ex &, const ex &,  
const ex &, const ex &, const ex &, const symbol &)
```

5.1.1.108 power_funcp_7

```
typedef ex(* GiNaC::power_funcp_7) (const ex &, const ex &, const ex &, const ex &, const ex  
&, const ex &, const ex &, const ex &)
```

5.1.1.109 series_funcp_7

```
typedef ex(* GiNaC::series_funcp_7) (const ex &, const ex &, const ex &, const ex &, const ex  
&, const ex &, const ex &, const relational &, int, unsigned)
```

5.1.1.110 print_funcp_7

```
typedef void(* GiNaC::print_funcp_7) (const ex &, const ex &, const ex &, const ex &, const ex  
&, const ex &, const ex &, const print_context &)
```

5.1.1.111 info_funcp_7

```
typedef bool(* GiNaC::info_funcp_7) (const ex &, const ex &, const ex &, const ex &, const ex  
&, const ex &, const ex &, unsigned)
```

5.1.1.112 eval_funcp_8

```
typedef ex(* GiNaC::eval_funcp_8) (const ex &, const ex &, const ex &, const ex &, const ex &,  
const ex &, const ex &, const ex &)
```

5.1.1.113 evalf_funcp_8

```
typedef ex(* GiNaC::evalf_funcp_8) (const ex &, const ex &, const ex &, const ex &, const ex  
&, const ex &, const ex &, const ex &)
```

5.1.1.114 conjugate_funcp_8

```
typedef ex(* GiNaC::conjugate_funcp_8) (const ex &, const ex &, const ex &, const ex &, const  
ex &, const ex &, const ex &, const ex &)
```

5.1.1.115 real_part_funcp_8

```
typedef ex(* GiNaC::real_part_funcp_8) (const ex &, const ex &, const ex &, const ex &, const  
ex &, const ex &, const ex &, const ex &)
```

5.1.1.116 imag_part_funcp_8

```
typedef ex(* GiNaC::imag_part_funcp_8) (const ex &, const ex &, const ex &, const ex &, const  
ex &, const ex &, const ex &, const ex &)
```

5.1.1.117 expand_funcp_8

```
typedef ex(* GiNaC::expand_funcp_8) (const ex &, const ex &, const ex &, const ex &, const ex  
&, const ex &, const ex &, const ex &, unsigned)
```

5.1.1.118 derivative_funcp_8

```
typedef ex(* GiNaC::derivative_funcp_8) (const ex &, const ex &, const ex &, const ex &, const  
ex &, const ex &, const ex &, const ex &, unsigned)
```

5.1.1.119 expl_derivative_funcp_8

```
typedef ex(* GiNaC::expl_derivative_funcp_8) (const ex &, const ex &, const ex &, const ex &,
const ex &, const ex &, const ex &, const ex &, const symbol &)
```

5.1.1.120 power_funcp_8

```
typedef ex(* GiNaC::power_funcp_8) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.121 series_funcp_8

```
typedef ex(* GiNaC::series_funcp_8) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const relational &, int, unsigned)
```

5.1.1.122 print_funcp_8

```
typedef void(* GiNaC::print_funcp_8) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const print_context &)
```

5.1.1.123 info_funcp_8

```
typedef bool(* GiNaC::info_funcp_8) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, unsigned)
```

5.1.1.124 eval_funcp_9

```
typedef ex(* GiNaC::eval_funcp_9) (const ex &, const ex &, const ex &, const ex &, const ex &,
const ex &, const ex &, const ex &, const ex &)
```

5.1.1.125 evalf_funcp_9

```
typedef ex(* GiNaC::evalf_funcp_9) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.126 conjugate_funcp_9

```
typedef ex(* GiNaC::conjugate_funcp_9) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.127 real_part_funcp_9

```
typedef ex(* GiNaC::real_part_funcp_9) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.128 imag_part_funcp_9

```
typedef ex(* GiNaC::imag_part_funcp_9) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.129 expand_funcp_9

```
typedef ex(* GiNaC::expand_funcp_9) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, unsigned)
```

5.1.1.130 derivative_funcp_9

```
typedef ex(* GiNaC::derivative_funcp_9) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const ex &, const ex &, const ex &, unsigned)
```

5.1.1.131 expl_derivative_funcp_9

```
typedef ex(* GiNaC::expl_derivative_funcp_9) (const ex &, const ex &, const ex &, const ex &,
const ex &, const ex &, const ex &, const ex &, const ex &, const symbol &)
```

5.1.1.132 power_funcp_9

```
typedef ex(* GiNaC::power_funcp_9) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.133 series_funcp_9

```
typedef ex(* GiNaC::series_funcp_9) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, const relational &, int, unsigned)
```

5.1.1.134 print_funcp_9

```
typedef void(* GiNaC::print_funcp_9) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, const print_context &)
```

5.1.1.135 info_funcp_9

```
typedef bool(* GiNaC::info_funcp_9) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, unsigned)
```

5.1.1.136 eval_funcp_10

```
typedef ex(* GiNaC::eval_funcp_10) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.137 evalf_funcp_10

```
typedef ex(* GiNaC::evalf_funcp_10) (const ex &, const ex &, const ex &, const ex &, const ex  
&, const ex &, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.138 conjugate_funcp_10

```
typedef ex(* GiNaC::conjugate_funcp_10) (const ex &, const ex &, const ex &, const ex &, const  
ex &, const ex &, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.139 real_part_funcp_10

```
typedef ex(* GiNaC::real_part_funcp_10) (const ex &, const ex &, const ex &, const ex &, const  
ex &, const ex &, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.140 imag_part_funcp_10

```
typedef ex(* GiNaC::imag_part_funcp_10) (const ex &, const ex &, const ex &, const ex &, const  
ex &, const ex &, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.141 expand_funcp_10

```
typedef ex(* GiNaC::expand_funcp_10) (const ex &, const ex &, const ex &, const ex &, const ex  
&, const ex &, const ex &, const ex &, const ex &, const ex &, unsigned)
```

5.1.1.142 derivative_funcp_10

```
typedef ex(* GiNaC::derivative_funcp_10) (const ex &, const ex &, const ex &, const ex &,  
const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, unsigned)
```

5.1.1.143 expl_derivative_funcp_10

```
typedef ex(* GiNaC::expl_derivative_funcp_10) (const ex &, const ex &, const ex &, const ex &,  
const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const symbol &)
```

5.1.1.144 power_funcp_10

```
typedef ex(* GiNaC::power_funcp_10) (const ex &, const ex &, const ex &, const ex &, const ex  
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.145 series_funcp_10

```
typedef ex(* GiNaC::series_funcp_10) (const ex &, const ex &, const ex &, const ex &, const  
ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const relational &, int,  
unsigned)
```

5.1.1.146 print_funcp_10

```
typedef void(* GiNaC::print_funcp_10) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const print_context &)
```

5.1.1.147 info_funcp_10

```
typedef bool(* GiNaC::info_funcp_10) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, const ex &, unsigned)
```

5.1.1.148 eval_funcp_11

```
typedef ex(* GiNaC::eval_funcp_11) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.149 evalf_funcp_11

```
typedef ex(* GiNaC::evalf_funcp_11) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.150 conjugate_funcp_11

```
typedef ex(* GiNaC::conjugate_funcp_11) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.151 real_part_funcp_11

```
typedef ex(* GiNaC::real_part_funcp_11) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.152 imag_part_funcp_11

```
typedef ex(* GiNaC::imag_part_funcp_11) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.153 expand_funcp_11

```
typedef ex(* GiNaC::expand_funcp_11) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, unsigned)
```

5.1.1.154 derivative_funcp_11

```
typedef ex(* GiNaC::derivative_funcp_11) (const ex &, const ex &, const ex &, const ex &,
const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, unsigned)
```


5.1.1.155 expl_derivative_funcp_11

```
typedef ex(* GiNaC::expl_derivative_funcp_11) (const ex &, const ex &, const ex &, const ex  
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const  
symbol &)
```

5.1.1.156 power_funcp_11

```
typedef ex(* GiNaC::power_funcp_11) (const ex &, const ex &, const ex &, const ex &, const ex  
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.157 series_funcp_11

```
typedef ex(* GiNaC::series_funcp_11) (const ex &, const ex &, const ex &, const ex &, const ex  
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const relational &,  
int, unsigned)
```

5.1.1.158 print_funcp_11

```
typedef void(* GiNaC::print_funcp_11) (const ex &, const ex &, const ex &, const ex &, const  
ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const print_context  
&)
```

5.1.1.159 info_funcp_11

```
typedef bool(* GiNaC::info_funcp_11) (const ex &, const ex &, const ex &, const ex &, const ex  
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, unsigned)
```

5.1.1.160 eval_funcp_12

```
typedef ex(* GiNaC::eval_funcp_12) (const ex &, const ex &, const ex &, const ex &, const ex  
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.161 evalf_funcp_12

```
typedef ex(* GiNaC::evalf_funcp_12) (const ex &, const ex &, const ex &, const ex &, const ex  
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.162 conjugate_funcp_12

```
typedef ex(* GiNaC::conjugate_funcp_12) (const ex &, const ex &, const ex &, const ex &, const  
ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.163 real_part_funcp_12

```
typedef ex(* GiNaC::real_part_funcp_12) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.164 imag_part_funcp_12

```
typedef ex(* GiNaC::imag_part_funcp_12) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &)
```

5.1.1.165 expand_funcp_12

```
typedef ex(* GiNaC::expand_funcp_12) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &,
unsigned)
```

5.1.1.166 derivative_funcp_12

```
typedef ex(* GiNaC::derivative_funcp_12) (const ex &, const ex &, const ex &, const ex &,
const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex
&, unsigned)
```

5.1.1.167 expl_derivative_funcp_12

```
typedef ex(* GiNaC::expl_derivative_funcp_12) (const ex &, const ex &, const ex &, const ex &,
const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex
&, const symbol &)
```

5.1.1.168 power_funcp_12

```
typedef ex(* GiNaC::power_funcp_12) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const
ex &)
```

5.1.1.169 series_funcp_12

```
typedef ex(* GiNaC::series_funcp_12) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const
relational &, int, unsigned)
```

5.1.1.170 print_funcp_12

```
typedef void(* GiNaC::print_funcp_12) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &,
const print_context &)
```

5.1.1.171 info_funcp_12

```
typedef bool(* GiNaC::info_funcp_12) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &,
unsigned)
```

5.1.1.172 eval_funcp_13

```
typedef ex(* GiNaC::eval_funcp_13) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const
ex &)
```

5.1.1.173 evalf_funcp_13

```
typedef ex(* GiNaC::evalf_funcp_13) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const
ex &)
```

5.1.1.174 conjugate_funcp_13

```
typedef ex(* GiNaC::conjugate_funcp_13) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &,
const ex &)
```

5.1.1.175 real_part_funcp_13

```
typedef ex(* GiNaC::real_part_funcp_13) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &,
const ex &)
```

5.1.1.176 imag_part_funcp_13

```
typedef ex(* GiNaC::imag_part_funcp_13) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &,
const ex &)
```

5.1.1.177 expand_funcp_13

```
typedef ex(* GiNaC::expand_funcp_13) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const
ex &, unsigned)
```

5.1.1.178 derivative_funcp_13

```
typedef ex(* GiNaC::derivative_funcp_13) (const ex &, const ex &, const ex &, const ex &,
const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, unsigned)
```

5.1.1.179 expl_derivative_funcp_13

```
typedef ex(* GiNaC::expl_derivative_funcp_13) (const ex &, const ex &, const ex &, const ex &,
const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const symbol &)
```

5.1.1.180 power_funcp_13

```
typedef ex(* GiNaC::power_funcp_13) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &)
```

5.1.1.181 series_funcp_13

```
typedef ex(* GiNaC::series_funcp_13) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const
ex &, const relational &, int, unsigned)
```

5.1.1.182 print_funcp_13

```
typedef void(* GiNaC::print_funcp_13) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &,
const ex &, const print_context &)
```

5.1.1.183 info_funcp_13

```
typedef bool(* GiNaC::info_funcp_13) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const
ex &, unsigned)
```

5.1.1.184 eval_funcp_14

```
typedef ex(* GiNaC::eval_funcp_14) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &)
```

5.1.1.185 evalf_funcp_14

```
typedef ex(* GiNaC::evalf_funcp_14) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &)
```

5.1.1.186 conjugate_funcp_14

```
typedef ex(* GiNaC::conjugate_funcp_14) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &,
const ex &, const ex &)
```

5.1.1.187 real_part_funcp_14

```
typedef ex(* GiNaC::real_part_funcp_14) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &,
const ex &, const ex &)
```

5.1.1.188 imag_part_funcp_14

```
typedef ex(* GiNaC::imag_part_funcp_14) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &,
const ex &, const ex &)
```

5.1.1.189 expand_funcp_14

```
typedef ex(* GiNaC::expand_funcp_14) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, unsigned)
```

5.1.1.190 derivative_funcp_14

```
typedef ex(* GiNaC::derivative_funcp_14) (const ex &, const ex &, const ex &, const ex &,
const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, unsigned)
```

5.1.1.191 expl_derivative_funcp_14

```
typedef ex(* GiNaC::expl_derivative_funcp_14) (const ex &, const ex &, const ex &, const ex &,
const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const symbol &)
```

5.1.1.192 power_funcp_14

```
typedef ex(* GiNaC::power_funcp_14) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const ex &)
```

5.1.1.193 series_funcp_14

```
typedef ex(* GiNaC::series_funcp_14) (const ex &, const ex &, const ex &, const ex &, const ex
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const relational &, int, unsigned)
```

5.1.1.194 print_funcp_14

```
typedef void(* GiNaC::print_funcp_14) (const ex &, const ex &, const ex &, const ex &, const
ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &,
const ex &, const ex &, const print_context &)
```

5.1.1.195 info_funcp_14

```
typedef bool(* GiNaC::info_funcp_14) (const ex &, const ex &, const ex &, const ex &, const ex  
&, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const ex &, const  
ex &, const ex &, unsigned)
```

5.1.1.196 eval_funcp_exvector

```
typedef ex(* GiNaC::eval_funcp_exvector) (const exvector &)
```

5.1.1.197 evalf_funcp_exvector

```
typedef ex(* GiNaC::evalf_funcp_exvector) (const exvector &)
```

5.1.1.198 conjugate_funcp_exvector

```
typedef ex(* GiNaC::conjugate_funcp_exvector) (const exvector &)
```

5.1.1.199 real_part_funcp_exvector

```
typedef ex(* GiNaC::real_part_funcp_exvector) (const exvector &)
```

5.1.1.200 imag_part_funcp_exvector

```
typedef ex(* GiNaC::imag_part_funcp_exvector) (const exvector &)
```

5.1.1.201 expand_funcp_exvector

```
typedef ex(* GiNaC::expand_funcp_exvector) (const exvector &, unsigned)
```

5.1.1.202 derivative_funcp_exvector

```
typedef ex(* GiNaC::derivative_funcp_exvector) (const exvector &, unsigned)
```

5.1.1.203 expl_derivative_funcp_exvector

```
typedef ex(* GiNaC::expl_derivative_funcp_exvector) (const exvector &, const symbol &)
```

5.1.1.204 power_funcp_exvector

```
typedef ex(* GiNaC::power_funcp_exvector) (const exvector &, const ex &)
```

5.1.1.205 series_funcp_exvector

```
typedef ex(* GiNaC::series_funcp_exvector) (const exvector &, const relational &, int, unsigned)
```

5.1.1.206 print_funcp_exvector

```
typedef void(* GiNaC::print_funcp_exvector) (const exvector &, const print_context &)
```

5.1.1.207 info_funcp_exvector

```
typedef bool(* GiNaC::info_funcp_exvector) (const exvector &, unsigned)
```

5.1.1.208 exhashmap

```
template<typename T , class Hash = std::hash<ex>, class KeyEqual = std::equal_to<ex>, class  
Allocator = std::allocator<std::pair<const ex, T>>>  
using GiNaC::exhashmap = typedef std::unordered_map<ex, T, Hash, KeyEqual, Allocator>
```

5.1.1.209 spmap

```
typedef std::map<spmapkey, ex> GiNaC::spmap
```

5.1.1.210 lookup_map

```
typedef map<error_and_integral, ex, error_and_integral_is_less> GiNaC::lookup_map
```

5.1.1.211 lst

```
typedef container< std::list > GiNaC::lst
```

5.1.1.212 uintvector

```
typedef std::vector<std::size_t> GiNaC::uintvector
```

5.1.1.213 unsignedvector

```
typedef std::vector<unsigned> GiNaC::unsignedvector
```

5.1.1.214 exvectorvector

```
typedef std::vector<exvector> GiNaC::exvectorvector
```

5.1.1.215 sym_desc_vec

```
typedef std::vector<sym_desc> GiNaC::sym_desc_vec
```

5.1.1.216 digits_changed_callback

```
typedef void(* GiNaC::digits_changed_callback) (long)
```

Function pointer to implement callbacks in the case 'Digits' gets changed.

Main purpose of such callbacks is to adjust look-up tables of certain functions to the new precision. Parameter contains the signed difference between new Digits and old Digits.

5.1.1.217 print_context_class_info

```
typedef class_info<print_context_options> GiNaC::print_context_class_info
```

5.1.1.218 registered_class_info

```
typedef class_info<registered_class_options> GiNaC::registered_class_info
```

5.1.2 Enumeration Type Documentation

5.1.2.1 anonymous enum

anonymous enum

Enumerator

callback_registered

5.1.3 Function Documentation

5.1.3.1 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [1/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    add ,
    expairseq ,
    print_func< print_context > &::do_print. print_func< print_latex > &::do_↵
print_latex. print_func< print_csrc > &::do_print_csrc. print_func< print_tree > &::do_↵
print_tree. print_func< print_python_repr > &::do_print_python_repr )
```

5.1.3.2 GINAC_BIND_UNARCHIVER() [1/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    add )
```


5.1.3.3 GINAC_DECLARE_UNARCHIVER() [1/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    add )
```

5.1.3.4 write_unsigned()

```
static void GiNaC::write_unsigned (
    std::ostream & os,
    unsigned val ) [static]
```

Write unsigned integer quantity to stream.

5.1.3.5 read_unsigned()

```
static unsigned GiNaC::read_unsigned (
    std::istream & is ) [static]
```

Read unsigned integer quantity from stream.

5.1.3.6 operator<<() [1/16]

```
std::ostream & GiNaC::operator<< (
    std::ostream & os,
    const archive_node & n )
```

Write [archive_node](#) to binary data stream.

5.1.3.7 operator<<() [2/16]

```
std::ostream & GiNaC::operator<< (
    std::ostream & os,
    const archive & ar )
```

Write archive to binary data stream.

5.1.3.8 operator>>() [1/3]

```
std::istream & GiNaC::operator>> (
    std::istream & is,
    archive_node & n )
```

Read [archive_node](#) from binary data stream.

5.1.3.9 operator>>() [2/3]

```
std::istream & GiNaC::operator>> (
    std::istream & is,
    archive & ar )
```

Read archive from binary data stream.

5.1.3.10 find_factory_fcn()

```
static synthesize_func GiNaC::find_factory_fcn (
    const std::string & name ) [static]
```

References [GiNaC::unarchive_table_t::find\(\)](#).

Referenced by [GiNaC::archive_node::unarchive\(\)](#).

5.1.3.11 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [2/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    basic ,
    void ,
    print_func< print_context > &::do_print. print_func< print_tree > &::do_print↔
    _tree. print_func< print_python_repr > &::do_print_python_repr )
```

basic copy constructor: implicitly assumes that the other class is of the exact same type (as it's used by duplicate()), so it can copy the tinfo_key and the hash value.

5.1.3.12 is_a() [1/3]

```
template<class T >
bool GiNaC::is_a (
    const basic & obj ) [inline]
```

Check if obj is a T, including base classes.

Referenced by [GiNaC::container< C >::subs\(\)](#).

5.1.3.13 is_exactly_a() [1/2]

```
template<class T >
bool GiNaC::is_exactly_a (
    const basic & obj ) [inline]
```

Check if obj is a T, not including base classes.

5.1.3.14 dynallocate() [1/2]

```
template<class B , typename... Args>
B & GiNaC::dynallocate (
    Args &&... args ) [inline]
```

Constructs a new (class basic or derived) B object on the heap.

This function picks the object's ctor based on the given argument types.

This helps the constructor of ex from basic (or a derived class B) because then the constructor doesn't have to duplicate the object onto the heap. See [ex::construct_from_basic\(const basic &\)](#) for more information.

References [GiNaC::status_flags::dynallocated](#).

5.1.3.15 dynallocate() [2/2]

```
template<class B >
B & GiNaC::dynallocate (
    std::initializer_list< ex > il ) [inline]
```

Constructs a new (class basic or derived) B object on the heap.

This function is needed for [GiNaC](#) classes which have public ctors from initializer lists of expressions (which are not a type and not captured by the variadic template version).

References [GiNaC::status_flags::dynallocated](#).

5.1.3.16 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [3/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    clifford ,
    indexed ,
    print_func< print_dflt > &::do_print_dflt, print_func< print_latex > &::do_↵
print_latex, print_func< print_tree > &::do_print_tree )
```

5.1.3.17 print_func< print_dflt >() [1/3]

```
GiNaC::print_func< print_dflt > (
    &diracone::do_print ) &
```

5.1.3.18 GINAC_BIND_UNARCHIVER() [2/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    clifford )
```

5.1.3.19 GINAC_BIND_UNARCHIVER() [3/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    cliffordunit )
```

5.1.3.20 GINAC_BIND_UNARCHIVER() [4/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    diracone )
```

5.1.3.21 GINAC_BIND_UNARCHIVER() [5/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    diracgamma )
```

5.1.3.22 GINAC_BIND_UNARCHIVER() [6/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    diracgamma5 )
```

5.1.3.23 GINAC_BIND_UNARCHIVER() [7/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    diracgammaL )
```

5.1.3.24 GINAC_BIND_UNARCHIVER() [8/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    diracgammaR )
```

5.1.3.25 is_dirac_slash()

```
static bool GiNaC::is_dirac_slash (
    const ex & seq0 ) [static]
```

Referenced by [GiNaC::clifford::do_print_dflt\(\)](#), and [GiNaC::clifford::do_print_latex\(\)](#).

5.1.3.26 base_and_index()

```
static void GiNaC::base_and_index (
    const ex & c,
    ex & b,
    ex & i ) [static]
```

This function decomposes $\gamma_\mu \rightarrow (1, \mu)$ and $a_\lambda \rightarrow (a_{i\lambda}, i\lambda)$

5.1.3.27 dirac_ONE()

```
ex GiNaC::dirac_ONE (
    unsigned char rl = 0 )
```

Create a Clifford unity object.

Parameters

<i>rl</i>	Representation label

Returns

newly constructed object

Referenced by [GiNaC::add::coeff\(\)](#).

5.1.3.28 get_dim_uint()

```
static unsigned GiNaC::get_dim_uint (
    const ex & e ) [static]
```

5.1.3.29 clifford_unit()

```
ex GiNaC::clifford_unit (
    const ex & mu,
    const ex & metr,
    unsigned char rl = 0 )
```

Create a Clifford unit object.

Parameters

<i>mu</i>	Index (must be of class varidx or a derived class)
<i>metr</i>	Metric (should be indexed, tensmetric or a derived class, or a matrix)
<i>rl</i>	Representation label

Returns

newly constructed Clifford unit object

5.1.3.30 dirac_gamma()

```
ex GiNaC::dirac_gamma (
    const ex & mu,
    unsigned char rl = 0 )
```

Create a Dirac gamma object.

Parameters

<i>mu</i>	Index (must be of class varidx or a derived class)
<i>rl</i>	Representation label

Returns

newly constructed gamma object

5.1.3.31 `dirac_gamma5()`

```
ex GiNaC::dirac_gamma5 (
    unsigned char rl = 0 )
```

Create a Dirac gamma5 object.

Parameters

<i>rl</i>	Representation label

Returns

newly constructed object

5.1.3.32 `dirac_gammaL()`

```
ex GiNaC::dirac_gammaL (
    unsigned char rl = 0 )
```

Create a Dirac gammaL object.

Parameters

<i>rl</i>	Representation label

Returns

newly constructed object

5.1.3.33 `dirac_gammaR()`

```
ex GiNaC::dirac_gammaR (
    unsigned char rl = 0 )
```

Create a Dirac gammaR object.

Parameters

<i>rl</i>	Representation label

Returns

newly constructed object

5.1.3.34 `dirac_slash()`

```
ex GiNaC::dirac_slash (
    const ex & e,
    const ex & dim,
    unsigned char rl = 0 )
```

Create a term of the form $e_\mu * \gamma_\mu$ with a unique index μ .

Parameters

<i>e</i>	Original expression
<i>dim</i>	Dimension of index
<i>rl</i>	Representation label

5.1.3.35 `get_representation_label()` [1/2]

```
static unsigned char GiNaC::get_representation_label (
    const return_type_t & ti ) [static]
```

Extract representation label from tinfo key (as returned by `return_type_tinfo()`).

Referenced by [color_trace\(\)](#), [GiNaC::su3f::contract_with\(\)](#), and [GiNaC::su3d::contract_with\(\)](#).

5.1.3.36 `trace_string()`

```
static ex GiNaC::trace_string (
    exvector::const_iterator ix,
    size_t num ) [static]
```

Take trace of a string of an even number of Dirac gammas given a vector of indices.

5.1.3.37 `dirac_trace()` [1/3]

```
ex GiNaC::dirac_trace (
    const ex & e,
    const std::set< unsigned char > & rls,
    const ex & trONE = 4 )
```

Calculate dirac traces over the specified set of representation labels.

The computed trace is a linear functional that is equal to the usual trace only in $D = 4$ dimensions. In particular, the functional is not always cyclic in $D \neq 4$ dimensions when γ_5 is involved.

Parameters

<i>e</i>	Expression to take the trace of
<i>rls</i>	Set of representation labels
<i>trONE</i>	Expression to be returned as the trace of the unit matrix

5.1.3.38 `dirac_trace()` [2/3]

```
ex GiNaC::dirac_trace (
    const ex & e,
    const lst & rll,
    const ex & trONE = 4 )
```

Calculate dirac traces over the specified list of representation labels.

The computed trace is a linear functional that is equal to the usual trace only in $D = 4$ dimensions. In particular, the functional is not always cyclic in $D \neq 4$ dimensions when γ_5 is involved.

Parameters

<i>e</i>	Expression to take the trace of
<i>rll</i>	List of representation labels
<i>trONE</i>	Expression to be returned as the trace of the unit matrix

5.1.3.39 `dirac_trace()` [3/3]

```
ex GiNaC::dirac_trace (
    const ex & e,
    unsigned char rl = 0,
    const ex & trONE = 4 )
```

Calculate the trace of an expression containing gamma objects with a specified representation label.

The computed trace is a linear functional that is equal to the usual trace only in $D = 4$ dimensions. In particular, the functional is not always cyclic in $D \neq 4$ dimensions when γ_5 is involved.

Parameters

<i>e</i>	Expression to take the trace of
<i>rl</i>	Representation label
<i>trONE</i>	Expression to be returned as the trace of the unit matrix

5.1.3.40 `canonicalize_clifford()`

```
ex GiNaC::canonicalize_clifford (
    const ex & e )
```

Bring all products of clifford objects in an expression into a canonical order.

This is not necessarily the most simple form but it will allow to check two expressions for equality.

5.1.3.41 clifford_star_bar()

```
ex GiNaC::clifford_star_bar (
    const ex & e,
    bool do_bar,
    unsigned options )
```

An auxillary function performing [clifford_star\(\)](#) and [clifford_bar\(\)](#).

Referenced by [clifford_bar\(\)](#), and [clifford_star\(\)](#).

5.1.3.42 clifford_prime()

```
ex GiNaC::clifford_prime (
    const ex & e )
```

Automorphism of the Clifford algebra, simply changes signs of all clifford units.

5.1.3.43 remove_dirac_ONE()

```
ex GiNaC::remove_dirac_ONE (
    const ex & e,
    unsigned char rl = 0,
    unsigned options = 0 )
```

Replaces `dirac_ONE`'s (with a `representation_label` no less than `rl`) in `e` with 1.

For the default value `rl = 0` remove all of them. Aborts if `e` contains any `clifford_unit` with `representation_label` to be removed.

Parameters

<i>e</i>	Expression to be processed
<i>rl</i>	Value of representation label
<i>options</i>	Defines some internal use

5.1.3.44 clifford_max_label()

```
int GiNaC::clifford_max_label (
    const ex & e,
    bool ignore_ONE = false )
```

Returns the maximal representation label of a clifford object if `e` contains at least one, otherwise returns -1.

Parameters

<i>e</i>	Expressed to be processed
<i>ignore_ONE</i>	defines if <code>clifford_ONE</code> should be ignored in the search

Referenced by [GiNaC::add::coeff\(\)](#).

5.1.3.45 clifford_norm()

```
ex GiNaC::clifford_norm (
    const ex & e )
```

Calculation of the norm in the Clifford algebra.

5.1.3.46 clifford_inverse()

```
ex GiNaC::clifford_inverse (
    const ex & e )
```

Calculation of the inverse in the Clifford algebra.

5.1.3.47 lst_to_clifford() [1/2]

```
ex GiNaC::lst_to_clifford (
    const ex & v,
    const ex & mu,
    const ex & metr,
    unsigned char rl = 0 )
```

List or vector conversion into the Clifford vector.

Parameters

<i>v</i>	List or vector of coordinates
<i>mu</i>	Index (must be of class varidx or a derived class)
<i>metr</i>	Metric (should be indexed, tensmetric or a derived class, or a matrix)
<i>rl</i>	Representation label

Returns

Clifford vector with given components

5.1.3.48 lst_to_clifford() [2/2]

```
ex GiNaC::lst_to_clifford (
    const ex & v,
    const ex & e )
```

List or vector conversion into the Clifford vector.

Parameters

<i>v</i>	List or vector of coordinates
<i>e</i>	Clifford unit object

Returns

Clifford vector with given components

5.1.3.49 `get_clifford_comp()`

```
static ex GiNaC::get_clifford_comp (
    const ex & e,
    const ex & c,
    bool root = true ) [static]
```

Auxiliary structure to define a function for stripping one Clifford unit from vectors.

Used in [clifford_to_lst\(\)](#).

5.1.3.50 `clifford_to_lst()`

```
lst GiNaC::clifford_to_lst (
    const ex & e,
    const ex & c,
    bool algebraic = true )
```

An inverse function to [lst_to_clifford\(\)](#).

For given Clifford vector extracts its components with respect to given Clifford unit. Obtained components may contain Clifford units with a different metric. Extraction is based on the algebraic formula $(e * c.i + c.i * e) / \text{pow}(e.i, 2)$ for non-degenerate cases (i.e. neither $\text{pow}(e.i, 2) = 0$).

Parameters

<i>e</i>	Clifford expression to be decomposed into components
<i>c</i>	Clifford unit defining the metric for splitting (should have numeric dimension of indices)
<i>algebraic</i>	Use algebraic or symbolic algorithm for extractions

Returns

List of components of a Clifford vector

5.1.3.51 `clifford_moebius_map()` [1/2]

```
ex GiNaC::clifford_moebius_map (
    const ex & a,
    const ex & b,
    const ex & c,
    const ex & d,
    const ex & v,
    const ex & G,
    unsigned char rl = 0 )
```

Calculations of Moebius transformations (conformal map) defined by a 2x2 Clifford matrix (a b/c d) in linear spaces with arbitrary signature.

The expression is $(a * x + b)/(c * x + d)$, where x is a vector build from list v with metric G . (see Jan Cnops. An introduction to {D}irac operators on manifolds, v.24 of Progress in Mathematical Physics. Birkhauser Boston Inc., Boston, MA, 2002.)

Parameters

a	(1,1) entry of the defining matrix
b	(1,2) entry of the defining matrix
c	(2,1) entry of the defining matrix
d	(2,2) entry of the defining matrix
v	Vector to be transformed
G	Metric of the surrounding space, may be a Clifford unit then the next parameter is ignored
rl	Representation label

Returns

List of components of the transformed vector

5.1.3.52 clifford_moebius_map() [2/2]

```
ex GiNaC::clifford_moebius_map (
    const ex & M,
    const ex & v,
    const ex & G,
    unsigned char rl = 0 )
```

The second form of Moebius transformations defined by a 2x2 Clifford matrix M This function takes the transformation matrix M as a single entity.

Parameters

M	the defining matrix
v	Vector to be transformed
G	Metric of the surrounding space, may be a Clifford unit then the next parameter is ignored
rl	Representation label

Returns

List of components of the transformed vector

5.1.3.53 GINAC_DECLARE_UNARCHIVER() [2/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    clifford )
```

5.1.3.54 GINAC_DECLARE_UNARCHIVER() [3/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    diracone )
```

5.1.3.55 GINAC_DECLARE_UNARCHIVER() [4/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    cliffordunit )
```

5.1.3.56 GINAC_DECLARE_UNARCHIVER() [5/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    diracgamma )
```

5.1.3.57 GINAC_DECLARE_UNARCHIVER() [6/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    diracgamma5 )
```

5.1.3.58 GINAC_DECLARE_UNARCHIVER() [7/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    diracgammaL )
```

5.1.3.59 GINAC_DECLARE_UNARCHIVER() [8/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    diracgammaR )
```

5.1.3.60 is_clifford_tinfo()

```
bool GiNaC::is_clifford_tinfo (
    const return_type_t & ti ) [inline]
```

Check whether a given [return_type_t](#) object (as returned by [return_type_tinfo\(\)](#)) is that of a clifford object (with an arbitrary representation label).

Parameters

<i>ti</i>	tinfo key

References [GiNaC::return_type_t::tinfo](#).

Referenced by [GiNaC::ncmul::conjugate\(\)](#).

5.1.3.61 clifford_bar()

```
ex GiNaC::clifford_bar (
    const ex & e ) [inline]
```

Main anti-automorphism of the Clifford algebra: makes reversion and changes signs of all clifford units.

References [clifford_star_bar\(\)](#).

5.1.3.62 clifford_star()

```
ex GiNaC::clifford_star (
    const ex & e ) [inline]
```

Reversion of the Clifford algebra, reverse the order of all clifford units in ncmul.

References [clifford_star_bar\(\)](#).

5.1.3.63 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [4/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    su3one ,
    tensor ,
    print_func< print_dflt > &::do_print. print_func< print_latex > &::do_print_↵
    latex )
```

5.1.3.64 print_func< print_dflt >() [2/3]

```
GiNaC::print_func< print_dflt > (
    &su3t::do_print ) &
```

5.1.3.65 GINAC_BIND_UNARCHIVER() [9/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    color )
```

5.1.3.66 GINAC_BIND_UNARCHIVER() [10/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    su3one )
```

5.1.3.67 GINAC_BIND_UNARCHIVER() [11/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    su3t )
```

5.1.3.68 GINAC_BIND_UNARCHIVER() [12/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    su3f )
```

5.1.3.69 GINAC_BIND_UNARCHIVER() [13/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    su3d )
```

5.1.3.70 permute_free_index_to_front()

```
static ex GiNaC::permute_free_index_to_front (
    const exvector & iv3,
    const exvector & iv2,
    int & sig ) [static]
```

Given a vector iv3 of three indices and a vector iv2 of two indices that is a subset of iv3, return the (free) index that is in iv3 but not in iv2 and the sign introduced by permuting that index to the front.

Parameters

<i>iv3</i>	Vector of 3 indices
<i>iv2</i>	Vector of 2 indices, must be a subset of iv3
<i>sig</i>	Returns sign introduced by index permutation

Returns

the free index (the one that is in iv3 but not in iv2)

References [GINAC_ASSERT](#), and [TEST_PERMUTATION](#).

Referenced by [GiNaC::su3f::contract_with\(\)](#), and [GiNaC::su3d::contract_with\(\)](#).

5.1.3.71 color_ONE()

```
ex GiNaC::color_ONE (
    unsigned char rl = 0 )
```

Create the su(3) unity element.

This is an indexed object, although it has no indices.

Parameters

<i>rl</i>	Representation label

Returns

newly constructed unity element

Referenced by [GiNaC::su3t::contract_with\(\)](#).

5.1.3.72 `color_T()`

```
ex GiNaC::color_T (
    const ex & a,
    unsigned char rl = 0 )
```

Create an $su(3)$ generator.

Parameters

<i>a</i>	Index
<i>rl</i>	Representation label

Returns

newly constructed unity generator

Referenced by [color_trace\(\)](#), [GiNaC::su3f::contract_with\(\)](#), and [GiNaC::su3d::contract_with\(\)](#).

5.1.3.73 `color_f()`

```
ex GiNaC::color_f (
    const ex & a,
    const ex & b,
    const ex & c )
```

Create an $su(3)$ antisymmetric structure constant.

Parameters

<i>a</i>	First index
<i>b</i>	Second index
<i>c</i>	Third index

Returns

newly constructed structure constant

References [antisymmetric3\(\)](#), and [c](#).

Referenced by [color_h\(\)](#).

5.1.3.74 `color_d()`

```
ex GiNaC::color_d (
    const ex & a,
    const ex & b,
    const ex & c )
```

Create an $su(3)$ symmetric structure constant.

Parameters

<i>a</i>	First index
<i>b</i>	Second index
<i>c</i>	Third index

Returns

newly constructed structure constant

References [c](#), and [symmetric3\(\)](#).

Referenced by [color_h\(\)](#).

5.1.3.75 color_h()

```
ex GiNaC::color_h (
    const ex & a,
    const ex & b,
    const ex & c )
```

This returns the linear combination d.a.b.c+l*f.a.b.c.

References [c](#), [color_d\(\)](#), [color_f\(\)](#), and [l](#).

Referenced by [color_trace\(\)](#).

5.1.3.76 is_color_tinfo()

```
static bool GiNaC::is_color_tinfo (
    const return_type_t & ti ) [static]
```

Check whether a given tinfo key (as returned by [return_type_tinfo\(\)](#)) is that of a color object (with an arbitrary representation label).

References [GiNaC::return_type_t::tinfo](#).

Referenced by [color_trace\(\)](#).

5.1.3.77 get_representation_label() [2/2]

```
static unsigned char GiNaC::get_representation_label (
    const return_type_t & ti ) [static]
```

Extract representation label from tinfo key (as returned by [return_type_tinfo\(\)](#)).

References [GiNaC::return_type_t::rl](#).

5.1.3.78 color_trace() [1/3]

```
ex GiNaC::color_trace (
    const ex & e,
    const std::set< unsigned char > & rls )
```

Calculate color traces over the specified set of representation labels.

Parameters

<i>e</i>	Expression to take the trace of
<i>rls</i>	Set of representation labels

References [_ex0](#), [_ex1](#), [_ex3](#), [color_h\(\)](#), [color_T\(\)](#), [color_trace\(\)](#), [delta_tensor\(\)](#), [GiNaC::ex::expand\(\)](#), [get_representation_label\(\)](#), [is_color_tinfo\(\)](#), [GiNaC::ex::map\(\)](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), and [GiNaC::ex::return_type_tinfo\(\)](#)

Referenced by [color_trace\(\)](#), [color_trace\(\)](#), [color_trace\(\)](#), and [GiNaC::su3t::contract_with\(\)](#).

5.1.3.79 color_trace() [2/3]

```
ex GiNaC::color_trace (
    const ex & e,
    const lst & rll )
```

Calculate color traces over the specified list of representation labels.

Parameters

<i>e</i>	Expression to take the trace of
<i>rll</i>	List of representation labels

References [color_trace\(\)](#), and [GiNaC::info_flags::nonnegint](#).

5.1.3.80 color_trace() [3/3]

```
ex GiNaC::color_trace (
    const ex & e,
    unsigned char rl = 0 )
```

Calculate the trace of an expression containing color objects with a specified representation label.

Parameters

<i>e</i>	Expression to take the trace of
<i>rl</i>	Representation label

References [color_trace\(\)](#).

5.1.3.81 GINAC_DECLARE_UNARCHIVER() [9/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    color )
```

5.1.3.82 GINAC_DECLARE_UNARCHIVER() [10/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    su3one )
```

5.1.3.83 GINAC_DECLARE_UNARCHIVER() [11/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    su3t )
```

5.1.3.84 GINAC_DECLARE_UNARCHIVER() [12/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    su3f )
```

5.1.3.85 GINAC_DECLARE_UNARCHIVER() [13/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    su3d )
```

5.1.3.86 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [5/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    constant ,
    basic ,
    print_func< print_context > &::do_print, print_func< print_latex > &::do_↵
print_latex, print_func< print_tree > &::do_print_tree, print_func< print_python_repr > &↵
::do_print_python_repr ) const
```

References [GiNaC::status_flags::evaluated](#), and [GiNaC::status_flags::expanded](#).

5.1.3.87 GINAC_BIND_UNARCHIVER() [14/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    constant )
```

5.1.3.88 GINAC_DECLARE_UNARCHIVER() [14/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    constant )
```

5.1.3.89 crc32()

```
static unsigned GiNaC::crc32 (
    const char * c,
    const unsigned len,
    const unsigned crcinit ) [static]
```

References [c](#), [crctab](#), and [len](#).

5.1.3.90 are_ex_trivially_equal()

```
bool GiNaC::are_ex_trivially_equal (
    const ex & e1,
    const ex & e2 ) [inline]
```

Compare two objects of class quickly without doing a deep tree traversal.

Returns

"true" if they are equal "false" if equality cannot be established quickly (e1 and e2 may still be equal, in this case).

Referenced by [GiNaC::expair::conjugate\(\)](#), [GiNaC::add::conjugate\(\)](#), [GiNaC::container< C >::conjugate\(\)](#), [GiNaC::expairseq::conjugate\(\)](#), [GiNaC::integral::conjugate\(\)](#), [GiNaC::matrix::conjugate\(\)](#), [GiNaC::mul::conjugate\(\)](#), [GiNaC::power::conjugate\(\)](#), [GiNaC::pseries::conjugate\(\)](#), [GiNaC::pseries::eval_integ\(\)](#), [GiNaC::Gt::evalf\(\)](#), [GiNaC::integral::evalf\(\)](#), [GiNaC::pseries::evalm\(\)](#), [GiNaC::indexed::expand\(\)](#), [GiNaC::integral::expand\(\)](#), [GiNaC::power::expand\(\)](#), [GiNaC::expairseq::expandchildren\(\)](#), [GiNaC::mul::expandchildren\(\)](#), [GiNaC::ncmul::expandchildren\(\)](#), [GiNaC::expairseq::make_flat\(\)](#), [GiNaC::make_flat_inserter::make_flat_inserter\(\)](#), [GiNaC::basic::map\(\)](#), [GiNaC::idx::map\(\)](#), [GiNaC::power::map\(\)](#), [GiNaC::relational::map\(\)](#), [GiNaC::internal::_iter_rep::operator==\(\)](#), [GiNaC::const_iterator::operator==\(\)](#), [GiNaC::basic::subs\(\)](#), [GiNaC::clifford::subs\(\)](#), [GiNaC::idx::subs\(\)](#), [GiNaC::power::subs\(\)](#), [GiNaC::relational::subs\(\)](#), [GiNaC::container< C >::subschildren\(\)](#), and [GiNaC::expairseq::subschildren\(\)](#).

5.1.3.91 operator<<() [3/16]

```
std::ostream & GiNaC::operator<< (
    std::ostream & os,
    const exvector & e )
```

References [get_print_context\(\)](#).

5.1.3.92 operator<<() [4/16]

```
std::ostream & GiNaC::operator<< (
    std::ostream & os,
    const exset & e )
```

References [get_print_context\(\)](#).

5.1.3.93 operator<<() [5/16]

```
std::ostream & GiNaC::operator<< (
    std::ostream & os,
    const exmap & e )
```

References [get_print_context\(\)](#).

5.1.3.94 nops() [1/2]

```
size_t GiNaC::nops (
    const ex & thisex ) [inline]
```

References [GiNaC::ex::nops\(\)](#).

Referenced by [GiNaC::basic::calchash\(\)](#), [GiNaC::basic::derivative\(\)](#), [GiNaC::basic::do_print_tree\(\)](#), [GiNaC::container< C >::do_print\(\)](#), [GiNaC::basic::eval_integ\(\)](#), [GiNaC::basic::evalf\(\)](#), [GiNaC::basic::evalm\(\)](#), [GiNaC::basic::expand\(\)](#), [GiNaC::basic::has\(\)](#), [GiNaC::container< C >::let_op\(\)](#), [GiNaC::basic::map\(\)](#), [GiNaC::basic::match\(\)](#), [GiNaC::container< C >::op\(\)](#), and [GiNaC::basic::subs\(\)](#).

5.1.3.95 expand() [1/2]

```
ex GiNaC::expand (
    const ex & thisex,
    unsigned options = 0 ) [inline]
```

References [GiNaC::ex::expand\(\)](#), and [options](#).

Referenced by [GiNaC::basic::collect\(\)](#), [divide\(\)](#), [divide_in_z\(\)](#), [gcd\(\)](#), [gcd_pf_pow\(\)](#), [log_expand\(\)](#), [prem\(\)](#), [quo\(\)](#), [rem\(\)](#), and [sprem\(\)](#).

5.1.3.96 conjugate()

```
ex GiNaC::conjugate (
    const ex & thisex ) [inline]
```

References [GiNaC::ex::conjugate\(\)](#).

Referenced by [conjugate_expl_derivative\(\)](#).

5.1.3.97 real_part()

```
ex GiNaC::real_part (
    const ex & thisex ) [inline]
```

References [GiNaC::ex::real_part\(\)](#).

Referenced by [cos_imag_part\(\)](#), [cos_real_part\(\)](#), [cosh_imag_part\(\)](#), [cosh_real_part\(\)](#), [exp_imag_part\(\)](#), [exp_real_part\(\)](#), [log_imag_part\(\)](#), [real_part_expl_derivative\(\)](#), [sin_imag_part\(\)](#), [sin_real_part\(\)](#), [sinh_imag_part\(\)](#), [sinh_real_part\(\)](#), [tan_imag_part\(\)](#), [tan_real_part\(\)](#), [tanh_imag_part\(\)](#), and [tanh_real_part\(\)](#).

5.1.3.98 imag_part()

```
ex GiNaC::imag_part (
    const ex & thisex ) [inline]
```

References [GiNaC::ex::imag_part\(\)](#).

Referenced by [cos_imag_part\(\)](#), [cos_real_part\(\)](#), [cosh_imag_part\(\)](#), [cosh_real_part\(\)](#), [exp_imag_part\(\)](#), [exp_real_part\(\)](#), [imag_part_expl_derivative\(\)](#), [log_imag_part\(\)](#), [sin_imag_part\(\)](#), [sin_real_part\(\)](#), [sinh_imag_part\(\)](#), [sinh_real_part\(\)](#), [tan_imag_part\(\)](#), [tan_real_part\(\)](#), [tanh_imag_part\(\)](#), and [tanh_real_part\(\)](#).

5.1.3.99 has()

```
bool GiNaC::has (
    const ex & thisex,
    const ex & pattern,
    unsigned options = 0 ) [inline]
```

References [GiNaC::ex::has\(\)](#), and [options](#).

Referenced by [GiNaC::basic::is_polynomial\(\)](#).

5.1.3.100 find()

```
bool GiNaC::find (
    const ex & thisex,
    const ex & pattern,
    exset & found ) [inline]
```

References [GiNaC::ex::find\(\)](#).

5.1.3.101 is_polynomial()

```
bool GiNaC::is_polynomial (
    const ex & thisex,
    const ex & vars ) [inline]
```

References [GiNaC::ex::is_polynomial\(\)](#).

5.1.3.102 degree()

```
int GiNaC::degree (
    const ex & thisex,
    const ex & s ) [inline]
```

References [GiNaC::ex::degree\(\)](#).

Referenced by [GiNaC::basic::collect\(\)](#), [replace_with_symbol\(\)](#), and [sqrfree_parfrac\(\)](#).

5.1.3.103 ldegree()

```
int GiNaC::ldegree (
    const ex & thisex,
    const ex & s ) [inline]
```

References [GiNaC::ex::ldegree\(\)](#).

Referenced by [GiNaC::basic::collect\(\)](#).

5.1.3.104 `coeff()`

```
ex GiNaC::coeff (
    const ex & thisex,
    const ex & s,
    int n = 1 ) [inline]
```

References [GiNaC::ex::coeff\(\)](#), and [n](#).

Referenced by [GiNaC::basic::collect\(\)](#), [GiNaC::make_flat_inserter::handle_factor\(\)](#), [iterated_integral_evalf_impl\(\)](#), [log_series\(\)](#), and [sqrfree_parfrac\(\)](#).

5.1.3.105 `numer()` [1/2]

```
ex GiNaC::numer (
    const ex & thisex ) [inline]
```

References [GiNaC::ex::numer\(\)](#).

Referenced by [decomp_rational\(\)](#), [GiNaC::power::imag_part\(\)](#), [GiNaC::add::integer_content\(\)](#), [print_real_csrc\(\)](#), [GiNaC::power::real_part\(\)](#), [replace_with_symbol\(\)](#), and [sqrfree_parfrac\(\)](#).

5.1.3.106 `denom()` [1/2]

```
ex GiNaC::denom (
    const ex & thisex ) [inline]
```

References [GiNaC::ex::denom\(\)](#).

Referenced by [decomp_rational\(\)](#), [GiNaC::add::integer_content\(\)](#), [lcmcoeff\(\)](#), [print_real_csrc\(\)](#), [replace_with_symbol\(\)](#), and [sqrfree_parfrac\(\)](#).

5.1.3.107 `numer_denom()`

```
ex GiNaC::numer_denom (
    const ex & thisex ) [inline]
```

References [GiNaC::ex::numer_denom\(\)](#).

Referenced by [decomp_rational\(\)](#), and [sqrfree_parfrac\(\)](#).

5.1.3.108 `normal()`

```
ex GiNaC::normal (
    const ex & thisex ) [inline]
```

References [GiNaC::ex::normal\(\)](#).

Referenced by [fsolve\(\)](#), [GiNaC::normal_map_function::operator\(\)\(\)](#), and [replace_with_symbol\(\)](#).

5.1.3.109 to_rational()

```
ex GiNaC::to_rational (
    const ex & thisex,
    exmap & repl ) [inline]
```

References [GiNaC::ex::to_rational\(\)](#).

5.1.3.110 to_polynomial()

```
ex GiNaC::to_polynomial (
    const ex & thisex,
    exmap & repl ) [inline]
```

References [GiNaC::ex::to_polynomial\(\)](#).

5.1.3.111 collect()

```
ex GiNaC::collect (
    const ex & thisex,
    const ex & s,
    bool distributed = false ) [inline]
```

References [GiNaC::ex::collect\(\)](#).

Referenced by [GiNaC::basic::collect\(\)](#).

5.1.3.112 eval()

```
ex GiNaC::eval (
    const ex & thisex ) [inline]
```

References [GiNaC::ex::eval\(\)](#).

5.1.3.113 evalf() [1/2]

```
ex GiNaC::evalf (
    const ex & thisex ) [inline]
```

References [GiNaC::ex::evalf\(\)](#).

Referenced by [beta_evalf\(\)](#), [eta_evalf\(\)](#), [GiNaC::evalf_map_function::operator\(\)\(\)](#), and [zeta2_evalf\(\)](#).

5.1.3.114 evalm()

```
ex GiNaC::evalm (
    const ex & thisex ) [inline]
```

References [GiNaC::ex::evalm\(\)](#).

Referenced by [GiNaC::evalm_map_function::operator\(\)\(\)](#).

5.1.3.115 eval_integ()

```
ex GiNaC::eval_integ (
    const ex & thisex ) [inline]
```

References [GiNaC::ex::eval_integ\(\)](#).

Referenced by [GiNaC::eval_integ_map_function::operator\(\)\(\)](#).

5.1.3.116 diff()

```
ex GiNaC::diff (
    const ex & thisex,
    const symbol & s,
    unsigned nth = 1 ) [inline]
```

References [GiNaC::ex::diff\(\)](#).

Referenced by [GiNaC::derivative_map_function::operator\(\)\(\)](#).

5.1.3.117 series()

```
ex GiNaC::series (
    const ex & thisex,
    const ex & r,
    int order,
    unsigned options = 0 ) [inline]
```

References [options](#), [order](#), [r](#), and [GiNaC::ex::series\(\)](#).

Referenced by [atan_series\(\)](#), [atanh_series\(\)](#), [Li2_series\(\)](#), and [log_series\(\)](#).

5.1.3.118 match()

```
bool GiNaC::match (
    const ex & thisex,
    const ex & pattern,
    exmap & repl_lst ) [inline]
```

References [GiNaC::ex::match\(\)](#).

Referenced by [GiNaC::basic::has\(\)](#), and [GiNaC::basic::subs_one_level\(\)](#).

5.1.3.119 simplify_indexed() [1/3]

```
ex GiNaC::simplify_indexed (
    const ex & thisex,
    unsigned options = 0 ) [inline]
```

References [options](#), and [GiNaC::ex::simplify_indexed\(\)](#).

Referenced by [GiNaC::ex::simplify_indexed\(\)](#), and [GiNaC::ex::simplify_indexed\(\)](#).

5.1.3.120 `simplify_indexed()` [2/3]

```
ex GiNaC::simplify_indexed (
    const ex & thisex,
    const scalar_products & sp,
    unsigned options = 0 ) [inline]
```

References [options](#), and [GiNaC::ex::simplify_indexed\(\)](#).

5.1.3.121 `symmetrize()` [1/4]

```
ex GiNaC::symmetrize (
    const ex & thisex ) [inline]
```

References [GiNaC::ex::symmetrize\(\)](#).

Referenced by [idx_symmetrization\(\)](#), [GiNaC::ex::symmetrize\(\)](#), [symmetrize\(\)](#), and [symmetrize_cyclic\(\)](#).

5.1.3.122 `symmetrize()` [2/4]

```
ex GiNaC::symmetrize (
    const ex & thisex,
    const lst & l ) [inline]
```

References [GiNaC::ex::symmetrize\(\)](#).

5.1.3.123 `antisymmetrize()` [1/4]

```
ex GiNaC::antisymmetrize (
    const ex & thisex ) [inline]
```

References [GiNaC::ex::antisymmetrize\(\)](#).

Referenced by [GiNaC::ex::antisymmetrize\(\)](#), and [antisymmetrize\(\)](#).

5.1.3.124 `antisymmetrize()` [2/4]

```
ex GiNaC::antisymmetrize (
    const ex & thisex,
    const lst & l ) [inline]
```

References [GiNaC::ex::antisymmetrize\(\)](#).

5.1.3.125 `symmetrize_cyclic()` [1/4]

```
ex GiNaC::symmetrize_cyclic (
    const ex & thisex ) [inline]
```

References [GiNaC::ex::symmetrize_cyclic\(\)](#).

Referenced by [GiNaC::ex::symmetrize_cyclic\(\)](#), and [GiNaC::ex::symmetrize_cyclic\(\)](#).

5.1.3.126 symmetrize_cyclic() [2/4]

```
ex GiNaC::symmetrize_cyclic (
    const ex & thisex,
    const lst & l ) [inline]
```

References [GiNaC::ex::symmetrize_cyclic\(\)](#).

5.1.3.127 op()

```
ex GiNaC::op (
    const ex & thisex,
    size_t i ) [inline]
```

References [GiNaC::ex::op\(\)](#).

Referenced by [GiNaC::basic::calchash\(\)](#), [GiNaC::basic::do_print_tree\(\)](#), [GiNaC::basic::has\(\)](#), [Li2_series\(\)](#), [GiNaC::basic::map\(\)](#), [GiNaC::basic::match\(\)](#), [GiNaC::basic::operator\[\]\(\)](#), [GiNaC::basic::operator\[\]\(\)](#), [rename_dummy_indices\(\)](#), and [GiNaC::basic::subs\(\)](#).

5.1.3.128 lhs()

```
ex GiNaC::lhs (
    const ex & thisex ) [inline]
```

References [GiNaC::ex::lhs\(\)](#).

5.1.3.129 rhs()

```
ex GiNaC::rhs (
    const ex & thisex ) [inline]
```

References [GiNaC::ex::rhs\(\)](#).

Referenced by [lsolve\(\)](#), [GiNaC::ptr< T >::operator!=\(\)](#), [GiNaC::ptr< T >::operator==\(\)](#), [GiNaC::matrix::solve\(\)](#), and [sqrtfree_parfrac\(\)](#).

5.1.3.130 is_zero() [1/2]

```
bool GiNaC::is_zero (
    const ex & thisex ) [inline]
```

References [GiNaC::ex::is_zero\(\)](#).

Referenced by [cosh_eval\(\)](#), [GiNaC::matrix::determinant\(\)](#), [GiNaC::matrix::determinant_minor\(\)](#), [GiNaC::tensepsilon::eval_indexed\(\)](#), [GiNaC::matrix::gauss_elimination\(\)](#), [GiNaC::matrix::markowitz_elimination\(\)](#), [GiNaC::matrix::mul\(\)](#), [GiNaC::relational::operator safe_b](#), [GiNaC::matrix::pivot\(\)](#), [GiNaC::Gt::regularise\(\)](#), [sinh_eval\(\)](#), [tanh_eval\(\)](#), and [GiNaC::Gt::zisToFundamental\(\)](#).

5.1.3.131 swap() [1/2]

```
void GiNaC::swap (
    ex & e1,
    ex & e2 ) [inline]
```

References [GiNaC::ex::swap\(\)](#).

Referenced by [permutation_sign\(\)](#), and [GiNaC::matrix::pivot\(\)](#).

5.1.3.132 subs() [1/3]

```
ex GiNaC::subs (
    const ex & thisex,
    const exmap & m,
    unsigned options = 0 ) [inline]
```

References [m](#), [options](#), and [GiNaC::ex::subs\(\)](#).

Referenced by [Li2_series\(\)](#).

5.1.3.133 subs() [2/3]

```
ex GiNaC::subs (
    const ex & thisex,
    const lst & ls,
    const lst & lr,
    unsigned options = 0 ) [inline]
```

References [lr](#), [options](#), and [GiNaC::ex::subs\(\)](#).

5.1.3.134 subs() [3/3]

```
ex GiNaC::subs (
    const ex & thisex,
    const ex & e,
    unsigned options = 0 ) [inline]
```

References [options](#), and [GiNaC::ex::subs\(\)](#).

5.1.3.135 is_a() [2/3]

```
template<class T >
bool GiNaC::is_a (
    const ex & obj ) [inline]
```

Check if ex is a handle to a T, including base classes.

5.1.3.136 is_exactly_a() [2/2]

```
template<class T >
bool GiNaC::is_exactly_a (
    const ex & obj ) [inline]
```

Check if *ex* is a handle to a T, not including base classes.

5.1.3.137 ex_to()

```
template<class T >
attribute_pure const T & GiNaC::ex_to (
    const ex & e ) [inline]
```

Return a reference to the basic-derived class T object embedded in an expression.

This is fast but unsafe: the result is undefined if the expression does not contain a T object at its top level. Hence, you should generally check the type of *e* first. Also, you shouldn't cache the returned reference because GiNaC's garbage collector may destroy the referenced object any time it's used in another expression.

Parameters

<i>e</i>	expression

Returns

reference to object of class T

See also

[is_exactly_a<class T>\(\)](#)

5.1.3.138 compile_ex() [1/3]

```
void GiNaC::compile_ex (
    const ex & expr,
    const symbol & sym,
    FUNCP_1P & fp,
    const std::string filename = "" )
```

Takes an expression and produces a function pointer to the compiled and linked C code equivalent in double precision.

The function pointer has type FUNCP_1P.

Parameters

<i>expr</i>	Expression to be compiled
<i>sym</i>	Symbol from the expression to become the function parameter
<i>fp</i>	Returned function pointer
<i>filename</i>	Name of the intermediate source code and so-file. If supplied, these intermediate files will not be deleted

5.1.3.139 compile_ex() [2/3]

```
void GiNaC::compile_ex (
    const ex & expr,
    const symbol & sym1,
    const symbol & sym2,
    FUNCP_2P & fp,
    const std::string filename = "" )
```

Takes an expression and produces a function pointer to the compiled and linked C code equivalent in double precision.

The function pointer has type FUNCP_2P.

Parameters

<i>expr</i>	Expression to be compiled
<i>sym1</i>	Symbol from the expression to become the first function parameter
<i>sym2</i>	Symbol from the expression to become the second function parameter
<i>fp</i>	Returned function pointer
<i>filename</i>	Name of the intermediate source code and so-file. If supplied, these intermediate files will not be deleted

5.1.3.140 compile_ex() [3/3]

```
void GiNaC::compile_ex (
    const lst & exprs,
    const lst & syms,
    FUNCP_CUBA & fp,
    const std::string filename = "" )
```

Takes an expression and produces a function pointer to the compiled and linked C code equivalent in double precision.

The function pointer has type FUNCP_CUBA.

Parameters

<i>exprs</i>	List of expression to be compiled
<i>syms</i>	Symbols from the expression to become the function parameters
<i>fp</i>	Returned function pointer
<i>filename</i>	Name of the intermediate source code and so-file. If supplied, these intermediate files will not be deleted

5.1.3.141 link_ex() [1/3]

```
void GiNaC::link_ex (
    const std::string filename,
    FUNCP_1P & fp )
```

Opens an existing so-file and returns a function pointer of type FUNC_P_1P to the contained function.

The so-file has to be generated by compile_ex in advance.

Parameters

<i>filename</i>	Name of the so-file to open and link
<i>fp</i>	Returned function pointer

5.1.3.142 link_ex() [2/3]

```
void GiNaC::link_ex (
    const std::string filename,
    FUNC_P_2P & fp )
```

Opens an existing so-file and returns a function pointer of type FUNC_P_2P to the contained function.

The so-file has to be generated by compile_ex in advance.

Parameters

<i>filename</i>	Name of the so-file to open and link
<i>fp</i>	Returned function pointer

5.1.3.143 link_ex() [3/3]

```
void GiNaC::link_ex (
    const std::string filename,
    FUNC_P_CUBA & fp )
```

Opens an existing so-file and returns a function pointer of type FUNC_P_CUBA to the contained function.

The so-file has to be generated by compile_ex in advance.

Parameters

<i>filename</i>	Name of the so-file to open and link
<i>fp</i>	Returned function pointer

5.1.3.144 unlink_ex()

```
void GiNaC::unlink_ex (
    const std::string filename )
```

Closes all linked .so files that have the supplied filename.

Parameters

<i>filename</i>	Name of the so-file to close

5.1.3.145 swap() [2/2]

```
void GiNaC::swap (
    expair & e1,
    expair & e2 ) [inline]
```

References [GiNaC::expair::swap\(\)](#).

5.1.3.146 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [6/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    expairseq ,
    basic ,
    print_func< print_context > &::do_print. print_func< print_tree > &::do_print←
    _tree )
```

5.1.3.147 conjugateepvector()

```
epvector * GiNaC::conjugateepvector (
    const epvector & )
```

Complex conjugate every element of an epvector.

Returns zero if this does not change anything.

References [GiNaC::ex::conjugate\(\)](#), [GiNaC::ex::is_equal\(\)](#), and [x](#).

Referenced by [GiNaC::expairseq::conjugate\(\)](#), and [GiNaC::pseries::conjugate\(\)](#).

5.1.3.148 factor()

```
ex GiNaC::factor (
    const ex & poly,
    unsigned options )
```

Interface function to the outside world.

Factorizes univariate and multivariate polynomials.

It uses [factor1\(\)](#) on each of the explicitly present factors of poly.

The default option is [factor_options::polynomial](#), which means that [factor\(\)](#) will only factorize an expression if it is a proper polynomial (i.e. the flag [info_flags::polynomial](#) is set). Given the option [factor_options::all](#), [factor\(\)](#) will factorize all subexpressions, e.g. polynomials containing functions or polynomials inside function arguments.

Parameters

in	<i>poly</i>	expression to factorize
in	<i>options</i>	see GiNaC::factor_options

Returns

factorized expression

References [options](#), [poly](#), and [pow\(\)](#).

Referenced by [algebraic_match_mul_with_mul\(\)](#), [collect_common_factors\(\)](#), [GiNaC::ncmul::eval\(\)](#), [GiNaC::power::expand_add\(\)](#), [GiNaC::mul::expandchildren\(\)](#), [find_common_factor\(\)](#), [GiNaC::mul::find_real_imag\(\)](#), [GiNaC::mul::info\(\)](#), [GiNaC::Gt::qExpand\(\)](#), [GiNaC::mul::series\(\)](#), [sqrfree_parfrac\(\)](#), and [GiNaC::Gt::zisToFundamental\(\)](#).

5.1.3.149 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [7/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    fail ,
    basic ,
    print_func< print_context > &::do_print. print_func< print_tree > &::do_print↵
    _tree )
```

5.1.3.150 GINAC_DECLARE_UNARCHIVER() [15/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    fail )
```

5.1.3.151 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [8/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    fderivative ,
    function ,
    print_func< print_context > &::do_print. print_func< print_latex > &::do_↵
    print_latex. print_func< print_csrc > &::do_print_csrc. print_func< print_tree > &::do_↵
    print_tree )
```

5.1.3.152 GINAC_BIND_UNARCHIVER() [15/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    fderivative )
```

5.1.3.153 GINAC_DECLARE_UNARCHIVER() [16/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    fderivative )
```

5.1.3.154 GINAC_BIND_UNARCHIVER() [16/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    function )
```

5.1.3.155 GINAC_DECLARE_UNARCHIVER() [17/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    function )
```

5.1.3.156 is_the_function()

```
template<typename T >
bool GiNaC::is_the_function (
    const ex & x ) [inline]
```

References [x](#).

5.1.3.157 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [9/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    Gt ,
    basic ,
    print_func< print_dflt > &::do_print, print_func< print_python > &::do_print,
    print_func< print_latex > &::do_print_latex )
```

5.1.3.158 GINAC_BIND_UNARCHIVER() [17/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    Gt )
```

5.1.3.159 GINAC_DECLARE_UNARCHIVER() [18/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    Gt )
```

5.1.3.160 make_hash_seed()

```
static unsigned GiNaC::make_hash_seed (
    const std::type_info & tinfo ) [inline], [static]
```

We need a hash function which gives different values for objects of different types.

Hence we need some unique integer for each type. Fortunately, standard C++ RTTI class 'type_info' stores a pointer to mangled type name. Normally this pointer is the same for all objects of the same type (although it changes from run to run), so it can be used for computing hashes. However, on some platforms (such as woe32) the pointer returned by type_info::name() might be different even for objects of the same type! Hence we need to resort to comparing string representation of the (mangled) type names. This is quite expensive, so we compare crc32 hashes of those strings. We might get more hash collisions (and slower evaluation as a result), but being a bit slower is much better than being wrong.

References [golden_ratio_hash\(\)](#).

Referenced by [GiNaC::expairseq::calchash\(\)](#), [GiNaC::function::calchash\(\)](#), [GiNaC::Gt::calchash\(\)](#), [GiNaC::idx::calchash\(\)](#), [GiNaC::relational::calchash\(\)](#), [GiNaC::symbol::calchash\(\)](#), [GiNaC::symmetry::calchash\(\)](#), and [GiNaC::wildcard::calchash\(\)](#).

5.1.3.161 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [10/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    idx ,
    basic ,
    print_func< print_context > &::do_print. print_func< print_latex > &::do_↵
print_latex. print_func< print_csrc > &::do_print_csrc. print_func< print_tree > &::do_↵
print_tree )
```

5.1.3.162 print_func< print_context >()

```
GiNaC::print_func< print_context > (
    &varidx::do_print ) &
```

5.1.3.163 GINAC_BIND_UNARCHIVER() [18/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    idx )
```

5.1.3.164 GINAC_BIND_UNARCHIVER() [19/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    varidx )
```

5.1.3.165 GINAC_BIND_UNARCHIVER() [20/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    spinidx )
```

5.1.3.166 is_dummy_pair() [1/2]

```
bool GiNaC::is_dummy_pair (
    const idx & i1,
    const idx & i2 )
```

Check whether two indices form a dummy pair.

References [GiNaC::idx::is_dummy_pair_same_type\(\)](#).

Referenced by [GiNaC::matrix::contract_with\(\)](#), [GiNaC::spinmetric::contract_with\(\)](#), [GiNaC::matrix::eval_indexed\(\)](#), [GiNaC::tensdelta::eval_indexed\(\)](#), [find_free_and_dummy\(\)](#), [GiNaC::indexed::has_dummy_index_for\(\)](#), [is_dummy_pair\(\)](#), [GiNaC::is_summation_idx::operator\(\)\(\)](#), and [GiNaC::tensor::replace_contr_index\(\)](#).

5.1.3.167 is_dummy_pair() [2/2]

```
bool GiNaC::is_dummy_pair (
    const ex & e1,
    const ex & e2 )
```

Check whether two expressions form a dummy index pair.

References [is_dummy_pair\(\)](#).

5.1.3.168 find_free_and_dummy() [1/2]

```
void GiNaC::find_free_and_dummy (
    exvector::const_iterator it,
    exvector::const_iterator itend,
    exvector & out_free,
    exvector & out_dummy )
```

Given a vector of indices, split them into two vectors, one containing the free indices, the other containing the dummy indices (numeric indices are neither free nor dummy ones).

Parameters

<i>it</i>	Pointer to start of index vector
<i>itend</i>	Pointer to end of index vector
<i>out_free</i>	Vector of free indices (returned, sorted)
<i>out_dummy</i>	Vector of dummy indices (returned, sorted)

References [is_dummy_pair\(\)](#), [last](#), and [shaker_sort\(\)](#).

Referenced by [GiNaC::su3d::contract_with\(\)](#), [count_dummy_indices\(\)](#), [count_free_indices\(\)](#), [find_dummy_indices\(\)](#), [find_free_and_dummy\(\)](#), [get_all_dummy_indices_safely\(\)](#), [GiNaC::indexed::get_dummy_indices\(\)](#), [GiNaC::indexed::get_free_indices\(\)](#), [GiNaC::mul::get_free_indices\(\)](#), and [GiNaC::ncmul::get_free_indices\(\)](#).

5.1.3.169 minimal_dim()

```
ex GiNaC::minimal_dim (
    const ex & dim1,
    const ex & dim2 )
```

Return the minimum of two index dimensions.

If this is undecidable, throw an exception. Numeric dimensions are always considered "smaller" than symbolic dimensions.

References [GiNaC::ex::is_equal\(\)](#).

Referenced by [GiNaC::idx::minimal_dim\(\)](#).

5.1.3.170 GINAC_DECLARE_UNARCHIVER() [19/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    idx )
```

5.1.3.171 GINAC_DECLARE_UNARCHIVER() [20/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    varidx )
```

5.1.3.172 GINAC_DECLARE_UNARCHIVER() [21/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    spinidx )
```

5.1.3.173 find_free_and_dummy() [2/2]

```
void GiNaC::find_free_and_dummy (
    const exvector & v,
    exvector & out_free,
    exvector & out_dummy ) [inline]
```

Given a vector of indices, split them into two vectors, one containing the free indices, the other containing the dummy indices (numeric indices are neither free nor dummy ones).

Parameters

<i>v</i>	Index vector
<i>out_free</i>	Vector of free indices (returned, sorted)
<i>out_dummy</i>	Vector of dummy indices (returned, sorted)

References [find_free_and_dummy\(\)](#).

5.1.3.174 find_dummy_indices()

```
void GiNaC::find_dummy_indices (
    const exvector & v,
    exvector & out_dummy ) [inline]
```

Given a vector of indices, find the dummy indices.

Parameters

<i>v</i>	Index vector
<i>out_dummy</i>	Vector of dummy indices (returned, sorted)

References [find_free_and_dummy\(\)](#).

Referenced by [GiNaC::indexed::get_dummy_indices\(\)](#).

5.1.3.175 count_dummy_indices()

```
size_t GiNaC::count_dummy_indices (
    const exvector & v ) [inline]
```

Count the number of dummy index pairs in an index vector.

References [find_free_and_dummy\(\)](#).

5.1.3.176 count_free_indices()

```
size_t GiNaC::count_free_indices (
    const exvector & v ) [inline]
```

Count the number of dummy index pairs in an index vector.

References [find_free_and_dummy\(\)](#).

5.1.3.177 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [11/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    indexed ,
    exprseq ,
    print_func< print\_context > &::do_print. print_func< print\_latex > &::do_←
    print_latex. print_func< print\_tree > &::do_print_tree )
```

5.1.3.178 GINAC_BIND_UNARCHIVER() [21/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    indexed )
```

5.1.3.179 indices_consistent()

```
static bool GiNaC::indices_consistent (
    const exvector & v1,
    const exvector & v2 ) [static]
```

Check whether two sorted index vectors are consistent (i.e. equal).

Referenced by [GiNaC::add::get_free_indices\(\)](#).

5.1.3.180 number_of_type()

```
template<class T >
size_t GiNaC::number_of_type (
    const exvector & v )
```

5.1.3.181 rename_dummy_indices()

```
template<class T >
static ex GiNaC::rename_dummy_indices (
    const ex & e,
    exvector & global_dummy_indices,
    exvector & local_dummy_indices ) [static]
```

Rename dummy indices in an expression.

Parameters

<i>e</i>	Expression to work on
<i>local_dummy_indices</i>	The set of dummy indices that appear in the expression "e"
<i>global_dummy_indices</i>	The set of dummy indices that have appeared before and which we would like to use in "e", too. This gets updated by the function

References [GiNaC::ex::begin\(\)](#), [GINAC_ASSERT](#), [GiNaC::subs_options::no_pattern](#), [op\(\)](#), [shaker_sort\(\)](#), and [GiNaC::ex::subs\(\)](#).

5.1.3.182 find_variant_indices()

```
static void GiNaC::find_variant_indices (
    const exvector & v,
    exvector & variant_indices ) [static]
```

Given a set of indices, extract those of class varidx.

5.1.3.183 reposition_dummy_indices()

```
bool GiNaC::reposition_dummy_indices (
    ex & e,
    exvector & variant_dummy_indices,
    exvector & moved_indices )
```

Raise/lower dummy indices in a single indexed objects to canonicalize their variance.

Parameters

<i>e</i>	Object to work on
<i>variant_dummy_indices</i>	The set of indices that might need repositioning (will be changed by this function)
<i>moved_indices</i>	The set of indices that have been repositioned (will be changed by this function)

Returns

true if 'e' was changed

5.1.3.184 product_to_exvector()

```
static void GiNaC::product_to_exvector (
    const ex & e,
    exvector & v,
    bool & non_commutative ) [static]
```

References [_ex2](#), [GINAC_ASSERT](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::nops\(\)](#), and [GiNaC::ex::op\(\)](#).

Referenced by [get_all_dummy_indices\(\)](#).

5.1.3.185 idx_symmetrization()

```
template<class T >
ex GiNaC::idx_symmetrization (
    const ex & r,
    const exvector & local_dummy_indices )
```

References [r](#), and [symmetrize\(\)](#).

5.1.3.186 simplify_indexed() [3/3]

```
ex GiNaC::simplify_indexed (
    const ex & e,
    exvector & free_indices,
    exvector & dummy_indices,
    const scalar_products & sp )
```

Simplify indexed expression, return list of free indices.

5.1.3.187 simplify_indexed_product()

```
ex GiNaC::simplify_indexed_product (
    const ex & e,
    exvector & free_indices,
    exvector & dummy_indices,
    const scalar_products & sp )
```

Simplify product of indexed expressions (commutative, noncommutative and simple squares), return list of free indices.

5.1.3.188 hasindex()

```
bool GiNaC::hasindex (
    const ex & x,
    const ex & sym )
```

References [hasindex\(\)](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), and [x](#).

Referenced by [hasindex\(\)](#).

5.1.3.189 get_all_dummy_indices_safely()

```
exvector GiNaC::get_all_dummy_indices_safely (
    const ex & e )
```

More reliable version of the form.

The former assumes that e is an expanded expression.

References [find_free_and_dummy\(\)](#), [get_all_dummy_indices_safely\(\)](#), [GiNaC::ex::get_free_indices\(\)](#), [GiNaC::ex::nops\(\)](#), and [GiNaC::ex::op\(\)](#).

Referenced by [GiNaC::mul::expand\(\)](#), [get_all_dummy_indices_safely\(\)](#), [GiNaC::make_flat_inserter::handle_factor\(\)](#), [rename_dummy_indices_uniquely\(\)](#), and [rename_dummy_indices_uniquely\(\)](#).

5.1.3.190 `get_all_dummy_indices()`

```
exvector GiNaC::get_all_dummy_indices (
    const ex & e )
```

Returns all dummy indices from the exvector.

Returns all dummy indices from the expression.

References [product_to_exvector\(\)](#).

Referenced by [expand_dummy_sum\(\)](#), and [GiNaC::power::expand_mul\(\)](#).

5.1.3.191 `rename_dummy_indices_uniquely()` [1/4]

```
lst GiNaC::rename_dummy_indices_uniquely (
    const exvector & va,
    const exvector & vb )
```

Similar to above, where va and vb are the same and the return value is a list of two lists for substitution in b.

References [GiNaC::container_storage< C >::reserve\(\)](#).

Referenced by [GiNaC::expairseq::construct_from_2_ex\(\)](#), [GiNaC::mul::expand\(\)](#), [GiNaC::ncmul::expand\(\)](#), [GiNaC::power::expand_mul\(\)](#), [GiNaC::make_flat_inserter::handle_factor\(\)](#), [rename_dummy_indices_uniquely\(\)](#), [rename_dummy_indices_uniquely\(\)](#), and [rename_dummy_indices_uniquely\(\)](#).

5.1.3.192 `rename_dummy_indices_uniquely()` [2/4]

```
ex GiNaC::rename_dummy_indices_uniquely (
    const exvector & va,
    const exvector & vb,
    const ex & b )
```

Same as above, where va and vb contain the indices of a and b and are sorted.

References [GiNaC::subs_options::no_index_renaming](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::ex::nops\(\)](#), [GiNaC::container< C >::op\(\)](#), [rename_dummy_indices_uniquely\(\)](#), and [GiNaC::ex::subs\(\)](#).

5.1.3.193 `rename_dummy_indices_uniquely()` [3/4]

```
ex GiNaC::rename_dummy_indices_uniquely (
    const ex & a,
    const ex & b )
```

Returns b with all dummy indices, which are common with a, renamed.

References [get_all_dummy_indices_safely\(\)](#), [GiNaC::subs_options::no_index_renaming](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::ex::nops\(\)](#), [GiNaC::container< C >::op\(\)](#), [rename_dummy_indices_uniquely\(\)](#), and [GiNaC::ex::subs\(\)](#).

5.1.3.194 rename_dummy_indices_uniquely() [4/4]

```
ex GiNaC::rename_dummy_indices_uniquely (
    exvector & va,
    const ex & b,
    bool modify_va )
```

Returns b with all dummy indices, which are listed in va, renamed if modify_va is set to TRUE all dummy indices of b will be appended to va.

References [GiNaC::ex::begin\(\)](#), [GiNaC::ex::end\(\)](#), [get_all_dummy_indices_safely\(\)](#), [GiNaC::subs_options::no_index_renaming](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::ex::nops\(\)](#), [GiNaC::container< C >::op\(\)](#), [rename_dummy_indices_uniquely\(\)](#), and [GiNaC::ex::subs\(\)](#).

5.1.3.195 expand_dummy_sum()

```
ex GiNaC::expand_dummy_sum (
    const ex & e,
    bool subs_idx = false )
```

This function returns the given expression with expanded sums for all dummy index summations, where the dimensionality of the dummy index is a nonnegative integer.

Optionally all indices with a variance will be substituted by indices with the corresponding numeric values without variance.

Parameters

<i>e</i>	the given expression
<i>subs_idx</i>	indicates if variance of dummy indices should be neglected

References [GiNaC::ex::expand\(\)](#), [expand_dummy_sum\(\)](#), [get_all_dummy_indices\(\)](#), [GiNaC::ex::map\(\)](#), [GiNaC::info_flags::nonnegint](#), and [GiNaC::ex::subs\(\)](#).

Referenced by [expand_dummy_sum\(\)](#).

5.1.3.196 GINAC_DECLARE_UNARCHIVER() [22/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    indexed )
```

5.1.3.197 conjugate_evalf()

```
static ex GiNaC::conjugate_evalf (
    const ex & arg ) [static]
```

References [GiNaC::ex::conjugate\(\)](#).

5.1.3.198 conjugate_eval()

```
static ex GiNaC::conjugate_eval (
    const ex & arg ) [static]
```

References [GiNaC::ex::conjugate\(\)](#).

5.1.3.199 conjugate_print_latex()

```
static void GiNaC::conjugate_print_latex (
    const ex & arg,
    const print\_context & c ) [static]
```

References [c](#), and [GiNaC::ex::print\(\)](#).

5.1.3.200 conjugate_conjugate()

```
static ex GiNaC::conjugate_conjugate (
    const ex & arg ) [static]
```

5.1.3.201 conjugate_expl_derivative()

```
static ex GiNaC::conjugate_expl_derivative (
    const ex & arg,
    const symbol & s ) [static]
```

References [conjugate\(\)](#), [GiNaC::ex::diff\(\)](#), [GiNaC::basic::hold\(\)](#), [GiNaC::symbol::info\(\)](#), and [GiNaC::info_flags::real](#).

5.1.3.202 conjugate_real_part()

```
static ex GiNaC::conjugate_real_part (
    const ex & arg ) [static]
```

References [GiNaC::ex::real_part\(\)](#).

5.1.3.203 conjugate_imag_part()

```
static ex GiNaC::conjugate_imag_part (
    const ex & arg ) [static]
```

References [GiNaC::ex::imag_part\(\)](#).

5.1.3.204 func_arg_info()

```
static bool GiNaC::func_arg_info (
    const ex & arg,
    unsigned inf ) [static]
```

References [GiNaC::info_flags::cinteger](#), [GiNaC::info_flags::cinteger_polynomial](#), [GiNaC::info_flags::crational](#), [GiNaC::info_flags::crational_polynomial](#), [GiNaC::info_flags::even](#), [GiNaC::info_flags::has_indices](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::info_flags::integer_polynomial](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::negint](#), [GiNaC::info_flags::nonnegative](#), [GiNaC::info_flags::nonnegint](#), [GiNaC::info_flags::odd](#), [GiNaC::info_flags::polynomial](#), [GiNaC::info_flags::posint](#), [GiNaC::info_flags::positive](#), [GiNaC::info_flags::prime](#), [GiNaC::info_flags::rational](#), [GiNaC::info_flags::rational_function](#), [GiNaC::info_flags::rational_polynomial](#), and [GiNaC::info_flags::real](#).

Referenced by [conjugate_info\(\)](#).

5.1.3.205 conjugate_info()

```
static bool GiNaC::conjugate_info (
    const ex & arg,
    unsigned inf ) [static]
```

References [func_arg_info\(\)](#).

5.1.3.206 REGISTER_FUNCTION() [1/36]

```
GiNaC::REGISTER_FUNCTION (
    conjugate_function ,
    eval_func(conjugate\_eval). evalf_func(conjugate\_evalf). expl_derivative_↔
func(conjugate\_expl\_derivative). info_func(conjugate\_info). print_func< print\_latex >(conjugate\_print\_latex
conjugate_func(conjugate\_conjugate). real_part_func(conjugate\_real\_part). imag_part_func(conjugate\_imag\_part
set_name("conjugate","conjugate") )
```

5.1.3.207 real_part_evalf()

```
static ex GiNaC::real_part_evalf (
    const ex & arg ) [static]
```

5.1.3.208 real_part_eval()

```
static ex GiNaC::real_part_eval (
    const ex & arg ) [static]
```

References [GiNaC::ex::real_part\(\)](#).

5.1.3.209 real_part_print_latex()

```
static void GiNaC::real_part_print_latex (
    const ex & arg,
    const print\_context & c ) [static]
```

References [c](#), and [GiNaC::ex::print\(\)](#).

5.1.3.210 real_part_conjugate()

```
static ex GiNaC::real_part_conjugate (
    const ex & arg ) [static]
```

5.1.3.211 real_part_real_part()

```
static ex GiNaC::real_part_real_part (
    const ex & arg ) [static]
```

5.1.3.212 real_part_imag_part()

```
static ex GiNaC::real_part_imag_part (
    const ex & arg ) [static]
```

5.1.3.213 real_part_expl_derivative()

```
static ex GiNaC::real_part_expl_derivative (
    const ex & arg,
    const symbol & s ) [static]
```

References [GiNaC::ex::diff\(\)](#), [GiNaC::basic::hold\(\)](#), [GiNaC::symbol::info\(\)](#), [GiNaC::info_flags::real](#), and [real_part\(\)](#).

5.1.3.214 REGISTER_FUNCTION() [2/36]

```
GiNaC::REGISTER_FUNCTION (
    real_part_function ,
    eval_func(real\_part\_eval). evalf_func(real\_part\_evalf). expl_derivative_↔
func(real\_part\_expl\_derivative). print_func< print\_latex >(real\_part\_print\_latex). conjugate_↔
_func(real\_part\_conjugate). real_part_func(real\_part\_real\_part). imag_part_func(real\_part\_imag\_part).
set_name("real_part", "real_part") )
```

5.1.3.215 imag_part_evalf()

```
static ex GiNaC::imag_part_evalf (
    const ex & arg ) [static]
```

5.1.3.216 imag_part_eval()

```
static ex GiNaC::imag_part_eval (
    const ex & arg ) [static]
```

References [GiNaC::ex::imag_part\(\)](#).

5.1.3.217 imag_part_print_latex()

```
static void GiNaC::imag_part_print_latex (
    const ex & arg,
    const print\_context & c ) [static]
```

References [c](#), and [GiNaC::ex::print\(\)](#).

5.1.3.218 imag_part_conjugate()

```
static ex GiNaC::imag_part_conjugate (
    const ex & arg ) [static]
```

5.1.3.219 imag_part_real_part()

```
static ex GiNaC::imag_part_real_part (
    const ex & arg ) [static]
```

5.1.3.220 imag_part_imag_part()

```
static ex GiNaC::imag_part_imag_part (
    const ex & arg ) [static]
```

5.1.3.221 imag_part_expl_derivative()

```
static ex GiNaC::imag_part_expl_derivative (
    const ex & arg,
    const symbol & s ) [static]
```

References [GiNaC::ex::diff\(\)](#), [GiNaC::basic::hold\(\)](#), [imag_part\(\)](#), [GiNaC::symbol::info\(\)](#), and [GiNaC::info_flags::real](#).

5.1.3.222 REGISTER_FUNCTION() [3/36]

```
GiNaC::REGISTER_FUNCTION (
    imag_part_function ,
    eval_func(imag\_part\_eval). evalf_func(imag\_part\_evalf). expl_derivative_↔
func(imag\_part\_expl\_derivative). print_func< print\_latex >(imag\_part\_print\_latex). conjugate↔
_func(imag\_part\_conjugate). real_part_func(imag\_part\_real\_part). imag_part_func(imag\_part\_imag\_part).
set_name("imag_part","imag_part") )
```

5.1.3.223 abs_evalf()

```
static ex GiNaC::abs_evalf (
    const ex & arg ) [static]
```

References [abs\(\)](#), and [GiNaC::basic::hold\(\)](#).

5.1.3.224 abs_eval()

```
static ex GiNaC::abs_eval (
    const ex & arg ) [static]
```

References [abs\(\)](#), [exp\(\)](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::info\(\)](#), [is_ex_the_function](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::nonnegative](#), [GiNaC::ex::op\(\)](#), [GiNaC::info_flags::positive](#), [pow\(\)](#), [GiNaC::info_flags::real](#), [GiNaC::ex::real_part\(\)](#), and [step\(\)](#).

5.1.3.225 abs_expand()

```
static ex GiNaC::abs_expand (
    const ex & arg,
    unsigned options ) [static]
```

References [abs\(\)](#), [GiNaC::ex::begin\(\)](#), [GiNaC::ex::end\(\)](#), [GiNaC::ex::expand\(\)](#), [GiNaC::expand_options::expand_function_args](#), [GiNaC::expand_options::expand_transcendental](#), [GiNaC::status_flags::expanded](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::nops\(\)](#), and [options](#).

5.1.3.226 abs_expl_derivative()

```
static ex GiNaC::abs_expl_derivative (
    const ex & arg,
    const symbol & s ) [static]
```

References [abs\(\)](#), [GiNaC::ex::conjugate\(\)](#), and [GiNaC::ex::diff\(\)](#).

5.1.3.227 abs_print_latex()

```
static void GiNaC::abs_print_latex (
    const ex & arg,
    const print\_context & c ) [static]
```

References [c](#), and [GiNaC::ex::print\(\)](#).

5.1.3.228 abs_print_csrc_float()

```
static void GiNaC::abs_print_csrc_float (
    const ex & arg,
    const print\_context & c ) [static]
```

References [c](#), and [GiNaC::ex::print\(\)](#).

5.1.3.229 abs_conjugate()

```
static ex GiNaC::abs_conjugate (
    const ex & arg ) [static]
```

References [abs\(\)](#), and [GiNaC::basic::hold\(\)](#).

5.1.3.230 abs_real_part()

```
static ex GiNaC::abs_real_part (
    const ex & arg ) [static]
```

References [abs\(\)](#), and [GiNaC::basic::hold\(\)](#).

5.1.3.231 abs_imag_part()

```
static ex GiNaC::abs_imag_part (
    const ex & arg ) [static]
```

5.1.3.232 abs_power()

```
static ex GiNaC::abs_power (
    const ex & arg,
    const ex & exp ) [static]
```

References [abs\(\)](#), [GiNaC::ex::conjugate\(\)](#), [GiNaC::info_flags::even](#), [exp\(\)](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::numeric::info\(\)](#), [GiNaC::ex::is_equal\(\)](#), [is_even\(\)](#), [pow\(\)](#), and [GiNaC::info_flags::real](#).

5.1.3.233 abs_info()

```
bool GiNaC::abs_info (
    const ex & arg,
    unsigned inf )
```

References [GiNaC::info_flags::even](#), [GiNaC::info_flags::has_indices](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::nonnegative](#), [GiNaC::info_flags::nonnegint](#), [GiNaC::info_flags::odd](#), [GiNaC::info_flags::positive](#), [GiNaC::info_flags::prime](#), and [GiNaC::info_flags::real](#).

5.1.3.234 REGISTER_FUNCTION() [4/36]

```
GiNaC::REGISTER_FUNCTION (
    abs ,
    eval_func(abs\_eval). evalf_func(abs\_evalf). expand_func(abs\_expand). expl\_↵
    derivative_func(abs\_expl\_derivative). info_func(abs\_info). print_func< print\_latex >(abs\_print\_latex).
    print_func< print\_csrc\_float >(abs\_print\_csrc\_float). print_func< print\_csrc\_double >(abs\_print\_csrc\_double).
    conjugate_func(abs\_conjugate). real_part_func(abs\_real\_part). imag_part_func(abs\_imag\_part).
    power_func(abs\_power) )
```

5.1.3.235 step_evalf()

```
static ex GiNaC::step_evalf (
    const ex & arg ) [static]
```

References [GiNaC::basic::hold\(\)](#), and [step\(\)](#).

5.1.3.236 step_eval()

```
static ex GiNaC::step_eval (
    const ex & arg ) [static]
```

References [GiNaC::basic::hold\(\)](#), [l](#), [GiNaC::numeric::imag\(\)](#), [GiNaC::numeric::is_real\(\)](#), [GiNaC::numeric::is_zero\(\)](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), [GiNaC::numeric::real\(\)](#), and [step\(\)](#).

5.1.3.237 step_series()

```
static ex GiNaC::step_series (
    const ex & arg,
    const relational & rel,
    int order,
    unsigned options ) [static]
```

References [_ex0](#), [GiNaC::ex::info\(\)](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::info_flags::numeric](#), [options](#), [step\(\)](#), [GiNaC::ex::subs\(\)](#), and [GiNaC::series_options::suppress_branchcut](#).

5.1.3.238 step_conjugate()

```
static ex GiNaC::step_conjugate (
    const ex & arg ) [static]
```

References [GiNaC::basic::hold\(\)](#), and [step\(\)](#).

5.1.3.239 step_real_part()

```
static ex GiNaC::step_real_part (
    const ex & arg ) [static]
```

References [GiNaC::basic::hold\(\)](#), and [step\(\)](#).

5.1.3.240 step_imag_part()

```
static ex GiNaC::step_imag_part (
    const ex & arg ) [static]
```

5.1.3.241 REGISTER_FUNCTION() [5/36]

```
GiNaC::REGISTER_FUNCTION (
    step ,
    eval_func(step\_eval). evalf_func(step\_evalf). series_func(step\_series). conjugate←
    _func(step\_conjugate). real_part_func(step\_real\_part). imag_part_func(step\_imag\_part) )
```

5.1.3.242 csgn_evalf()

```
static ex GiNaC::csgn_evalf (
    const ex & arg ) [static]
```

References [csgn\(\)](#).

5.1.3.243 csgn_eval()

```
static ex GiNaC::csgn_eval (
    const ex & arg ) [static]
```

References [csgn\(\)](#), [I](#), [GiNaC::numeric::imag\(\)](#), [GiNaC::numeric::is_real\(\)](#), [GiNaC::numeric::is_zero\(\)](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), and [GiNaC::numeric::real\(\)](#).

5.1.3.244 csgn_series()

```
static ex GiNaC::csgn_series (
    const ex & arg,
    const relational & rel,
    int order,
    unsigned options ) [static]
```

References [_ex0](#), [csgn\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::info_flags::numeric](#), [options](#), [GiNaC::ex::subs\(\)](#), and [GiNaC::series_options::suppress_branchcut](#).

5.1.3.245 csgn_conjugate()

```
static ex GiNaC::csgn_conjugate (
    const ex & arg ) [static]
```

References [csgn\(\)](#).

5.1.3.246 csgn_real_part()

```
static ex GiNaC::csgn_real_part (
    const ex & arg ) [static]
```

References [csgn\(\)](#).

5.1.3.247 csgn_imag_part()

```
static ex GiNaC::csgn_imag_part (
    const ex & arg ) [static]
```

5.1.3.248 csgn_power()

```
static ex GiNaC::csgn_power (
    const ex & arg,
    const ex & exp ) [static]
```

References [_ex2](#), [csgn\(\)](#), [exp\(\)](#), [GiNaC::basic::hold\(\)](#), [GiNaC::numeric::info\(\)](#), [is_odd\(\)](#), and [GiNaC::info_flags::positive](#).

5.1.3.249 REGISTER_FUNCTION() [6/36]

```
GiNaC::REGISTER_FUNCTION (
    csgn ,
    eval_func(csgn_eval). evalf_func(csgn_evalf). series_func(csgn_series). conjugate↵
    _func(csgn_conjugate). real_part_func(csgn_real_part). imag_part_func(csgn_imag_part). power↵
    _func(csgn_power) )
```

5.1.3.250 eta_evalf()

```
static ex GiNaC::eta_evalf (
    const ex & x,
    const ex & y ) [static]
```

References [_ex0](#), [csgn\(\)](#), [evalf\(\)](#), [I](#), [imag\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::numeric::is_negative\(\)](#), [GiNaC::numeric::is_real\(\)](#), [GiNaC::info_flags::numeric](#), [Pi](#), [GiNaC::info_flags::positive](#), and [x](#).

5.1.3.251 eta_eval()

```
static ex GiNaC::eta_eval (
    const ex & x,
    const ex & y ) [static]
```

References [_ex0](#), [csgn\(\)](#), [I](#), [imag\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::numeric::is_negative\(\)](#), [GiNaC::numeric::is_real\(\)](#), [GiNaC::info_flags::numeric](#), [Pi](#), [GiNaC::info_flags::positive](#), and [x](#).

5.1.3.252 eta_series()

```
static ex GiNaC::eta_series (
    const ex & x,
    const ex & y,
    const relational & rel,
    int order,
    unsigned options ) [static]
```

References [_ex0](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::info_flags::numeric](#), [GiNaC::ex::subs\(\)](#), and [x](#).

5.1.3.253 eta_conjugate()

```
static ex GiNaC::eta_conjugate (
    const ex & x,
    const ex & y ) [static]
```

References [x](#).

5.1.3.254 eta_real_part()

```
static ex GiNaC::eta_real_part (
    const ex & x,
    const ex & y ) [static]
```

5.1.3.255 eta_imag_part()

```
static ex GiNaC::eta_imag_part (
    const ex & x,
    const ex & y ) [static]
```

References [GiNaC::basic::hold\(\)](#), [I](#), and [x](#).

5.1.3.256 REGISTER_FUNCTION() [7/36]

```
GiNaC::REGISTER_FUNCTION (
    eta ,
    eval_func(eta\_eval). evalf_func(eta\_evalf). series_func(eta\_series). latex↵
_name("\\eta"). set_symmetry(sy\_symm(0, 1)). conjugate_func(eta\_conjugate). real_part_↵
func(eta\_real\_part). imag_part_func(eta\_imag\_part) )
```

5.1.3.257 Li2_evalf()

```
static ex GiNaC::Li2_evalf (
    const ex & x ) [static]
```

References [GiNaC::basic::hold\(\)](#), [Li2\(\)](#), and [x](#).

5.1.3.258 Li2_eval()

```
static ex GiNaC::Li2_eval (
    const ex & x ) [static]
```

References [_ex0](#), [_ex1](#), [_ex12](#), [_ex1_2](#), [_ex2](#), [_ex6](#), [_ex_1](#), [_ex_1_2](#), [_ex_48](#), Catalan, [GiNaC::info_flags::crational](#), [GiNaC::basic::hold\(\)](#), [I](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::is_zero\(\)](#), [Li2\(\)](#), [log\(\)](#), [GiNaC::info_flags::numeric](#), [Pi](#), and [x](#).

5.1.3.259 Li2_deriv()

```
static ex GiNaC::Li2_deriv (
    const ex & x,
    unsigned deriv_param ) [static]
```

References [_ex1](#), [GINAC_ASSERT](#), [log\(\)](#), and [x](#).

5.1.3.260 Li2_series() [1/2]

```
static ex GiNaC::Li2_series (
    const ex & x,
    const relational & rel,
    int order,
    unsigned options ) [static]
```

References [_ex0](#), [_ex1](#), [_ex2](#), [_num2_p](#), [l](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_equal\(\)](#), [is_real\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::relational::lhs\(\)](#), [Li2\(\)](#), [log\(\)](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::ex::nops\(\)](#), [GiNaC::info_flags::numeric](#), [op\(\)](#), [GiNaC::ex::op\(\)](#), [options](#), [order](#), [Pi](#), [pow\(\)](#), [GiNaC::relational::rhs\(\)](#), [GiNaC::ex::series\(\)](#), [series\(\)](#), [subs\(\)](#), [GiNaC::ex::subs\(\)](#), [GiNaC::series_options::suppress_branchcut](#), [x](#), and [zeta\(\)](#).

Referenced by [Li2_projection\(\)](#).

5.1.3.261 Li2_conjugate()

```
static ex GiNaC::Li2_conjugate (
    const ex & x ) [static]
```

References [_num1_p](#), [GiNaC::ex::conjugate\(\)](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::imag_part\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_zero\(\)](#), [Li2\(\)](#), [GiNaC::info_flags::negative](#), and [x](#).

5.1.3.262 REGISTER_FUNCTION() [8/36]

```
GiNaC::REGISTER_FUNCTION (
    Li2 ,
    eval_func(Li2\_eval). evalf_func(Li2\_evalf). derivative_func(Li2\_deriv). series←
_func(Li2\_series). conjugate_func(Li2\_conjugate). latex_name("\mathrm{Li}_2" )
```

5.1.3.263 Li3_eval()

```
static ex GiNaC::Li3_eval (
    const ex & x ) [static]
```

References [GiNaC::ex::is_zero\(\)](#), and [x](#).

5.1.3.264 REGISTER_FUNCTION() [9/36]

```
GiNaC::REGISTER_FUNCTION (
    Li3 ,
    eval_func(Li3\_eval). latex_name("\mathrm{Li}_3" )
```

5.1.3.265 zetaderiv_eval()

```
static ex GiNaC::zetaderiv_eval (
    const ex & n,
    const ex & x ) [static]
```

References [GiNaC::basic::hold\(\)](#), [n](#), [GiNaC::info_flags::numeric](#), [x](#), and [zeta\(\)](#).

5.1.3.266 zetaderiv_deriv()

```
static ex GiNaC::zetaderiv_deriv (
    const ex & n,
    const ex & x,
    unsigned deriv_param ) [static]
```

References [GINAC_ASSERT](#), [n](#), and [x](#).

5.1.3.267 REGISTER_FUNCTION() [10/36]

```
GiNaC::REGISTER_FUNCTION (
    zetaderiv ,
    eval_func(zetaderiv\_eval). derivative_func(zetaderiv\_deriv). latex_name("\\zeta^\\prime")
)
```

5.1.3.268 factorial_evalf()

```
static ex GiNaC::factorial_evalf (
    const ex & x ) [static]
```

References [factorial\(\)](#), [GiNaC::basic::hold\(\)](#), and [x](#).

5.1.3.269 factorial_eval()

```
static ex GiNaC::factorial_eval (
    const ex & x ) [static]
```

References [factorial\(\)](#), [GiNaC::basic::hold\(\)](#), and [x](#).

5.1.3.270 factorial_print_dflt_latex()

```
static void GiNaC::factorial_print_dflt_latex (
    const ex & x,
    const print\_context & c ) [static]
```

References [c](#), [GiNaC::ex::print\(\)](#), and [x](#).

5.1.3.271 factorial_conjugate()

```
static ex GiNaC::factorial_conjugate (
    const ex & x ) [static]
```

References [factorial\(\)](#), [GiNaC::basic::hold\(\)](#), and [x](#).

5.1.3.272 factorial_real_part()

```
static ex GiNaC::factorial_real_part (
    const ex & x ) [static]
```

References [factorial\(\)](#), [GiNaC::basic::hold\(\)](#), and [x](#).

5.1.3.273 factorial_imag_part()

```
static ex GiNaC::factorial_imag_part (
    const ex & x ) [static]
```

5.1.3.274 REGISTER_FUNCTION() [11/36]

```
GiNaC::REGISTER_FUNCTION (
    factorial ,
    eval_func(factorial\_eval). evalf_func(factorial\_evalf). print_func< print\_dflt
>(factorial\_print\_dflt\_latex). print_func< print\_latex >(factorial\_print\_dflt\_latex). conjugate←
_func(factorial\_conjugate). real_part_func(factorial\_real\_part). imag_part_func(factorial\_imag\_part)
)
```

5.1.3.275 binomial_evalf()

```
static ex GiNaC::binomial_evalf (
    const ex & x,
    const ex & y ) [static]
```

References [binomial\(\)](#), [GiNaC::basic::hold\(\)](#), and [x](#).

5.1.3.276 binomial_sym()

```
static ex GiNaC::binomial_sym (
    const ex & x,
    const numeric & y ) [static]
```

References [_ex0](#), [_ex1](#), [binomial\(\)](#), [GiNaC::ex::expand\(\)](#), [GiNaC::basic::hold\(\)](#), [GiNaC::numeric::is_integer\(\)](#), [GiNaC::numeric::is_nonneg_integer\(\)](#), [GiNaC::numeric::to_int\(\)](#), and [x](#).

Referenced by [binomial_evalf\(\)](#).

5.1.3.277 binomial_eval()

```
static ex GiNaC::binomial_eval (
    const ex & x,
    const ex & y ) [static]
```

References [binomial\(\)](#), [binomial_sym\(\)](#), [GiNaC::basic::hold\(\)](#), [is_integer\(\)](#), and [x](#).

5.1.3.278 binomial_conjugate()

```
static ex GiNaC::binomial_conjugate (
    const ex & x,
    const ex & y ) [static]
```

References [binomial\(\)](#), [GiNaC::basic::hold\(\)](#), and [x](#).

5.1.3.279 binomial_real_part()

```
static ex GiNaC::binomial_real_part (
    const ex & x,
    const ex & y ) [static]
```

References [binomial\(\)](#), [GiNaC::basic::hold\(\)](#), and [x](#).

5.1.3.280 binomial_imag_part()

```
static ex GiNaC::binomial_imag_part (
    const ex & x,
    const ex & y ) [static]
```

5.1.3.281 REGISTER_FUNCTION() [12/36]

```
GiNaC::REGISTER_FUNCTION (
    binomial ,
    eval_func(binomial\_eval) . evalf_func(binomial\_evalf) . conjugate_func(binomial\_conjugate) .
    real_part_func(binomial\_real\_part) . imag_part_func(binomial\_imag\_part) )
```

5.1.3.282 Order_eval()

```
static ex GiNaC::Order_eval (
    const ex & x ) [static]
```

References [_ex0](#), [_ex1](#), [GiNaC::ex::is_zero\(\)](#), [m](#), [GiNaC::ex::op\(\)](#), and [x](#).

5.1.3.283 Order_series()

```
static ex GiNaC::Order_series (
    const ex & x,
    const relational & r,
    int order,
    unsigned options ) [static]
```

References [_ex1](#), [GINAC_ASSERT](#), [GiNaC::ex::ldegree\(\)](#), [order](#), [r](#), and [x](#).

5.1.3.284 Order_conjugate()

```
static ex GiNaC::Order_conjugate (
    const ex & x ) [static]
```

References [x](#).

5.1.3.285 Order_real_part()

```
static ex GiNaC::Order_real_part (
    const ex & x ) [static]
```

References [x](#).

5.1.3.286 Order_imag_part()

```
static ex GiNaC::Order_imag_part (
    const ex & x ) [static]
```

References [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::real](#), and [x](#).

5.1.3.287 Order_power()

```
static ex GiNaC::Order_power (
    const ex & x,
    const ex & e ) [static]
```

References [GiNaC::basic::hold\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::posint](#), [pow\(\)](#), and [x](#).

5.1.3.288 Order_expl_derivative()

```
static ex GiNaC::Order_expl_derivative (
    const ex & arg,
    const symbol & s ) [static]
```

References [GiNaC::ex::diff\(\)](#).

5.1.3.289 REGISTER_FUNCTION() [13/36]

```
GiNaC::REGISTER_FUNCTION (
    Order ,
    eval_func(Order_eval). series_func(Order_series). latex_name("\\mathcal{O}").
    expl_derivative_func(Order_expl_derivative). conjugate_func(Order_conjugate). real_part_func(Order_real_part). imag_part_func(Order_imag_part). power_func(Order_power) )
```

5.1.3.290 Isolve()

```
ex GiNaC::lsolve (
    const ex & eqns,
    const ex & symbols,
    unsigned options = solve_algo::automatic )
```

Factorial function.

Binomial function. Order term function (for truncated power series).

References [GiNaC::container< C >::append\(\)](#), [c](#), [GiNaC::ex::coeff\(\)](#), [GiNaC::matrix::cols\(\)](#), [GiNaC::ex::expand\(\)](#), [GiNaC::info_flags::exprseq](#), [GINAC_ASSERT](#), [GiNaC::symbolset::has\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::list](#), [Isolve\(\)](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), [options](#), [r](#), [GiNaC::info_flags::relation_equal](#), [rhs\(\)](#), [GiNaC::matrix::rows\(\)](#), [GiNaC::matrix::solve\(\)](#), [GiNaC::info_flags::symbol](#), and [syms](#).

Referenced by [Isolve\(\)](#).

5.1.3.291 fsolve()

```
const numeric GiNaC::fsolve (
    const ex & f,
    const symbol & x,
    const numeric & x1,
    const numeric & x2 )
```

Find a real root of real-valued function f(x) numerically within a given interval.

The function must change sign across interval. Uses Newton- Raphson method combined with bisection in order to guarantee convergence.

Parameters

<i>f</i>	Function f(x)
<i>x</i>	Symbol f(x)
<i>x1</i>	lower interval limit
<i>x2</i>	upper interval limit

Exceptions

<i>runtime_error</i>	(if interval is invalid).

References [GiNaC::ex::diff\(\)](#), [GiNaC::ex::evalf\(\)](#), [GiNaC::numeric::is_real\(\)](#), [is_real\(\)](#), [GiNaC::numeric::is_zero\(\)](#),

[GiNaC::ex::lhs\(\)](#), [normal\(\)](#), [GiNaC::ex::rhs\(\)](#), [GiNaC::ex::subs\(\)](#), and [x](#).

5.1.3.292 [zeta\(\)](#) [1/3]

```
template<typename T1 >
function GiNaC::zeta (
    const T1 & p1 ) [inline]
```

References [GiNaC::zeta1_SERIAL::serial](#).

Referenced by [Li2_series\(\)](#), [Li_eval\(\)](#), [psi2_eval\(\)](#), [GiNaC::Gt::qExpand\(\)](#), [S_eval\(\)](#), [zeta1_eval\(\)](#), [zeta1_evalf\(\)](#), [zeta2_eval\(\)](#), [zeta2_evalf\(\)](#), and [zetaderiv_eval\(\)](#).

5.1.3.293 [zeta\(\)](#) [2/3]

```
template<typename T1 , typename T2 >
function GiNaC::zeta (
    const T1 & p1,
    const T2 & p2 ) [inline]
```

References [GiNaC::zeta2_SERIAL::serial](#).

5.1.3.294 [is_the_function< zeta_SERIAL >\(\)](#)

```
template<>
bool GiNaC::is_the_function< zeta_SERIAL > (
    const ex & x ) [inline]
```

References [x](#).

5.1.3.295 [G\(\)](#) [1/2]

```
template<typename T1 , typename T2 >
function GiNaC::G (
    const T1 & x,
    const T2 & y ) [inline]
```

References [GiNaC::G2_SERIAL::serial](#), and [x](#).

Referenced by [G2_eval\(\)](#), [G2_evalf\(\)](#), [G3_eval\(\)](#), [G3_evalf\(\)](#), and [GiNaC::Gt_detail::pathintegral_term::G_path\(\)](#).

5.1.3.296 [G\(\)](#) [2/2]

```
template<typename T1 , typename T2 , typename T3 >
function GiNaC::G (
    const T1 & x,
    const T2 & s,
    const T3 & y ) [inline]
```

References [GiNaC::G3_SERIAL::serial](#), and [x](#).

5.1.3.297 is_the_function< G_SERIAL >()

```
template<>
bool GiNaC::is_the_function< G_SERIAL > (
    const ex & x ) [inline]
```

References [x](#).

5.1.3.298 psi() [1/4]

```
template<typename T1 >
function GiNaC::psi (
    const T1 & p1 ) [inline]
```

References [GiNaC::psi1_SERIAL::serial](#).

Referenced by [beta_deriv\(\)](#), [lgamma_deriv\(\)](#), [psi1_deriv\(\)](#), [psi1_eval\(\)](#), [psi1_evalf\(\)](#), [psi1_series\(\)](#), [psi2_deriv\(\)](#), [psi2_eval\(\)](#), [psi2_evalf\(\)](#), [psi2_series\(\)](#), [sr_gcd\(\)](#), and [tgamma_deriv\(\)](#).

5.1.3.299 psi() [2/4]

```
template<typename T1 , typename T2 >
function GiNaC::psi (
    const T1 & p1,
    const T2 & p2 ) [inline]
```

References [GiNaC::psi2_SERIAL::serial](#).

5.1.3.300 is_the_function< psi_SERIAL >()

```
template<>
bool GiNaC::is_the_function< psi_SERIAL > (
    const ex & x ) [inline]
```

References [x](#).

5.1.3.301 iterated_integral() [1/2]

```
template<typename T1 , typename T2 >
function GiNaC::iterated_integral (
    const T1 & kernel_lst,
    const T2 & lambda ) [inline]
```

References [GiNaC::iterated_integral2_SERIAL::serial](#).

Referenced by [iterated_integral2_eval\(\)](#), [iterated_integral3_eval\(\)](#), and [iterated_integral_evalf_impl\(\)](#).

5.1.3.302 iterated_integral() [2/2]

```
template<typename T1 , typename T2 , typename T3 >
function GiNaC::iterated_integral (
    const T1 & kernel_lst,
    const T2 & lambda,
    const T3 & N_trunc ) [inline]
```

References [GiNaC::iterated_integral3_SERIAL::serial](#).

5.1.3.303 is_the_function< iterated_integral_SERIAL >()

```
template<>
bool GiNaC::is_the_function< iterated_integral_SERIAL > (
    const ex & x ) [inline]
```

References [x](#).

5.1.3.304 is_order_function()

```
bool GiNaC::is_order_function (
    const ex & e ) [inline]
```

Check whether a function is the Order (O(n)) function.

References [is_ex_the_function](#).

Referenced by [GiNaC::pseries::add_series\(\)](#), [GiNaC::pseries::convert_to_poly\(\)](#), [GiNaC::pseries::derivative\(\)](#), [GiNaC::pseries::is_terminating\(\)](#), [GiNaC::pseries::mul_const\(\)](#), [GiNaC::pseries::mul_series\(\)](#), [GiNaC::pseries::op\(\)](#), [GiNaC::pseries::power_const\(\)](#), [GiNaC::pseries::print_series\(\)](#), [GiNaC::pseries::pseries\(\)](#), [GiNaC::pseries::pseries\(\)](#), and [GiNaC::integral::series\(\)](#).

5.1.3.305 convert_H_to_Li()

```
ex GiNaC::convert_H_to_Li (
    const ex & parameterlst,
    const ex & arg )
```

Converts a given list containing parameters for H in Remiddi/Vermaseren notation into the corresponding [GiNaC](#) functions.

References [m](#), and [x](#).

5.1.3.306 EllipticK_evalf()

```
static ex GiNaC::EllipticK_evalf (
    const ex & k ) [static]
```

References [GiNaC::ex::evalf\(\)](#), [k](#), [GiNaC::info_flags::numeric](#), [Pi](#), and [sqrt\(\)](#).

5.1.3.307 EllipticK_eval()

```
static ex GiNaC::EllipticK_eval (
    const ex & k ) [static]
```

References [_ex0](#), [GiNaC::info_flags::crational](#), [GiNaC::constant::evalf\(\)](#), [k](#), [GiNaC::info_flags::numeric](#), and [Pi](#).

5.1.3.308 EllipticK_deriv()

```
static ex GiNaC::EllipticK_deriv (
    const ex & k,
    unsigned deriv_param ) [static]
```

References [k](#).

5.1.3.309 EllipticK_series()

```
static ex GiNaC::EllipticK_series (
    const ex & k,
    const relational & rel,
    int order,
    unsigned options ) [static]
```

References [_ex0](#), [_ex1](#), [_ex_1](#), [k](#), [GiNaC::subs_options::no_pattern](#), [order](#), [Pi](#), [pow\(\)](#), [GiNaC::ex::series\(\)](#), [GiNaC::symbol::series\(\)](#), and [GiNaC::ex::subs\(\)](#).

5.1.3.310 EllipticK_print_latex()

```
static void GiNaC::EllipticK_print_latex (
    const ex & k,
    const print\_context & c ) [static]
```

References [c](#), and [k](#).

5.1.3.311 REGISTER_FUNCTION() [14/36]

```
GiNaC::REGISTER_FUNCTION (
    EllipticK ,
    evalf_func(EllipticK\_evalf). eval_func(EllipticK\_eval). derivative_func(EllipticK\_deriv).
    series_func(EllipticK\_series). print_func< print\_latex >(EllipticK\_print\_latex). do_not_↵
    evalf_params() )
```

5.1.3.312 EllipticE_evalf()

```
static ex GiNaC::EllipticE_evalf (
    const ex & k ) [static]
```

References [GiNaC::ex::evalf\(\)](#), [k](#), [GiNaC::info_flags::numeric](#), [Pi](#), and [sqrt\(\)](#).

5.1.3.313 EllipticE_eval()

```
static ex GiNaC::EllipticE_eval (
    const ex & k ) [static]
```

References [_ex0](#), [_ex1](#), [_ex_1](#), [GiNaC::info_flags::crational](#), [GiNaC::constant::evalf\(\)](#), [k](#), [GiNaC::info_flags::numeric](#), and [Pi](#).

5.1.3.314 EllipticE_deriv()

```
static ex GiNaC::EllipticE_deriv (
    const ex & k,
    unsigned deriv_param ) [static]
```

References [k](#).

5.1.3.315 EllipticE_series()

```
static ex GiNaC::EllipticE_series (
    const ex & k,
    const relational & rel,
    int order,
    unsigned options ) [static]
```

References [_ex0](#), [_ex1](#), [_ex_1](#), [k](#), [GiNaC::subs_options::no_pattern](#), [order](#), [Pi](#), [pow\(\)](#), [GiNaC::ex::series\(\)](#), [GiNaC::symbol::series\(\)](#), and [GiNaC::ex::subs\(\)](#).

5.1.3.316 EllipticE_print_latex()

```
static void GiNaC::EllipticE_print_latex (
    const ex & k,
    const print\_context & c ) [static]
```

References [c](#), and [k](#).

5.1.3.317 REGISTER_FUNCTION() [15/36]

```
GiNaC::REGISTER_FUNCTION (
    EllipticE ,
    evalf_func(EllipticE\_evalf). eval_func(EllipticE\_eval). derivative_func(EllipticE\_deriv).
    series_func(EllipticE\_series). print_func< print\_latex >(EllipticE\_print\_latex). do_not_↔
    evalf_params() )
```

5.1.3.318 iterated_integral_evalf_impl()

```
static ex GiNaC::iterated_integral_evalf_impl (
    const ex & kernel_lst,
    const ex & lambda,
    const ex & N_trunc ) [static]
```

References [coeff\(\)](#), [Digits](#), [GiNaC::ex::evalf\(\)](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::info\(\)](#), [iterated_integral\(\)](#), [GiNaC::info_flags::list](#), [GiNaC::info_flags::nonnegint](#), [GiNaC::container< C >::nops\(\)](#), [GiNaC::info_flags::numeric](#), and [one](#).

Referenced by [iterated_integral2_evalf\(\)](#), and [iterated_integral3_evalf\(\)](#).

5.1.3.319 iterated_integral2_evalf()

```
static ex GiNaC::iterated_integral2_evalf (
    const ex & kernel_lst,
    const ex & lambda ) [static]
```

References [iterated_integral_evalf_impl\(\)](#).

5.1.3.320 iterated_integral3_evalf()

```
static ex GiNaC::iterated_integral3_evalf (
    const ex & kernel_lst,
    const ex & lambda,
    const ex & N_trunc ) [static]
```

References [iterated_integral_evalf_impl\(\)](#).

5.1.3.321 iterated_integral2_eval()

```
static ex GiNaC::iterated_integral2_eval (
    const ex & kernel_lst,
    const ex & lambda ) [static]
```

References [GiNaC::info_flags::crational](#), [GiNaC::function::evalf\(\)](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::info\(\)](#), [iterated_integral\(\)](#), and [GiNaC::info_flags::numeric](#).

5.1.3.322 iterated_integral3_eval()

```
static ex GiNaC::iterated_integral3_eval (
    const ex & kernel_lst,
    const ex & lambda,
    const ex & N_trunc ) [static]
```

References [GiNaC::info_flags::crational](#), [GiNaC::function::evalf\(\)](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::info\(\)](#), [iterated_integral\(\)](#), and [GiNaC::info_flags::numeric](#).

5.1.3.323 lgamma_evalf()

```
static ex GiNaC::lgamma_evalf (
    const ex & x ) [static]
```

References [lgamma\(\)](#), and [x](#).

5.1.3.324 lgamma_eval()

```
static ex GiNaC::lgamma_eval (
    const ex & x ) [static]
```

Evaluation of $\lgamma(x)$, the natural logarithm of the Gamma function.

Handles integer arguments as a special case.

Exceptions

<i>GiNaC::pole_error("lgamma_eval()"</i>	logarithmic pole",0)

References [_ex_1](#), [factorial\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [is_rational\(\)](#), [lgamma\(\)](#), [log\(\)](#), [GiNaC::info_flags::numeric](#), [GiNaC::info_flags::posint](#), and [x](#).

5.1.3.325 lgamma_deriv()

```
static ex GiNaC::lgamma_deriv (
    const ex & x,
    unsigned deriv_param ) [static]
```

References [GINAC_ASSERT](#), [psi\(\)](#), and [x](#).

5.1.3.326 lgamma_series()

```
static ex GiNaC::lgamma_series (
    const ex & arg,
    const relational & rel,
    int order,
    unsigned options ) [static]
```

References [_ex1](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [lgamma\(\)](#), [log\(\)](#), [m](#), [GiNaC::subs_options::no_pattern](#), [options](#), [order](#), [GiNaC::info_flags::positive](#), [GiNaC::basic::series\(\)](#), and [GiNaC::ex::subs\(\)](#).

5.1.3.327 lgamma_conjugate()

```
static ex GiNaC::lgamma_conjugate (
    const ex & x ) [static]
```

References [GiNaC::ex::conjugate\(\)](#), [GiNaC::ex::imag_part\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_zero\(\)](#), [lgamma\(\)](#), [GiNaC::info_flags::positive](#), and [x](#).

5.1.3.328 REGISTER_FUNCTION() [16/36]

```
GiNaC::REGISTER_FUNCTION (
    lgamma ,
    eval_func(lgamma\_eval). evalf_func(lgamma\_evalf). derivative_func(lgamma\_deriv).
    series_func(lgamma\_series). conjugate_func(lgamma\_conjugate). latex_name("\\log \\Gamma") )
```

5.1.3.329 tgamma_evalf()

```
static ex GiNaC::tgamma_evalf (
    const ex & x ) [static]
```

References [tgamma\(\)](#), and [x](#).

5.1.3.330 tgamma_eval()

```
static ex GiNaC::tgamma_eval (
    const ex & x ) [static]
```

Evaluation of tgamma(x), the true Gamma function.

Knows about integer arguments, half-integer arguments and that's it. Somebody ought to provide some good numerical evaluation some day...

Exceptions

<code>pole_error("tgamma_eval() simple pole",0)</code>

References [_num1_2_p](#), [_num1_p](#), [_num2_p](#), [_num_2_p](#), [abs\(\)](#), [GiNaC::numeric::div\(\)](#), [doublefactorial\(\)](#), [factorial\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::numeric::is_even\(\)](#), [GiNaC::numeric::is_integer\(\)](#), [GiNaC::numeric::is_positive\(\)](#), [is_rational\(\)](#), [n](#), [GiNaC::info_flags::numeric](#), [Pi](#), [pow\(\)](#), [sqrt\(\)](#), [GiNaC::numeric::sub\(\)](#), [tgamma\(\)](#), and [x](#).

5.1.3.331 tgamma_deriv()

```
static ex GiNaC::tgamma_deriv (
    const ex & x,
    unsigned deriv_param ) [static]
```

References [GINAC_ASSERT](#), [psi\(\)](#), [tgamma\(\)](#), and [x](#).

5.1.3.332 tgamma_series()

```
static ex GiNaC::tgamma_series (
    const ex & arg,
    const relational & rel,
    int order,
    unsigned options ) [static]
```

References [_ex1](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [m](#), [GiNaC::subs_options::no_pattern](#), [options](#), [order](#), [GiNaC::info_flags::positive](#), [GiNaC::ex::series\(\)](#), [GiNaC::ex::subs\(\)](#), and [tgamma\(\)](#).

5.1.3.333 tgamma_conjugate()

```
static ex GiNaC::tgamma_conjugate (
    const ex & x ) [static]
```

References [GiNaC::ex::conjugate\(\)](#), [tgamma\(\)](#), and [x](#).

5.1.3.334 REGISTER_FUNCTION() [17/36]

```
GiNaC::REGISTER_FUNCTION (
    tgamma ,
    eval_func(tgamma_eval). evalf_func(tgamma_evalf). derivative_func(tgamma_deriv).
    series_func(tgamma_series). conjugate_func(tgamma_conjugate). latex_name("\\Gamma") )
```

5.1.3.335 beta_evalf()

```
static ex GiNaC::beta_evalf (
    const ex & x,
    const ex & y ) [static]
```

References [exp\(\)](#), [lgamma\(\)](#), and [x](#).

5.1.3.336 beta_eval()

```
static ex GiNaC::beta_eval (
    const ex & x,
    const ex & y ) [static]
```

References [_ex0](#), [_ex1](#), [_num_1_p](#), [evalf\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::numeric::is_integer\(\)](#), [is_integer\(\)](#), [GiNaC::numeric::is_negative\(\)](#), [is_positive\(\)](#), [is_rational\(\)](#), [GiNaC::numeric::is_real\(\)](#), [is_real\(\)](#), [GiNaC::info_flags::numeric](#), [pow\(\)](#), [tgamma\(\)](#), and [x](#).

5.1.3.337 beta_deriv()

```
static ex GiNaC::beta_deriv (
    const ex & x,
    const ex & y,
    unsigned deriv_param ) [static]
```

References [GINAC_ASSERT](#), [psi\(\)](#), and [x](#).

5.1.3.338 beta_series()

```
static ex GiNaC::beta_series (
    const ex & arg1,
    const ex & arg2,
    const relational & rel,
    int order,
    unsigned options ) [static]
```

References [GINAC_ASSERT](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::relational::lhs\(\)](#), [GiNaC::subs_options::no_pattern](#), [options](#), [order](#), [GiNaC::info_flags::positive](#), [GiNaC::ex::subs\(\)](#), and [tgamma\(\)](#).

5.1.3.339 REGISTER_FUNCTION() [18/36]

```
GiNaC::REGISTER_FUNCTION (
    beta ,
    eval_func(beta_eval). evalf_func(beta_evalf). derivative_func(beta_deriv).
    series_func(beta_series). latex_name("\\mathrm{B}"). set_symmetry(sy_symm(0, 1)) )
```

5.1.3.340 psi1_evalf()

```
static ex GiNaC::psi1_evalf (
    const ex & x ) [static]
```

References [GiNaC::basic::hold\(\)](#), [psi\(\)](#), and [x](#).

5.1.3.341 `psi1_eval()`

```
static ex GiNaC::psi1_eval (
    const ex & x ) [static]
```

Evaluation of digamma-function $\psi(x)$.

Somebody ought to provide some good numerical evaluation some day...

References [_ex1_2](#), [_ex2](#), [_num2_p](#), [_num_1_p](#), [Euler](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::numeric::inverse\(\)](#), [GiNaC::numeric::is_integer\(\)](#), [is_integer\(\)](#), [GiNaC::numeric::is_positive\(\)](#), [log\(\)](#), [GiNaC::info_flags::numeric](#), [pow\(\)](#), [psi\(\)](#), and [x](#).

5.1.3.342 `psi1_deriv()`

```
static ex GiNaC::psi1_deriv (
    const ex & x,
    unsigned deriv_param ) [static]
```

References [_ex1](#), [GINAC_ASSERT](#), [psi\(\)](#), and [x](#).

5.1.3.343 `psi1_series()`

```
static ex GiNaC::psi1_series (
    const ex & arg,
    const relational & rel,
    int order,
    unsigned options ) [static]
```

References [_ex1](#), [_ex_1](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [m](#), [GiNaC::subs_options::no_pattern](#), [options](#), [order](#), [GiNaC::info_flags::positive](#), [psi\(\)](#), [GiNaC::power::series\(\)](#), and [GiNaC::ex::subs\(\)](#).

5.1.3.344 `psi2_evalf()`

```
static ex GiNaC::psi2_evalf (
    const ex & n,
    const ex & x ) [static]
```

References [GiNaC::basic::hold\(\)](#), [n](#), [psi\(\)](#), and [x](#).

5.1.3.345 `psi2_eval()`

```
static ex GiNaC::psi2_eval (
    const ex & n,
    const ex & x ) [static]
```

Evaluation of polygamma-function $\psi(n,x)$.

Somebody ought to provide some good numerical evaluation some day...

References [_ex1](#), [_ex1_2](#), [_ex_1](#), [_num1_2_p](#), [_num1_p](#), [_num2_p](#), [_num_1_p](#), [factorial\(\)](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::numeric::is_equal\(\)](#), [GiNaC::numeric::is_integer\(\)](#), [is_integer\(\)](#), [GiNaC::numeric::is_positive\(\)](#), [log\(\)](#), [m](#), [n](#), [GiNaC::info_flags::numeric](#), [GiNaC::info_flags::posint](#), [pow\(\)](#), [psi\(\)](#), [tgamma\(\)](#), [x](#), and [zeta\(\)](#).

5.1.3.346 psi2_deriv()

```
static ex GiNaC::psi2_deriv (
    const ex & n,
    const ex & x,
    unsigned deriv_param ) [static]
```

References [_ex1](#), [GINAC_ASSERT](#), [n](#), [psi\(\)](#), and [x](#).

5.1.3.347 psi2_series()

```
static ex GiNaC::psi2_series (
    const ex & n,
    const ex & arg,
    const relational & rel,
    int order,
    unsigned options ) [static]
```

References [_ex1](#), [_ex_1](#), [factorial\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [m](#), [n](#), [GiNaC::subs_options::no_pattern](#), [options](#), [order](#), [GiNaC::info_flags::positive](#), [psi\(\)](#), and [GiNaC::ex::subs\(\)](#).

5.1.3.348 G2_evalf()

```
static ex GiNaC::G2_evalf (
    const ex & x_,
    const ex & y ) [static]
```

References [_ex0](#), [_ex1](#), [factorial\(\)](#), [G\(\)](#), [GiNaC::basic::hold\(\)](#), [imag\(\)](#), [GiNaC::ex::info\(\)](#), [is_real\(\)](#), [log\(\)](#), [GiNaC::ex::nops\(\)](#), [GiNaC::info_flags::numeric](#), [GiNaC::ex::op\(\)](#), [GiNaC::info_flags::positive](#), [pow\(\)](#), and [x](#).

5.1.3.349 G2_eval()

```
static ex GiNaC::G2_eval (
    const ex & x_,
    const ex & y ) [static]
```

References [_ex0](#), [_ex1](#), [GiNaC::info_flags::crational](#), [factorial\(\)](#), [G\(\)](#), [GiNaC::basic::hold\(\)](#), [imag\(\)](#), [GiNaC::ex::info\(\)](#), [is_real\(\)](#), [log\(\)](#), [GiNaC::ex::nops\(\)](#), [GiNaC::info_flags::numeric](#), [GiNaC::ex::op\(\)](#), [GiNaC::info_flags::positive](#), [pow\(\)](#), and [x](#).

5.1.3.350 G3_evalf()

```
static ex GiNaC::G3_evalf (
    const ex & x_,
    const ex & s_,
    const ex & y ) [static]
```

References [_ex0](#), [_ex1](#), [GiNaC::container< C >::begin\(\)](#), [GiNaC::ex::begin\(\)](#), [GiNaC::ex::end\(\)](#), [factorial\(\)](#), [G\(\)](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::info\(\)](#), [log\(\)](#), [GiNaC::ex::nops\(\)](#), [GiNaC::container< C >::nops\(\)](#), [GiNaC::info_flags::numeric](#), [GiNaC::ex::op\(\)](#), [GiNaC::info_flags::positive](#), [pow\(\)](#), [GiNaC::info_flags::real](#), and [x](#).

5.1.3.351 G3_eval()

```
static ex GiNaC::G3_eval (
    const ex & x_,
    const ex & s_,
    const ex & y ) [static]
```

References [_ex0](#), [_ex1](#), [GiNaC::container< C >::begin\(\)](#), [GiNaC::ex::begin\(\)](#), [GiNaC::info_flags::crational](#), [GiNaC::ex::end\(\)](#), [factorial\(\)](#), [G\(\)](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::info\(\)](#), [log\(\)](#), [GiNaC::ex::nops\(\)](#), [GiNaC::container< C >::nops\(\)](#), [GiNaC::info_flags::numeric](#), [GiNaC::ex::op\(\)](#), [GiNaC::info_flags::positive](#), [pow\(\)](#), [GiNaC::info_flags::real](#), and [x](#).

5.1.3.352 Li_evalf()

```
static ex GiNaC::Li_evalf (
    const ex & m_,
    const ex & x_ ) [static]
```

References [_ex0](#), [_ex1](#), [GiNaC::ex::begin\(\)](#), [GiNaC::ex::evalf\(\)](#), [GiNaC::ex::info\(\)](#), [m](#), [GiNaC::ex::nops\(\)](#), [GiNaC::info_flags::numeric](#), [GiNaC::ex::op\(\)](#), [GiNaC::info_flags::posint](#), and [x](#).

5.1.3.353 Li_eval()

```
static ex GiNaC::Li_eval (
    const ex & m_,
    const ex & x_ ) [static]
```

References [_ex0](#), [_ex1](#), [_ex2](#), [_ex_1](#), [_ex_48](#), [GiNaC::container< C >::append\(\)](#), [GiNaC::ex::begin\(\)](#), [Catalan](#), [GiNaC::info_flags::crational](#), [GINAC_ASSERT](#), [I](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_equal\(\)](#), [log\(\)](#), [m](#), [GiNaC::ex::nops\(\)](#), [GiNaC::info_flags::numeric](#), [Pi](#), [GiNaC::info_flags::posint](#), [pow\(\)](#), [GiNaC::numeric::to_int\(\)](#), [x](#), and [zeta\(\)](#).

5.1.3.354 Li_series()

```
static ex GiNaC::Li_series (
    const ex & m,
    const ex & x,
    const relational & rel,
    int order,
    unsigned options ) [static]
```

References [_ex1](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_zero\(\)](#), [m](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::info_flags::numeric](#), [order](#), [pow\(\)](#), [GiNaC::ex::series\(\)](#), [GiNaC::ex::subs\(\)](#), and [x](#).

5.1.3.355 Li_deriv()

```
static ex GiNaC::Li_deriv (
    const ex & m_,
    const ex & x_,
    unsigned deriv_param ) [static]
```

References [_ex0](#), [GINAC_ASSERT](#), [m](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), and [x](#).

5.1.3.356 Li_print_latex()

```
static void GiNaC::Li_print_latex (
    const ex & m_,
    const ex & x_,
    const print_context & c ) [static]
```

References [GiNaC::ex::begin\(\)](#), [c](#), [GiNaC::ex::end\(\)](#), [m](#), and [x](#).

5.1.3.357 REGISTER_FUNCTION() [19/36]

```
GiNaC::REGISTER_FUNCTION (
    Li ,
    evalf_func(Li_evalf). eval_func(Li_eval). series_func(Li_series). derivative_↔
func(Li_deriv). print_func< print_latex >(Li_print_latex). do_not_evalf_params() )
```

5.1.3.358 S_evalf()

```
static ex GiNaC::S_evalf (
    const ex & n,
    const ex & p,
    const ex & x ) [static]
```

References [GiNaC::ex::evalf\(\)](#), [GiNaC::ex::info\(\)](#), [n](#), [GiNaC::info_flags::posint](#), and [x](#).

5.1.3.359 S_eval()

```
static ex GiNaC::S_eval (
    const ex & n,
    const ex & p,
    const ex & x ) [static]
```

References [_ex0](#), [GiNaC::container< C >::append\(\)](#), [GiNaC::info_flags::crational](#), [factorial\(\)](#), [GiNaC::ex::info\(\)](#), [log\(\)](#), [m](#), [n](#), [GiNaC::info_flags::numeric](#), [GiNaC::info_flags::posint](#), [pow\(\)](#), [to_int\(\)](#), [x](#), and [zeta\(\)](#).

5.1.3.360 S_series()

```
static ex GiNaC::S_series (
    const ex & n,
    const ex & p,
    const ex & x,
    const relational & rel,
    int order,
    unsigned options ) [static]
```

References [_ex1](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_zero\(\)](#), [n](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::info_flags::numeric](#), [options](#), [order](#), [GiNaC::info_flags::posint](#), [pow\(\)](#), [GiNaC::ex::series\(\)](#), [GiNaC::ex::subs\(\)](#), and [x](#).

5.1.3.361 S_deriv()

```
static ex GiNaC::S_deriv (
    const ex & n,
    const ex & p,
    const ex & x,
    unsigned deriv_param ) [static]
```

References [_ex0](#), [GINAC_ASSERT](#), [n](#), and [x](#).

5.1.3.362 S_print_latex()

```
static void GiNaC::S_print_latex (
    const ex & n,
    const ex & p,
    const ex & x,
    const print\_context & c ) [static]
```

References [c](#), [n](#), [GiNaC::ex::print\(\)](#), and [x](#).

5.1.3.363 REGISTER_FUNCTION() [20/36]

```
GiNaC::REGISTER_FUNCTION (
    S ,
    evalf_func(S_evalf). eval_func(S_eval). series_func(S_series). derivative_↔
func(S_deriv). print_func< print\_latex >(S_print_latex). do_not_evalf_params() )
```

5.1.3.364 H_evalf()

```
static ex GiNaC::H_evalf (
    const ex & x1,
    const ex & x2 ) [static]
```

References [GiNaC::container< C >::append\(\)](#), [GiNaC::container< C >::begin\(\)](#), [GiNaC::container< C >::end\(\)](#), [GiNaC::ex::evalf\(\)](#), [l](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::ex::let_op\(\)](#), [m](#), [GiNaC::ex::nops\(\)](#), [GiNaC::container< C >::nops\(\)](#), [GiNaC::ex::op\(\)](#), [GiNaC::container< C >::op\(\)](#), [Pi](#), [GiNaC::ex::subs\(\)](#), and [x](#).

5.1.3.365 H_eval()

```
static ex GiNaC::H_eval (
    const ex & m_,
    const ex & x ) [static]
```

References [_ex0](#), [_ex1](#), [_ex_1](#), [GiNaC::ex::begin\(\)](#), [GiNaC::info_flags::crational](#), [GiNaC::ex::evalf\(\)](#), [factorial\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [log\(\)](#), [m](#), [n](#), [GiNaC::basic::nops\(\)](#), [GiNaC::info_flags::numeric](#), [pow\(\)](#), [step\(\)](#), and [x](#).

5.1.3.366 H_series()

```
static ex GiNaC::H_series (
    const ex & m,
    const ex & x,
    const relational & rel,
    int order,
    unsigned options ) [static]
```

References [m](#), and [x](#).

5.1.3.367 H_deriv()

```
static ex GiNaC::H_deriv (
    const ex & m_,
    const ex & x,
    unsigned deriv_param ) [static]
```

References [_ex0](#), [_ex1](#), [_ex_1](#), [GiNaC::ex::begin\(\)](#), [GINAC_ASSERT](#), [m](#), and [x](#).

5.1.3.368 H_print_latex()

```
static void GiNaC::H_print_latex (
    const ex & m_,
    const ex & x,
    const print_context & c ) [static]
```

References [GiNaC::ex::begin\(\)](#), [c](#), [m](#), [GiNaC::ex::print\(\)](#), and [x](#).

5.1.3.369 REGISTER_FUNCTION() [21/36]

```
GiNaC::REGISTER_FUNCTION (
    H ,
    evalf_func(H_evalf). eval_func(H_eval). series_func(H_series). derivative_↔
func(H_deriv). print_func< print_latex >(H_print_latex). do_not_evalf_params() )
```

5.1.3.370 zeta1_evalf()

```
static ex GiNaC::zeta1_evalf (
    const ex & x ) [static]
```

References [GiNaC::container< C >::begin\(\)](#), [Digits](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::nops\(\)](#), [GiNaC::info_flags::posint](#), [r](#), [x](#), and [zeta\(\)](#).

Referenced by [zeta1_eval\(\)](#).

5.1.3.371 zeta1_eval()

```
static ex GiNaC::zeta1_eval (
    const ex & m ) [static]
```

References [_ex0](#), [_ex_1_2](#), [_num1_p](#), [_num2_p](#), [abs\(\)](#), [bernoulli\(\)](#), [GiNaC::info_flags::crational](#), [factorial\(\)](#), [GiNaC::basic::hold\(\)](#), [GiNaC::numeric::info\(\)](#), [GiNaC::numeric::is_equal\(\)](#), [GiNaC::numeric::is_integer\(\)](#), [GiNaC::numeric::is_zero\(\)](#), [m](#), [GiNaC::info_flags::numeric](#), [GiNaC::info_flags::odd](#), [Pi](#), [GiNaC::info_flags::posint](#), [pow\(\)](#), [zeta\(\)](#), and [zeta1_evalf\(\)](#).

5.1.3.372 zeta1_deriv()

```
static ex GiNaC::zeta1_deriv (
    const ex & m,
    unsigned deriv\_param ) [static]
```

References [_ex0](#), [_ex1](#), [GINAC_ASSERT](#), and [m](#).

5.1.3.373 zeta1_print_latex()

```
static void GiNaC::zeta1_print_latex (
    const ex & m_,
    const print\_context & c ) [static]
```

References [c](#), [m](#), and [GiNaC::ex::print\(\)](#).

5.1.3.374 zeta2_evalf()

```
static ex GiNaC::zeta2_evalf (
    const ex & x,
    const ex & s ) [static]
```

References [GiNaC::container< C >::begin\(\)](#), [evalf\(\)](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::nops\(\)](#), [GiNaC::info_flags::posint](#), [x](#), and [zeta\(\)](#).

5.1.3.375 zeta2_eval()

```
static ex GiNaC::zeta2_eval (
    const ex & m,
    const ex & s_ ) [static]
```

References [GiNaC::basic::hold\(\)](#), [GiNaC::ex::info\(\)](#), [m](#), [GiNaC::info_flags::positive](#), and [zeta\(\)](#).

5.1.3.376 zeta2_deriv()

```
static ex GiNaC::zeta2_deriv (
    const ex & m,
    const ex & s,
    unsigned deriv\_param ) [static]
```

References [_ex0](#), [_ex1](#), [GINAC_ASSERT](#), [GiNaC::ex::info\(\)](#), [m](#), [GiNaC::ex::op\(\)](#), and [GiNaC::info_flags::positive](#).

5.1.3.377 zeta2_print_latex()

```
static void GiNaC::zeta2_print_latex (
    const ex & m_,
    const ex & s_,
    const print\_context & c ) [static]
```

References [GiNaC::container< C >::begin\(\)](#), [GiNaC::ex::begin\(\)](#), [c](#), and [m](#).

5.1.3.378 exp_evalf()

```
static ex GiNaC::exp_evalf (
    const ex & x ) [static]
```

References [exp\(\)](#), [GiNaC::basic::hold\(\)](#), and [x](#).

5.1.3.379 exp_eval()

```
static ex GiNaC::exp_eval (
    const ex & x ) [static]
```

References [_ex1](#), [_ex2](#), [_ex_1](#), [_num0_p](#), [_num1_p](#), [_num2_p](#), [_num3_p](#), [_num4_p](#), [GiNaC::info_flags::crational](#), [exp\(\)](#), [GiNaC::basic::hold\(\)](#), [l](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::numeric::is_equal\(\)](#), [is_ex_the_function](#), [GiNaC::ex::is_zero\(\)](#), [log\(\)](#), [mod\(\)](#), [GiNaC::info_flags::numeric](#), [GiNaC::ex::op\(\)](#), [Pi](#), and [x](#).

5.1.3.380 exp_expand()

```
static ex GiNaC::exp_expand (
    const ex & arg,
    unsigned options ) [static]
```

References [GiNaC::ex::begin\(\)](#), [GiNaC::ex::end\(\)](#), [exp\(\)](#), [GiNaC::ex::expand\(\)](#), [GiNaC::expand_options::expand_function_args](#), [GiNaC::expand_options::expand_transcendental](#), [GiNaC::status_flags::expanded](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::nops\(\)](#), and [options](#).

5.1.3.381 exp_deriv()

```
static ex GiNaC::exp_deriv (
    const ex & x,
    unsigned deriv_param ) [static]
```

References [exp\(\)](#), [GINAC_ASSERT](#), and [x](#).

5.1.3.382 exp_real_part()

```
static ex GiNaC::exp_real_part (
    const ex & x ) [static]
```

References [cos\(\)](#), [exp\(\)](#), [imag_part\(\)](#), [real_part\(\)](#), and [x](#).

5.1.3.383 exp_imag_part()

```
static ex GiNaC::exp_imag_part (
    const ex & x ) [static]
```

References [exp\(\)](#), [imag_part\(\)](#), [real_part\(\)](#), [sin\(\)](#), and [x](#).

5.1.3.384 exp_conjugate()

```
static ex GiNaC::exp_conjugate (
    const ex & x ) [static]
```

References [GiNaC::ex::conjugate\(\)](#), [exp\(\)](#), and [x](#).

5.1.3.385 exp_power()

```
static ex GiNaC::exp_power (
    const ex & x,
    const ex & a ) [static]
```

References [_ex_1](#), [exp\(\)](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::nonnegative](#), [GiNaC::info_flags::real](#), and [x](#).

5.1.3.386 exp_info()

```
static bool GiNaC::exp_info (
    const ex & x,
    unsigned inf ) [static]
```

References [GiNaC::info_flags::expanded](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::nonnegative](#), [GiNaC::info_flags::positive](#), [GiNaC::info_flags::real](#), and [x](#).

5.1.3.387 REGISTER_FUNCTION() [22/36]

```
GiNaC::REGISTER_FUNCTION (
    exp ,
    eval_func(exp\_eval). evalf_func(exp\_evalf). info_func(exp\_info). expand_↵
func(exp\_expand). derivative_func(exp\_deriv). real_part_func(exp\_real\_part). imag_part_↵
func(exp\_imag\_part). conjugate_func(exp\_conjugate). power_func(exp\_power). latex_name("\\exp")
)
```

5.1.3.388 log_evalf()

```
static ex GiNaC::log_evalf (
    const ex & x ) [static]
```

References [GiNaC::basic::hold\(\)](#), [log\(\)](#), and [x](#).

5.1.3.389 log_eval()

```
static ex GiNaC::log_eval (
    const ex & x ) [static]
```

References [_ex0](#), [_ex1](#), [_ex1_2](#), [_ex_1_2](#), [GiNaC::info_flags::crational](#), [exp\(\)](#), [GiNaC::basic::hold\(\)](#), [I](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_equal\(\)](#), [is_ex_the_function](#), [GiNaC::ex::is_zero\(\)](#), [log\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::numeric](#), [GiNaC::ex::op\(\)](#), [Pi](#), [GiNaC::info_flags::rational](#), [GiNaC::info_flags::real](#), and [x](#).

5.1.3.390 log_deriv()

```
static ex GiNaC::log_deriv (
    const ex & x,
    unsigned deriv_param ) [static]
```

References [_ex_1](#), [GINAC_ASSERT](#), and [x](#).

5.1.3.391 log_series()

```
static ex GiNaC::log_series (
    const ex & arg,
    const relational & rel,
    int order,
    unsigned options ) [static]
```

References [_ex0](#), [_ex1](#), [_ex_1](#), [GiNaC::pseries::add_series\(\)](#), [GiNaC::pseries::coeff\(\)](#), [coeff\(\)](#), [csgn\(\)](#), [GiNaC::ex::diff\(\)](#), [GINAC_ASSERT](#), [I](#), [GiNaC::ex::info\(\)](#), [GiNaC::pseries::is_terminating\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::pseries::ldegree\(\)](#), [GiNaC::relational::lhs\(\)](#), [log\(\)](#), [n](#), [GiNaC::info_flags::negative](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::pseries::nops\(\)](#), [options](#), [order](#), [Pi](#), [GiNaC::info_flags::positive](#), [pow\(\)](#), [GiNaC::relational::rhs\(\)](#), [GiNaC::ex::series\(\)](#), [series\(\)](#), [GiNaC::ex::subs\(\)](#), and [GiNaC::series_options::suppress_branchcut](#).

5.1.3.392 log_real_part()

```
static ex GiNaC::log_real_part (
    const ex & x ) [static]
```

References [abs\(\)](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::info\(\)](#), [log\(\)](#), [GiNaC::info_flags::nonnegative](#), and [x](#).

5.1.3.393 log_imag_part()

```
static ex GiNaC::log_imag_part (
    const ex & x ) [static]
```

References [imag_part\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::nonnegative](#), [real_part\(\)](#), and [x](#).

5.1.3.394 log_expand()

```
static ex GiNaC::log_expand (
    const ex & arg,
    unsigned options ) [static]
```

References [_ex1](#), [_ex_1](#), [GiNaC::ex::begin\(\)](#), [GiNaC::ex::end\(\)](#), [expand\(\)](#), [GiNaC::ex::expand\(\)](#), [GiNaC::expand_options::expand_func](#), [GiNaC::expand_options::expand_transcendental](#), [GiNaC::basic::hold\(\)](#), [GiNaC::info_flags::indefinite](#), [GiNaC::ex::info\(\)](#), [log\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::ex::nops\(\)](#), [options](#), [GiNaC::info_flags::positive](#), [GiNaC::status_flags::purely_indefinite](#), and [GiNaC::basic::setflag\(\)](#).

5.1.3.395 log_conjugate()

```
static ex GiNaC::log_conjugate (
    const ex & x ) [static]
```

References [GiNaC::ex::conjugate\(\)](#), [GiNaC::ex::imag_part\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_zero\(\)](#), [log\(\)](#), [GiNaC::info_flags::positive](#), and [x](#).

5.1.3.396 log_info()

```
static bool GiNaC::log_info (
    const ex & x,
    unsigned inf ) [static]
```

References [GiNaC::info_flags::expanded](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::positive](#), [GiNaC::info_flags::real](#), and [x](#).

5.1.3.397 REGISTER_FUNCTION() [23/36]

```
GiNaC::REGISTER_FUNCTION (
    log ,
    eval_func(log\_eval). evalf_func(log\_evalf). info_func(log\_info). expand_↵
func(log\_expand). derivative_func(log\_deriv). series_func(log\_series). real_part_func(log\_real\_part).
imag_part_func(log\_imag\_part). conjugate_func(log\_conjugate). latex_name("\\ln" )
```

5.1.3.398 sin_evalf()

```
static ex GiNaC::sin_evalf (
    const ex & x ) [static]
```

References [GiNaC::basic::hold\(\)](#), [sin\(\)](#), and [x](#).

5.1.3.399 sin_eval()

```
static ex GiNaC::sin_eval (
    const ex & x ) [static]
```

References [_ex0](#), [_ex1](#), [_ex1_2](#), [_ex1_3](#), [_ex1_4](#), [_ex2](#), [_ex3](#), [_ex5](#), [_ex6](#), [_ex60](#), [_ex_1](#), [_ex_1_2](#), [_ex_1_3](#), [_ex_1_4](#), [_num0_p](#), [_num10_p](#), [_num120_p](#), [_num15_p](#), [_num18_p](#), [_num20_p](#), [_num25_p](#), [_num30_p](#), [_num5_p](#), [_num60_p](#), [_num6_p](#), [acos\(\)](#), [asin\(\)](#), [atan\(\)](#), [GiNaC::info_flags::crational](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::numeric::is_equal\(\)](#), [is_ex_the_function](#), [mod\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::numeric](#), [GiNaC::ex::op\(\)](#), [Pi](#), [sin\(\)](#), [sqrt\(\)](#), and [x](#).

5.1.3.400 sin_deriv()

```
static ex GiNaC::sin_deriv (
    const ex & x,
    unsigned deriv_param ) [static]
```

References [cos\(\)](#), [GINAC_ASSERT](#), and [x](#).

5.1.3.401 sin_real_part()

```
static ex GiNaC::sin_real_part (
    const ex & x ) [static]
```

References [cosh\(\)](#), [imag_part\(\)](#), [real_part\(\)](#), [sin\(\)](#), and [x](#).

5.1.3.402 sin_imag_part()

```
static ex GiNaC::sin_imag_part (
    const ex & x ) [static]
```

References [cos\(\)](#), [imag_part\(\)](#), [real_part\(\)](#), [sinh\(\)](#), and [x](#).

5.1.3.403 sin_conjugate()

```
static ex GiNaC::sin_conjugate (
    const ex & x ) [static]
```

References [GiNaC::ex::conjugate\(\)](#), [sin\(\)](#), and [x](#).

5.1.3.404 trig_info()

```
static bool GiNaC::trig_info (
    const ex & x,
    unsigned inf ) [static]
```

References [GiNaC::info_flags::expanded](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::real](#), and [x](#).

5.1.3.405 REGISTER_FUNCTION() [24/36]

```
GiNaC::REGISTER_FUNCTION (
    sin ,
    eval_func(sin\_eval). evalf_func(sin\_evalf). info_func(trig\_info). derivative_↔
func(sin\_deriv). real_part_func(sin\_real\_part). imag_part_func(sin\_imag\_part). conjugate_↔
func(sin\_conjugate). latex_name("\\sin" ) )
```

5.1.3.406 `cos_evalf()`

```
static ex GiNaC::cos_evalf (
    const ex & x ) [static]
```

References [cos\(\)](#), [GiNaC::basic::hold\(\)](#), and [x](#).

5.1.3.407 `cos_eval()`

```
static ex GiNaC::cos_eval (
    const ex & x ) [static]
```

References [_ex0](#), [_ex1](#), [_ex1_2](#), [_ex1_3](#), [_ex1_4](#), [_ex2](#), [_ex3](#), [_ex5](#), [_ex6](#), [_ex60](#), [_ex_1](#), [_ex_1_2](#), [_ex_1_3](#), [_ex_1_4](#), [_num0_p](#), [_num10_p](#), [_num120_p](#), [_num12_p](#), [_num15_p](#), [_num20_p](#), [_num24_p](#), [_num25_p](#), [_num30_p](#), [_num5_p](#), [_num60_p](#), [acos\(\)](#), [asin\(\)](#), [atan\(\)](#), [cos\(\)](#), [GiNaC::info_flags::crational](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::numeric::is_equal\(\)](#), [is_ex_the_function](#), [mod\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::numeric](#), [GiNaC::ex::op\(\)](#), [Pi](#), [sqrt\(\)](#), and [x](#).

5.1.3.408 `cos_deriv()`

```
static ex GiNaC::cos_deriv (
    const ex & x,
    unsigned deriv_param ) [static]
```

References [GINAC_ASSERT](#), [sin\(\)](#), and [x](#).

5.1.3.409 `cos_real_part()`

```
static ex GiNaC::cos_real_part (
    const ex & x ) [static]
```

References [cos\(\)](#), [cosh\(\)](#), [imag_part\(\)](#), [real_part\(\)](#), and [x](#).

5.1.3.410 `cos_imag_part()`

```
static ex GiNaC::cos_imag_part (
    const ex & x ) [static]
```

References [imag_part\(\)](#), [real_part\(\)](#), [sin\(\)](#), [sinh\(\)](#), and [x](#).

5.1.3.411 `cos_conjugate()`

```
static ex GiNaC::cos_conjugate (
    const ex & x ) [static]
```

References [GiNaC::ex::conjugate\(\)](#), [cos\(\)](#), and [x](#).

5.1.3.412 REGISTER_FUNCTION() [25/36]

```
GiNaC::REGISTER_FUNCTION (
    cos ,
    eval_func(cos_eval). info_func(trig_info). evalf_func(cos_evalf). derivative_↵
func(cos_deriv). real_part_func(cos_real_part). imag_part_func(cos_imag_part). conjugate_↵
func(cos_conjugate). latex_name("\\cos") )
```

5.1.3.413 tan_evalf()

```
static ex GiNaC::tan_evalf (
    const ex & x ) [static]
```

References [GiNaC::basic::hold\(\)](#), [tan\(\)](#), and [x](#).

5.1.3.414 tan_eval()

```
static ex GiNaC::tan_eval (
    const ex & x ) [static]
```

References [_ex0](#), [_ex1](#), [_ex1_3](#), [_ex2](#), [_ex3](#), [_ex60](#), [_ex_1](#), [_ex_1_2](#), [_num0_p](#), [_num10_p](#), [_num15_p](#), [_num20_p](#), [_num25_p](#), [_num30_p](#), [_num5_p](#), [_num60_p](#), [acos\(\)](#), [asin\(\)](#), [atan\(\)](#), [GiNaC::info_flags::crational](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::numeric::is_equal\(\)](#), [is_ex_the_function](#), [mod\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::numeric](#), [GiNaC::ex::op\(\)](#), [Pi](#), [sqrt\(\)](#), [tan\(\)](#), and [x](#).

5.1.3.415 tan_deriv()

```
static ex GiNaC::tan_deriv (
    const ex & x,
    unsigned deriv_param ) [static]
```

References [_ex1](#), [_ex2](#), [GINAC_ASSERT](#), [tan\(\)](#), and [x](#).

5.1.3.416 tan_real_part()

```
static ex GiNaC::tan_real_part (
    const ex & x ) [static]
```

References [imag_part\(\)](#), [real_part\(\)](#), [tan\(\)](#), and [x](#).

5.1.3.417 tan_imag_part()

```
static ex GiNaC::tan_imag_part (
    const ex & x ) [static]
```

References [imag_part\(\)](#), [real_part\(\)](#), [tan\(\)](#), [tanh\(\)](#), and [x](#).

5.1.3.418 tan_series()

```
static ex GiNaC::tan_series (
    const ex & x,
    const relational & rel,
    int order,
    unsigned options ) [static]
```

References [cos\(\)](#), [GINAC_ASSERT](#), [GiNaC::relational::lhs\(\)](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::info_flags::odd](#), [options](#), [order](#), [Pi](#), [sin\(\)](#), [GiNaC::ex::subs\(\)](#), and [x](#).

5.1.3.419 tan_conjugate()

```
static ex GiNaC::tan_conjugate (
    const ex & x ) [static]
```

References [GiNaC::ex::conjugate\(\)](#), [tan\(\)](#), and [x](#).

5.1.3.420 REGISTER_FUNCTION() [26/36]

```
GiNaC::REGISTER_FUNCTION (
    tan ,
    eval_func(tan_eval). evalf_func(tan_evalf). info_func(trig_info). derivative↵
    _func(tan_deriv). series_func(tan_series). real_part_func(tan_real_part). imag_part_↵
    func(tan_imag_part). conjugate_func(tan_conjugate). latex_name("\\tan" )
```

5.1.3.421 asin_evalf()

```
static ex GiNaC::asin_evalf (
    const ex & x ) [static]
```

References [asin\(\)](#), [GiNaC::basic::hold\(\)](#), and [x](#).

5.1.3.422 asin_eval()

```
static ex GiNaC::asin_eval (
    const ex & x ) [static]
```

References [_ex1](#), [_ex1_2](#), [_ex_1](#), [_ex_1_2](#), [asin\(\)](#), [GiNaC::info_flags::crational](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::numeric](#), [Pi](#), and [x](#).

5.1.3.423 asin_deriv()

```
static ex GiNaC::asin_deriv (
    const ex & x,
    unsigned deriv_param ) [static]
```

References [_ex2](#), [_ex_1_2](#), [GINAC_ASSERT](#), and [x](#).

5.1.3.424 asin_conjugate()

```
static ex GiNaC::asin_conjugate (
    const ex & x ) [static]
```

References [_num1_p](#), [_num_1_p](#), [asin\(\)](#), [GiNaC::ex::conjugate\(\)](#), [GiNaC::ex::imag_part\(\)](#), [GiNaC::ex::is_zero\(\)](#), and [x](#).

5.1.3.425 asin_info()

```
static bool GiNaC::asin_info (
    const ex & x,
    unsigned inf ) [static]
```

References [GiNaC::info_flags::expanded](#), [GiNaC::ex::info\(\)](#), and [x](#).

5.1.3.426 REGISTER_FUNCTION() [27/36]

```
GiNaC::REGISTER_FUNCTION (
    asin ,
    eval_func(asin\_eval). evalf_func(asin\_evalf). info_func(asin\_info). derivative↔
_func(asin\_deriv). conjugate_func(asin\_conjugate). latex_name("\\arcsin") )
```

5.1.3.427 acos_evalf()

```
static ex GiNaC::acos_evalf (
    const ex & x ) [static]
```

References [acos\(\)](#), [GiNaC::basic::hold\(\)](#), and [x](#).

5.1.3.428 acos_eval()

```
static ex GiNaC::acos_eval (
    const ex & x ) [static]
```

References [_ex0](#), [_ex1](#), [_ex1_2](#), [_ex1_3](#), [_ex_1](#), [_ex_1_2](#), [acos\(\)](#), [GiNaC::info_flags::crational](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::numeric](#), [Pi](#), and [x](#).

5.1.3.429 acos_deriv()

```
static ex GiNaC::acos_deriv (
    const ex & x,
    unsigned deriv_param ) [static]
```

References [_ex2](#), [_ex_1_2](#), [GINAC_ASSERT](#), and [x](#).

5.1.3.430 `acos_conjugate()`

```
static ex GiNaC::acos_conjugate (
    const ex & x ) [static]
```

References [_num1_p](#), [_num_1_p](#), [acos\(\)](#), [GiNaC::ex::conjugate\(\)](#), [GiNaC::ex::imag_part\(\)](#), [GiNaC::ex::is_zero\(\)](#), and [x](#).

5.1.3.431 `REGISTER_FUNCTION()` [28/36]

```
GiNaC::REGISTER_FUNCTION (
    acos ,
    eval_func(acos\_eval). evalf_func(acos\_evalf). info_func(asin\_info). derivative←
_func(acos\_deriv). conjugate_func(acos\_conjugate). latex_name("\\arccos") )
```

5.1.3.432 `atan_evalf()`

```
static ex GiNaC::atan_evalf (
    const ex & x ) [static]
```

References [atan\(\)](#), [GiNaC::basic::hold\(\)](#), and [x](#).

5.1.3.433 `atan_eval()`

```
static ex GiNaC::atan_eval (
    const ex & x ) [static]
```

References [_ex0](#), [_ex1](#), [_ex1_4](#), [_ex_1](#), [_ex_1_4](#), [atan\(\)](#), [GiNaC::info_flags::crational](#), [GiNaC::basic::hold\(\)](#), [I](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::numeric](#), [Pi](#), and [x](#).

5.1.3.434 `atan_deriv()`

```
static ex GiNaC::atan_deriv (
    const ex & x,
    unsigned deriv_param ) [static]
```

References [_ex1](#), [_ex2](#), [_ex_1](#), [GINAC_ASSERT](#), and [x](#).

5.1.3.435 `atan_series()`

```
static ex GiNaC::atan_series (
    const ex & arg,
    const relational & rel,
    int order,
    unsigned options ) [static]
```

References [_ex0](#), [_ex1](#), [_ex1_2](#), [_ex_1](#), [_ex_1_2](#), [abs\(\)](#), [atan\(\)](#), [csgn\(\)](#), [GINAC_ASSERT](#), [I](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::relational::lhs\(\)](#), [log\(\)](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::ex::op\(\)](#), [options](#), [order](#), [Pi](#), [GiNaC::info_flags::real](#), [GiNaC::relational::rhs\(\)](#), [series\(\)](#), [GiNaC::ex::subs\(\)](#), and [GiNaC::series_options::suppress_branchcut](#).

5.1.3.436 atan_conjugate()

```
static ex GiNaC::atan_conjugate (
    const ex & x ) [static]
```

References [_num1_p](#), [_num_1_p](#), [atan\(\)](#), [GiNaC::ex::conjugate\(\)](#), [GiNaC::ex::imag_part\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::numeric::is_zero\(\)](#), [GiNaC::info_flags::real](#), [GiNaC::ex::real_part\(\)](#), and [x](#).

5.1.3.437 atan_info()

```
static bool GiNaC::atan_info (
    const ex & x,
    unsigned inf ) [static]
```

References [GiNaC::info_flags::expanded](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::nonnegative](#), [GiNaC::info_flags::positive](#), [GiNaC::info_flags::real](#), and [x](#).

5.1.3.438 REGISTER_FUNCTION() [29/36]

```
GiNaC::REGISTER_FUNCTION (
    atan ,
    eval_func(atan\_eval). evalf_func(atan\_evalf). info_func(atan\_info). derivative↔
    _func(atan\_deriv). series_func(atan\_series). conjugate_func(atan\_conjugate). latex_name("\\arctan")
)
```

5.1.3.439 atan2_evalf()

```
static ex GiNaC::atan2_evalf (
    const ex & y,
    const ex & x ) [static]
```

References [atan\(\)](#), and [x](#).

5.1.3.440 atan2_eval()

```
static ex GiNaC::atan2_eval (
    const ex & y,
    const ex & x ) [static]
```

References [_ex0](#), [_ex1_2](#), [_ex1_4](#), [_ex_1_2](#), [_ex_1_4](#), [atan\(\)](#), [GiNaC::info_flags::crational](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::info_flags::negative](#), [Pi](#), [GiNaC::info_flags::positive](#), [GiNaC::info_flags::real](#), and [x](#).

5.1.3.441 atan2_deriv()

```
static ex GiNaC::atan2_deriv (
    const ex & y,
    const ex & x,
    unsigned deriv_param ) [static]
```

References [_ex2](#), [_ex_1](#), [GINAC_ASSERT](#), and [x](#).

5.1.3.442 atan2_info()

```
static bool GiNaC::atan2_info (
    const ex & y,
    const ex & x,
    unsigned inf ) [static]
```

References [GiNaC::info_flags::expanded](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::nonnegative](#), [GiNaC::info_flags::positive](#), [GiNaC::info_flags::real](#), and [x](#).

5.1.3.443 REGISTER_FUNCTION() [30/36]

```
GiNaC::REGISTER_FUNCTION (
    atan2 ,
    eval_func(atan2\_eval). evalf_func(atan2\_evalf). info_func(atan2\_info). evalf_↔
func(atan2\_evalf). derivative_func(atan2\_deriv) )
```

5.1.3.444 sinh_evalf()

```
static ex GiNaC::sinh_evalf (
    const ex & x ) [static]
```

References [GiNaC::basic::hold\(\)](#), [sinh\(\)](#), and [x](#).

5.1.3.445 sinh_eval()

```
static ex GiNaC::sinh_eval (
    const ex & x ) [static]
```

References [_ex0](#), [_ex1](#), [_ex2](#), [_ex_1_2](#), [acosh\(\)](#), [asinh\(\)](#), [atanh\(\)](#), [GiNaC::info_flags::crational](#), [GiNaC::basic::hold\(\)](#), [I](#), [GiNaC::ex::info\(\)](#), [is_ex_the_function](#), [GiNaC::ex::is_zero\(\)](#), [is_zero\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::numeric](#), [GiNaC::ex::op\(\)](#), [Pi](#), [real\(\)](#), [sin\(\)](#), [sinh\(\)](#), [sqrt\(\)](#), and [x](#).

5.1.3.446 sinh_deriv()

```
static ex GiNaC::sinh_deriv (
    const ex & x,
    unsigned deriv_param ) [static]
```

References [cosh\(\)](#), [GINAC_ASSERT](#), and [x](#).

5.1.3.447 sinh_real_part()

```
static ex GiNaC::sinh_real_part (
    const ex & x ) [static]
```

References [cos\(\)](#), [imag_part\(\)](#), [real_part\(\)](#), [sinh\(\)](#), and [x](#).

5.1.3.448 sinh_imag_part()

```
static ex GiNaC::sinh_imag_part (
    const ex & x ) [static]
```

References [cosh\(\)](#), [imag_part\(\)](#), [real_part\(\)](#), [sin\(\)](#), and [x](#).

5.1.3.449 sinh_conjugate()

```
static ex GiNaC::sinh_conjugate (
    const ex & x ) [static]
```

References [GiNaC::ex::conjugate\(\)](#), [sinh\(\)](#), and [x](#).

5.1.3.450 REGISTER_FUNCTION() [31/36]

```
GiNaC::REGISTER_FUNCTION (
    sinh ,
    eval_func(sinh\_eval). evalf_func(sinh\_evalf). info_func(atan\_info). derivative↔
    _func(sinh\_deriv). real_part_func(sinh\_real\_part). imag_part_func(sinh\_imag\_part). conjugate↔
    _func(sinh\_conjugate). latex_name("\\sinh") )
```

5.1.3.451 cosh_evalf()

```
static ex GiNaC::cosh_evalf (
    const ex & x ) [static]
```

References [cosh\(\)](#), [GiNaC::basic::hold\(\)](#), and [x](#).

5.1.3.452 cosh_eval()

```
static ex GiNaC::cosh_eval (
    const ex & x ) [static]
```

References [_ex1](#), [_ex2](#), [_ex_1_2](#), [acosh\(\)](#), [asinh\(\)](#), [atanh\(\)](#), [cos\(\)](#), [cosh\(\)](#), [GiNaC::info_flags::crational](#), [GiNaC::basic::hold\(\)](#), [I](#), [GiNaC::ex::info\(\)](#), [is_ex_the_function](#), [GiNaC::ex::is_zero\(\)](#), [is_zero\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::numeric](#), [GiNaC::ex::op\(\)](#), [Pi](#), [real\(\)](#), [sqrt\(\)](#), and [x](#).

5.1.3.453 cosh_deriv()

```
static ex GiNaC::cosh_deriv (
    const ex & x,
    unsigned deriv_param ) [static]
```

References [GINAC_ASSERT](#), [sinh\(\)](#), and [x](#).

5.1.3.454 cosh_real_part()

```
static ex GiNaC::cosh_real_part (
    const ex & x ) [static]
```

References [cos\(\)](#), [cosh\(\)](#), [imag_part\(\)](#), [real_part\(\)](#), and [x](#).

5.1.3.455 cosh_imag_part()

```
static ex GiNaC::cosh_imag_part (
    const ex & x ) [static]
```

References [imag_part\(\)](#), [real_part\(\)](#), [sin\(\)](#), [sinh\(\)](#), and [x](#).

5.1.3.456 cosh_conjugate()

```
static ex GiNaC::cosh_conjugate (
    const ex & x ) [static]
```

References [GiNaC::ex::conjugate\(\)](#), [cosh\(\)](#), and [x](#).

5.1.3.457 REGISTER_FUNCTION() [32/36]

```
GiNaC::REGISTER_FUNCTION (
    cosh ,
    eval_func(cosh\_eval). evalf_func(cosh\_evalf). info_func(exp\_info). derivative↔
    _func(cosh\_deriv). real_part_func(cosh\_real\_part). imag_part_func(cosh\_imag\_part). conjugate↔
    _func(cosh\_conjugate). latex_name("\\cosh") )
```

5.1.3.458 tanh_evalf()

```
static ex GiNaC::tanh_evalf (
    const ex & x ) [static]
```

References [GiNaC::basic::hold\(\)](#), [tanh\(\)](#), and [x](#).

5.1.3.459 tanh_eval()

```
static ex GiNaC::tanh_eval (
    const ex & x ) [static]
```

References [_ex0](#), [_ex1](#), [_ex2](#), [_ex_1](#), [_ex_1_2](#), [acosh\(\)](#), [asinh\(\)](#), [atanh\(\)](#), [GiNaC::info_flags::crational](#), [GiNaC::basic::hold\(\)](#), [l](#), [GiNaC::ex::info\(\)](#), [is_ex_the_function](#), [GiNaC::ex::is_zero\(\)](#), [is_zero\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::numeric](#), [GiNaC::ex::op\(\)](#), [Pi](#), [real\(\)](#), [sqrt\(\)](#), [tan\(\)](#), [tanh\(\)](#), and [x](#).

5.1.3.460 tanh_deriv()

```
static ex GiNaC::tanh_deriv (
    const ex & x,
    unsigned deriv_param ) [static]
```

References [_ex1](#), [_ex2](#), [GINAC_ASSERT](#), [tanh\(\)](#), and [x](#).

5.1.3.461 tanh_series()

```
static ex GiNaC::tanh_series (
    const ex & x,
    const relational & rel,
    int order,
    unsigned options ) [static]
```

References [cosh\(\)](#), [GINAC_ASSERT](#), [I](#), [GiNaC::relational::lhs\(\)](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::info_flags::odd](#), [options](#), [order](#), [Pi](#), [sinh\(\)](#), [GiNaC::ex::subs\(\)](#), and [x](#).

5.1.3.462 tanh_real_part()

```
static ex GiNaC::tanh_real_part (
    const ex & x ) [static]
```

References [imag_part\(\)](#), [real_part\(\)](#), [tan\(\)](#), [tanh\(\)](#), and [x](#).

5.1.3.463 tanh_imag_part()

```
static ex GiNaC::tanh_imag_part (
    const ex & x ) [static]
```

References [imag_part\(\)](#), [real_part\(\)](#), [tan\(\)](#), [tanh\(\)](#), and [x](#).

5.1.3.464 tanh_conjugate()

```
static ex GiNaC::tanh_conjugate (
    const ex & x ) [static]
```

References [GiNaC::ex::conjugate\(\)](#), [tanh\(\)](#), and [x](#).

5.1.3.465 REGISTER_FUNCTION() [33/36]

```
GiNaC::REGISTER_FUNCTION (
    tanh ,
    eval_func(tanh\_eval). evalf_func(tanh\_evalf). info_func(atan\_info). derivative↔
_func(tanh\_deriv). series_func(tanh\_series). real_part_func(tanh\_real\_part). imag_part_↔
func(tanh\_imag\_part). conjugate_func(tanh\_conjugate). latex_name("\\tanh" ) )
```

5.1.3.466 asinh_evalf()

```
static ex GiNaC::asinh_evalf (
    const ex & x ) [static]
```

References [asinh\(\)](#), [GiNaC::basic::hold\(\)](#), and [x](#).

5.1.3.467 asinh_eval()

```
static ex GiNaC::asinh_eval (
    const ex & x ) [static]
```

References [_ex0](#), [asinh\(\)](#), [GiNaC::info_flags::crational](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::numeric](#), and [x](#).

5.1.3.468 asinh_deriv()

```
static ex GiNaC::asinh_deriv (
    const ex & x,
    unsigned deriv_param ) [static]
```

References [_ex1](#), [_ex2](#), [_ex_1_2](#), [GINAC_ASSERT](#), and [x](#).

5.1.3.469 asinh_conjugate()

```
static ex GiNaC::asinh_conjugate (
    const ex & x ) [static]
```

References [_num1_p](#), [_num_1_p](#), [asinh\(\)](#), [GiNaC::ex::conjugate\(\)](#), [GiNaC::ex::imag_part\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::numeric::is_zero\(\)](#), [GiNaC::info_flags::real](#), [GiNaC::ex::real_part\(\)](#), and [x](#).

5.1.3.470 REGISTER_FUNCTION() [34/36]

```
GiNaC::REGISTER_FUNCTION (
    asinh ,
    eval_func(asinh\_eval) . evalf_func(asinh\_evalf) . info_func(atan\_info) . derivative←
    _func(asinh\_deriv) . conjugate_func(asinh\_conjugate) )
```

5.1.3.471 acosh_evalf()

```
static ex GiNaC::acosh_evalf (
    const ex & x ) [static]
```

References [acosh\(\)](#), [GiNaC::basic::hold\(\)](#), and [x](#).

5.1.3.472 `acosh_eval()`

```
static ex GiNaC::acosh_eval (
    const ex & x ) [static]
```

References [_ex0](#), [_ex1](#), [_ex_1](#), [acosh\(\)](#), [GiNaC::info_flags::crational](#), [GiNaC::basic::hold\(\)](#), [I](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::numeric](#), [Pi](#), and [x](#).

5.1.3.473 `acosh_deriv()`

```
static ex GiNaC::acosh_deriv (
    const ex & x,
    unsigned deriv_param ) [static]
```

References [_ex1](#), [_ex_1](#), [_ex_1_2](#), [GINAC_ASSERT](#), and [x](#).

5.1.3.474 `acosh_conjugate()`

```
static ex GiNaC::acosh_conjugate (
    const ex & x ) [static]
```

References [_num1_p](#), [acosh\(\)](#), [GiNaC::ex::conjugate\(\)](#), [GiNaC::ex::imag_part\(\)](#), [GiNaC::ex::is_zero\(\)](#), and [x](#).

5.1.3.475 `REGISTER_FUNCTION()` [35/36]

```
GiNaC::REGISTER_FUNCTION (
    acosh ,
    eval_func(acosh\_eval). evalf_func(acosh\_evalf). info_func(asin\_info). derivative←
    _func(acosh\_deriv). conjugate_func(acosh\_conjugate) )
```

5.1.3.476 `atanh_evalf()`

```
static ex GiNaC::atanh_evalf (
    const ex & x ) [static]
```

References [atanh\(\)](#), [GiNaC::basic::hold\(\)](#), and [x](#).

5.1.3.477 `atanh_eval()`

```
static ex GiNaC::atanh_eval (
    const ex & x ) [static]
```

References [_ex0](#), [_ex1](#), [_ex_1](#), [atanh\(\)](#), [GiNaC::info_flags::crational](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::numeric](#), and [x](#).

5.1.3.478 atanh_deriv()

```
static ex GiNaC::atanh_deriv (
    const ex & x,
    unsigned deriv_param ) [static]
```

References [_ex1](#), [_ex2](#), [_ex_1](#), [GINAC_ASSERT](#), and [x](#).

5.1.3.479 atanh_series()

```
static ex GiNaC::atanh_series (
    const ex & arg,
    const relational & rel,
    int order,
    unsigned options ) [static]
```

References [_ex0](#), [_ex1](#), [_ex1_2](#), [_ex_1](#), [_ex_1_2](#), [abs\(\)](#), [atanh\(\)](#), [csgn\(\)](#), [GINAC_ASSERT](#), [I](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::relational::lhs\(\)](#), [log\(\)](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::ex::op\(\)](#), [options](#), [order](#), [Pi](#), [GiNaC::info_flags::real](#), [GiNaC::relational::rhs\(\)](#), [series\(\)](#), [GiNaC::ex::subs\(\)](#), and [GiNaC::series_options::suppress_branchcut](#).

5.1.3.480 atanh_conjugate()

```
static ex GiNaC::atanh_conjugate (
    const ex & x ) [static]
```

References [_num1_p](#), [_num_1_p](#), [atanh\(\)](#), [GiNaC::ex::conjugate\(\)](#), [GiNaC::ex::imag_part\(\)](#), [GiNaC::ex::is_zero\(\)](#), and [x](#).

5.1.3.481 REGISTER_FUNCTION() [36/36]

```
GiNaC::REGISTER_FUNCTION (
    atanh ,
    eval_func(atanh_eval). evalf_func(atanh_evalf). info_func(asin_info). derivative←
_func(atanh_deriv). series_func(atanh_series). conjugate_func(atanh_conjugate) )
```

5.1.3.482 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [12/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    integral ,
    basic ,
    print_func< print_dflt > &::do_print. print_func< print_python > &::do_print.
print_func< print_latex > &::do_print_latex )
```

5.1.3.483 subsvalue()

```
ex GiNaC::subsvalue (
    const ex & var,
    const ex & value,
    const ex & fun )
```

References [GiNaC::ex::evalf\(\)](#), [GiNaC::ex::subs\(\)](#), and [value](#).

Referenced by [adaptivesimpson\(\)](#).

5.1.3.484 adaptivesimpson()

```
GiNaC::ex GiNaC::adaptivesimpson (
    const ex & x,
    const ex & a_in,
    const ex & b_in,
    const ex & f,
    const ex & error )
```

Numeric integration routine based upon the "Adaptive Quadrature" one in "Numerical Analysis" by Burden and Faires.

Parameters are integration variable, left boundary, right boundary, function to be integrated and the relative integration error. The function should evalf into a number after substituting the integration variable by a number. Another thing to note is that this implementation is no good at integrating functions with discontinuities.

References [abs\(\)](#), [GiNaC::ex::evalf\(\)](#), [GiNaC::integral::max_integration_level](#), [GiNaC::ex::subs\(\)](#), [subsvalue\(\)](#), and [x](#).

Referenced by [GiNaC::integral::evalf\(\)](#).

5.1.3.485 GINAC_BIND_UNARCHIVER() [22/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    integral )
```

5.1.3.486 GINAC_DECLARE_UNARCHIVER() [23/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    integral )
```

5.1.3.487 ifactor()

```
ex GiNaC::ifactor (
    const numeric & n )
```

Returns the decomposition of the positive integer n into prime numbers in the form $\text{lst}(\text{lst}(p_1, \dots, p_r), \text{lst}(a_1, \dots, a_r))$ such that $n = p_1^{a_1} \dots p_r^{a_r}$.

References [GiNaC::container< C >::append\(\)](#), [irem\(\)](#), [n](#), and [GiNaC::info_flags::prime](#).

Referenced by [is_discriminant_of_quadratic_number_field\(\)](#), and [kronecker_symbol\(\)](#).

5.1.3.488 is_discriminant_of_quadratic_number_field()

```
bool GiNaC::is_discriminant_of_quadratic_number_field (
    const numeric & n )
```

Returns true if the integer n is either one or the discriminant of a quadratic number field.

Returns false otherwise.

Ref.: Toshitsune Miyake, Modular Forms, Chapter 3.1

References [abs\(\)](#), [ifactor\(\)](#), [is_discriminant_of_quadratic_number_field\(\)](#), [GiNaC::numeric::is_odd\(\)](#), [mod\(\)](#), [n](#), [GiNaC::container< C >::nops\(\)](#), and [GiNaC::container< C >::op\(\)](#).

Referenced by [is_discriminant_of_quadratic_number_field\(\)](#).

5.1.3.489 kronecker_symbol()

```
numeric GiNaC::kronecker_symbol (
    const numeric & a,
    const numeric & n )
```

Returns the Kronecker symbol a : integer n : integer.

This routine defines $\text{kronecker\_symbol}(1,0) = 1$ $\text{kronecker\_symbol}(-1,0) = 1$ $\text{kronecker\_symbol}(a,0) = 0$, $a \neq 1, -1$

In particular $\text{kronecker\_symbol}(-1,0) = 1$ (in agreement with Sage)

Ref.: Toshitsune Miyake, Modular Forms, Chapter 3.1

References [GiNaC::container< C >::begin\(\)](#), [GiNaC::container< C >::end\(\)](#), [ifactor\(\)](#), [GiNaC::numeric::is_even\(\)](#), [n](#), [GiNaC::container< C >::op\(\)](#), [pow\(\)](#), and [unit](#).

Referenced by [primitive_dirichlet_character\(\)](#).

5.1.3.490 primitive_dirichlet_character()

```
numeric GiNaC::primitive_dirichlet_character (
    const numeric & n,
    const numeric & a )
```

Defines a primitive Dirichlet character through the Kronecker symbol.

n : integer a : discriminant of a quadratic field $|a|$: conductor

The character takes the values -1,0,1.

References [kronecker_symbol\(\)](#), and [n](#).

Referenced by [dirichlet_character\(\)](#), and [generalised_Bernoulli_number\(\)](#).

5.1.3.491 dirichlet_character()

```
numeric GiNaC::dirichlet_character (
    const numeric & n,
    const numeric & a,
    const numeric & N )
```

Defines a Dirichlet character through the Kronecker symbol.

n : integer a : discriminant of a quadratic field $|a|$: conductor N : modulus, needs to be multiple of $|a|$

The character takes the values -1,0,1.

References [gcd\(\)](#), [n](#), and [primitive_dirichlet_character\(\)](#).

5.1.3.492 generalised_Bernoulli_number()

```
numeric GiNaC::generalised_Bernoulli_number (
    const numeric & k,
    const numeric & b )
```

The generalised Bernoulli number.

k: index / modular weight

b: discriminant of a quadratic field, defines primitive character ψ $M=|b|$: conductor of primitive character ψ

The generalised Bernoulli number is computed from the series expansion of the generating function. The generating function is given in eq.(34), arXiv:1704.08895

References [abs\(\)](#), [GiNaC::ex::coeff\(\)](#), [exp\(\)](#), [factorial\(\)](#), [k](#), [primitive_dirichlet_character\(\)](#), [GiNaC::ex::series\(\)](#), [series_to_poly\(\)](#), [GiNaC::numeric::to_int\(\)](#), and [x](#).

5.1.3.493 Bernoulli_polynomial()

```
ex GiNaC::Bernoulli_polynomial (
    const numeric & k,
    const ex & x )
```

The Bernoulli polynomials.

References [GiNaC::ex::coeff\(\)](#), [exp\(\)](#), [factorial\(\)](#), [k](#), [GiNaC::ex::series\(\)](#), [series_to_poly\(\)](#), and [x](#).

Referenced by [GiNaC::Eisenstein_h_kernel::coefficient_a0\(\)](#).

5.1.3.494 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [13/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    integration_kernel ,
    basic ,
    print_func< print_context > &::do_print )
```

5.1.3.495 GINAC_BIND_UNARCHIVER() [23/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    integration_kernel )
```

5.1.3.496 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [14/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    basic_log_kernel ,
    integration_kernel ,
    print_func< print_context > &::do_print )
```

5.1.3.497 GINAC_BIND_UNARCHIVER() [24/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    basic_log_kernel )
```

5.1.3.498 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [15/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    multiple_polylog_kernel ,
    integration_kernel ,
    print_func< print_context > &::do_print )
```

References [_ex1](#).

5.1.3.499 GINAC_BIND_UNARCHIVER() [25/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    multiple_polylog_kernel )
```

5.1.3.500 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [16/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    ELi_kernel ,
    integration_kernel ,
    print_func< print_context > &::do_print )
```

5.1.3.501 GINAC_BIND_UNARCHIVER() [26/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    ELi_kernel )
```

5.1.3.502 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [17/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    Ebar_kernel ,
    integration_kernel ,
    print_func< print_context > &::do_print )
```

5.1.3.503 GINAC_BIND_UNARCHIVER() [27/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    Ebar_kernel )
```


5.1.3.504 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [18/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    Kronecker_dtau_kernel ,
    integration_kernel ,
    print_func< print_context > &::do_print )
```

5.1.3.505 GINAC_BIND_UNARCHIVER() [28/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    Kronecker_dtau_kernel )
```

5.1.3.506 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [19/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    Kronecker_dz_kernel ,
    integration_kernel ,
    print_func< print_context > &::do_print )
```

5.1.3.507 GINAC_BIND_UNARCHIVER() [29/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    Kronecker_dz_kernel )
```

5.1.3.508 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [20/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    Eisenstein_kernel ,
    integration_kernel ,
    print_func< print_context > &::do_print )
```

5.1.3.509 GINAC_BIND_UNARCHIVER() [30/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    Eisenstein_kernel )
```

5.1.3.510 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [21/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    Eisenstein_h_kernel ,
    integration_kernel ,
    print_func< print_context > &::do_print )
```

5.1.3.511 GINAC_BIND_UNARCHIVER() [31/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    Eisenstein_h_kernel )
```

5.1.3.512 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [22/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    modular_form_kernel ,
    integration_kernel ,
    print_func< print_context > &::do_print )
```

5.1.3.513 GINAC_BIND_UNARCHIVER() [32/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    modular_form_kernel )
```

5.1.3.514 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [23/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    user_defined_kernel ,
    integration_kernel ,
    print_func< print_context > &::do_print )
```

5.1.3.515 GINAC_BIND_UNARCHIVER() [33/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    user_defined_kernel )
```

5.1.3.516 GINAC_DECLARE_UNARCHIVER() [24/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    integration_kernel )
```

5.1.3.517 GINAC_DECLARE_UNARCHIVER() [25/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    basic_log_kernel )
```

5.1.3.518 GINAC_DECLARE_UNARCHIVER() [26/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    multiple_polylog_kernel )
```

5.1.3.519 GINAC_DECLARE_UNARCHIVER() [27/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    ELi_kernel )
```

5.1.3.520 GINAC_DECLARE_UNARCHIVER() [28/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    Ebar_kernel )
```

5.1.3.521 GINAC_DECLARE_UNARCHIVER() [29/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    Kronecker_dtau_kernel )
```

5.1.3.522 GINAC_DECLARE_UNARCHIVER() [30/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    Kronecker_dz_kernel )
```

5.1.3.523 GINAC_DECLARE_UNARCHIVER() [31/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    Eisenstein_kernel )
```

5.1.3.524 GINAC_DECLARE_UNARCHIVER() [32/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    Eisenstein_h_kernel )
```

5.1.3.525 GINAC_DECLARE_UNARCHIVER() [33/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    modular_form_kernel )
```

5.1.3.526 GINAC_DECLARE_UNARCHIVER() [34/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    user_defined_kernel )
```

5.1.3.527 GINAC_DECLARE_UNARCHIVER() [35/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    lst )
```

5.1.3.528 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [24/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    matrix ,
    basic ,
    print_func< print_context > &::do_print. print_func< print_latex > &::do_↵
print_latex. print_func< print_tree > &::do_print_tree. print_func< print_python_repr > &↵
::do_print_python_repr )
```

Default ctor.

Initializes to 1 x 1-dimensional zero-matrix.

References [GiNaC::status_flags::not_shareable](#).

5.1.3.529 GINAC_BIND_UNARCHIVER() [34/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    matrix )
```

5.1.3.530 lst_to_matrix()

```
ex GiNaC::lst_to_matrix (
    const lst & l )
```

Convert list of lists to matrix.

References [cols\(\)](#), [GiNaC::container< C >::nops\(\)](#), and [rows\(\)](#).

5.1.3.531 diag_matrix() [1/2]

```
ex GiNaC::diag_matrix (
    const lst & l )
```

Convert list of diagonal elements to matrix.

References [GiNaC::container< C >::nops\(\)](#).

5.1.3.532 diag_matrix() [2/2]

```
ex GiNaC::diag_matrix (
    std::initializer_list< ex > l )
```

5.1.3.533 unit_matrix() [1/2]

```
ex GiNaC::unit_matrix (
    unsigned r,
    unsigned c )
```

Create an r times c unit matrix.

References [_ex1](#), [c](#), [GiNaC::status_flags::evaluated](#), [r](#), and [GiNaC::basic::setflag\(\)](#).

Referenced by [unit_matrix\(\)](#).

5.1.3.534 symbolic_matrix() [1/2]

```
ex GiNaC::symbolic_matrix (
    unsigned r,
    unsigned c,
    const std::string & base_name,
    const std::string & tex_base_name )
```

Create an r times c matrix of newly generated symbols consisting of the given base name plus the numeric row/column position of each element.

The base name for LaTeX output is specified separately.

References [c](#), [GiNaC::status_flags::evaluated](#), [r](#), and [GiNaC::basic::setflag\(\)](#).

Referenced by [symbolic_matrix\(\)](#).

5.1.3.535 reduced_matrix()

```
ex GiNaC::reduced_matrix (
    const matrix & m,
    unsigned r,
    unsigned c )
```

Return the reduced matrix that is formed by deleting the r th row and c th column of matrix m .

The determinant of the result is the Minor r, c .

References [c](#), [cols\(\)](#), [GiNaC::status_flags::evaluated](#), [m](#), [r](#), [rows\(\)](#), and [GiNaC::basic::setflag\(\)](#).

5.1.3.536 sub_matrix()

```
ex GiNaC::sub_matrix (
    const matrix & m,
    unsigned r,
    unsigned nr,
    unsigned c,
    unsigned nc )
```

Return the nr times nc submatrix starting at position r, c of matrix m .

References [c](#), [GiNaC::status_flags::evaluated](#), [m](#), [r](#), and [GiNaC::basic::setflag\(\)](#).

5.1.3.537 GINAC_DECLARE_UNARCHIVER() [36/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    matrix )
```

5.1.3.538 nops() [2/2]

```
size_t GiNaC::nops (
    const matrix & m ) [inline]
```

References [m](#).

5.1.3.539 `expand()` [2/2]

```
ex GiNaC::expand (
    const matrix & m,
    unsigned options = 0 ) [inline]
```

References [GiNaC::ex::expand\(\)](#), [m](#), and [options](#).

5.1.3.540 `evalf()` [2/2]

```
ex GiNaC::evalf (
    const matrix & m ) [inline]
```

References [GiNaC::ex::evalf\(\)](#), and [m](#).

5.1.3.541 `rows()`

```
unsigned GiNaC::rows (
    const matrix & m ) [inline]
```

References [m](#).

Referenced by [lst_to_matrix\(\)](#), and [reduced_matrix\(\)](#).

5.1.3.542 `cols()`

```
unsigned GiNaC::cols (
    const matrix & m ) [inline]
```

References [m](#).

Referenced by [lst_to_matrix\(\)](#), and [reduced_matrix\(\)](#).

5.1.3.543 `transpose()`

```
matrix GiNaC::transpose (
    const matrix & m ) [inline]
```

References [m](#), and [GiNaC::matrix::transpose\(\)](#).

5.1.3.544 `determinant()`

```
ex GiNaC::determinant (
    const matrix & m,
    unsigned options = determinant_algo::automatic ) [inline]
```

References [m](#), and [options](#).

5.1.3.545 `trace()`

```
ex GiNaC::trace (
    const matrix & m ) [inline]
```

References [m](#).

5.1.3.546 `charpoly()`

```
ex GiNaC::charpoly (
    const matrix & m,
    const ex & lambda ) [inline]
```

References [m](#).

5.1.3.547 `inverse()` [1/3]

```
matrix GiNaC::inverse (
    const matrix & m ) [inline]
```

References [GiNaC::solve_algo::automatic](#), [GiNaC::matrix::inverse\(\)](#), and [m](#).

5.1.3.548 `inverse()` [2/3]

```
matrix GiNaC::inverse (
    const matrix & m,
    unsigned algo ) [inline]
```

References [GiNaC::matrix::inverse\(\)](#), and [m](#).

5.1.3.549 `rank()` [1/2]

```
unsigned GiNaC::rank (
    const matrix & m ) [inline]
```

References [m](#), and [GiNaC::matrix::rank\(\)](#).

5.1.3.550 `rank()` [2/2]

```
unsigned GiNaC::rank (
    const matrix & m,
    unsigned solve_algo ) [inline]
```

References [m](#).

5.1.3.551 unit_matrix() [2/2]

```
ex GiNaC::unit_matrix (
    unsigned x ) [inline]
```

Create a x times x unit matrix.

References [unit_matrix\(\)](#), and [x](#).

5.1.3.552 symbolic_matrix() [2/2]

```
ex GiNaC::symbolic_matrix (
    unsigned r,
    unsigned c,
    const std::string & base_name ) [inline]
```

Create an r times c matrix of newly generated symbols consisting of the given base name plus the numeric row/column position of each element.

References [c](#), [r](#), and [symbolic_matrix\(\)](#).

5.1.3.553 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [25/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    mul ,
    expairseq ,
    print_func< print_context > &::do_print. print_func< print_latex > &::do_↵
print_latex. print_func< print_csrc > &::do_print_csrc. print_func< print_tree > &::do_↵
print_tree. print_func< print_python_repr > &::do_print_python_repr )
```

5.1.3.554 tryfactsubs()

```
bool GiNaC::tryfactsubs (
    const ex & origfactor,
    const ex & patternfactor,
    int & nummatches,
    exmap & repls )
```

References [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::ex::match\(\)](#), and [GiNaC::ex::op\(\)](#).

Referenced by [algebraic_match_mul_with_mul\(\)](#), [GiNaC::mul::algebraic_subs_mul\(\)](#), and [GiNaC::power::subs\(\)](#).

5.1.3.555 algebraic_match_mul_with_mul()

```
bool GiNaC::algebraic_match_mul_with_mul (
    const mul & e,
    const ex & pat,
    exmap & repls,
    int factor,
    int & nummatches,
    const std::vector< bool > & substed,
    std::vector< bool > & matched )
```

Checks whether e matches to the pattern pat and the (possibly to be updated) list of replacements repls.

This matching is in the sense of algebraic substitutions. Matching starts with pat.op(factor) of the pattern because the factors before this one have already been matched. The (possibly updated) number of matches is in nummatches. substed[i] is true for factors that already have been replaced by previous substitutions and matched[i] is true for factors that have been matched by the current match.

References [algebraic_match_mul_with_mul\(\)](#), [factor\(\)](#), [GINAC_ASSERT](#), [GiNaC::ex::nops\(\)](#), [GiNaC::expairseq::nops\(\)](#), [GiNaC::ex::op\(\)](#), [GiNaC::expairseq::op\(\)](#), and [tryfactsubs\(\)](#).

Referenced by [algebraic_match_mul_with_mul\(\)](#), [GiNaC::mul::algebraic_subs_mul\(\)](#), and [GiNaC::mul::has\(\)](#).

5.1.3.556 GINAC_BIND_UNARCHIVER() [35/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    mul )
```

5.1.3.557 GINAC_DECLARE_UNARCHIVER() [37/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    mul )
```

5.1.3.558 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [26/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    ncmul ,
    exprseq ,
    print_func< print_context > &::do_print. print_func< print_tree > &::do_print↵
    _tree. print_func< print_csrc > &::do_print_csrc. print_func< print_python_repr > &::do_↵
    print_csrc )
```

5.1.3.559 reeval_ncmul()

```
ex GiNaC::reeval_ncmul (
    const exvector & v )
```

5.1.3.560 hold_ncmul()

```
ex GiNaC::hold_ncmul (
    const exvector & v )
```

Referenced by [GiNaC::basic::eval_ncmul\(\)](#), [GiNaC::color::eval_ncmul\(\)](#), and [GiNaC::structure< T, ComparisonPolicy >::eval_ncmul\(\)](#).

5.1.3.561 GINAC_BIND_UNARCHIVER() [36/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    ncmul )
```

5.1.3.562 GINAC_DECLARE_UNARCHIVER() [38/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    ncmul )
```

5.1.3.563 get_first_symbol()

```
static bool GiNaC::get_first_symbol (
    const ex & e,
    ex & x ) [static]
```

Return pointer to first symbol found in expression.

Due to [GiNaC](#)'s internal ordering of terms, it may not be obvious which symbol this function returns for a given expression.

Parameters

<i>e</i>	expression to search
<i>x</i>	first symbol found (returned)

Returns

"false" if no symbol was found, "true" otherwise

References [get_first_symbol\(\)](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), and [x](#).

Referenced by [divide\(\)](#), [frac_cancel\(\)](#), [get_first_symbol\(\)](#), and [GiNaC::ex::unit\(\)](#).

5.1.3.564 add_symbol()

```
static void GiNaC::add_symbol (
    const ex & s,
    sym_desc_vec & v ) [static]
```

Referenced by [collect_symbols\(\)](#).

5.1.3.565 collect_symbols()

```
static void GiNaC::collect_symbols (
    const ex & e,
    sym\_desc\_vec & v ) [static]
```

References [add_symbol\(\)](#), [collect_symbols\(\)](#), [GiNaC::ex::nops\(\)](#), and [GiNaC::ex::op\(\)](#).

Referenced by [collect_symbols\(\)](#), and [get_symbol_stats\(\)](#).

5.1.3.566 get_symbol_stats()

```
static void GiNaC::get_symbol_stats (
    const ex & a,
    const ex & b,
    sym\_desc\_vec & v ) [static]
```

Collect statistical information about symbols in polynomials.

This function fills in a vector of "sym_desc" structs which contain information about the highest and lowest degrees of all symbols that appear in two polynomials. The vector is then sorted by minimum degree (lowest to highest). The information gathered by this function is used by the GCD routines to identify trivial factors and to determine which variable to choose as the main variable for GCD computation.

Parameters

<i>a</i>	first multivariate polynomial
<i>b</i>	second multivariate polynomial
<i>v</i>	vector of sym_desc structs (filled in)

References [collect_symbols\(\)](#), [GiNaC::ex::degree\(\)](#), [GiNaC::ex::lcoeff\(\)](#), [GiNaC::ex::ldegree\(\)](#), and [GiNaC::ex::nops\(\)](#).

Referenced by [divide_in_z\(\)](#), [gcd\(\)](#), and [sqrfree\(\)](#).

5.1.3.567 lcmcoeff()

```
static numeric GiNaC::lcmcoeff (
    const ex & e,
    const numeric & l ) [static]
```

References [_num1_p](#), [c](#), [denom\(\)](#), [GiNaC::ex::info\(\)](#), [lcm\(\)](#), [lcmcoeff\(\)](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), [pow\(\)](#), and [GiNaC::info_flags::rational](#).

Referenced by [heur_gcd\(\)](#), [lcm_of_coefficients_denominators\(\)](#), [lcmcoeff\(\)](#), and [multiply_lcm\(\)](#).

5.1.3.568 lcm_of_coefficients_denominators()

```
static numeric GiNaC::lcm_of_coefficients_denominators (
    const ex & e ) [static]
```

Compute LCM of denominators of coefficients of a polynomial.

Given a polynomial with rational coefficients, this function computes the LCM of the denominators of all coefficients. This can be used to bring a polynomial from $\mathbb{Q}[X]$ to $\mathbb{Z}[X]$.

Parameters

<i>e</i>	multivariate polynomial (need not be expanded)

Returns

LCM of denominators of coefficients

References [_num1_p](#), and [lcmcoeff\(\)](#).

Referenced by [frac_cancel\(\)](#), [heur_gcd\(\)](#), and [sqrfree\(\)](#).

5.1.3.569 multiply_lcm()

```
static ex GiNaC::multiply_lcm (
    const ex & e,
    const numeric & lcm ) [static]
```

Bring polynomial from $\mathbb{Q}[X]$ to $\mathbb{Z}[X]$ by multiplying in the previously determined LCM of the coefficient's denominators.

Parameters

<i>e</i>	multivariate polynomial (need not be expanded)
<i>lcm</i>	LCM to multiply in

References [_num1_p](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::numeric::is_rational\(\)](#), [lcm\(\)](#), [lcmcoeff\(\)](#), [multiply_lcm\(\)](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), and [pow\(\)](#).

Referenced by [frac_cancel\(\)](#), [multiply_lcm\(\)](#), and [sqrfree\(\)](#).

5.1.3.570 quo()

```
ex GiNaC::quo (
    const ex & a,
    const ex & b,
    const ex & x,
    bool check_args )
```

Quotient $q(x)$ of polynomials $a(x)$ and $b(x)$ in $\mathbb{Q}[x]$.

It satisfies $a(x)=b(x)*q(x)+r(x)$.

Parameters

<i>a</i>	first polynomial in x (dividend)
<i>b</i>	second polynomial in x (divisor)
<i>x</i>	a and b are polynomials in x
<i>check_args</i>	check whether a and b are polynomials with rational coefficients (defaults to "true")

Returns

quotient of a and b in $\mathbb{Q}[x]$

References [_ex1](#), [GiNaC::ex::coeff\(\)](#), [GiNaC::ex::degree\(\)](#), [divide\(\)](#), [expand\(\)](#), [GiNaC::ex::expand\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::is_zero\(\)](#), [pow\(\)](#), [r](#), [GiNaC::info_flags::rational_polynomial](#), and [x](#).

Referenced by [decomp_rational\(\)](#), [GiNaC::ex::primpart\(\)](#), [sqrfree_parfrac\(\)](#), [sqrfree_yun\(\)](#), and [GiNaC::ex::unitcontprim\(\)](#).

5.1.3.571 rem()

```
ex GiNaC::rem (
    const ex & a,
    const ex & b,
    const ex & x,
    bool check_args )
```

Remainder [r\(x\)](#) of polynomials a(x) and b(x) in $\mathbb{Q}[x]$.

It satisfies $a(x)=b(x)*q(x)+r(x)$.

Parameters

<i>a</i>	first polynomial in x (dividend)
<i>b</i>	second polynomial in x (divisor)
<i>x</i>	a and b are polynomials in x
<i>check_args</i>	check whether a and b are polynomials with rational coefficients (defaults to "true")

Returns

remainder of a(x) and b(x) in $\mathbb{Q}[x]$

References [_ex0](#), [GiNaC::ex::coeff\(\)](#), [GiNaC::ex::degree\(\)](#), [divide\(\)](#), [expand\(\)](#), [GiNaC::ex::expand\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::is_zero\(\)](#), [pow\(\)](#), [r](#), [GiNaC::info_flags::rational_polynomial](#), and [x](#).

Referenced by [decomp_rational\(\)](#), and [sqrfree_parfrac\(\)](#).

5.1.3.572 decomp_rational()

```
ex GiNaC::decomp_rational (
    const ex & a,
    const ex & x )
```

Decompose rational function $a(x)=N(x)/D(x)$ into $P(x)+n(x)/D(x)$ with $\text{degree}(n, x) < \text{degree}(D, x)$.

Parameters

<i>a</i>	rational function in x
<i>x</i>	a is a function of x

Returns

decomposed function.

References [denom\(\)](#), [numer\(\)](#), [numer_denom\(\)](#), [GiNaC::ex::op\(\)](#), [quo\(\)](#), [rem\(\)](#), and [x](#).

5.1.3.573 prem()

```
ex GiNaC::prem (
    const ex & a,
    const ex & b,
    const ex & x,
    bool check_args )
```

Pseudo-remainder of polynomials $a(x)$ and $b(x)$ in $\mathbb{Q}[x]$.

Parameters

<i>a</i>	first polynomial in x (dividend)
<i>b</i>	second polynomial in x (divisor)
<i>x</i>	a and b are polynomials in x
<i>check_args</i>	check whether a and b are polynomials with rational coefficients (defaults to "true")

Returns

pseudo-remainder of $a(x)$ and $b(x)$ in $\mathbb{Q}[x]$

References [_ex0](#), [_ex1](#), [GiNaC::ex::coeff\(\)](#), [GiNaC::ex::degree\(\)](#), [expand\(\)](#), [GiNaC::ex::expand\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_zero\(\)](#), [pow\(\)](#), [r](#), [GiNaC::info_flags::rational_polynomial](#), and [x](#).

Referenced by [sr_gcd\(\)](#).

5.1.3.574 sprem()

```
ex GiNaC::sprem (
    const ex & a,
    const ex & b,
    const ex & x,
    bool check_args )
```

Sparse pseudo-remainder of polynomials $a(x)$ and $b(x)$ in $\mathbb{Q}[x]$.

Parameters

<i>a</i>	first polynomial in x (dividend)
<i>b</i>	second polynomial in x (divisor)
<i>x</i>	a and b are polynomials in x
<i>check_args</i>	check whether a and b are polynomials with rational coefficients (defaults to "true")

Returns

sparse pseudo-remainder of $a(x)$ and $b(x)$ in $\mathbb{Q}[x]$

References [_ex0](#), [_ex1](#), [GiNaC::ex::coeff\(\)](#), [GiNaC::ex::degree\(\)](#), [expand\(\)](#), [GiNaC::ex::expand\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_zero\(\)](#), [pow\(\)](#), [r](#), [GiNaC::info_flags::rational_polynomial](#), and [x](#).

5.1.3.575 divide()

```
bool GiNaC::divide (
    const ex & a,
    const ex & b,
    ex & q,
    bool check_args )
```

Exact polynomial division of $a(X)$ by $b(X)$ in $\mathbb{Q}[X]$.

Parameters

<i>a</i>	first multivariate polynomial (dividend)
<i>b</i>	second multivariate polynomial (divisor)
<i>q</i>	quotient (returned)
<i>check_args</i>	check whether a and b are polynomials with rational coefficients (defaults to "true")

Returns

"true" when exact division succeeds (quotient returned in *q*), "false" otherwise (*q* left untouched)

References [_ex0](#), [_ex1](#), [GiNaC::ex::coeff\(\)](#), [GiNaC::ex::degree\(\)](#), [divide\(\)](#), [expand\(\)](#), [GiNaC::ex::expand\(\)](#), [get_first_symbol\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), [pow\(\)](#), [r](#), [GiNaC::info_flags::rational_polynomial](#), and [x](#).

Referenced by [divide\(\)](#), [find_common_factor\(\)](#), [GiNaC::matrix::fraction_free_elimination\(\)](#), [gcd\(\)](#), [quo\(\)](#), and [rem\(\)](#).

5.1.3.576 divide_in_z()

```
static bool GiNaC::divide_in_z (
    const ex & a,
    const ex & b,
    ex & q,
    sym_desc_vec::const_iterator var ) [static]
```

Exact polynomial division of $a(X)$ by $b(X)$ in $\mathbb{Z}[X]$.

This functions works like [divide\(\)](#) but the input and output polynomials are in $\mathbb{Z}[X]$ instead of $\mathbb{Q}[X]$ (i.e. they have integer coefficients). Unlike [divide\(\)](#), it doesn't check whether the input polynomials really are integer polynomials, so be careful of what you pass in. Also, you have to run [get_symbol_stats\(\)](#) over the input polynomials before calling this function and pass an iterator to the first element of the [sym_desc](#) vector. This function is used internally by the [heur_gcd\(\)](#).

Parameters

<i>a</i>	first multivariate polynomial (dividend)
<i>b</i>	second multivariate polynomial (divisor)
<i>q</i>	quotient (returned)
<i>var</i>	iterator to first element of vector of sym_desc structs

Returns

"true" when exact division succeeds (the quotient is returned in q), "false" otherwise.

See also

[get_symbol_stats](#), [heur_gcd](#)

References [_ex0](#), [_ex1](#), [_num0_p](#), [_num1_p](#), [GiNaC::ex::begin\(\)](#), [c](#), [GiNaC::ex::coeff\(\)](#), [GiNaC::ex::degree\(\)](#), [divide_in_z\(\)](#), [expand\(\)](#), [GiNaC::ex::expand\(\)](#), [GiNaC::ex::find\(\)](#), [get_symbol_stats\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::numeric::inverse\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::is_zero\(\)](#), [k](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::ex::op\(\)](#), [pow\(\)](#), [qbar](#), [r](#), [GiNaC::ex::subs\(\)](#), and [x](#).

Referenced by [divide_in_z\(\)](#), [heur_gcd_z\(\)](#), and [sr_gcd\(\)](#).

5.1.3.577 **sr_gcd()**

```
static ex GiNaC::sr_gcd (
    const ex & a,
    const ex & b,
    sym_desc_vec::const_iterator var ) [static]
```

Compute GCD of multivariate polynomials using the subresultant PRS algorithm.

This function is used internally by [gcd\(\)](#).

Parameters

<i>a</i>	first multivariate polynomial
<i>b</i>	second multivariate polynomial
<i>var</i>	iterator to first element of vector of sym_desc structs

Returns

the GCD as a new expression

See also

[gcd](#)

References [_ex0](#), [_ex1](#), [c](#), [GiNaC::ex::content\(\)](#), [GiNaC::ex::degree\(\)](#), [divide_in_z\(\)](#), [GiNaC::ex::expand\(\)](#), [gcd\(\)](#), [GiNaC::ex::lcoeff\(\)](#), [pow\(\)](#), [prem\(\)](#), [GiNaC::ex::primpart\(\)](#), [psi\(\)](#), [r](#), and [x](#).

Referenced by [gcd\(\)](#).

5.1.3.578 interpolate()

```
static ex GiNaC::interpolate (
    const ex & gamma,
    const numeric & xi,
    const ex & x,
    int degree_hint = 1 ) [static]
```

xi-adic polynomial interpolation

References [GiNaC::numeric::inverse\(\)](#), [GiNaC::ex::is_zero\(\)](#), [pow\(\)](#), [GiNaC::ex::smod\(\)](#), and [x](#).

Referenced by [heur_gcd_z\(\)](#).

5.1.3.579 heur_gcd_z()

```
static bool GiNaC::heur_gcd_z (
    ex & res,
    const ex & a,
    const ex & b,
    ex * ca,
    ex * cb,
    sym_desc_vec::const_iterator var ) [static]
```

Compute GCD of multivariate polynomials using the heuristic GCD algorithm.

[get_symbol_stats\(\)](#) must have been called previously with the input polynomials and an iterator to the first element of the [sym_desc](#) vector passed in. This function is used internally by [gcd\(\)](#).

Parameters

<i>a</i>	first integer multivariate polynomial (expanded)
<i>b</i>	second integer multivariate polynomial (expanded)
<i>ca</i>	cofactor of polynomial a (returned), nullptr to suppress calculation of cofactor
<i>cb</i>	cofactor of polynomial b (returned), nullptr to suppress calculation of cofactor
<i>var</i>	iterator to first element of vector of sym_desc structs
<i>res</i>	the GCD (returned)

Returns

true if GCD was computed, false otherwise.

See also

[gcd](#)

Exceptions

<code>gcdheu_failed()</code>

References [GiNaC::ex::degree\(\)](#), [divide_in_z\(\)](#), [GiNaC::ex::expand\(\)](#), [gcd\(\)](#), [heur_gcd_z\(\)](#), [GiNaC::numeric::int_length\(\)](#), [GiNaC::ex::integer_content\(\)](#), [interpolate\(\)](#), [GiNaC::numeric::inverse\(\)](#), [iquo\(\)](#), [GiNaC::ex::is_zero\(\)](#), [isqrt\(\)](#), [GiNaC::ex::max_coefficient\(\)](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::ex::subs\(\)](#), and [x](#).

Referenced by [heur_gcd\(\)](#), and [heur_gcd_z\(\)](#).

5.1.3.580 [heur_gcd\(\)](#)

```
static bool GiNaC::heur_gcd (
    ex & res,
    const ex & a,
    const ex & b,
    ex * ca,
    ex * cb,
    sym_desc_vec::const_iterator var ) [static]
```

Compute GCD of multivariate polynomials using the heuristic GCD algorithm.

[get_symbol_stats\(\)](#) must have been called previously with the input polynomials and an iterator to the first element of the [sym_desc](#) vector passed in. This function is used internally by [gcd\(\)](#).

Parameters

<i>a</i>	first rational multivariate polynomial (expanded)
<i>b</i>	second rational multivariate polynomial (expanded)
<i>ca</i>	cofactor of polynomial a (returned), nullptr to suppress calculation of cofactor
<i>cb</i>	cofactor of polynomial b (returned), nullptr to suppress calculation of cofactor
<i>var</i>	iterator to first element of vector of sym_desc structs
<i>res</i>	the GCD (returned)

Returns

true if GCD was computed, false otherwise.

See also

[heur_gcd_z](#)
[gcd](#)

References [heur_gcd_z\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer_polynomial](#), [lcm_of_coefficients_denominators\(\)](#), and [lcmcoeff\(\)](#).

Referenced by [gcd\(\)](#).

5.1.3.581 [gcd_pf_pow\(\)](#)

```
static ex GiNaC::gcd_pf_pow (
    const ex & a,
    const ex & b,
    ex * ca,
    ex * cb ) [static]
```

References [_ex1](#), [expand\(\)](#), [gcd\(\)](#), [gcd_pf_pow\(\)](#), [gcd_pf_pow_pow\(\)](#), [GINAC_ASSERT](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::ldegree\(\)](#), [GiNaC::ex::op\(\)](#), and [pow\(\)](#).

Referenced by [gcd\(\)](#), and [gcd_pf_pow\(\)](#).

5.1.3.582 gcd_pf_mul()

```
static ex GiNaC::gcd_pf_mul (
    const ex & a,
    const ex & b,
    ex * ca,
    ex * cb ) [static]
```

References [gcd\(\)](#), [gcd_pf_mul\(\)](#), [GINAC_ASSERT](#), [GiNaC::ex::nops\(\)](#), and [GiNaC::ex::op\(\)](#).

Referenced by [gcd\(\)](#), and [gcd_pf_mul\(\)](#).

5.1.3.583 gcd() [1/2]

```
ex GiNaC::gcd (
    const ex & a,
    const ex & b,
    ex * ca,
    ex * cb,
    bool check_args,
    unsigned options )
```

Compute GCD (Greatest Common Divisor) of multivariate polynomials $a(X)$ and $b(X)$ in $\mathbb{Z}[X]$.

Optionally also compute the cofactors of a and b , defined by $a = ca * \text{gcd}(a, b)$ and $b = cb * \text{gcd}(a, b)$.

Parameters

<i>a</i>	first multivariate polynomial
<i>b</i>	second multivariate polynomial
<i>ca</i>	pointer to expression that will receive the cofactor of a , or nullptr
<i>cb</i>	pointer to expression that will receive the cofactor of b , or nullptr
<i>check_args</i>	check whether a and b are polynomials with rational coefficients (defaults to "true")
<i>options</i>	see GiNaC::gcd_options

Returns

the GCD as a new expression

References [_ex0](#), [_ex1](#), [divide\(\)](#), [expand\(\)](#), [GiNaC::ex::expand\(\)](#), [gcd\(\)](#), [gcd_pf_mul\(\)](#), [gcd_pf_pow\(\)](#), [get_symbol_stats\(\)](#), [heur_gcd\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::integer_content\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::numeric::is_zero\(\)](#), [n](#), [GiNaC::gcd_options::no_heur_gcd](#), [GiNaC::gcd_options::no_part_factored](#), [GiNaC::subs_options::no_pattern](#), [options](#), [pow\(\)](#), [GiNaC::info_flags::rational_polynomial](#), [sr_gcd\(\)](#), [GiNaC::ex::subs\(\)](#), [GiNaC::ex::unitcontprim\(\)](#), [GiNaC::gcd_options::use_sr_gcd](#), and [x](#).

Referenced by [GiNaC::ex::content\(\)](#), [dirichlet_character\(\)](#), [find_common_factor\(\)](#), [frac_cancel\(\)](#), [gcd\(\)](#), [gcd_pf_mul\(\)](#), [gcd_pf_pow\(\)](#), [gcd_pf_pow_pow\(\)](#), [heur_gcd_z\(\)](#), [GiNaC::add::integer_content\(\)](#), [lcm\(\)](#), [GiNaC::add::normal\(\)](#), [sqrfree_yun\(\)](#), and [sr_gcd\(\)](#).

5.1.3.584 gcd_pf_pow_pow()

```
static ex GiNaC::gcd_pf_pow_pow (
    const ex & a,
    const ex & b,
    ex * ca,
    ex * cb ) [static]
```

References [_ex1](#), [gcd\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::op\(\)](#), and [pow\(\)](#).

Referenced by [gcd_pf_pow\(\)](#).

5.1.3.585 lcm() [1/2]

```
ex GiNaC::lcm (
    const ex & a,
    const ex & b,
    bool check_args )
```

Compute LCM (Least Common Multiple) of multivariate polynomials in $\mathbb{Z}[X]$.

Parameters

<i>a</i>	first multivariate polynomial
<i>b</i>	second multivariate polynomial
<i>check_args</i>	check whether a and b are polynomials with rational coefficients (defaults to "true")

Returns

the LCM as a new expression

References [gcd\(\)](#), [GiNaC::ex::info\(\)](#), [lcm\(\)](#), and [GiNaC::info_flags::rational_polynomial](#).

Referenced by [GiNaC::add::integer_content\(\)](#), [lcm\(\)](#), [lcmcoeff\(\)](#), [multiply_lcm\(\)](#), and [sqrfree\(\)](#).

5.1.3.586 sqrfree_yun()

```
static epvector GiNaC::sqrfree_yun (
    const ex & a,
    const symbol & x ) [static]
```

Compute square-free factorization of multivariate polynomial $a(x)$ using Yun's algorithm.

Used internally by [sqrfree\(\)](#).

Parameters

<i>a</i>	multivariate polynomial over $\mathbb{Z}[X]$, treated here as univariate polynomial in x (needs not be expanded).
<i>x</i>	variable to factor in

Returns

vector of expairs (factor, exponent), sorted by exponents

References [_ex1](#), [GiNaC::ex::diff\(\)](#), [factors](#), [gcd\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::is_zero\(\)](#), [quo\(\)](#), and [x](#).

Referenced by [sqrfree\(\)](#), and [sqrfree_parfrac\(\)](#).

5.1.3.587 sqrfree()

```
ex GiNaC::sqrfree (
    const ex & a,
    const lst & l )
```

Compute a square-free factorization of a multivariate polynomial in $\mathbb{Q}[X]$.

Parameters

<i>a</i>	multivariate polynomial over $\mathbb{Q}[X]$ (needs not be expanded)
<i>l</i>	lst of variables to factor in, may be left empty for autodetection

Returns

a square-free factorization of *a*.

Note

A polynomial $p(X) \in C[X]$ is said *square-free* if, whenever any two polynomials $q(X)$ and $r(X)$ are such that

$$p(X) = q(X)^2 r(X),$$

we have $q(X) \in C$. This means that $p(X)$ has no repeated factors, apart eventually from constants. Given a polynomial $p(X) \in C[X]$, we say that the decomposition

$$p(X) = b \cdot p_1(X)^{a_1} \cdot p_2(X)^{a_2} \cdots p_r(X)^{a_r}$$

is a *square-free factorization* of $p(X)$ if the following conditions hold:

1. $b \in C$ and $b \neq 0$;
2. a_i is a positive integer for $i = 1, \dots, r$;
3. the degree of the polynomial p_i is strictly positive for $i = 1, \dots, r$;
4. the polynomial $\prod_{i=1}^r p_i(X)$ is square-free.

Square-free factorizations need not be unique. For example, if a_i is even, we could change the polynomial $p_i(X)$ into $-p_i(X)$. Observe also that the factors $p_i(X)$ need not be irreducible polynomials.

References [_ex0](#), [GiNaC::container< C >::append\(\)](#), [factors](#), [get_symbol_stats\(\)](#), [lcm\(\)](#), [lcm_of_coefficients_denominators\(\)](#), [multiply_lcm\(\)](#), [GiNaC::container< C >::nops\(\)](#), [GiNaC::container< C >::op\(\)](#), [GiNaC::container< C >::remove_first\(\)](#), [sqrfree\(\)](#), [sqrfree_yun\(\)](#), and [x](#).

Referenced by [sqrfree\(\)](#).

5.1.3.588 sqrfree_parfrac()

```
ex GiNaC::sqrfree_parfrac (
    const ex & a,
    const symbol & x )
```

Compute square-free partial fraction decomposition of rational function $a(x)$.

Parameters

<i>a</i>	rational function over $\mathbb{Z}[x]$, treated as univariate polynomial in x
<i>x</i>	variable to factor in

Returns

decomposed rational function

References [_ex1](#), [GiNaC::basic::coeff\(\)](#), [GiNaC::ex::coeff\(\)](#), [coeff\(\)](#), [degree\(\)](#), [denom\(\)](#), [GiNaC::ex::expand\(\)](#), [factor\(\)](#), [GINAC_ASSERT](#), [k](#), [n](#), [numer\(\)](#), [numer_denom\(\)](#), [GiNaC::ex::op\(\)](#), [pow\(\)](#), [quo\(\)](#), [r](#), [rem\(\)](#), [rhs\(\)](#), [GiNaC::matrix::solve\(\)](#), [sqrfree_yun\(\)](#), [to_int\(\)](#), and [x](#).

5.1.3.589 `replace_with_symbol()` [1/2]

```
static ex GiNaC::replace_with_symbol (
    const ex & e,
    exmap & repl,
    exmap & rev_lookup,
    lst & modifier ) [static]
```

Create a symbol for replacing the expression "e" (or return a previously assigned symbol).

The symbol and expression are appended to `repl`, for a later application of [subs\(\)](#). An entry in the replacement table `repl` can be changed in some cases. If it was altered, we need to provide the modifier for the previously build expressions. The modifier is an (ordered) list, because those substitutions need to be done in the incremental order. As an example let us consider a rationalisation of the expression $e = \exp(2x) \cdot \cos(\exp(2x)+1) \cdot \exp(x)$. The first factor [GiNaC](#) denotes by something like `symbol1` and will record: $e = \text{symbol1} \cdot \cos(\text{symbol1} + 1) \cdot \exp(x)$ `repl = {symbol1 :  $\exp(2x)$ }`. Similarly, the second factor would be denoted as `symbol2` and we will have $e = \text{symbol1} \cdot \text{symbol2} \cdot \exp(x)$ `repl = {symbol1 :  $\exp(2x)$ , symbol2 :  $\cos(\text{symbol1} + 1)$ }`. Denoting the third term as `symbol3` [GiNaC](#) is willing to re-think $\exp(2x)$ as symbol3^2 rather than just `symbol1`. Here are two issues: 1) The replacement "symbol1 -> symbol3^2 " in the previous part of the expression needs to be done outside of the present routine; 2) The pair "symbol1 : $\exp(2x)$ " shall be deleted from the replacement table `repl`. However, this will create illegal substitution "symbol2 : $\cos(\text{symbol1} + 1)$ " with undefined `symbol1`. These both problems are mitigated through the additions of the record "symbol1== symbol3^2 " to the list modifier. Changed length of the modifier signals to the calling code that the previous portion of the expression needs to be altered (it solves 1). Thus [GiNaC](#) can record now $e = \text{symbol3}^2 \cdot \text{symbol2} \cdot \text{symbol3}$ `repl = {symbol2 :  $\cos(\text{symbol1} + 1)$ , symbol3 :  $\exp(x)$ }` `modifier = {symbol1== $\text{symbol3}^2$ }`. Then, doing the backward substitutions the list modifier will be used to restore such iterative substitutions in the right way (this solves 2).

See also

[ex::normal](#)

References [_ex_1](#), [GiNaC::container< C >::append\(\)](#), [degree\(\)](#), [denom\(\)](#), [exp\(\)](#), [GiNaC::ex::find\(\)](#), [GiNaC::ex::is_equal\(\)](#), [is_ex_the_function](#), [is_integer\(\)](#), [is_rational\(\)](#), [GiNaC::subs_options::no_pattern](#), [normal\(\)](#), [numer\(\)](#), [GiNaC::ex::op\(\)](#), [pow\(\)](#), and [GiNaC::ex::subs\(\)](#).

Referenced by [GiNaC::basic::normal\(\)](#), [GiNaC::numeric::normal\(\)](#), [GiNaC::power::normal\(\)](#), [GiNaC::pseries::normal\(\)](#), [GiNaC::basic::to_polynomial\(\)](#), [GiNaC::numeric::to_polynomial\(\)](#), [GiNaC::power::to_polynomial\(\)](#), [GiNaC::basic::to_rational\(\)](#), [GiNaC::numeric::to_rational\(\)](#), and [GiNaC::power::to_rational\(\)](#).

5.1.3.590 replace_with_symbol() [2/2]

```
static ex GiNaC::replace_with_symbol (
    const ex & e,
    exmap & repl ) [static]
```

Create a symbol for replacing the expression "e" (or return a previously assigned symbol).

The symbol and expression are appended to repl, and the symbol is returned.

See also

[basic::to_rational](#)

[basic::to_polynomial](#)

References [GiNaC::subs_options::no_pattern](#), and [GiNaC::ex::subs\(\)](#).

5.1.3.591 frac_cancel()

```
static ex GiNaC::frac_cancel (
    const ex & n,
    const ex & d ) [static]
```

Fraction cancellation.

Parameters

<i>n</i>	numerator
<i>d</i>	denominator

Returns

cancelled fraction {n, d} as a list

References [_ex1](#), [_ex1](#), [_num1_p](#), [GiNaC::numeric::denom\(\)](#), [GiNaC::ex::expand\(\)](#), [gcd\(\)](#), [get_first_symbol\(\)](#), [GINAC_ASSERT](#), [GiNaC::ex::is_equal\(\)](#), [is_negative\(\)](#), [GiNaC::ex::is_zero\(\)](#), [lcm_of_coefficients_denominators\(\)](#), [multiply_lcm\(\)](#), [n](#), [GiNaC::numeric::numer\(\)](#), [GiNaC::ex::unit\(\)](#), and [x](#).

Referenced by [GiNaC::add::normal\(\)](#), and [GiNaC::mul::normal\(\)](#).

5.1.3.592 find_common_factor()

```
static ex GiNaC::find_common_factor (
    const ex & e,
    ex & factor,
    exmap & repl ) [static]
```

Remove the common factor in the terms of a sum 'e' by calculating the GCD, and multiply it into the expression 'factor' (which needs to be initialized to 1, unless you're accumulating factors).

References [_ex0](#), [_ex1](#), [divide\(\)](#), [factor\(\)](#), [find_common_factor\(\)](#), [gcd\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::is_zero\(\)](#), [k](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), [pow\(\)](#), [GiNaC::ex::to_polynomial\(\)](#), and [x](#).

Referenced by [collect_common_factors\(\)](#), and [find_common_factor\(\)](#).

5.1.3.593 collect_common_factors()

```
ex GiNaC::collect_common_factors (
    const ex & e )
```

Collect common factors in sums.

This converts expressions like 'a*(b*x+b*y)' to 'a*b*(x+y)'.

References [factor\(\)](#), [find_common_factor\(\)](#), [GiNaC::subs_options::no_pattern](#), [r](#), and [GiNaC::ex::subs\(\)](#).

Referenced by [GiNaC::power::to_polynomial\(\)](#).

5.1.3.594 resultant()

```
ex GiNaC::resultant (
    const ex & e1,
    const ex & e2,
    const ex & s )
```

Resultant of two expressions e1,e2 with respect to symbol s.

Method: Compute determinant of Sylvester matrix of e1,e2,s.

References [GiNaC::ex::coeff\(\)](#), [GiNaC::ex::degree\(\)](#), [GiNaC::ex::expand\(\)](#), [GiNaC::ex::info\(\)](#), [k](#), [GiNaC::ex::ldegree\(\)](#), [m](#), and [GiNaC::info_flags::polynomial](#).

5.1.3.595 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [27/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    numeric ,
    basic ,
    print_func< print_context > &::do_print. print_func< print_latex > &::do_↔
print_latex. print_func< print_csrc > &::do_print_csrc. print_func< print_csrc_cl_N > &↔
::do_print_csrc_cl_N. print_func< print_tree > &::do_print_tree. print_func< print_python_repr
> &::do_print_python_repr )
```

default ctor.

Numerically it initializes to an integer zero.

References [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::expanded](#), [GiNaC::basic::setflag\(\)](#), and [GiNaC::numeric::value](#).

5.1.3.596 make_real_float()

```
static const cln::cl_F GiNaC::make_real_float (
    const cln::cl_idcoded_float & dec ) [static]
```

Construct a floating point number from sign, mantissa, and exponent.

References [x](#).

Referenced by [read_real_float\(\)](#).

5.1.3.597 read_real_float()

```
static const cln::cl_F GiNaC::read_real_float (
    std::istream & s ) [static]
```

Read serialized floating point number.

References [make_real_float\(\)](#), and [x](#).

Referenced by [GiNaC::numeric::read_archive\(\)](#).

5.1.3.598 GINAC_BIND_UNARCHIVER() [37/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    numeric )
```

5.1.3.599 write_real_float()

```
static void GiNaC::write_real_float (
    std::ostream & s,
    const cln::cl_R & n ) [static]
```

References [n](#).

Referenced by [GiNaC::numeric::archive\(\)](#).

5.1.3.600 print_real_number()

```
static void GiNaC::print_real_number (
    const print_context & c,
    const cln::cl_R & x ) [static]
```

Helper function to print a real number in a nicer way than is CLN's default.

Instead of printing 42.0L0 this just prints 42.0 to ostream os and instead of 3.99168L7 it prints 3.99168E7. This is fine in [GiNaC](#) as long as it only uses `cl_LF` and no other floating point types that we might want to visibly distinguish from `cl_LF`.

See also

[numeric::print\(\)](#)

References [c](#), and [x](#).

Referenced by [GiNaC::numeric::print_numeric\(\)](#), and [print_real_cl_N\(\)](#).

5.1.3.601 print_integer_csrc()

```
static void GiNaC::print_integer_csrc (
    const print\_context & c,
    const cln::cl\_I & x ) [static]
```

Helper function to print integer number in C++ source format.

See also

[numeric::print\(\)](#)

References [c](#), and [x](#).

Referenced by [print_real_csrc\(\)](#).

5.1.3.602 print_real_csrc()

```
static void GiNaC::print_real_csrc (
    const print\_context & c,
    const cln::cl\_R & x ) [static]
```

Helper function to print real number in C++ source format.

See also

[numeric::print\(\)](#)

References [c](#), [denom\(\)](#), [numer\(\)](#), [print_integer_csrc\(\)](#), and [x](#).

Referenced by [GiNaC::numeric::do_print_csrc\(\)](#).

5.1.3.603 coerce()

```
template<typename T1 , typename T2 >
static bool GiNaC::coerce (
    T1 & dst,
    const T2 & arg ) [inline], [static]
```

Referenced by [print_real_cl_N\(\)](#).

5.1.3.604 coerce< int, [cln::cl_I](#) >()

```
template<>
bool GiNaC::coerce< int, cln::cl\_I > (
    int & dst,
    const cln::cl\_I & arg ) [inline]
```

Check if CLN integer can be converted into int.

See also

<https://www.ginac.de/pipermail/cln-list/2006-October/000248.html>

5.1.3.605 `coerce< unsigned int, cln::cl_I >()`

```
template<>
bool GiNaC::coerce< unsigned int, cln::cl_I > (
    unsigned int & dst,
    const cln::cl_I & arg ) [inline]
```

5.1.3.606 `print_real_cl_N()`

```
static void GiNaC::print_real_cl_N (
    const print_context & c,
    const cln::cl_R & x ) [static]
```

Helper function to print real number in C++ source format using `cl_N` types.

See also

[numeric::print\(\)](#)

References [c](#), [coerce\(\)](#), [Digits](#), [print_real_number\(\)](#), and [x](#).

Referenced by [GiNaC::numeric::do_print_csrc_cl_N\(\)](#).

5.1.3.607 `exp()`

```
const numeric GiNaC::exp (
    const numeric & x )
```

Exponential function.

Returns

arbitrary precision numerical $\exp(x)$.

References [x](#).

Referenced by [abs_eval\(\)](#), [abs_power\(\)](#), [Bernoulli_polynomial\(\)](#), [beta_evalf\(\)](#), [GiNaC::Eisenstein_h_kernel::coefficient_an\(\)](#), [csgn_power\(\)](#), [GiNaC::power::do_print_csrc\(\)](#), [exp_conjugate\(\)](#), [exp_deriv\(\)](#), [exp_eval\(\)](#), [exp_evalf\(\)](#), [exp_expand\(\)](#), [exp_imag_part\(\)](#), [exp_power\(\)](#), [exp_real_part\(\)](#), [generalised_Bernoulli_number\(\)](#), [GiNaC::Kronecker_dtau_kernel::get_numerical_val\(\)](#), [GiNaC::power::imag_part\(\)](#), [log_eval\(\)](#), [print_sym_pow\(\)](#), [GiNaC::Gt::qExpand\(\)](#), [GiNaC::power::real_part\(\)](#), [replace_with_symbol\(\)](#), [GiNaC::power::series\(\)](#), [GiNaC::Kronecker_dtau_kernel::series_coeff_impl\(\)](#), [GiNaC::Kronecker_dz_kernel::series_coeff_impl\(\)](#), and [tgamma\(\)](#).

5.1.3.608 `log()`

```
const numeric GiNaC::log (
    const numeric & x )
```

Natural logarithm.

Parameters

<i>x</i>	complex number

Returns

arbitrary precision numerical $\log(x)$.

Exceptions

<i>pole_error("log()"</i>	<i>logarithmic pole",0)</i>

References [GiNaC::ex::is_zero\(\)](#), and [x](#).

Referenced by [atan_series\(\)](#), [atanh_series\(\)](#), [GiNaC::power::derivative\(\)](#), [GiNaC::integral::eval_integ\(\)](#), [exp_eval\(\)](#), [G2_eval\(\)](#), [G2_evalf\(\)](#), [G3_eval\(\)](#), [G3_evalf\(\)](#), [H_eval\(\)](#), [GiNaC::power::imag_part\(\)](#), [lgamma\(\)](#), [lgamma_eval\(\)](#), [lgamma_series\(\)](#), [Li2_deriv\(\)](#), [Li2_eval\(\)](#), [Li2_series\(\)](#), [Li_eval\(\)](#), [log_conjugate\(\)](#), [log_eval\(\)](#), [log_evalf\(\)](#), [log_expand\(\)](#), [log_real_part\(\)](#), [log_series\(\)](#), [psi1_eval\(\)](#), [psi2_eval\(\)](#), [GiNaC::Gt::qExpand\(\)](#), [GiNaC::power::real_part\(\)](#), [S_eval\(\)](#), [GiNaC::power::series\(\)](#), and [GiNaC::Gt::tauToFundamental\(\)](#).

5.1.3.609 sin()

```
const numeric GiNaC::sin (
    const numeric & x )
```

Numeric sine (trigonometric function).

Returns

arbitrary precision numerical $\sin(x)$.

References [x](#).

Referenced by [GiNaC::Eisenstein_h_kernel::coefficient_a0\(\)](#), [cos_deriv\(\)](#), [cos_imag_part\(\)](#), [cosh_imag_part\(\)](#), [exp_imag_part\(\)](#), [GiNaC::power::imag_part\(\)](#), [lgamma\(\)](#), [sin_conjugate\(\)](#), [sin_eval\(\)](#), [sin_evalf\(\)](#), [sin_real_part\(\)](#), [sinh_eval\(\)](#), [sinh_imag_part\(\)](#), and [tan_series\(\)](#).

5.1.3.610 cos()

```
const numeric GiNaC::cos (
    const numeric & x )
```

Numeric cosine (trigonometric function).

Returns

arbitrary precision numerical $\cos(x)$.

References [x](#).

Referenced by [GiNaC::Eisenstein_h_kernel::coefficient_a0\(\)](#), [cos_conjugate\(\)](#), [cos_eval\(\)](#), [cos_evalf\(\)](#), [cos_real_part\(\)](#), [cosh_eval\(\)](#), [cosh_real_part\(\)](#), [exp_real_part\(\)](#), [GiNaC::power::real_part\(\)](#), [sin_deriv\(\)](#), [sin_imag_part\(\)](#), [sinh_real_part\(\)](#), and [tan_series\(\)](#).

5.1.3.611 `tan()`

```
const numeric GiNaC::tan (
    const numeric & x )
```

Numeric tangent (trigonometric function).

Returns

arbitrary precision numerical $\tan(x)$.

References [x](#).

Referenced by [tan_conjugate\(\)](#), [tan_deriv\(\)](#), [tan_eval\(\)](#), [tan_evalf\(\)](#), [tan_imag_part\(\)](#), [tan_real_part\(\)](#), [tanh_eval\(\)](#), [tanh_imag_part\(\)](#), and [tanh_real_part\(\)](#).

5.1.3.612 `asin()`

```
const numeric GiNaC::asin (
    const numeric & x )
```

Numeric inverse sine (trigonometric function).

Returns

arbitrary precision numerical $\arcsin(x)$.

References [x](#).

Referenced by [asin_conjugate\(\)](#), [asin_eval\(\)](#), [asin_evalf\(\)](#), [cos_eval\(\)](#), [sin_eval\(\)](#), and [tan_eval\(\)](#).

5.1.3.613 `acos()`

```
const numeric GiNaC::acos (
    const numeric & x )
```

Numeric inverse cosine (trigonometric function).

Returns

arbitrary precision numerical $\arccos(x)$.

References [x](#).

Referenced by [acos_conjugate\(\)](#), [acos_eval\(\)](#), [acos_evalf\(\)](#), [cos_eval\(\)](#), [sin_eval\(\)](#), and [tan_eval\(\)](#).

5.1.3.614 `atan()` [1/2]

```
const numeric GiNaC::atan (
    const numeric & x )
```

Numeric arcustangent.

Parameters

<i>x</i>	complex number

Returns

`atan(x)`

Exceptions

<code>pole_error("atan()"</code>	<code>logarithmic pole",0)</code> if <code>x==1</code> or <code>x==-1</code> .

References [_num1_p](#), [abs\(\)](#), [GiNaC::numeric::is_equal\(\)](#), [GiNaC::ex::is_zero\(\)](#), and [x](#).

Referenced by [atan2_eval\(\)](#), [atan2_evalf\(\)](#), [atan_conjugate\(\)](#), [atan_eval\(\)](#), [atan_evalf\(\)](#), [atan_series\(\)](#), [cos_eval\(\)](#), [sin_eval\(\)](#), and [tan_eval\(\)](#).

5.1.3.615 atan() [2/2]

```
const numeric GiNaC::atan (
    const numeric & y,
    const numeric & x )
```

Numeric arcustangent of two arguments, analytically continued in a suitable way.

Parameters

<i>y</i>	complex number
<i>x</i>	complex number

Returns

$-i \log((x+iy)/\sqrt{x^2+y^2})$, which is equal to $\operatorname{atan}(y/x)$ if y and x are both real.

Exceptions

<code>pole_error("atan()"</code>	<code>logarithmic pole",0)</code> if <code>y/x==1</code> or <code>y/x==-1</code> .

References [_num0_p](#), [GiNaC::numeric::is_real\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::numeric::is_zero\(\)](#), [GiNaC::numeric::to_cl_N\(\)](#), and [x](#).

5.1.3.616 sinh()

```
const numeric GiNaC::sinh (
    const numeric & x )
```

Numeric hyperbolic sine (trigonometric function).

Returns

arbitrary precision numerical $\sinh(x)$.

References [x](#).

Referenced by [cos_imag_part\(\)](#), [cosh_deriv\(\)](#), [cosh_imag_part\(\)](#), [sin_imag_part\(\)](#), [sinh_conjugate\(\)](#), [sinh_eval\(\)](#), [sinh_evalf\(\)](#), [sinh_real_part\(\)](#), and [tanh_series\(\)](#).

5.1.3.617 cosh()

```
const numeric GiNaC::cosh (  
    const numeric & x )
```

Numeric hyperbolic cosine (trigonometric function).

Returns

arbitrary precision numerical $\cosh(x)$.

References [x](#).

Referenced by [cos_real_part\(\)](#), [cosh_conjugate\(\)](#), [cosh_eval\(\)](#), [cosh_evalf\(\)](#), [cosh_real_part\(\)](#), [sin_real_part\(\)](#), [sinh_deriv\(\)](#), [sinh_imag_part\(\)](#), and [tanh_series\(\)](#).

5.1.3.618 tanh()

```
const numeric GiNaC::tanh (  
    const numeric & x )
```

Numeric hyperbolic tangent (trigonometric function).

Returns

arbitrary precision numerical $\tanh(x)$.

References [x](#).

Referenced by [tan_imag_part\(\)](#), [tanh_conjugate\(\)](#), [tanh_deriv\(\)](#), [tanh_eval\(\)](#), [tanh_evalf\(\)](#), [tanh_imag_part\(\)](#), and [tanh_real_part\(\)](#).

5.1.3.619 asinh()

```
const numeric GiNaC::asinh (  
    const numeric & x )
```

Numeric inverse hyperbolic sine (trigonometric function).

Returns

arbitrary precision numerical $\operatorname{asinh}(x)$.

References [x](#).

Referenced by [asinh_conjugate\(\)](#), [asinh_eval\(\)](#), [asinh_evalf\(\)](#), [cosh_eval\(\)](#), [sinh_eval\(\)](#), and [tanh_eval\(\)](#).

5.1.3.620 acosh()

```
const numeric GiNaC::acosh (
    const numeric & x )
```

Numeric inverse hyperbolic cosine (trigonometric function).

Returns

arbitrary precision numerical acosh(x).

References [x](#).

Referenced by [acosh_conjugate\(\)](#), [acosh_eval\(\)](#), [acosh_evalf\(\)](#), [cosh_eval\(\)](#), [sinh_eval\(\)](#), and [tanh_eval\(\)](#).

5.1.3.621 atanh()

```
const numeric GiNaC::atanh (
    const numeric & x )
```

Numeric inverse hyperbolic tangent (trigonometric function).

Returns

arbitrary precision numerical atanh(x).

References [x](#).

Referenced by [atanh_conjugate\(\)](#), [atanh_eval\(\)](#), [atanh_evalf\(\)](#), [atanh_series\(\)](#), [cosh_eval\(\)](#), [sinh_eval\(\)](#), and [tanh_eval\(\)](#).

5.1.3.622 Li2_series() [2/2]

```
static cln::cl_N GiNaC::Li2_series (
    const cln::cl_N & x,
    const cln::float_format_t & prec ) [static]
```

Numeric evaluation of Dilogarithm within circle of convergence (unit circle) using a power series.

References [x](#).

5.1.3.623 Li2_projection()

```
static cln::cl_N GiNaC::Li2_projection (
    const cln::cl_N & x,
    const cln::float_format_t & prec ) [static]
```

Folds Li2's argument inside a small rectangle to enhance convergence.

References [Li2_projection\(\)](#), [Li2_series\(\)](#), and [x](#).

Referenced by [Li2_\(\)](#), and [Li2_projection\(\)](#).

5.1.3.624 Li2_()

```
const cln::cl_N GiNaC::Li2_ (
    const cln::cl_N & value )
```

Numeric evaluation of Dilogarithm.

The domain is the entire complex plane, the branch cut lies along the positive real axis, starting at 1 and continuous with quadrant IV.

Returns

arbitrary precision numerical Li2(x).

References [Li2_projection\(\)](#), and [value](#).

Referenced by [Li2\(\)](#).

5.1.3.625 Li2()

```
const numeric GiNaC::Li2 (
    const numeric & x )
```

References [_num0_p](#), [Li2_\(\)](#), and [x](#).

Referenced by [Li2_conjugate\(\)](#), [Li2_eval\(\)](#), [Li2_evalf\(\)](#), and [Li2_series\(\)](#).

5.1.3.626 zeta() [3/3]

```
const numeric GiNaC::zeta (
    const numeric & x )
```

Numeric evaluation of Riemann's Zeta function.

Currently works only for integer arguments.

References [x](#).

5.1.3.627 guess_precision()

```
static cln::float_format_t GiNaC::guess_precision (
    const cln::cl_N & x ) [static]
```

References [x](#).

Referenced by [lgamma\(\)](#), and [tgamma\(\)](#).

5.1.3.628 lgamma() [1/2]

```
const cln::cl_N GiNaC::lgamma (
    const cln::cl_N & x )
```

The Gamma function.

Use the Lanczos approximation. If the coefficients used here are not sufficiently many or sufficiently accurate, more can be calculated using the program `doc/examples/lanczos.cpp`. In that case, be sure to read the comments in that file.

References [GiNaC::lanczos_coeffs::calc_lanczos_A\(\)](#), [GiNaC::lanczos_coeffs::get_order\(\)](#), [guess_precision\(\)](#), [lgamma\(\)](#), [log\(\)](#), [sin\(\)](#), [GiNaC::lanczos_coeffs::sufficiently_accurate\(\)](#), and [x](#).

Referenced by [beta_evalf\(\)](#), [lgamma\(\)](#), [lgamma\(\)](#), [lgamma_conjugate\(\)](#), [lgamma_eval\(\)](#), [lgamma_evalf\(\)](#), and [lgamma_series\(\)](#).

5.1.3.629 lgamma() [2/2]

```
const numeric GiNaC::lgamma (
    const numeric & x )
```

References [lgamma\(\)](#), and [x](#).

5.1.3.630 tgamma() [1/2]

```
const cln::cl_N GiNaC::tgamma (
    const cln::cl_N & x )
```

References [GiNaC::lanczos_coeffs::calc_lanczos_A\(\)](#), [exp\(\)](#), [GiNaC::lanczos_coeffs::get_order\(\)](#), [guess_precision\(\)](#), [sqrt\(\)](#), [GiNaC::lanczos_coeffs::sufficiently_accurate\(\)](#), [tgamma\(\)](#), and [x](#).

Referenced by [beta_eval\(\)](#), [beta_series\(\)](#), [psi2_eval\(\)](#), [tgamma\(\)](#), [tgamma\(\)](#), [tgamma_conjugate\(\)](#), [tgamma_deriv\(\)](#), [tgamma_eval\(\)](#), [tgamma_evalf\(\)](#), and [tgamma_series\(\)](#).

5.1.3.631 tgamma() [2/2]

```
const numeric GiNaC::tgamma (
    const numeric & x )
```

References [tgamma\(\)](#), and [x](#).

5.1.3.632 psi() [3/4]

```
const numeric GiNaC::psi (
    const numeric & x )
```

The psi function (aka polygamma function).

This is only a stub!

5.1.3.633 psi() [4/4]

```
const numeric GiNaC::psi (
    const numeric & n,
    const numeric & x )
```

The psi functions (aka polygamma functions).

This is only a stub!

5.1.3.634 factorial()

```
const numeric GiNaC::factorial (
    const numeric & n )
```

Factorial combinatorial function.

Parameters

n	integer argument ≥ 0

Exceptions

<i>range_error</i>	(argument must be integer ≥ 0)

References [n](#).

Referenced by [Bernoulli_polynomial\(\)](#), [GiNaC::Gt::eval\(\)](#), [factorial_conjugate\(\)](#), [factorial_eval\(\)](#), [factorial_evalf\(\)](#), [factorial_real_part\(\)](#), [G2_eval\(\)](#), [G2_evalf\(\)](#), [G3_eval\(\)](#), [G3_evalf\(\)](#), [GiNaC::Gt_detail::pathintegral_term::G_path\(\)](#), [generalised_Bernoulli_number\(\)](#), [H_eval\(\)](#), [lgamma_eval\(\)](#), [multinomial_coefficient\(\)](#), [psi2_eval\(\)](#), [psi2_series\(\)](#), [GiNaC::Gt::qExpand\(\)](#), [GiNaC::Gt::regularise\(\)](#), [S_eval\(\)](#), [GiNaC::Kronecker_dtau_kernel::series_coeff_impl\(\)](#), [GiNaC::Kronecker_dz_kernel::series_coeff_impl\(\)](#), [symm\(\)](#), [GiNaC::Gt::tauToFundamental\(\)](#), [tgamma_eval\(\)](#), [zeta1_eval\(\)](#), and [GiNaC::Gt::zisToFundamental\(\)](#).

5.1.3.635 doublefactorial()

```
const numeric GiNaC::doublefactorial (
    const numeric & n )
```

The double factorial combinatorial function.

(Scarcely used, but still useful in cases, like for exact results of $\text{tgamma}(n+1/2)$ for instance.)

Parameters

n	integer argument ≥ -1

Returns

$n!! == n * (n-2) * (n-4) * \dots * (\{1|2\})$ with $0!! == (-1)!! == 1$

Exceptions

<i>range_error</i>	(argument must be integer ≥ -1)

References [_num1_p](#), [_num_1_p](#), and [n](#).

Referenced by [tgamma_eval\(\)](#).

5.1.3.636 binomial()

```
const numeric GiNaC::binomial (
    const numeric & n,
    const numeric & k )
```

The Binomial coefficients.

It computes the binomial coefficients. For integer n and k and positive n this is the number of ways of choosing k objects from n distinct objects. If n is a negative integer, the formula $\text{binomial}(n,k) == (-1)^k * \text{binomial}(k-n-1,k)$ (if $k \geq 0$) $\text{binomial}(n,k) == (-1)^{(n-k)} * \text{binomial}(-k-1,n-k)$ (otherwise) is used to compute the result.

References [_num0_p](#), [_num1_p](#), [_num_1_p](#), [binomial\(\)](#), [k](#), [n](#), and [GiNaC::numeric::power\(\)](#).

Referenced by [binomial\(\)](#), [binomial_conjugate\(\)](#), [binomial_eval\(\)](#), [binomial_evalf\(\)](#), [binomial_real_part\(\)](#), [binomial_sym\(\)](#), [GiNaC::power::expand_add\(\)](#), [GiNaC::power::imag_part\(\)](#), [GiNaC::Gt::qExpand\(\)](#), [GiNaC::power::real_part\(\)](#), and [GiNaC::Gt::tauToFundamental\(\)](#).

5.1.3.637 bernoulli()

```
const numeric GiNaC::bernoulli (
    const numeric & nn )
```

Bernoulli number.

The n th Bernoulli number is the coefficient of $x^n/n!$ in the expansion of the function $x/(e^x-1)$.

Returns

the n th Bernoulli number (a rational number).

Exceptions

<i>range_error</i>	(argument must be integer ≥ 0)

References [_num1_p](#), [c](#), [GiNaC::numeric::is_integer\(\)](#), [GiNaC::numeric::is_negative\(\)](#), [k](#), [n](#), and [GiNaC::numeric::to_int\(\)](#).

Referenced by [GiNaC::Kronecker_dtau_kernel::series_coeff_impl\(\)](#), [GiNaC::Kronecker_dz_kernel::series_coeff_impl\(\)](#), and [zeta1_eval\(\)](#).

5.1.3.638 fibonacci()

```
const numeric GiNaC::fibonacci (
    const numeric & n )
```

Fibonacci number.

The nth Fibonacci number $F(n)$ is defined by the recurrence formula $F(n)=F(n-1)+F(n-2)$ with $F(0)=0$ and $F(1)=1$.

Parameters

<i>n</i>	an integer

Returns

the nth Fibonacci number $F(n)$ (an integer number)

Exceptions

<i>range_error</i>	(argument must be an integer)

References [_num0_p](#), [fibonacci\(\)](#), [m](#), and [n](#).

Referenced by [fibonacci\(\)](#).

5.1.3.639 abs()

```
const numeric GiNaC::abs (
    const numeric & x )
```

Absolute value.

References [x](#).

Referenced by [abs_conjugate\(\)](#), [abs_eval\(\)](#), [abs_evalf\(\)](#), [abs_expand\(\)](#), [abs_expl_derivative\(\)](#), [abs_power\(\)](#), [abs_real_part\(\)](#), [adaptivesimpson\(\)](#), [atan\(\)](#), [atan_series\(\)](#), [atanh_series\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::Gt::findMoebiusTransform\(\)](#), [generalised_Bernoulli_number\(\)](#), [GiNaC::power::imag_part\(\)](#), [GiNaC::mul::integer_content\(\)](#), [GiNaC::numeric::integer_content\(\)](#), [is_discriminant_of_quadratic_number_field\(\)](#), [log_real_part\(\)](#), [GiNaC::add::max_coefficient\(\)](#), [GiNaC::mul::max_coefficient\(\)](#), [GiNaC::numeric::max_coefficient\(\)](#), [GiNaC::matrix::pivot\(\)](#), [GiNaC::Gt::qExpand\(\)](#), [GiNaC::power::real_part\(\)](#), [tgamma_eval\(\)](#), [GiNaC::ex::unitcontprim\(\)](#), and [zeta1_eval\(\)](#).

5.1.3.640 mod()

```
const numeric GiNaC::mod (
    const numeric & a,
    const numeric & b )
```

Modulus (in positive representation).

In general, $\text{mod}(a,b)$ has the sign of b or is zero, and $\text{rem}(a,b)$ has the sign of a or is zero. This is different from Maple's modp , where the sign of b is ignored. It is in agreement with Mathematica's Mod .

Returns

$a \bmod b$ in the range $[0, \text{abs}(b)-1]$ with sign of b if both are integer, 0 otherwise.

References [_num0_p](#), [GiNaC::numeric::is_integer\(\)](#), and [GiNaC::numeric::to_cl_N\(\)](#).

Referenced by [GiNaC::Eisenstein_h_kernel::coefficient_a0\(\)](#), [GiNaC::Eisenstein_h_kernel::coefficient_an\(\)](#), [cos_eval\(\)](#), [exp_eval\(\)](#), [is_discriminant_of_quadratic_number_field\(\)](#), [sin_eval\(\)](#), and [tan_eval\(\)](#).

5.1.3.641 smod()

```
const numeric GiNaC::smod (
    const numeric & a_,
    const numeric & b_ )
```

Modulus (in symmetric representation).

Returns

$a \bmod b$ in the range $[-\text{iquo}(\text{abs}(b), 2), \text{iquo}(\text{abs}(b), 2)]$.

References [_num0_p](#), [GiNaC::numeric::is_integer\(\)](#), [m](#), and [GiNaC::numeric::to_cl_N\(\)](#).

Referenced by [GiNaC::add::smod\(\)](#), [GiNaC::mul::smod\(\)](#), and [GiNaC::numeric::smod\(\)](#).

5.1.3.642 irem() [1/2]

```
const numeric GiNaC::irem (
    const numeric & a,
    const numeric & b )
```

Numeric integer remainder.

Equivalent to Maple's `irem(a,b)` as far as sign conventions are concerned. In general, `mod(a,b)` has the sign of b or is zero, and `irem(a,b)` has the sign of a or is zero.

Returns

remainder of a/b if both are integer, 0 otherwise.

Exceptions

<code>overflow_error</code>	(division by zero) if b is zero.

References [_num0_p](#), [GiNaC::numeric::is_integer\(\)](#), [GiNaC::numeric::is_zero\(\)](#), and [GiNaC::numeric::to_cl_N\(\)](#).

Referenced by [GiNaC::Eisenstein_h_kernel::coefficient_a0\(\)](#), [GiNaC::Eisenstein_h_kernel::coefficient_an\(\)](#), and [ifactor\(\)](#).

5.1.3.643 `irem()` [2/2]

```
const numeric GiNaC::irem (
    const numeric & a,
    const numeric & b,
    numeric & q )
```

Numeric integer remainder.

Equivalent to Maple's `irem(a,b,q)` it obeys the relation `irem(a,b,q) == a - q*b`. In general, `mod(a,b)` has the sign of `b` or is zero, and `irem(a,b)` has the sign of `a` or is zero.

Returns

remainder of `a/b` and quotient stored in `q` if both are integer, 0 otherwise.

Exceptions

<code>overflow_error</code>	(division by zero) if <code>b</code> is zero.

References [_num0_p](#), [GiNaC::numeric::is_integer\(\)](#), [GiNaC::numeric::is_zero\(\)](#), and [GiNaC::numeric::to_cl_N\(\)](#).

5.1.3.644 `iquo()` [1/2]

```
const numeric GiNaC::iquo (
    const numeric & a,
    const numeric & b )
```

Numeric integer quotient.

Equivalent to Maple's `iquo` as far as sign conventions are concerned.

Returns

truncated quotient of `a/b` if both are integer, 0 otherwise.

Exceptions

<code>overflow_error</code>	(division by zero) if <code>b</code> is zero.

References [_num0_p](#), [GiNaC::numeric::is_integer\(\)](#), [GiNaC::numeric::is_zero\(\)](#), and [GiNaC::numeric::to_cl_N\(\)](#).

Referenced by [GiNaC::power::eval\(\)](#), and [heur_gcd_z\(\)](#).

5.1.3.645 `iquo()` [2/2]

```
const numeric GiNaC::iquo (
    const numeric & a,
```

```
const numeric & b,
numeric & r )
```

Numeric integer quotient.

Equivalent to Maple's `iquo(a,b,r)` it obeys the relation $r == a - \text{iquo}(a,b,r)*b$.

Returns

truncated quotient of a/b and remainder stored in r if both are integer, 0 otherwise.

Exceptions

<code>overflow_error</code>	(division by zero) if b is zero.

References `_num0_p`, `GiNaC::numeric::is_integer()`, `GiNaC::numeric::is_zero()`, `r`, and `GiNaC::numeric::to_cl_N()`.

5.1.3.646 `gcd()` [2/2]

```
const numeric GiNaC::gcd (
    const numeric & a,
    const numeric & b )
```

Greatest Common Divisor.

Returns

The GCD of two numbers if both are integer, a numerical 1 if they are not.

References `_num1_p`, `GiNaC::numeric::is_integer()`, and `GiNaC::numeric::to_cl_N()`.

5.1.3.647 `lcm()` [2/2]

```
const numeric GiNaC::lcm (
    const numeric & a,
    const numeric & b )
```

Least Common Multiple.

Returns

The LCM of two numbers if both are integer, the product of those two numbers if they are not.

References `GiNaC::numeric::is_integer()`, `GiNaC::numeric::mul()`, and `GiNaC::numeric::to_cl_N()`.

5.1.3.648 `sqrt()` [1/2]

```
const numeric GiNaC::sqrt (
    const numeric & x )
```

Numeric square root.

If possible, `sqrt(x)` should respect squares of exact numbers, i.e. `sqrt(4)` should return integer 2.

Parameters

<code>x</code>	numeric argument

Returns

square root of x . Branch cut along negative real axis, the negative real axis itself where $\text{imag}(x)=0$ and $\text{real}(x)<0$ belongs to the upper part where $\text{imag}(x)>0$.

References [x](#).

Referenced by [cos_eval\(\)](#), [cosh_eval\(\)](#), [EllipticE_evalf\(\)](#), [EllipticK_evalf\(\)](#), [GiNaC::su3f::eval_indexed\(\)](#), [GiNaC::su3d::eval_indexed\(\)](#), [sin_eval\(\)](#), [sinh_eval\(\)](#), [tan_eval\(\)](#), [tanh_eval\(\)](#), [tgamma\(\)](#), and [tgamma_eval\(\)](#).

5.1.3.649 isqrt()

```
const numeric GiNaC::isqrt (
    const numeric & x )
```

Integer numeric square root.

References [_num0_p](#), and [x](#).

Referenced by [heur_gcd_z\(\)](#).

5.1.3.650 PiEvalf()

```
ex GiNaC::PiEvalf ( )
```

Floating point evaluation of Archimedes' constant Pi.

5.1.3.651 EulerEvalf()

```
ex GiNaC::EulerEvalf ( )
```

Floating point evaluation of Euler's constant gamma.

5.1.3.652 CatalanEvalf()

```
ex GiNaC::CatalanEvalf ( )
```

Floating point evaluation of Catalan's constant.

5.1.3.653 operator<<() [6/16]

```
std::ostream & GiNaC::operator<< (
    std::ostream & os,
    const _numeric_digits & e )
```

References [GiNaC::_numeric_digits::print\(\)](#).

5.1.3.654 GINAC_DECLARE_UNARCHIVER() [39/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    numeric )
```

5.1.3.655 pow() [1/3]

```
const numeric GiNaC::pow (
    const numeric & x,
    const numeric & y ) [inline]
```

References [x](#).

Referenced by [abs_eval\(\)](#), [abs_power\(\)](#), [GiNaC::mul::algebraic_subs_mul\(\)](#), [beta_eval\(\)](#), [GiNaC::Eisenstein_h_kernel::coefficient_and_derivative\(\)](#), [GiNaC::basic::collect\(\)](#), [GiNaC::mul::combine_ex_with_coeff_to_pair\(\)](#), [GiNaC::mul::combine_pair_with_coeff_to_pair\(\)](#), [GiNaC::pseries::convert_to_poly\(\)](#), [GiNaC::mul::derivative\(\)](#), [GiNaC::power::derivative\(\)](#), [divide\(\)](#), [divide_in_z\(\)](#), [EllipticE_series\(\)](#), [EllipticK_series\(\)](#), [GiNaC::Gt::eval\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::power::evalm\(\)](#), [GiNaC::Gt_detail::pathintegral_term::eval\(\)](#), [GiNaC::power::expand\(\)](#), [GiNaC::power::expand_add\(\)](#), [factor\(\)](#), [find_common_factor\(\)](#), [G2_eval\(\)](#), [G2_evalf\(\)](#), [G3_eval\(\)](#), [G3_evalf\(\)](#), [GiNaC::Gt_detail::pathintegral_term::G_path\(\)](#), [gcd\(\)](#), [gcd_pf_pow\(\)](#), [gcd_pf_pow_pow\(\)](#), [GiNaC::Kronecker_dtau_kernel::get_numerical_value\(\)](#), [GiNaC::Kronecker_dz_kernel::get_numerical_value\(\)](#), [H_eval\(\)](#), [GiNaC::power::imag_part\(\)](#), [GiNaC::Gt_detail::pathintegral_term::integrate\(\)](#), [interpolate\(\)](#), [kronecker_symbol\(\)](#), [GiNaC::integration_kernel::Laurent_series\(\)](#), [lcmcoeff\(\)](#), [Li2_series\(\)](#), [Li_eval\(\)](#), [Li_series\(\)](#), [log_series\(\)](#), [multiply_lcm\(\)](#), [GiNaC::power::normal\(\)](#), [GiNaC::pseries::op\(\)](#), [Order_power\(\)](#), [GiNaC::pseries::power_const\(\)](#), [prem\(\)](#), [GiNaC::pseries::print_series\(\)](#), [psi1_eval\(\)](#), [psi2_eval\(\)](#), [GiNaC::Eisenstein_h_kernel::q_expansion_modular_form\(\)](#), [GiNaC::Gt::qExpand\(\)](#), [quo\(\)](#), [GiNaC::power::real_part\(\)](#), [GiNaC::Gt::regularise\(\)](#), [rem\(\)](#), [replace_with_symbol\(\)](#), [S_eval\(\)](#), [S_series\(\)](#), [GiNaC::modular_form_kernel::series\(\)](#), [GiNaC::Kronecker_dz_kernel::series_coeff_impl\(\)](#), [sprem\(\)](#), [sqrfree_parfrac\(\)](#), [sr_gcd\(\)](#), [GiNaC::power::subs\(\)](#), [GiNaC::Gt::tauToFundamental\(\)](#), [tgamma_eval\(\)](#), [GiNaC::power::to_polynomial\(\)](#), [GiNaC::power::to_rational\(\)](#), [zeta1_eval\(\)](#), and [GiNaC::Gt::zisToFundamental\(\)](#).

5.1.3.656 inverse() [3/3]

```
const numeric GiNaC::inverse (
    const numeric & x ) [inline]
```

References [x](#).

5.1.3.657 step()

```
numeric GiNaC::step (
    const numeric & x ) [inline]
```

References [x](#).

Referenced by [abs_eval\(\)](#), [H_eval\(\)](#), [step_conjugate\(\)](#), [step_eval\(\)](#), [step_evalf\(\)](#), [step_real_part\(\)](#), and [step_series\(\)](#).

5.1.3.658 csgn()

```
int GiNaC::csgn (
    const numeric & x ) [inline]
```

References [x](#).

Referenced by [atan_series\(\)](#), [atanh_series\(\)](#), [csgn_conjugate\(\)](#), [csgn_eval\(\)](#), [csgn_evalf\(\)](#), [csgn_power\(\)](#), [csgn_real_part\(\)](#), [csgn_series\(\)](#), [eta_eval\(\)](#), [eta_evalf\(\)](#), and [log_series\(\)](#).

5.1.3.659 is_zero() [2/2]

```
bool GiNaC::is_zero (
    const numeric & x ) [inline]
```

References [GiNaC::ex::is_zero\(\)](#), and [x](#).

5.1.3.660 is_positive()

```
bool GiNaC::is_positive (
    const numeric & x ) [inline]
```

References [x](#).

Referenced by [beta_eval\(\)](#).

5.1.3.661 is_negative()

```
bool GiNaC::is_negative (
    const numeric & x ) [inline]
```

References [x](#).

Referenced by [GiNaC::power::do_print_latex\(\)](#), [GiNaC::power::eval\(\)](#), and [frac_cancel\(\)](#).

5.1.3.662 is_integer()

```
bool GiNaC::is_integer (
    const numeric & x ) [inline]
```

References [x](#).

Referenced by [beta_eval\(\)](#), [binomial_eval\(\)](#), [GiNaC::power::coeff\(\)](#), [GiNaC::power::degree\(\)](#), [GiNaC::power::expand\(\)](#), [GiNaC::power::lddegree\(\)](#), [GiNaC::pseries::power_const\(\)](#), [psi1_eval\(\)](#), [psi2_eval\(\)](#), [replace_with_symbol\(\)](#), and [GiNaC::power::series\(\)](#).

5.1.3.663 is_pos_integer()

```
bool GiNaC::is_pos_integer (
    const numeric & x ) [inline]
```

References [x](#).

Referenced by [GiNaC::power::expand_add\(\)](#), and [GiNaC::power::expand_add_2\(\)](#).

5.1.3.664 is_nonneg_integer()

```
bool GiNaC::is_nonneg_integer (
    const numeric & x ) [inline]
```

References [x](#).

5.1.3.665 `is_even()`

```
bool GiNaC::is_even (
    const numeric & x ) [inline]
```

References [x](#).

Referenced by [abs_power\(\)](#), and [GiNaC::Kronecker_dz_kernel::series_coeff_impl\(\)](#).

5.1.3.666 `is_odd()`

```
bool GiNaC::is_odd (
    const numeric & x ) [inline]
```

References [x](#).

Referenced by [csgn_power\(\)](#).

5.1.3.667 `is_prime()`

```
bool GiNaC::is_prime (
    const numeric & x ) [inline]
```

References [x](#).

5.1.3.668 `is_rational()`

```
bool GiNaC::is_rational (
    const numeric & x ) [inline]
```

References [x](#).

Referenced by [beta_eval\(\)](#), [lgamma_eval\(\)](#), [replace_with_symbol\(\)](#), and [tgamma_eval\(\)](#).

5.1.3.669 `is_real()`

```
bool GiNaC::is_real (
    const numeric & x ) [inline]
```

References [x](#).

Referenced by [beta_eval\(\)](#), [fsolve\(\)](#), [G2_eval\(\)](#), [G2_evalf\(\)](#), and [Li2_series\(\)](#).

5.1.3.670 `is_cinteger()`

```
bool GiNaC::is_cinteger (
    const numeric & x ) [inline]
```

References [x](#).

5.1.3.671 is_crational()

```
bool GiNaC::is_crational (
    const numeric & x ) [inline]
```

References [x](#).

5.1.3.672 to_int()

```
int GiNaC::to_int (
    const numeric & x ) [inline]
```

References [x](#).

Referenced by [GiNaC::basic::operator\[\]\(\)](#), [GiNaC::basic::operator\[\]\(\)](#), [GiNaC::pseries::power_const\(\)](#), [S_eval\(\)](#), [GiNaC::power::series\(\)](#), and [sqrfree_parfrac\(\)](#).

5.1.3.673 to_long()

```
long GiNaC::to_long (
    const numeric & x ) [inline]
```

References [x](#).

5.1.3.674 to_double()

```
double GiNaC::to_double (
    const numeric & x ) [inline]
```

References [x](#).

Referenced by [GiNaC::Gt::qExpand\(\)](#).

5.1.3.675 real()

```
const numeric GiNaC::real (
    const numeric & x ) [inline]
```

References [x](#).

Referenced by [cosh_eval\(\)](#), [GiNaC::Gt::qExpand\(\)](#), [sinh_eval\(\)](#), and [tanh_eval\(\)](#).

5.1.3.676 imag()

```
const numeric GiNaC::imag (
    const numeric & x ) [inline]
```

References [x](#).

Referenced by [eta_eval\(\)](#), [eta_evalf\(\)](#), [G2_eval\(\)](#), and [G2_evalf\(\)](#).

5.1.3.677 numer() [2/2]

```
const numeric GiNaC::numer (
    const numeric & x ) [inline]
```

References [GiNaC::ex::numer\(\)](#), and [x](#).

5.1.3.678 denom() [2/2]

```
const numeric GiNaC::denom (
    const numeric & x ) [inline]
```

References [GiNaC::ex::denom\(\)](#), and [x](#).

5.1.3.679 exadd()

```
static const ex GiNaC::exadd (
    const ex & lh,
    const ex & rh ) [inline], [static]
```

Used internally by [operator+\(\)](#) to add two ex objects.

Referenced by [operator+\(\)](#), [operator++\(\)](#), [operator++\(\)](#), [operator+=\(\)](#), [operator-\(\)](#), [operator--\(\)](#), [operator--\(\)](#), and [operator-=\(\)](#).

5.1.3.680 exmul()

```
static const ex GiNaC::exmul (
    const ex & lh,
    const ex & rh ) [inline], [static]
```

Used internally by [operator*\(\)](#) to multiply two ex objects.

References [GiNaC::return_types::commutative](#), and [GiNaC::ex::return_type\(\)](#).

Referenced by [operator*\(\)](#), [operator*=\(\(\)\)](#), [operator/\(\)](#), and [operator/=\(\(\)\)](#).

5.1.3.681 exminus()

```
static const ex GiNaC::exminus (
    const ex & lh ) [inline], [static]
```

Used internally by [operator-\(\)](#) and friends to change the sign of an argument.

References [_ex_1](#).

Referenced by [operator-\(\)](#), [operator-\(\)](#), and [operator-=\(\)](#).

5.1.3.682 operator+() [1/4]

```
const ex GiNaC::operator+ (
    const ex & lh,
    const ex & rh )
```

References [exadd\(\)](#).

5.1.3.683 operator-() [1/4]

```
const ex GiNaC::operator- (
    const ex & lh,
    const ex & rh )
```

References [exadd\(\)](#), and [exminus\(\)](#).

5.1.3.684 operator*() [1/2]

```
const ex GiNaC::operator* (
    const ex & lh,
    const ex & rh )
```

References [exmul\(\)](#).

5.1.3.685 operator/() [1/2]

```
const ex GiNaC::operator/ (
    const ex & lh,
    const ex & rh )
```

References [_ex_1](#), and [exmul\(\)](#).

5.1.3.686 operator+() [2/4]

```
const numeric GiNaC::operator+ (
    const numeric & lh,
    const numeric & rh )
```

References [GiNaC::numeric::add\(\)](#).

5.1.3.687 operator-() [2/4]

```
const numeric GiNaC::operator- (
    const numeric & lh,
    const numeric & rh )
```

References [GiNaC::numeric::sub\(\)](#).

5.1.3.688 operator*() [2/2]

```
const numeric GiNaC::operator* (
    const numeric & lh,
    const numeric & rh )
```

References [GiNaC::numeric::mul\(\)](#).

5.1.3.689 operator/() [2/2]

```
const numeric GiNaC::operator/ (
    const numeric & lh,
    const numeric & rh )
```

References [GiNaC::numeric::div\(\)](#).

5.1.3.690 operator+=() [1/2]

```
ex & GiNaC::operator+= (
    ex & lh,
    const ex & rh )
```

References [exadd\(\)](#).

5.1.3.691 operator-=() [1/2]

```
ex & GiNaC::operator-= (
    ex & lh,
    const ex & rh )
```

References [exadd\(\)](#), and [exminus\(\)](#).

5.1.3.692 operator*=() [1/2]

```
ex & GiNaC::operator*= (
    ex & lh,
    const ex & rh )
```

References [exmul\(\)](#).

5.1.3.693 operator/=() [1/2]

```
ex & GiNaC::operator/= (
    ex & lh,
    const ex & rh )
```

References [_ex_1](#), and [exmul\(\)](#).

5.1.3.694 operator+=() [2/2]

```
numeric & GiNaC::operator+= (
    numeric & lh,
    const numeric & rh )
```

References [GiNaC::numeric::add\(\)](#).

5.1.3.695 operator-=() [2/2]

```
numeric & GiNaC::operator-= (
    numeric & lh,
    const numeric & rh )
```

References [GiNaC::numeric::sub\(\)](#).

5.1.3.696 operator*=() [2/2]

```
numeric & GiNaC::operator*= (
    numeric & lh,
    const numeric & rh )
```

References [GiNaC::numeric::mul\(\)](#).

5.1.3.697 operator/=() [2/2]

```
numeric & GiNaC::operator/= (
    numeric & lh,
    const numeric & rh )
```

References [GiNaC::numeric::div\(\)](#).

5.1.3.698 operator+() [3/4]

```
const ex GiNaC::operator+ (
    const ex & lh )
```

5.1.3.699 operator-() [3/4]

```
const ex GiNaC::operator- (
    const ex & lh )
```

References [exminus\(\)](#).

5.1.3.700 operator+() [4/4]

```
const numeric GiNaC::operator+ (
    const numeric & lh )
```

5.1.3.701 operator-() [4/4]

```
const numeric GiNaC::operator- (
    const numeric & lh )
```

References [_num_1_p](#), and [GiNaC::numeric::mul\(\)](#).

5.1.3.702 operator++() [1/4]

```
ex & GiNaC::operator++ (
    ex & rh )
```

Expression prefix increment.

Adds 1 and returns incremented ex.

References [_ex1](#), and [exadd\(\)](#).

5.1.3.703 operator--() [1/4]

```
ex & GiNaC::operator-- (
    ex & rh )
```

Expression prefix decrement.

Subtracts 1 and returns decremented ex.

References [_ex_1](#), and [exadd\(\)](#).

5.1.3.704 operator++() [2/4]

```
const ex GiNaC::operator++ (
    ex & lh,
    int )
```

Expression postfix increment.

Returns the ex and leaves the original incremented by 1.

References [_ex1](#), and [exadd\(\)](#).

5.1.3.705 operator--() [2/4]

```
const ex GiNaC::operator-- (
    ex & lh,
    int )
```

Expression postfix decrement.

Returns the ex and leaves the original decremented by 1.

References [_ex_1](#), and [exadd\(\)](#).

5.1.3.706 operator++() [3/4]

```
numeric & GiNaC::operator++ (
    numeric & rh )
```

Numeric prefix increment.

Adds 1 and returns incremented number.

References [_num1_p](#), and [GiNaC::numeric::add\(\)](#).

5.1.3.707 operator--() [3/4]

```
numeric & GiNaC::operator-- (
    numeric & rh )
```

Numeric prefix decrement.

Subtracts 1 and returns decremented number.

References [_num_1_p](#), and [GiNaC::numeric::add\(\)](#).

5.1.3.708 operator++() [4/4]

```
const numeric GiNaC::operator++ (
    numeric & lh,
    int )
```

Numeric postfix increment.

Returns the number and leaves the original incremented by 1.

References [_num1_p](#), and [GiNaC::numeric::add\(\)](#).

5.1.3.709 operator--() [4/4]

```
const numeric GiNaC::operator-- (
    numeric & lh,
    int )
```

Numeric postfix decrement.

Returns the number and leaves the original decremented by 1.

References [_num_1_p](#), and [GiNaC::numeric::add\(\)](#).

5.1.3.710 operator==()

```
const relational GiNaC::operator== (
    const ex & lh,
    const ex & rh )
```

References [GiNaC::relational::equal](#).

5.1.3.711 operator"!=()

```
const relational GiNaC::operator!= (
    const ex & lh,
    const ex & rh )
```

References [GiNaC::relational::not_equal](#).

5.1.3.712 operator<()

```
const relational GiNaC::operator< (
    const ex & lh,
    const ex & rh )
```

References [GiNaC::relational::less](#).

5.1.3.713 operator<=()

```
const relational GiNaC::operator<= (
    const ex & lh,
    const ex & rh )
```

References [GiNaC::relational::less_or_equal](#).

5.1.3.714 operator>()

```
const relational GiNaC::operator> (
    const ex & lh,
    const ex & rh )
```

References [GiNaC::relational::greater](#).

5.1.3.715 operator>=()

```
const relational GiNaC::operator>= (
    const ex & lh,
    const ex & rh )
```

References [GiNaC::relational::greater_or_equal](#).

5.1.3.716 my_ios_index()

```
static int GiNaC::my_ios_index ( ) [static]
```

Referenced by [get_print_context\(\)](#), and [set_print_context\(\)](#).

5.1.3.717 `my_ios_callback()`

```
static void GiNaC::my_ios_callback (
    std::ios_base::event ev,
    std::ios_base & s,
    int i ) [static]
```

Referenced by [set_print_context\(\)](#).

5.1.3.718 `get_print_context()`

```
static print\_context * GiNaC::get_print_context (
    std::ios_base & s ) [inline], [static]
```

References [my_ios_index\(\)](#).

Referenced by [get_print_options\(\)](#), [operator<<\(\)](#), [operator<<\(\)](#), [operator<<\(\)](#), [operator<<\(\)](#), and [set_print_options\(\)](#).

5.1.3.719 `set_print_context()`

```
static void GiNaC::set_print_context (
    std::ios_base & s,
    const print\_context & c ) [static]
```

References [c](#), [callback_registered](#), [my_ios_callback\(\)](#), [my_ios_index\(\)](#), [options](#), and [GiNaC::print_context::options](#).

Referenced by [csrc\(\)](#), [csrc_cl_N\(\)](#), [csrc_double\(\)](#), [csrc_float\(\)](#), [dflt\(\)](#), [latex\(\)](#), [python\(\)](#), [python_repr\(\)](#), [set_print_options\(\)](#), and [tree\(\)](#).

5.1.3.720 `get_print_options()`

```
static unsigned GiNaC::get_print_options (
    std::ios_base & s ) [inline], [static]
```

References [get_print_context\(\)](#), and [GiNaC::print_context::options](#).

Referenced by [index_dimensions\(\)](#), and [no_index_dimensions\(\)](#).

5.1.3.721 `set_print_options()`

```
static void GiNaC::set_print_options (
    std::ostream & s,
    unsigned options ) [static]
```

References [get_print_context\(\)](#), [options](#), [GiNaC::print_context::options](#), and [set_print_context\(\)](#).

Referenced by [dflt\(\)](#), [index_dimensions\(\)](#), and [no_index_dimensions\(\)](#).

5.1.3.722 operator<<() [7/16]

```
std::ostream & GiNaC::operator<< (
    std::ostream & os,
    const ex & e )
```

References [get_print_context\(\)](#), and [GiNaC::ex::print\(\)](#).

5.1.3.723 operator>>() [3/3]

```
std::istream & GiNaC::operator>> (
    std::istream & is,
    ex & e )
```

5.1.3.724 dflt()

```
std::ostream & GiNaC::dflt (
    std::ostream & os )
```

References [set_print_context\(\)](#), and [set_print_options\(\)](#).

5.1.3.725 latex()

```
std::ostream & GiNaC::latex (
    std::ostream & os )
```

References [set_print_context\(\)](#).

5.1.3.726 python()

```
std::ostream & GiNaC::python (
    std::ostream & os )
```

References [set_print_context\(\)](#).

5.1.3.727 python_repr()

```
std::ostream & GiNaC::python_repr (
    std::ostream & os )
```

References [set_print_context\(\)](#).

5.1.3.728 tree()

```
std::ostream & GiNaC::tree (
    std::ostream & os )
```

References [set_print_context\(\)](#).

Referenced by [GiNaC::class_info< OPT >::dump_hierarchy\(\)](#).

5.1.3.729 csrc()

```
std::ostream & GiNaC::csrc (
    std::ostream & os )
```

References [set_print_context\(\)](#).

5.1.3.730 csrc_float()

```
std::ostream & GiNaC::csrc_float (
    std::ostream & os )
```

References [set_print_context\(\)](#).

5.1.3.731 csrc_double()

```
std::ostream & GiNaC::csrc_double (
    std::ostream & os )
```

References [set_print_context\(\)](#).

5.1.3.732 csrc_cl_N()

```
std::ostream & GiNaC::csrc_cl_N (
    std::ostream & os )
```

References [set_print_context\(\)](#).

5.1.3.733 index_dimensions()

```
std::ostream & GiNaC::index_dimensions (
    std::ostream & os )
```

References [get_print_options\(\)](#), [GiNaC::print_options::print_index_dimensions](#), and [set_print_options\(\)](#).

5.1.3.734 no_index_dimensions()

```
std::ostream & GiNaC::no_index_dimensions (
    std::ostream & os )
```

References [get_print_options\(\)](#), [GiNaC::print_options::print_index_dimensions](#), and [set_print_options\(\)](#).

5.1.3.735 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [28/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    power ,
    basic ,
    print_func< print_dflt > &::do_print_dflt. print_func< print_latex > &::do←
_print_latex. print_func< print_csrc > &::do_print_csrc. print_func< print_python > &←
::do_print_python. print_func< print_python_repr > &::do_print_python_repr. print_func<
print_csrc_cl_N > &::do_print_csrc_cl_N )
```

5.1.3.736 print_sym_pow()

```
static void GiNaC::print_sym_pow (
    const print_context & c,
    const symbol & x,
    int exp ) [static]
```

References [c](#), [exp\(\)](#), [GiNaC::ex::print\(\)](#), [print_sym_pow\(\)](#), and [x](#).

Referenced by [GiNaC::power::do_print_csrc\(\)](#), and [print_sym_pow\(\)](#).

5.1.3.737 GINAC_BIND_UNARCHIVER() [38/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    power )
```

5.1.3.738 GINAC_DECLARE_UNARCHIVER() [40/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    power )
```

5.1.3.739 pow() [2/3]

```
ex GiNaC::pow (
    const ex & b,
    const ex & e ) [inline]
```

Symbolic exponentiation.

Returns a power-object as a new expression.

Parameters

<i>b</i>	the basis expression
<i>e</i>	the exponent expression

5.1.3.740 pow() [3/3]

```
template<typename T1 , typename T2 >
ex GiNaC::pow (
    const T1 & b,
    const T2 & e ) [inline]
```

5.1.3.741 sqrt() [2/2]

```
ex GiNaC::sqrt (
    const ex & a ) [inline]
```


Square root expression.

Returns a power-object with exponent 1/2.

References [_ex1_2](#).

5.1.3.742 `is_a()` [3/3]

```
template<class T >
bool GiNaC::is_a (
    const print\_context & obj ) [inline]
```

Check if obj is a T, including base classes.

5.1.3.743 `GINAC_IMPLEMENT_REGISTERED_CLASS_OPT()` [29/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    pseries ,
    basic ,
    print_func< print\_context > &::do_print. print_func< print\_latex > &::do_↵
    print_latex. print_func< print\_tree > &::do_print_tree. print_func< print\_python > &::do_↵
    print_python. print_func< print\_python\_repr > &::do_print_python_repr )
```

5.1.3.744 `GINAC_BIND_UNARCHIVER()` [39/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    pseries )
```

5.1.3.745 `GINAC_DECLARE_UNARCHIVER()` [41/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    pseries )
```

5.1.3.746 `series_to_poly()`

```
ex GiNaC::series_to_poly (
    const ex & e ) [inline]
```

Convert the pseries object embedded in an expression to an ordinary polynomial in the expansion variable.

The result is undefined if the expression does not contain a pseries object at its top level.

Parameters

<code>e</code>	expression

Returns

polynomial expression

See also

[is_a<>](#)

[pseries::convert_to_poly](#)

Referenced by [Bernoulli_polynomial\(\)](#), [generalised_Bernoulli_number\(\)](#), [GiNaC::modular_form_kernel::is_numeric\(\)](#), and [GiNaC::modular_form_kernel::Laurent_series\(\)](#).

5.1.3.747 is_terminating()

```
bool GiNaC::is_terminating (
    const pseries & s ) [inline]
```

References [GiNaC::pseries::is_terminating\(\)](#).

5.1.3.748 make_return_type_t()

```
template<typename T >
return_type_t GiNaC::make_return_type_t (
    const unsigned rl = 0 ) [inline]
```

References [GiNaC::return_type_t::rl](#), and [GiNaC::return_type_t::tinfo](#).

5.1.3.749 set_print_func() [1/2]

```
template<class Alg , class Ctx , class T , class C >
void GiNaC::set_print_func (
    void f(const T &, const C &c, unsigned) ) [extern]
```

Add or replace a print method.

References [GiNaC::class_info< OPT >::options](#), and [options](#).

5.1.3.750 set_print_func() [2/2]

```
template<class Alg , class Ctx , class T , class C >
void GiNaC::set_print_func (
    void(T::*)(const C &, unsigned) f ) [extern]
```

Add or replace a print method.

References [options](#).

5.1.3.751 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [30/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    relational ,
    basic ,
    print_func< print_context > &::do_print. print_func< print_tree > &::do_print↵
    _tree. print_func< print_python_repr > &::do_print_python_repr )
```

5.1.3.752 GINAC_BIND_UNARCHIVER() [40/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    relational )
```

5.1.3.753 print_operator()

```
static void GiNaC::print_operator (
    const print_context & c,
    relational::operators o ) [static]
```

References [c](#), [GiNaC::relational::equal](#), [GiNaC::relational::greater](#), [GiNaC::relational::greater_or_equal](#), [GiNaC::relational::less](#), [GiNaC::relational::less_or_equal](#), and [GiNaC::relational::not_equal](#).

Referenced by [GiNaC::relational::do_print\(\)](#), and [GiNaC::relational::do_print_python_repr\(\)](#).

5.1.3.754 GINAC_DECLARE_UNARCHIVER() [42/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    relational )
```

5.1.3.755 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [31/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    symbol ,
    basic ,
    print_func< print_context > &::do_print. print_func< print_latex > &::do_↵
    print_latex. print_func< print_tree > &::do_print_tree. print_func< print_python_repr > &↵
    ::do_print_python_repr )
```

References [GiNaC::status_flags::evaluated](#), and [GiNaC::status_flags::expanded](#).

5.1.3.756 get_default_TeX_name()

```
static const std::string & GiNaC::get_default_TeX_name (
    const std::string & name ) [static]
```

Return default TeX name for symbol.

This recognizes some greek letters.

Referenced by [GiNaC::symbol::do_print_latex\(\)](#), and [GiNaC::symbol::get_TeX_name\(\)](#).

5.1.3.757 GINAC_BIND_UNARCHIVER() [41/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    symbol )
```

5.1.3.758 GINAC_BIND_UNARCHIVER() [42/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    realsymbol )
```

5.1.3.759 GINAC_BIND_UNARCHIVER() [43/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    possymbol )
```

5.1.3.760 GINAC_DECLARE_UNARCHIVER() [43/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    symbol )
```

5.1.3.761 GINAC_DECLARE_UNARCHIVER() [44/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    realsymbol )
```

5.1.3.762 GINAC_DECLARE_UNARCHIVER() [45/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    possymbol )
```

5.1.3.763 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [32/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    symmetry ,
    basic ,
    print_func< print_context > &::do_print, print_func< print_tree > &::do_print↵
    _tree )
```

References [GiNaC::status_flags::evaluated](#), and [GiNaC::status_flags::expanded](#).

5.1.3.764 GINAC_BIND_UNARCHIVER() [44/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    symmetry )
```

5.1.3.765 index0()

```
static const symmetry & GiNaC::index0 ( ) [static]
```

Referenced by [antisymmetric2\(\)](#), [antisymmetric3\(\)](#), [antisymmetric4\(\)](#), [symmetric2\(\)](#), [symmetric3\(\)](#), and [symmetric4\(\)](#).

5.1.3.766 index1()

```
static const symmetry & GiNaC::index1 ( ) [static]
```

Referenced by [antisymmetric2\(\)](#), [antisymmetric3\(\)](#), [antisymmetric4\(\)](#), [symmetric2\(\)](#), [symmetric3\(\)](#), and [symmetric4\(\)](#).

5.1.3.767 index2()

```
static const symmetry & GiNaC::index2 ( ) [static]
```

Referenced by [antisymmetric3\(\)](#), [antisymmetric4\(\)](#), [symmetric3\(\)](#), and [symmetric4\(\)](#).

5.1.3.768 index3()

```
static const symmetry & GiNaC::index3 ( ) [static]
```

Referenced by [antisymmetric4\(\)](#), and [symmetric4\(\)](#).

5.1.3.769 not_symmetric()

```
const symmetry & GiNaC::not_symmetric ( )
```

Referenced by [GiNaC::indexed::indexed\(\)](#), and [GiNaC::indexed::read_archive\(\)](#).

5.1.3.770 symmetric2()

```
const symmetry & GiNaC::symmetric2 ( )
```

References [index0\(\)](#), [index1\(\)](#), and [GiNaC::symmetry::symmetric](#).

Referenced by [delta_tensor\(\)](#), [GiNaC::clifford::get_metric\(\)](#), [lorentz_g\(\)](#), and [metric_tensor\(\)](#).

5.1.3.771 symmetric3()

```
const symmetry & GiNaC::symmetric3 ( )
```

References [index0\(\)](#), [index1\(\)](#), [index2\(\)](#), and [GiNaC::symmetry::symmetric](#).

Referenced by [color_d\(\)](#).

5.1.3.772 symmetric4()

```
const symmetry & GiNaC::symmetric4 ( )
```

References [index0\(\)](#), [index1\(\)](#), [index2\(\)](#), [index3\(\)](#), and [GiNaC::symmetry::symmetric](#).

5.1.3.773 antisymmetric2()

```
const symmetry & GiNaC::antisymmetric2 ( )
```

References [GiNaC::symmetry::antisymmetric](#), [index0\(\)](#), and [index1\(\)](#).

Referenced by [epsilon_tensor\(\)](#), and [spinor_metric\(\)](#).

5.1.3.774 antisymmetric3()

```
const symmetry & GiNaC::antisymmetric3 ( )
```

References [GiNaC::symmetry::antisymmetric](#), [index0\(\)](#), [index1\(\)](#), and [index2\(\)](#).

Referenced by [color_f\(\)](#), and [epsilon_tensor\(\)](#).

5.1.3.775 antisymmetric4()

```
const symmetry & GiNaC::antisymmetric4 ( )
```

References [GiNaC::symmetry::antisymmetric](#), [index0\(\)](#), [index1\(\)](#), [index2\(\)](#), and [index3\(\)](#).

Referenced by [lorentz_eps\(\)](#).

5.1.3.776 canonicalize()

```
int GiNaC::canonicalize (
    exvector::iterator v,
    const symmetry & symm )
```

Canonicalize the order of elements of an expression vector, according to the symmetry properties defined in a symmetry tree.

Parameters

<i>v</i>	Start of expression vector
<i>symm</i>	Root node of symmetry tree

Returns

the overall sign introduced by the reordering (+1, -1 or 0) or `numeric_limits<int>::max()` if nothing changed

Referenced by [GiNaC::function::eval\(\)](#), and [GiNaC::indexed::eval\(\)](#).

5.1.3.777 symm()

```
static ex GiNaC::symm (
    const ex & e,
    exvector::const_iterator first,
    exvector::const_iterator last,
    bool asymmetric ) [static]
```

References [GiNaC::container< C >::append\(\)](#), [factorial\(\)](#), [last](#), [GiNaC::subs_options::no_index_renaming](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::container< C >::op\(\)](#), [permutation_sign\(\)](#), and [GiNaC::ex::subs\(\)](#).

Referenced by [antisymmetrize\(\)](#), [GiNaC::ex::antisymmetrize\(\)](#), [GiNaC::indexed::read_archive\(\)](#), [symmetrize\(\)](#), and [GiNaC::ex::symmetrize\(\)](#).

5.1.3.778 symmetrize() [3/4]

```
ex GiNaC::symmetrize (
    const ex & e,
    exvector::const_iterator first,
    exvector::const_iterator last )
```

Symmetrize expression over a set of objects (symbols, indices).

References [last](#), and [symm\(\)](#).

5.1.3.779 antisymmetrize() [3/4]

```
ex GiNaC::antisymmetrize (
    const ex & e,
    exvector::const_iterator first,
    exvector::const_iterator last )
```

Antisymmetrize expression over a set of objects (symbols, indices).

References [last](#), and [symm\(\)](#).

5.1.3.780 symmetrize_cyclic() [3/4]

```
ex GiNaC::symmetrize_cyclic (
    const ex & e,
    exvector::const_iterator first,
    exvector::const_iterator last )
```

Symmetrize expression by cyclic permutation over a set of objects (symbols, indices).

References [GiNaC::container< C >::append\(\)](#), [last](#), [GiNaC::subs_options::no_index_renaming](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::container< C >::op\(\)](#), [GiNaC::container< C >::remove_first\(\)](#), and [GiNaC::ex::subs\(\)](#).

5.1.3.781 GINAC_DECLARE_UNARCHIVER() [46/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    symmetry )
```

5.1.3.782 sy_none() [1/4]

```
symmetry GiNaC::sy_none ( ) [inline]
```

5.1.3.783 sy_none() [2/4]

```
symmetry GiNaC::sy_none (
    const symmetry & c1,
    const symmetry & c2 ) [inline]
```

References [GiNaC::symmetry::none](#).

5.1.3.784 sy_none() [3/4]

```
symmetry GiNaC::sy_none (
    const symmetry & c1,
    const symmetry & c2,
    const symmetry & c3 ) [inline]
```

References [GiNaC::symmetry::add\(\)](#), and [GiNaC::symmetry::none](#).

5.1.3.785 sy_none() [4/4]

```
symmetry GiNaC::sy_none (
    const symmetry & c1,
    const symmetry & c2,
    const symmetry & c3,
    const symmetry & c4 ) [inline]
```

References [GiNaC::symmetry::add\(\)](#), and [GiNaC::symmetry::none](#).

5.1.3.786 sy_symm() [1/4]

```
symmetry GiNaC::sy_symm ( ) [inline]
```

References [GiNaC::symmetry::set_type\(\)](#), and [GiNaC::symmetry::symmetric](#).

Referenced by [GiNaC::indexed::read_archive\(\)](#).

5.1.3.787 sy_symm() [2/4]

```
symmetry GiNaC::sy_symm (
    const symmetry & c1,
    const symmetry & c2 ) [inline]
```

References [GiNaC::symmetry::symmetric](#).

5.1.3.788 sy_symm() [3/4]

```
symmetry GiNaC::sy_symm (
    const symmetry & c1,
    const symmetry & c2,
    const symmetry & c3 ) [inline]
```

References [GiNaC::symmetry::add\(\)](#), and [GiNaC::symmetry::symmetric](#).

5.1.3.789 sy_symm() [4/4]

```
symmetry GiNaC::sy_symm (
    const symmetry & c1,
    const symmetry & c2,
    const symmetry & c3,
    const symmetry & c4 ) [inline]
```

References [GiNaC::symmetry::add\(\)](#), and [GiNaC::symmetry::symmetric](#).

5.1.3.790 sy_anti() [1/4]

```
symmetry GiNaC::sy_anti ( ) [inline]
```

References [GiNaC::symmetry::antisymmetric](#), and [GiNaC::symmetry::set_type\(\)](#).

Referenced by [GiNaC::indexed::read_archive\(\)](#).

5.1.3.791 sy_anti() [2/4]

```
symmetry GiNaC::sy_anti (
    const symmetry & c1,
    const symmetry & c2 ) [inline]
```

References [GiNaC::symmetry::antisymmetric](#).

5.1.3.792 sy_anti() [3/4]

```
symmetry GiNaC::sy_anti (
    const symmetry & c1,
    const symmetry & c2,
    const symmetry & c3 ) [inline]
```

References [GiNaC::symmetry::add\(\)](#), and [GiNaC::symmetry::antisymmetric](#).

5.1.3.793 sy_anti() [4/4]

```
symmetry GiNaC::sy_anti (
    const symmetry & c1,
    const symmetry & c2,
    const symmetry & c3,
    const symmetry & c4 ) [inline]
```

References [GiNaC::symmetry::add\(\)](#), and [GiNaC::symmetry::antisymmetric](#).

5.1.3.794 sy_cycl() [1/4]

```
symmetry GiNaC::sy_cycl ( ) [inline]
```

References [GiNaC::symmetry::cyclic](#), and [GiNaC::symmetry::set_type\(\)](#).

5.1.3.795 sy_cycl() [2/4]

```
symmetry GiNaC::sy_cycl (
    const symmetry & c1,
    const symmetry & c2 ) [inline]
```

References [GiNaC::symmetry::cyclic](#).

5.1.3.796 sy_cycl() [3/4]

```
symmetry GiNaC::sy_cycl (
    const symmetry & c1,
    const symmetry & c2,
    const symmetry & c3 ) [inline]
```

References [GiNaC::symmetry::add\(\)](#), and [GiNaC::symmetry::cyclic](#).

5.1.3.797 sy_cycl() [4/4]

```
symmetry GiNaC::sy_cycl (
    const symmetry & c1,
    const symmetry & c2,
    const symmetry & c3,
    const symmetry & c4 ) [inline]
```

References [GiNaC::symmetry::add\(\)](#), and [GiNaC::symmetry::cyclic](#).

5.1.3.798 symmetrize() [4/4]

```
ex GiNaC::symmetrize (
    const ex & e,
    const exvector & v ) [inline]
```

Symmetrize expression over a set of objects (symbols, indices).

References [GiNaC::ex::begin\(\)](#), and [symmetrize\(\)](#).

5.1.3.799 antisymmetrize() [4/4]

```
ex GiNaC::antisymmetrize (
    const ex & e,
    const exvector & v ) [inline]
```

Antisymmetrize expression over a set of objects (symbols, indices).

References [antisymmetrize\(\)](#), and [GiNaC::ex::begin\(\)](#).

5.1.3.800 symmetrize_cyclic() [4/4]

```
ex GiNaC::symmetrize_cyclic (
    const ex & e,
    const exvector & v ) [inline]
```

Symmetrize expression by cyclic permutation over a set of objects (symbols, indices).

References [GiNaC::ex::begin\(\)](#), and [symmetrize\(\)](#).

5.1.3.801 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [33/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    tensdelta ,
    tensor ,
    print_func< print_dflt > &::do_print. print_func< print_latex > &::do_print_↵
    latex )
```

5.1.3.802 print_func< print_dflt >() [3/3]

```
GiNaC::print_func< print_dflt > (
    &tensmetric::do_print ) &
```

References [GiNaC::status_flags::evaluated](#), and [GiNaC::status_flags::expanded](#).

5.1.3.803 GINAC_BIND_UNARCHIVER() [45/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    minkmetric )
```

5.1.3.804 GINAC_BIND_UNARCHIVER() [46/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    tensepsilon )
```

5.1.3.805 GINAC_BIND_UNARCHIVER() [47/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    tensdelta )
```

5.1.3.806 GINAC_BIND_UNARCHIVER() [48/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    tensmetric )
```

5.1.3.807 GINAC_BIND_UNARCHIVER() [49/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    spinmetric )
```

5.1.3.808 delta_tensor()

```
ex GiNaC::delta_tensor (
    const ex & i1,
    const ex & i2 )
```

Create a delta tensor with specified indices.

The indices must be of class `idx` or a subclass. The delta tensor is always symmetric and its trace is the dimension of the index space.

Parameters

<i>i1</i>	First index
<i>i2</i>	Second index

Returns

newly constructed delta tensor

References [symmetric2\(\)](#).

Referenced by [color_trace\(\)](#), [GiNaC::su3f::contract_with\(\)](#), [GiNaC::su3d::contract_with\(\)](#), [GiNaC::spinmetric::contract_with\(\)](#), [GiNaC::tensepsilon::contract_with\(\)](#), and [GiNaC::tensmetric::eval_indexed\(\)](#).

5.1.3.809 metric_tensor()

```
ex GiNaC::metric_tensor (
    const ex & i1,
    const ex & i2 )
```

Create a symmetric metric tensor with specified indices.

The indices must be of class `varidx` or a subclass. A metric tensor with one covariant and one contravariant index is equivalent to the delta tensor.

Parameters

<i>i1</i>	First index
<i>i2</i>	Second index

Returns

newly constructed metric tensor

References [symmetric2\(\)](#).

Referenced by [GiNaC::tensepsilon::contract_with\(\)](#).

5.1.3.810 `lorentz_g()`

```
ex GiNaC::lorentz_g (
    const ex & i1,
    const ex & i2,
    bool pos_sig = false )
```

Create a Minkowski metric tensor with specified indices.

The indices must be of class `varidx` or a subclass. The Lorentz metric is a symmetric tensor with a matrix representation of $\text{diag}(1, -1, -1, \dots)$ (negative signature, the default) or $\text{diag}(-1, 1, 1, \dots)$ (positive signature).

Parameters

<i>i1</i>	First index
<i>i2</i>	Second index
<i>pos_sig</i>	Whether the signature is positive

Returns

newly constructed Lorentz metric tensor

References [symmetric2\(\)](#).

Referenced by [GiNaC::tensepsilon::contract_with\(\)](#).

5.1.3.811 `spinor_metric()`

```
ex GiNaC::spinor_metric (
    const ex & i1,
    const ex & i2 )
```

Create a spinor metric tensor with specified indices.

The indices must be of class `spinidx` or a subclass and have a dimension of 2. The spinor metric is an antisymmetric tensor with a matrix representation of $\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$.

Parameters

<i>i1</i>	First index
<i>i2</i>	Second index

Returns

newly constructed spinor metric tensor

References [antisymmetric2\(\)](#).

5.1.3.812 epsilon_tensor() [1/2]

```
ex GiNaC::epsilon_tensor (
    const ex & i1,
    const ex & i2 )
```

Create an epsilon tensor in a Euclidean space with two indices.

The indices must be of class `idx` or a subclass, and have a dimension of 2.

Parameters

<i>i1</i>	First index
<i>i2</i>	Second index

Returns

newly constructed epsilon tensor

References [_ex2](#), [antisymmetric2\(\)](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::is_equal\(\)](#), and [GiNaC::ex::op\(\)](#).

5.1.3.813 epsilon_tensor() [2/2]

```
ex GiNaC::epsilon_tensor (
    const ex & i1,
    const ex & i2,
    const ex & i3 )
```

Create an epsilon tensor in a Euclidean space with three indices.

The indices must be of class `idx` or a subclass, and have a dimension of 3.

Parameters

<i>i1</i>	First index
<i>i2</i>	Second index
<i>i3</i>	Third index

Returns

newly constructed epsilon tensor

References [_ex3](#), [antisymmetric3\(\)](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::is_equal\(\)](#), and [GiNaC::ex::op\(\)](#).

5.1.3.814 lorentz_eps()

```
ex GiNaC::lorentz_eps (
    const ex & i1,
    const ex & i2,
```

```
const ex & i3,
const ex & i4,
bool pos_sig = false )
```

Create an epsilon tensor in a Minkowski space with four indices.

The indices must be of class `varidx` or a subclass, and have a dimension of 4.

Parameters

<i>i1</i>	First index
<i>i2</i>	Second index
<i>i3</i>	Third index
<i>i4</i>	Fourth index
<i>pos_sig</i>	Whether the signature of the metric is positive

Returns

newly constructed epsilon tensor

References [_ex4](#), [antisymmetric4\(\)](#), [GiNaC::ex::is_equal\(\)](#), and [GiNaC::ex::op\(\)](#).

5.1.3.815 GINAC_DECLARE_UNARCHIVER() [47/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    tensdelta )
```

5.1.3.816 GINAC_DECLARE_UNARCHIVER() [48/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    tensmetric )
```

5.1.3.817 GINAC_DECLARE_UNARCHIVER() [49/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    minkmetric )
```

5.1.3.818 GINAC_DECLARE_UNARCHIVER() [50/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    spinmetric )
```

5.1.3.819 GINAC_DECLARE_UNARCHIVER() [51/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    tensepsilon )
```

5.1.3.820 log2()

```
unsigned GiNaC::log2 (
    unsigned n )
```

Integer binary logarithm.

References [k](#), and [n](#).

Referenced by [GiNaC::remember_table::remember_table\(\)](#).

5.1.3.821 multinomial_coefficient()

```
const numeric GiNaC::multinomial_coefficient (
    const std::vector< unsigned > & p )
```

Compute the multinomial coefficient $n!/(p_1! * p_2! * \dots * p_k!)$ where $n = p_1 + p_2 + \dots + p_k$, i.e.

p is a partition of n .

References [GiNaC::numeric::add\(\)](#), [GiNaC::numeric::div\(\)](#), [factorial\(\)](#), [GiNaC::numeric::mul\(\)](#), and [n](#).

Referenced by [GiNaC::power::expand_add\(\)](#).

5.1.3.822 rotate_left()

```
unsigned GiNaC::rotate_left (
    unsigned n ) [inline]
```

Rotate bits of unsigned value by one bit to the left.

This can be necessary if the user wants to define its own hashes.

References [n](#).

Referenced by [GiNaC::basic::calchash\(\)](#), [GiNaC::expairseq::calchash\(\)](#), [GiNaC::function::calchash\(\)](#), [GiNaC::Gt::calchash\(\)](#), [GiNaC::idx::calchash\(\)](#), [GiNaC::relational::calchash\(\)](#), and [GiNaC::symmetry::calchash\(\)](#).

5.1.3.823 compare_pointers()

```
template<class T >
int GiNaC::compare_pointers (
    const T * a,
    const T * b ) [inline]
```

Compare two pointers (just to establish some sort of canonical order).

Returns

-1, 0, or 1

Referenced by [GiNaC::basic::compare_same_type\(\)](#).

5.1.3.824 `golden_ratio_hash()`

```
unsigned GiNaC::golden_ratio_hash (
    uintptr_t n ) [inline]
```

Truncated multiplication with golden ratio, for computing hash values.

References [n](#).

Referenced by [GiNaC::constant::calchash\(\)](#), [GiNaC::function::calchash\(\)](#), [GiNaC::numeric::calchash\(\)](#), [GiNaC::symbol::calchash\(\)](#), [GiNaC::wildcard::calchash\(\)](#), and [make_hash_seed\(\)](#).

5.1.3.825 `permutation_sign()` [1/2]

```
template<class It >
int GiNaC::permutation_sign (
    It first,
    It last )
```

References [last](#), [swap\(\)](#), and [std::swap\(\)](#).

Referenced by [GiNaC::matrix::determinant\(\)](#), [GiNaC::tensepsilon::eval_indexed\(\)](#), and [symm\(\)](#).

5.1.3.826 `permutation_sign()` [2/2]

```
template<class It , class Cmp , class Swap >
int GiNaC::permutation_sign (
    It first,
    It last,
    Cmp comp,
    Swap swapit )
```

References [last](#).

5.1.3.827 `shaker_sort()`

```
template<class It , class Cmp , class Swap >
void GiNaC::shaker_sort (
    It first,
    It last,
    Cmp comp,
    Swap swapit )
```

References [last](#).

Referenced by [find_free_and_dummy\(\)](#), and [rename_dummy_indices\(\)](#).

5.1.3.828 cyclic_permutation()

```
template<class It , class Swap >
void GiNaC::cyclic_permutation (
    It first,
    It last,
    It new_first,
    Swap swapit )
```

References [last](#).

5.1.3.829 format_index_value() [1/2]

```
template<typename T >
std::enable_if< has\_distance< T >::value, typename std::iterator_traits< T >::difference_type
>::type GiNaC::format_index_value (
    const T & a,
    const T & b )
```

For printing a multi-index: If the templates are used, where T is an iterator, printing the address where the iterator points to is not meaningful.

However, we may print the difference to the starting point.

Referenced by [operator<<\(\)](#), [operator<<\(\)](#), [operator<<\(\)](#), [operator<<\(\)](#), [operator<<\(\)](#), [operator<<\(\)](#), [operator<<\(\)](#), and [operator<<\(\)](#).

5.1.3.830 format_index_value() [2/2]

```
template<typename T >
std::enable_if<!has\_distance< T >::value, T >::type GiNaC::format_index_value (
    const T & a,
    const T & b )
```

For all other cases we simply print the value.

5.1.3.831 operator<<() [8/16]

```
template<class T >
std::ostream & GiNaC::operator<< (
    std::ostream & os,
    const basic\_multi\_iterator< T > & v ) [inline]
```

Output operator.

A multi_iterator prints out as [basic_multi_iterator](#)($n_0, n_1, \dots$).

References [GiNaC::basic_multi_iterator< T >::B](#), [format_index_value\(\)](#), and [GiNaC::basic_multi_iterator< T >::size\(\)](#).

5.1.3.832 operator<<() [9/16]

```
template<class T >
std::ostream & GiNaC::operator<< (
    std::ostream & os,
    const multi_iterator_ordered< T > & v ) [inline]
```

Output operator.

A `multi_iterator_ordered` prints out as `multi_iterator_ordered(  $n_0, n_1, \dots$  )`.

References `GiNaC::basic_multi_iterator< T >::B`, `format_index_value()`, and `GiNaC::basic_multi_iterator< T >::size()`.

5.1.3.833 operator<<() [10/16]

```
template<class T >
std::ostream & GiNaC::operator<< (
    std::ostream & os,
    const multi_iterator_ordered_eq< T > & v ) [inline]
```

Output operator.

A `multi_iterator_ordered_eq` prints out as `multi_iterator_ordered_eq(  $n_0, n_1, \dots$  )`.

References `GiNaC::basic_multi_iterator< T >::B`, `format_index_value()`, and `GiNaC::basic_multi_iterator< T >::size()`.

5.1.3.834 operator<<() [11/16]

```
template<class T >
std::ostream & GiNaC::operator<< (
    std::ostream & os,
    const multi_iterator_ordered_eq_indv< T > & v ) [inline]
```

Output operator.

A `multi_iterator_ordered_eq_indv` prints out as `multi_iterator_ordered_eq_indv(  $n_0, n_1, \dots$  )`.

References `GiNaC::basic_multi_iterator< T >::B`, `format_index_value()`, and `GiNaC::basic_multi_iterator< T >::size()`.

5.1.3.835 operator<<() [12/16]

```
template<class T >
std::ostream & GiNaC::operator<< (
    std::ostream & os,
    const multi_iterator_counter< T > & v ) [inline]
```

Output operator.

A `multi_iterator_counter` prints out as `multi_iterator_counter(  $n_0, n_1, \dots$  )`.

References `GiNaC::basic_multi_iterator< T >::B`, `format_index_value()`, and `GiNaC::basic_multi_iterator< T >::size()`.

5.1.3.836 operator<<() [13/16]

```
template<class T >
std::ostream & GiNaC::operator<< (
    std::ostream & os,
    const multi_iterator_counter_indv< T > & v ) [inline]
```

Output operator.

A `multi_iterator_counter_indv` prints out as `multi_iterator_counter_indv(  $n_0, n_1, \dots$  )`.

References [GiNaC::basic_multi_iterator< T >::B](#), [format_index_value\(\)](#), and [GiNaC::basic_multi_iterator< T >::size\(\)](#).

5.1.3.837 operator<<() [14/16]

```
template<class T >
std::ostream & GiNaC::operator<< (
    std::ostream & os,
    const multi_iterator_permutation< T > & v ) [inline]
```

Output operator.

A `multi_iterator_permutation` prints out as `multi_iterator_permutation(  $n_0, n_1, \dots$  )`.

References [GiNaC::basic_multi_iterator< T >::B](#), [format_index_value\(\)](#), and [GiNaC::basic_multi_iterator< T >::size\(\)](#).

5.1.3.838 operator<<() [15/16]

```
template<class T >
std::ostream & GiNaC::operator<< (
    std::ostream & os,
    const multi_iterator_shuffle< T > & v ) [inline]
```

Output operator.

A `multi_iterator_shuffle` prints out as `multi_iterator_shuffle(  $n_0, n_1, \dots$  )`.

References [GiNaC::basic_multi_iterator< T >::B](#), [format_index_value\(\)](#), and [GiNaC::basic_multi_iterator< T >::size\(\)](#).

5.1.3.839 operator<<() [16/16]

```
template<class T >
std::ostream & GiNaC::operator<< (
    std::ostream & os,
    const multi_iterator_shuffle_prime< T > & v ) [inline]
```

Output operator.

A `multi_iterator_shuffle_prime` prints out as `multi_iterator_shuffle_prime(  $n_0, n_1, \dots$  )`.

References [GiNaC::basic_multi_iterator< T >::B](#), [format_index_value\(\)](#), and [GiNaC::basic_multi_iterator< T >::size\(\)](#).

5.1.3.840 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT() [34/34]

```
GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT (
    wildcard ,
    basic ,
    print_func< print_context > &::do_print, print_func< print_tree > &::do_print←
_tree, print_func< print_python_repr > &::do_print_python_repr )
```

References [GiNaC::status_flags::evaluated](#), and [GiNaC::status_flags::expanded](#).

5.1.3.841 GINAC_BIND_UNARCHIVER() [50/50]

```
GiNaC::GINAC_BIND_UNARCHIVER (
    wildcard )
```

5.1.3.842 haswild()

```
bool GiNaC::haswild (
    const ex & x )
```

Check whether x has a wildcard anywhere as a subexpression.

References [haswild\(\)](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), and [x](#).

Referenced by [GiNaC::integral::eval\(\)](#), and [haswild\(\)](#).

5.1.3.843 GINAC_DECLARE_UNARCHIVER() [52/52]

```
GiNaC::GINAC_DECLARE_UNARCHIVER (
    wildcard )
```

5.1.3.844 wild()

```
ex GiNaC::wild (
    unsigned label = 0 ) [inline]
```

Create a wildcard object with the specified label.

5.1.4 Variable Documentation**5.1.4.1 unarch_table_instance**

```
unarchive_table_t GiNaC::unarch_table_instance [static]
```

5.1.4.2 map_evalm

```
GiNaC::evalm_map_function GiNaC::map_evalm
```

Referenced by [GiNaC::basic::evalm\(\)](#).

5.1.4.3 map_eval_integ

`GiNaC::eval_integ_map_function` `GiNaC::map_eval_integ`

Referenced by [GiNaC::basic::eval_integ\(\)](#).

5.1.4.4 tensor

`GiNaC::tensor`

5.1.4.5 Pi

```
const constant GiNaC::Pi (
    "Pi" ,
    PiEvalf ,
    "\\pi" ,
    domain::positive )
```

Pi.

(3.14159...) Diverts straight into CLN for [evalf\(\)](#).

Referenced by [acos_eval\(\)](#), [acosh_eval\(\)](#), [asin_eval\(\)](#), [atan2_eval\(\)](#), [atan_eval\(\)](#), [atan_series\(\)](#), [atanh_series\(\)](#), [GiNaC::Eisenstein_h_kernel::coefficient_a0\(\)](#), [GiNaC::Eisenstein_h_kernel::coefficient_an\(\)](#), [cos_eval\(\)](#), [cosh_eval\(\)](#), [EllipticE_eval\(\)](#), [EllipticE_evalf\(\)](#), [EllipticE_series\(\)](#), [EllipticK_eval\(\)](#), [EllipticK_evalf\(\)](#), [EllipticK_series\(\)](#), [eta_eval\(\)](#), [eta_evalf\(\)](#), [exp_eval\(\)](#), [GiNaC::Kronecker_dtau_kernel::get_numerical_value\(\)](#), [GiNaC::Kronecker_dz_kernel::get_numerical_value\(\)](#), [H_evalf\(\)](#), [Li2_eval\(\)](#), [Li2_series\(\)](#), [Li_eval\(\)](#), [log_eval\(\)](#), [log_series\(\)](#), [GiNaC::Gt::qExpand\(\)](#), [GiNaC::constant::read_archive\(\)](#), [GiNaC::Kronecker_dtau_kernel::series_coeff_impl\(\)](#), [GiNaC::Kronecker_dz_kernel::series_coeff_impl\(\)](#), [sin_eval\(\)](#), [sinh_eval\(\)](#), [tan_eval\(\)](#), [tan_series\(\)](#), [tanh_eval\(\)](#), [tanh_series\(\)](#), [GiNaC::Gt::tauToFundamental\(\)](#), [tgamma_eval\(\)](#), [zeta1_eval\(\)](#), and [GiNaC::Gt::zisToFundamental\(\)](#).

5.1.4.6 Euler

```
const constant GiNaC::Euler (
    "Euler" ,
    EulerEvalf ,
    "\\gamma_E" ,
    domain::positive )
```

Euler's constant.

(0.57721...) Sometimes called Euler-Mascheroni constant. Diverts straight into CLN for [evalf\(\)](#).

Referenced by [psi1_eval\(\)](#), and [GiNaC::constant::read_archive\(\)](#).

5.1.4.7 Catalan

```
const constant GiNaC::Catalan (
    "Catalan" ,
    CatalanEvalf ,
    "G" ,
    domain::positive )
```

Catalan's constant.

(0.91597...) Diverts straight into CLN for [evalf\(\)](#).

Referenced by [Li2_eval\(\)](#), [Li_eval\(\)](#), and [GiNaC::constant::read_archive\(\)](#).

5.1.4.8 crctab

```
unsigned const GiNaC::crctab[256] [static]
```

Referenced by [crc32\(\)](#).

5.1.4.9 library_initializer

```
library_init GiNaC::library_initializer [static]
```

For construction of flyweights, etc.

5.1.4.10 _num0_bp

```
const basic * GiNaC::_num0_bp
```

Referenced by [GiNaC::library_init::library_init\(\)](#).

5.1.4.11 idx

```
GiNaC::idx
```

5.1.4.12 force_include_tgamma

```
unsigned GiNaC::force_include_tgamma = tgamma_SERIAL::serial
```

5.1.4.13 force_include_zeta1

```
unsigned GiNaC::force_include_zeta1 = zeta1_SERIAL::serial
```

5.1.4.14 GINAC_BIND_UNARCHIVER

```
template<>
GINAC_IMPLEMENT_REGISTERED_CLASS_OPT_T(lst, basic, print_func< print_context >(&lst::do_print).
print_func< print_tree >(&lst::do_print_tree)) template<> bool lst GiNaC::GINAC_BIND_UNARCHIVER(lst)
(
    lst )
```

Specialization of [container::info\(\)](#) for lst.

5.1.4.15 I

```
const numeric GiNaC::I = numeric(cln::complex(cln::cl_I(0),cln::cl_I(1)))
```

Imaginary unit.

This is not a constant but a numeric since we are natively handing complex numbers anyways, so in each expression containing an I it is automatically eval'ed away anyhow.

Referenced by [acosh_eval\(\)](#), [atan_eval\(\)](#), [atan_series\(\)](#), [atanh_series\(\)](#), [GiNaC::Eisenstein_h_kernel::coefficient_a0\(\)](#), [GiNaC::Eisenstein_h_kernel::coefficient_an\(\)](#), [color_h\(\)](#), [GiNaC::su3f::contract_with\(\)](#), [cosh_eval\(\)](#), [csgn_eval\(\)](#), [eta_eval\(\)](#), [eta_evalf\(\)](#), [eta_imag_part\(\)](#), [exp_eval\(\)](#), [GiNaC::Kronecker_dtau_kernel::get_numerical_value\(\)](#), [GiNaC::Kronecker_dz_kernel::get_numerical_value\(\)](#), [H_evalf\(\)](#), [GiNaC::numeric::has\(\)](#), [Li2_eval\(\)](#), [Li2_series\(\)](#), [Li_eval\(\)](#), [log_eval\(\)](#), [log_series\(\)](#), [GiNaC::numeric::normal\(\)](#), [GiNaC::Gt::qExpand\(\)](#), [GiNaC::Kronecker_dtau_kernel::series_coeff_im\(\)](#), [GiNaC::Kronecker_dz_kernel::series_coeff_impl\(\)](#), [sinh_eval\(\)](#), [step_eval\(\)](#), [tanh_eval\(\)](#), [tanh_series\(\)](#), [GiNaC::Gt::tauToFundamental\(\)](#), [GiNaC::numeric::to_polynomial\(\)](#), [GiNaC::numeric::to_rational\(\)](#), and [GiNaC::Gt::zisToFundamental\(\)](#).

5.1.4.16 Digits

```
_numeric_digits GiNaC::Digits
```

Accuracy in decimal digits.

Only object of this type! Can be set using assignment from C++ unsigned ints and evaluated like any built-in type.

Referenced by [GiNaC::integration_kernel::get_numerical_value_impl\(\)](#), [iterated_integral_evalf_impl\(\)](#), [GiNaC::numeric::numeric\(\)](#), [print_real_cl_N\(\)](#), [GiNaC::Gt::qExpand\(\)](#), and [zeta1_evalf\(\)](#).

5.1.4.17 next_print_context_id

```
unsigned GiNaC::next_print_context_id = 0
```

Next free ID for [print_context](#) types.

5.1.4.18 version_major

```
const int GiNaC::version_major = GINACLIB_MAJOR_VERSION
```

5.1.4.19 version_minor

```
const int GiNaC::version_minor = GINACLIB_MINOR_VERSION
```

5.1.4.20 version_micro

```
const int GiNaC::version_micro = GINACLIB_MICRO_VERSION
```


5.1.4.21 `_num_120_p`

```
const numeric * GiNaC::_num_120_p
```

Referenced by [GiNaC::library_init::library_init\(\)](#).

5.1.4.22 `_ex_120`

```
const ex GiNaC::_ex_120 = ex(*_num_120_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.23 `_num_60_p`

```
const numeric * GiNaC::_num_60_p
```

Referenced by [GiNaC::library_init::library_init\(\)](#).

5.1.4.24 `_ex_60`

```
const ex GiNaC::_ex_60 = ex(*_num_60_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.25 `_num_48_p`

```
const numeric * GiNaC::_num_48_p
```

Referenced by [GiNaC::library_init::library_init\(\)](#).

5.1.4.26 `_ex_48`

```
const ex GiNaC::_ex_48 = ex(*_num_48_p)
```

Referenced by [Li2_eval\(\)](#), [Li_eval\(\)](#), [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.27 `_num_30_p`

```
const numeric * GiNaC::_num_30_p
```

Referenced by [GiNaC::library_init::library_init\(\)](#).

5.1.4.28 `_ex_30`

```
const ex GiNaC::_ex_30 = ex(*_num_30_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.29 `_num_25_p`

```
const numeric * GiNaC::_num_25_p
```

Referenced by [GiNaC::library_init::library_init\(\)](#).

5.1.4.30 `_ex_25`

```
const ex GiNaC::_ex_25 = ex(*_num_25_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.31 `_num_24_p`

```
const numeric * GiNaC::_num_24_p
```

Referenced by [GiNaC::library_init::library_init\(\)](#).

5.1.4.32 `_ex_24`

```
const ex GiNaC::_ex_24 = ex(*_num_24_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.33 `_num_20_p`

```
const numeric * GiNaC::_num_20_p
```

Referenced by [GiNaC::library_init::library_init\(\)](#).

5.1.4.34 `_ex_20`

```
const ex GiNaC::_ex_20 = ex(*_num_20_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.35 `_num_18_p`

```
const numeric * GiNaC::_num_18_p
```

Referenced by [GiNaC::library_init::library_init\(\)](#).

5.1.4.36 `_ex_18`

```
const ex GiNaC::_ex_18 = ex(*_num_18_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.37 `_num_15_p`

```
const numeric * GiNaC::_num_15_p
```

Referenced by [GiNaC::library_init::library_init\(\)](#).

5.1.4.38 `_ex_15`

```
const ex GiNaC::_ex_15 = ex(*_num_15_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.39 `_num_12_p`

```
const numeric * GiNaC::_num_12_p
```

Referenced by [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), and [GiNaC::library_init::library_init\(\)](#).

5.1.4.40 `_ex_12`

```
const ex GiNaC::_ex_12 = ex(*_num_12_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.41 `_num_11_p`

```
const numeric * GiNaC::_num_11_p
```

Referenced by [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), and [GiNaC::library_init::library_init\(\)](#).

5.1.4.42 `_ex_11`

```
const ex GiNaC::_ex_11 = ex(*_num_11_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.43 `_num_10_p`

```
const numeric * GiNaC::_num_10_p
```

Referenced by [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), and [GiNaC::library_init::library_init\(\)](#).

5.1.4.44 `_ex_10`

```
const ex GiNaC::_ex_10 = ex(*_num_10_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.45 `_num_9_p`

```
const numeric * GiNaC::_num_9_p
```

Referenced by [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), and [GiNaC::library_init::library_init\(\)](#).

5.1.4.46 `_ex_9`

```
const ex GiNaC::_ex_9 = ex(*_num_9_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.47 `_num_8_p`

```
const numeric * GiNaC::_num_8_p
```

Referenced by [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), and [GiNaC::library_init::library_init\(\)](#).

5.1.4.48 `_ex_8`

```
const ex GiNaC::_ex_8 = ex(*_num_8_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.49 `_num_7_p`

```
const numeric * GiNaC::_num_7_p
```

Referenced by [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), and [GiNaC::library_init::library_init\(\)](#).

5.1.4.50 `_ex_7`

```
const ex GiNaC::_ex_7 = ex(*_num_7_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.51 `_num_6_p`

```
const numeric * GiNaC::_num_6_p
```

Referenced by [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), and [GiNaC::library_init::library_init\(\)](#).

5.1.4.52 `_ex_6`

```
const ex GiNaC::_ex_6 = ex(*_num_6_p)
```

Referenced by [GiNaC::su3d::eval_indexed\(\)](#), [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.53 `_num_5_p`

```
const numeric * GiNaC::_num_5_p
```

Referenced by [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), and [GiNaC::library_init::library_init\(\)](#).

5.1.4.54 `_ex_5`

```
const ex GiNaC::_ex_5 = ex(*_num_5_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.55 `_num_4_p`

```
const numeric * GiNaC::_num_4_p
```

Referenced by [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), and [GiNaC::library_init::library_init\(\)](#).

5.1.4.56 `_ex_4`

```
const ex GiNaC::_ex_4 = ex(*_num_4_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.57 `_num_3_p`

```
const numeric * GiNaC::_num_3_p
```

Referenced by [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), and [GiNaC::library_init::library_init\(\)](#).

5.1.4.58 `_ex_3`

```
const ex GiNaC::_ex_3 = ex(*_num_3_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.59 `_num_2_p`

```
const numeric * GiNaC::_num_2_p
```

Referenced by [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), [GiNaC::library_init::library_init\(\)](#), and [tgamma_eval\(\)](#).

5.1.4.60 `_ex_2`

```
const ex GiNaC::_ex_2 = ex(*_num_2_p)
```

Referenced by [GiNaC::spinmetric::contract_with\(\)](#), [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.61 `_num_1_p`

```
const numeric * GiNaC::_num_1_p
```

Referenced by [acos_conjugate\(\)](#), [asin_conjugate\(\)](#), [asinh_conjugate\(\)](#), [atan_conjugate\(\)](#), [atanh_conjugate\(\)](#), [beta_eval\(\)](#), [binomial\(\)](#), [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), [GiNaC::add::do_print_csrc\(\)](#), [doublefactorial\(\)](#), [GiNaC::mul::eval\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::library_init::library_init\(\)](#), [operator-\(\)](#), [operator--\(\)](#), [operator-\(\)](#), [GiNaC::add::print_add\(\)](#), [GiNaC::mul::print_overall_coeff\(\)](#), [psi1_eval\(\)](#), and [psi2_eval\(\)](#).

5.1.4.62 `_ex_1`

```
const ex GiNaC::_ex_1 = ex(*_num_1_p)
```

Referenced by [acos_eval\(\)](#), [acosh_deriv\(\)](#), [acosh_eval\(\)](#), [asin_eval\(\)](#), [atan2_deriv\(\)](#), [atan_deriv\(\)](#), [atan_eval\(\)](#), [atan_series\(\)](#), [atanh_deriv\(\)](#), [atanh_eval\(\)](#), [atanh_series\(\)](#), [cos_eval\(\)](#), [GiNaC::power::derivative\(\)](#), [GiNaC::add::do_print_csrc\(\)](#), [GiNaC::mul::do_print_csrc\(\)](#), [GiNaC::power::do_print_csrc\(\)](#), [GiNaC::power::do_print_csrc_cl_N\(\)](#), [EllipticE_eval\(\)](#), [EllipticE_series\(\)](#), [EllipticK_series\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::minkmetric::eval_indexed\(\)](#), [GiNaC::spinmetric::eval_indexed\(\)](#), [exminus\(\)](#), [exp_eval\(\)](#), [exp_power\(\)](#), [GiNaC::power::expand\(\)](#), [frac_cancel\(\)](#), [H_deriv\(\)](#), [H_eval\(\)](#), [lgamma_eval\(\)](#), [Li2_eval\(\)](#), [Li_eval\(\)](#), [GiNaC::library_init::library_init\(\)](#), [log_deriv\(\)](#), [log_expand\(\)](#), [log_series\(\)](#), [operator-\(\)](#), [operator--\(\)](#), [operator/\(\)](#), [operator/\(\)](#), [operator/\(\)](#), [psi1_series\(\)](#), [psi2_eval\(\)](#), [psi2_series\(\)](#), [replace_with_symbol\(\)](#), [sin_eval\(\)](#), [tan_eval\(\)](#), [tanh_eval\(\)](#), [GiNaC::power::to_polynomial\(\)](#), [GiNaC::ex::unit\(\)](#), [GiNaC::ex::unitcontprim\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.63 `_num_1_2_p`

```
const numeric * GiNaC::_num_1_2_p
```

Referenced by [GiNaC::library_init::library_init\(\)](#).

5.1.4.64 `_ex_1_2`

```
const ex GiNaC::_ex_1_2 = ex(*_num_1_2_p)
```

Referenced by [acos_deriv\(\)](#), [acos_eval\(\)](#), [acosh_deriv\(\)](#), [asin_deriv\(\)](#), [asin_eval\(\)](#), [asinh_deriv\(\)](#), [atan2_eval\(\)](#), [atan_series\(\)](#), [atanh_series\(\)](#), [cos_eval\(\)](#), [cosh_eval\(\)](#), [GiNaC::su3f::eval_indexed\(\)](#), [GiNaC::su3d::eval_indexed\(\)](#), [Li2_eval\(\)](#), [GiNaC::library_init::library_init\(\)](#), [log_eval\(\)](#), [sin_eval\(\)](#), [sinh_eval\(\)](#), [tan_eval\(\)](#), [tanh_eval\(\)](#), [zeta1_eval\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.65 `_num_1_3_p`

```
const numeric * GiNaC::_num_1_3_p
```

Referenced by [GiNaC::library_init::library_init\(\)](#).

5.1.4.66 `_ex_1_3`

```
const ex GiNaC::_ex_1_3 = ex(*_num_1_3_p)
```

Referenced by [cos_eval\(\)](#), [GiNaC::su3d::eval_indexed\(\)](#), [GiNaC::library_init::library_init\(\)](#), [sin_eval\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.67 `_num_1_4_p`

```
const numeric * GiNaC::_num_1_4_p
```

Referenced by [GiNaC::library_init::library_init\(\)](#).

5.1.4.68 `_ex_1_4`

```
const ex GiNaC::_ex_1_4 = ex(*_num_1_4_p)
```

Referenced by [atan2_eval\(\)](#), [atan_eval\(\)](#), [cos_eval\(\)](#), [GiNaC::library_init::library_init\(\)](#), [sin_eval\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.69 `_num0_p`

```
const numeric * GiNaC::_num0_p
```

Referenced by [GiNaC::numeric::add_dyn\(\)](#), [atan\(\)](#), [binomial\(\)](#), [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), [GiNaC::ex::construct_from_uint\(\)](#), [GiNaC::ex::construct_from_ulong\(\)](#), [cos_eval\(\)](#), [divide_in_z\(\)](#), [GiNaC::power::eval\(\)](#), [exp_eval\(\)](#), [GiNaC::mul::expand\(\)](#), [fibonacci\(\)](#), [GiNaC::numeric::has\(\)](#), [GiNaC::add::integer_content\(\)](#), [iquo\(\)](#), [iquo\(\)](#), [irem\(\)](#), [irem\(\)](#), [isqrt\(\)](#), [Li2\(\)](#), [GiNaC::library_init::library_init\(\)](#), [mod\(\)](#), [GiNaC::relational::operator safe_bool\(\)](#), [GiNaC::numeric::power\(\)](#), [GiNaC::numeric::power_dyn\(\)](#), [sin_eval\(\)](#), [smod\(\)](#), [GiNaC::numeric::sub_dyn\(\)](#), and [tan_eval\(\)](#).

5.1.4.70 `_ex0`

```
const ex GiNaC::_ex0 = ex(*_num0_p)
```

Referenced by [acos_eval\(\)](#), [acosh_eval\(\)](#), [GiNaC::add::add\(\)](#), [GiNaC::add::add\(\)](#), [GiNaC::add::add\(\)](#), [GiNaC::add::add\(\)](#), [GiNaC::pseries::add_series\(\)](#), [asinh_eval\(\)](#), [atan2_eval\(\)](#), [atan_eval\(\)](#), [atan_series\(\)](#), [atanh_eval\(\)](#), [atanh_series\(\)](#), [beta_eval\(\)](#), [binomial_sym\(\)](#), [GiNaC::relational::canonical\(\)](#), [GiNaC::basic::coeff\(\)](#), [GiNaC::add::coeff\(\)](#), [GiNaC::mul::coeff\(\)](#), [GiNaC::ncmul::coeff\(\)](#), [GiNaC::numeric::coeff\(\)](#), [GiNaC::power::coeff\(\)](#), [GiNaC::pseries::coeff\(\)](#), [GiNaC::basic::collect\(\)](#), [color_trace\(\)](#), [GiNaC::ex::content\(\)](#), [cos_eval\(\)](#), [csgn_series\(\)](#), [GiNaC::expairseq::default_overall_coeff\(\)](#), [GiNaC::basic::derivative\(\)](#), [GiNaC::constant::derivative\(\)](#), [GiNaC::idx::derivative\(\)](#), [GiNaC::indexed::derivative\(\)](#), [GiNaC::symbol::derivative\(\)](#), [GiNaC::matrix::determinant\(\)](#), [GiNaC::matrix::determinant_minor\(\)](#), [divide\(\)](#), [divide_in_z\(\)](#), [GiNaC::matrix::division_free_elimination\(\)](#), [EllipticE_eval\(\)](#), [EllipticE_series\(\)](#), [EllipticK_eval\(\)](#), [EllipticK_series\(\)](#), [eta_eval\(\)](#), [eta_evalf\(\)](#), [eta_series\(\)](#), [GiNaC::function::eval\(\)](#), [GiNaC::indexed::eval\(\)](#), [GiNaC::integral::eval\(\)](#), [GiNaC::mul::eval\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::su3f::eval_indexed\(\)](#), [GiNaC::su3d::eval_indexed\(\)](#), [GiNaC::tensdelta::eval_indexed\(\)](#), [GiNaC::minkmetric::eval_indexed\(\)](#), [GiNaC::spinmetric::eval_indexed\(\)](#), [GiNaC::tensepsilon::eval_indexed\(\)](#), [GiNaC::indexed::expand\(\)](#), [find_common_factor\(\)](#), [GiNaC::matrix::fraction_free_elimination\(\)](#), [G2_eval\(\)](#), [G2_evalf\(\)](#), [G3_eval\(\)](#), [G3_evalf\(\)](#), [GiNaC::matrix::gauss_elimination\(\)](#), [gcd\(\)](#), [H_deriv\(\)](#), [H_eval\(\)](#), [GiNaC::ex::is_zero\(\)](#), [Li2_eval\(\)](#), [Li2_series\(\)](#), [Li_deriv\(\)](#), [Li_eval\(\)](#), [Li_evalf\(\)](#), [GiNaC::library_init::library_init\(\)](#), [log_eval\(\)](#), [log_series\(\)](#), [GiNaC::matrix::markowitz_elimination\(\)](#), [GiNaC::matrix::matrix\(\)](#), [GiNaC::pseries::mul_series\(\)](#), [Order_eval\(\)](#), [GiNaC::pseries::power_const\(\)](#), [prem\(\)](#), [GiNaC::ex::primpart\(\)](#), [rem\(\)](#), [S_deriv\(\)](#), [S_eval\(\)](#), [GiNaC::ex::series\(\)](#), [GiNaC::basic::series\(\)](#), [GiNaC::symbol::series\(\)](#), [sin_eval\(\)](#), [sinh_eval\(\)](#), [sprem\(\)](#), [sqrtfree\(\)](#), [sr_gcd\(\)](#), [step_series\(\)](#), [tan_eval\(\)](#), [tanh_eval\(\)](#), [GiNaC::ex::unitcontprim\(\)](#), [zeta1_deriv\(\)](#), [zeta1_eval\(\)](#), [zeta2_deriv\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.71 `_num1_4_p`

```
const numeric * GiNaC::_num1_4_p
```

Referenced by [GiNaC::library_init::library_init\(\)](#).

5.1.4.72 `_ex1_4`

```
const ex GiNaC::_ex1_4 = ex(*_num1_4_p)
```

Referenced by [atan2_eval\(\)](#), [atan_eval\(\)](#), [cos_eval\(\)](#), [GiNaC::library_init::library_init\(\)](#), [sin_eval\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.73 `_num1_3_p`

```
const numeric * GiNaC::_num1_3_p
```

Referenced by [GiNaC::library_init::library_init\(\)](#).

5.1.4.74 `_ex1_3`

```
const ex GiNaC::_ex1_3 = ex(*_num1_3_p)
```

Referenced by [acos_eval\(\)](#), [cos_eval\(\)](#), [GiNaC::su3d::eval_indexed\(\)](#), [GiNaC::library_init::library_init\(\)](#), [sin_eval\(\)](#), [tan_eval\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.75 `_num1_2_p`

```
const numeric * GiNaC::_num1_2_p
```

Referenced by [GiNaC::library_init::library_init\(\)](#), [psi2_eval\(\)](#), and [tgamma_eval\(\)](#).

5.1.4.76 `_ex1_2`

```
const ex GiNaC::_ex1_2 = ex(*_num1_2_p)
```

Referenced by [acos_eval\(\)](#), [asin_eval\(\)](#), [atan2_eval\(\)](#), [atan_series\(\)](#), [atanh_series\(\)](#), [cos_eval\(\)](#), [GiNaC::power::do_print_dflt\(\)](#), [GiNaC::power::do_print_latex\(\)](#), [GiNaC::su3f::eval_indexed\(\)](#), [GiNaC::su3d::eval_indexed\(\)](#), [GiNaC::clifford::get_metric\(\)](#), [Li2_eval\(\)](#), [GiNaC::library_init::library_init\(\)](#), [log_eval\(\)](#), [psi1_eval\(\)](#), [psi2_eval\(\)](#), [sin_eval\(\)](#), [sqrt\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.77 `_num1_p`

```
const numeric * GiNaC::_num1_p
```

Referenced by [acos_conjugate\(\)](#), [acosh_conjugate\(\)](#), [asin_conjugate\(\)](#), [asinh_conjugate\(\)](#), [atan\(\)](#), [atan_conjugate\(\)](#), [atanh_conjugate\(\)](#), [bernoulli\(\)](#), [binomial\(\)](#), [GiNaC::add::combine_pair_with_coeff_to_pair\(\)](#), [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), [GiNaC::ex::construct_from_uint\(\)](#), [GiNaC::ex::construct_from_ulong\(\)](#), [GiNaC::numeric::denom\(\)](#), [GiNaC::numeric::div_dyn\(\)](#), [divide_in_z\(\)](#), [GiNaC::add::do_print_csrc\(\)](#), [doublefactorial\(\)](#), [GiNaC::mul::eval\(\)](#), [GiNaC::power::eval\(\)](#), [exp_eval\(\)](#), [GiNaC::power::expand_add\(\)](#), [frac_cancel\(\)](#), [gcd\(\)](#), [GiNaC::mul::info\(\)](#), [GiNaC::basic::integer_content\(\)](#), [GiNaC::add::integer_content\(\)](#), [lcm_of_coefficients_denominators\(\)](#), [lcmcoeff\(\)](#), [Li2_conjugate\(\)](#), [GiNaC::library_init::library_init\(\)](#), [GiNaC::basic::max_coefficient\(\)](#), [GiNaC::numeric::mul_dyn\(\)](#), [multiply_lcm\(\)](#), [operator++\(\)](#), [operator++\(\)](#), [GiNaC::matrix::pow\(\)](#), [GiNaC::numeric::power\(\)](#), [GiNaC::numeric::power_dyn\(\)](#), [GiNaC::add::print_add\(\)](#), [GiNaC::mul::print_overall_coeff\(\)](#), [psi2_eval\(\)](#), [GiNaC::add::recombine_pair_to_ex\(\)](#), [tgamma_eval\(\)](#), and [zeta1_eval\(\)](#).

5.1.4.78 `_ex1`

```
const ex GiNaC::_ex1 = ex(*_num1_p)
```

Referenced by `acos_eval()`, `acosh_deriv()`, `acosh_eval()`, `GiNaC::pseries::add_series()`, `asin_eval()`, `asinh_deriv()`, `atan_deriv()`, `atan_eval()`, `atan_series()`, `atanh_deriv()`, `atanh_eval()`, `atanh_series()`, `beta_eval()`, `binomial_sym()`, `GiNaC::basic::coeff()`, `GiNaC::ncmul::coeff()`, `GiNaC::power::coeff()`, `GiNaC::basic::collect()`, `color_trace()`, `GiNaC::add::combine_ex_with_coeff_to_pair()`, `GiNaC::mul::combine_ex_with_coeff_to_pair()`, `GiNaC::add::combine_pair_with_coeff_to_pair()`, `GiNaC::mul::combine_pair_with_coeff_to_pair()`, `GiNaC::su3t::contract_with()`, `GiNaC::su3f::contract_with()`, `GiNaC::su3d::contract_with()`, `GiNaC::matrix::contract_with()`, `GiNaC::spinmetric::contract_with()`, `GiNaC::tensepsilon::contract_with()`, `cos_eval()`, `cosh_eval()`, `GiNaC::mul::default_overall_coeff()`, `GiNaC::mul::derivative()`, `GiNaC::power::derivative()`, `GiNaC::symbol::derivative()`, `GiNaC::matrix::determinant_minor()`, `divide()`, `divide_in_z()`, `GiNaC::add::do_print_csrc()`, `GiNaC::mul::do_print_csrc()`, `EllipticE_eval()`, `EllipticE_series()`, `EllipticK_series()`, `GiNaC::mul::eval()`, `GiNaC::ncmul::eval()`, `GiNaC::power::eval()`, `GiNaC::tensdelta::eval_indexed()`, `GiNaC::minkmetric::eval_indexed()`, `GiNaC::spinmetric::eval_indexed()`, `GiNaC::mul::evalm()`, `exp_eval()`, `GiNaC::expairseq::expair_needs_further_processing()`, `GiNaC::mul::expair_needs_further_processing()`, `GiNaC::mul::expand()`, `GiNaC::ncmul::expand()`, `GiNaC::power::expand()`, `GiNaC::power::expand_add()`, `GiNaC::power::expand_add_mul()`, `find_common_factor()`, `frac_cancel()`, `GiNaC::matrix::fraction_free_elimination()`, `G2_eval()`, `G2_evalf()`, `G3_eval()`, `G3_evalf()`, `gcd()`, `gcd_pf_pow()`, `gcd_pf_pow_pow()`, `GINAC_IMPLEMENT_REGISTERED_CLASS`, `H_deriv()`, `H_eval()`, `GiNaC::power::imag_part()`, `GiNaC::matrix::inverse()`, `lgamma_series()`, `Li2_deriv()`, `Li2_eval()`, `Li2_series()`, `Li_eval()`, `Li_evalf()`, `Li_series()`, `GiNaC::library_init::library_init()`, `log_eval()`, `log_expand()`, `log_series()`, `GiNaC::expairseq::make_flat()`, `GiNaC::expairseq::make_flat()`, `GiNaC::mul::mul()`, `GiNaC::mul::mul()`, `GiNaC::mul::mul()`, `GiNaC::mul::mul()`, `GiNaC::pseries::mul_series()`, `GiNaC::basic::normal()`, `GiNaC::power::normal()`, `GiNaC::pseries::normal()`, `GiNaC::symbol::normal()`, `operator++()`, `operator++()`, `Order_eval()`, `Order_series()`, `GiNaC::matrix::pow()`, `GiNaC::pseries::power_const()`, `prem()`, `GiNaC::ex::primpart()`, `GiNaC::pseries::print_series()`, `psi1_deriv()`, `psi1_series()`, `psi2_deriv()`, `psi2_eval()`, `psi2_series()`, `quo()`, `GiNaC::power::real_part()`, `GiNaC::mul::recombine_pair_to_ex()`, `GiNaC::tensor::replace_contr_index()`, `S_series()`, `GiNaC::basic::series()`, `GiNaC::add::series()`, `GiNaC::integral::series()`, `GiNaC::pseries::series()`, `GiNaC::mul::series()`, `GiNaC::power::series()`, `GiNaC::symbol::series()`, `sin_eval()`, `sinh_eval()`, `GiNaC::expairseq::split_ex_to_pair()`, `GiNaC::add::split_ex_to_pair()`, `GiNaC::mul::split_ex_to_pair()`, `sprem()`, `sqrfree_parfrac()`, `sqrfree_yun()`, `sr_gcd()`, `tan_deriv()`, `tan_eval()`, `tanh_deriv()`, `tanh_eval()`, `tgamma_series()`, `GiNaC::expairseq::to_polynomial()`, `GiNaC::expairseq::to_rational()`, `GiNaC::ex::unit()`, `unit_matrix()`, `GiNaC::ex::unitcontprim()`, `zeta1_deriv()`, `zeta2_deriv()`, and `GiNaC::library_init::~~library_init()`.

5.1.4.79 `_num2_p`

```
const numeric * GiNaC::_num2_p
```

Referenced by `GiNaC::ex::construct_from_int()`, `GiNaC::ex::construct_from_long()`, `GiNaC::ex::construct_from_uint()`, `GiNaC::ex::construct_from_ulong()`, `exp_eval()`, `GiNaC::power::expand_add_2()`, `Li2_series()`, `GiNaC::library_init::library_init()`, `GiNaC::matrix::pow()`, `psi1_eval()`, `psi2_eval()`, `tgamma_eval()`, and `zeta1_eval()`.

5.1.4.80 `_ex2`

```
const ex GiNaC::_ex2 = ex(*_num2_p)
```

Referenced by `acos_deriv()`, `asin_deriv()`, `asinh_deriv()`, `atan2_deriv()`, `atan_deriv()`, `atanh_deriv()`, `GiNaC::spinmetric::contract_with()`, `cos_eval()`, `cosh_eval()`, `csgn_power()`, `epsilon_tensor()`, `exp_eval()`, `GiNaC::power::expand_add_2()`, `Li2_eval()`, `Li2_series()`, `Li_eval()`, `GiNaC::library_init::library_init()`, `product_to_exvector()`, `psi1_eval()`, `sin_eval()`, `sinh_eval()`, `tan_deriv()`, `tan_eval()`, `tanh_deriv()`, `tanh_eval()`, and `GiNaC::library_init::~~library_init()`.

5.1.4.81 `_num3_p`

```
const numeric * GiNaC::_num3_p
```

Referenced by `GiNaC::ex::construct_from_int()`, `GiNaC::ex::construct_from_long()`, `GiNaC::ex::construct_from_uint()`, `GiNaC::ex::construct_from_ulong()`, `exp_eval()`, and `GiNaC::library_init::library_init()`.

5.1.4.82 `_ex3`

```
const ex GiNaC::_ex3 = ex(*\_num3\_p)
```

Referenced by [color_trace\(\)](#), [cos_eval\(\)](#), [epsilon_tensor\(\)](#), [GiNaC::su3f::eval_indexed\(\)](#), [GiNaC::su3d::eval_indexed\(\)](#), [GiNaC::library_init::library_init\(\)](#), [sin_eval\(\)](#), [tan_eval\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.83 `_num4_p`

```
const numeric * GiNaC::_num4_p
```

Referenced by [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), [GiNaC::ex::construct_from_uint\(\)](#), [GiNaC::ex::construct_from_ulong\(\)](#), [exp_eval\(\)](#), and [GiNaC::library_init::library_init\(\)](#).

5.1.4.84 `_ex4`

```
const ex GiNaC::_ex4 = ex(*\_num4\_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), [lorentz_eps\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.85 `_num5_p`

```
const numeric * GiNaC::_num5_p
```

Referenced by [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), [GiNaC::ex::construct_from_uint\(\)](#), [GiNaC::ex::construct_from_ulong\(\)](#), [cos_eval\(\)](#), [GiNaC::library_init::library_init\(\)](#), [sin_eval\(\)](#), and [tan_eval\(\)](#).

5.1.4.86 `_ex5`

```
const ex GiNaC::_ex5 = ex(*\_num5\_p)
```

Referenced by [cos_eval\(\)](#), [GiNaC::library_init::library_init\(\)](#), [sin_eval\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.87 `_num6_p`

```
const numeric * GiNaC::_num6_p
```

Referenced by [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), [GiNaC::ex::construct_from_uint\(\)](#), [GiNaC::ex::construct_from_ulong\(\)](#), [GiNaC::library_init::library_init\(\)](#), and [sin_eval\(\)](#).

5.1.4.88 `_ex6`

```
const ex GiNaC::_ex6 = ex(*\_num6\_p)
```

Referenced by [cos_eval\(\)](#), [Li2_eval\(\)](#), [GiNaC::library_init::library_init\(\)](#), [sin_eval\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.89 `_num7_p`

```
const numeric * GiNaC::_num7_p
```

Referenced by [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), [GiNaC::ex::construct_from_uint\(\)](#), [GiNaC::ex::construct_from_ulong\(\)](#), and [GiNaC::library_init::library_init\(\)](#).

5.1.4.90 `_ex7`

```
const ex GiNaC::_ex7 = ex(*_num7_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.91 `_num8_p`

```
const numeric * GiNaC::_num8_p
```

Referenced by [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), [GiNaC::ex::construct_from_uint\(\)](#), [GiNaC::ex::construct_from_ulong\(\)](#), and [GiNaC::library_init::library_init\(\)](#).

5.1.4.92 `_ex8`

```
const ex GiNaC::_ex8 = ex(*_num8_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.93 `_num9_p`

```
const numeric * GiNaC::_num9_p
```

Referenced by [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), [GiNaC::ex::construct_from_uint\(\)](#), [GiNaC::ex::construct_from_ulong\(\)](#), and [GiNaC::library_init::library_init\(\)](#).

5.1.4.94 `_ex9`

```
const ex GiNaC::_ex9 = ex(*_num9_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.95 `_num10_p`

```
const numeric * GiNaC::_num10_p
```

Referenced by [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), [GiNaC::ex::construct_from_uint\(\)](#), [GiNaC::ex::construct_from_ulong\(\)](#), [cos_eval\(\)](#), [GiNaC::library_init::library_init\(\)](#), [sin_eval\(\)](#), and [tan_eval\(\)](#).

5.1.4.96 `_ex10`

```
const ex GiNaC::_ex10 = ex(*_num10_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.97 `_num11_p`

```
const numeric * GiNaC::_num11_p
```

Referenced by [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), [GiNaC::ex::construct_from_uint\(\)](#), [GiNaC::ex::construct_from_ulong\(\)](#), and [GiNaC::library_init::library_init\(\)](#).

5.1.4.98 `_ex11`

```
const ex GiNaC::_ex11 = ex(*_num11_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.99 `_num12_p`

```
const numeric * GiNaC::_num12_p
```

Referenced by [GiNaC::ex::construct_from_int\(\)](#), [GiNaC::ex::construct_from_long\(\)](#), [GiNaC::ex::construct_from_uint\(\)](#), [GiNaC::ex::construct_from_ulong\(\)](#), [cos_eval\(\)](#), and [GiNaC::library_init::library_init\(\)](#).

5.1.4.100 `_ex12`

```
const ex GiNaC::_ex12 = ex(*_num12_p)
```

Referenced by [Li2_eval\(\)](#), [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.101 `_num15_p`

```
const numeric * GiNaC::_num15_p
```

Referenced by [cos_eval\(\)](#), [GiNaC::library_init::library_init\(\)](#), [sin_eval\(\)](#), and [tan_eval\(\)](#).

5.1.4.102 `_ex15`

```
const ex GiNaC::_ex15 = ex(*_num15_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.103 `_num18_p`

```
const numeric * GiNaC::_num18_p
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [sin_eval\(\)](#).

5.1.4.104 `_ex18`

```
const ex GiNaC::_ex18 = ex(*_num18_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.105 `_num20_p`

```
const numeric * GiNaC::_num20_p
```

Referenced by [cos_eval\(\)](#), [GiNaC::library_init::library_init\(\)](#), [sin_eval\(\)](#), and [tan_eval\(\)](#).

5.1.4.106 `_ex20`

```
const ex GiNaC::_ex20 = ex(*_num20_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.107 `_num24_p`

```
const numeric * GiNaC::_num24_p
```

Referenced by [cos_eval\(\)](#), and [GiNaC::library_init::library_init\(\)](#).

5.1.4.108 `_ex24`

```
const ex GiNaC::_ex24 = ex(*_num24_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.109 `_num25_p`

```
const numeric * GiNaC::_num25_p
```

Referenced by [cos_eval\(\)](#), [GiNaC::library_init::library_init\(\)](#), [sin_eval\(\)](#), and [tan_eval\(\)](#).

5.1.4.110 `_ex25`

```
const ex GiNaC::_ex25 = ex(*_num25_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.111 `_num30_p`

```
const numeric * GiNaC::_num30_p
```

Referenced by [cos_eval\(\)](#), [GiNaC::library_init::library_init\(\)](#), [sin_eval\(\)](#), and [tan_eval\(\)](#).

5.1.4.112 `_ex30`

```
const ex GiNaC::_ex30 = ex(*_num30_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.113 `_num48_p`

```
const numeric * GiNaC::_num48_p
```

Referenced by [GiNaC::library_init::library_init\(\)](#).

5.1.4.114 `_ex48`

```
const ex GiNaC::_ex48 = ex(*_num48_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.115 `_num60_p`

```
const numeric * GiNaC::_num60_p
```

Referenced by [cos_eval\(\)](#), [GiNaC::library_init::library_init\(\)](#), [sin_eval\(\)](#), and [tan_eval\(\)](#).

5.1.4.116 `_ex60`

```
const ex GiNaC::_ex60 = ex(*_num60_p)
```

Referenced by [cos_eval\(\)](#), [GiNaC::library_init::library_init\(\)](#), [sin_eval\(\)](#), [tan_eval\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.1.4.117 `_num120_p`

```
const numeric * GiNaC::_num120_p
```

Referenced by [cos_eval\(\)](#), [GiNaC::library_init::library_init\(\)](#), and [sin_eval\(\)](#).

5.1.4.118 `_ex120`

```
const ex GiNaC::_ex120 = ex(*_num120_p)
```

Referenced by [GiNaC::library_init::library_init\(\)](#), and [GiNaC::library_init::~~library_init\(\)](#).

5.2 GiNaC::Gt_detail Namespace Reference

Classes

- class [pathintegral_term](#)
- class [pathintegral_term_list](#)
- class [TransformExpressionWithCache](#)

Functions

- `std::vector< std::vector< int > > integer\_partition (const int n, const int m)`
- `template<typename Kernel >
ex deconcatenate_path (const std::vector< Kernel > &args, const std::vector< ex > &endpoints, const std::function< ex(std::vector< Kernel > &new_args, const ex &start, const ex &end)> &construct)`
- `template ex deconcatenate\_path< ex > (const std::vector< ex > &args, const std::vector< ex > &endpoints, const std::function< ex(std::vector< ex > &new_args, const ex &start, const ex &end)> &construct)`
- `template ex deconcatenate\_path< Gt::kernel > (const std::vector< Gt::kernel > &args, const std::vector< ex > &endpoints, const std::function< ex(std::vector< Gt::kernel > &new_args, const ex &start, const ex &end)> &construct)`
- `std::ostream & operator<< (std::ostream &os, const pathintegral\_term &term)`
- `bool operator< (const pathintegral\_term &a, const pathintegral\_term &b)`

5.2.1 Function Documentation

5.2.1.1 [integer_partition\(\)](#)

```
std::vector< std::vector< int > > GiNaC::Gt_detail::integer_partition (
    const int n,
    const int m )
```

References [k](#), [m](#), [n](#), and [x](#).

Referenced by [deconcatenate_path\(\)](#).

5.2.1.2 [deconcatenate_path\(\)](#)

```
template<typename Kernel >
ex GiNaC::Gt_detail::deconcatenate_path (
    const std::vector< Kernel > & args,
    const std::vector< ex > & endpoints,
    const std::function< ex(std::vector< Kernel > &new_args, const ex &start, const
ex &end)> & construct )
```

References [factors](#), and [integer_partition\(\)](#).

5.2.1.3 [deconcatenate_path< ex >\(\)](#)

```
template ex GiNaC::Gt_detail::deconcatenate_path< ex > (
    const std::vector< ex > & args,
    const std::vector< ex > & endpoints,
    const std::function< ex(std::vector< ex > &new_args, const ex &start, const ex
&end)> & construct )
```

Referenced by [GiNaC::Gt_detail::pathintegral_term::G_path\(\)](#).

5.2.1.4 deconcatenate_path< Gt::kernel >()

```
template< ex GiNaC::Gt_detail::deconcatenate_path< Gt::kernel > (
    const std::vector< Gt::kernel > & args,
    const std::vector< ex > & endpoints,
    const std::function< ex(std::vector< Gt::kernel > &new_args, const ex &start,
const ex &end)> & construct )
```

5.2.1.5 operator<<()

```
std::ostream & GiNaC::Gt_detail::operator<< (
    std::ostream & os,
    const pathintegral_term & term )
```

5.2.1.6 operator<()

```
bool GiNaC::Gt_detail::operator< (
    const pathintegral_term & a,
    const pathintegral_term & b )
```

5.3 GiNaC::internal Namespace Reference

Classes

- struct [_iter_rep](#)

5.4 std Namespace Reference

Classes

- struct [equal_to< GiNaC::ex >](#)
Specialization of std::equal_to() for ex objects.
- struct [hash< GiNaC::ex >](#)
Specialization of std::hash() for ex objects.
- struct [less< GiNaC::ptr< T > >](#)
Specialization of std::less for ptr<T> to enable ordering of ptr<T> objects (e.g.

Functions

- template<> void [swap](#) (GiNaC::ex &a, GiNaC::ex &b)
Specialization of [std::swap\(\)](#) for ex objects.

5.4.1 Function Documentation

5.4.1.1 swap()

```
template<>
void std::swap (
    GiNaC::ex & a,
    GiNaC::ex & b ) [inline]
```

Specialization of [std::swap\(\)](#) for ex objects.

References [GiNaC::ex::swap\(\)](#).

Referenced by [GiNaC::su3f::eval_indexed\(\)](#), [GiNaC::su3d::eval_indexed\(\)](#), [GiNaC::matrix::markowitz_elimination\(\)](#), and [GiNaC::permutation_sign\(\)](#).

Chapter 6

Class Documentation

6.1 GiNaC::internal::_iter_rep Struct Reference

```
#include <ex.h>
```

Public Member Functions

- [_iter_rep](#) (const [ex](#) &[e](#)_, size_t [i](#)_, size_t [i_end](#)_)
- bool [operator==](#) (const [_iter_rep](#) &[other](#)) const noexcept
- bool [operator!=](#) (const [_iter_rep](#) &[other](#)) const noexcept

Public Attributes

- [ex](#) [e](#)
- size_t [i](#)
- size_t [i_end](#)

6.1.1 Constructor & Destructor Documentation

6.1.1.1 _iter_rep()

```
GiNaC::internal::_iter_rep::_iter_rep (  
    const ex & e_,  
    size_t i_,  
    size_t i\_end_ ) [inline]
```

6.1.2 Member Function Documentation

6.1.2.1 operator==()

```
bool GiNaC::internal::_iter_rep::operator== (  
    const \_iter\_rep & other ) const [inline], [noexcept]
```

References [GiNaC::are_ex_trivially_equal\(\)](#), [e](#), and [i](#).

6.1.2.2 operator"!=()

```
bool GiNaC::internal::_iter_rep::operator!= (
    const _iter_rep & other ) const [inline], [noexcept]
```

6.1.3 Member Data Documentation

6.1.3.1 e

```
ex GiNaC::internal::_iter_rep::e
```

Referenced by [GiNaC::const_postorder_iterator::descend\(\)](#), [GiNaC::const_preorder_iterator::increment\(\)](#), and [operator==\(\)](#).

6.1.3.2 i

```
size_t GiNaC::internal::_iter_rep::i
```

Referenced by [GiNaC::const_postorder_iterator::descend\(\)](#), [GiNaC::const_preorder_iterator::increment\(\)](#), and [operator==\(\)](#).

6.1.3.3 i_end

```
size_t GiNaC::internal::_iter_rep::i_end
```

Referenced by [GiNaC::const_preorder_iterator::increment\(\)](#).

The documentation for this struct was generated from the following file:

- [ex.h](#)

6.2 GiNaC::_numeric_digits Class Reference

This class is used to instantiate a global singleton object Digits which behaves just like Maple's Digits.

```
#include <numeric.h>
```

Public Member Functions

- [_numeric_digits](#) ()
_numeric_digits default ctor, checking for singleton invariance.
- [_numeric_digits](#) & [operator=](#) (long prec)
Assign a native long to global Digits object.
- [operator long](#) ()
Convert global Digits object to native type long.
- void [print](#) (std::ostream &os) const
Append global Digits object to ostream.
- void [add_callback](#) ([digits_changed_callback](#) callback)
Add a new callback function.

Private Attributes

- long [digits](#)
Number of decimal digits.
- std::vector< [digits_changed_callback](#) > [callbacklist](#)

Static Private Attributes

- static bool [too_late](#) = false
Already one object present.

6.2.1 Detailed Description

This class is used to instantiate a global singleton object Digits which behaves just like Maple's Digits.

We need an object rather than a dumb basic type since as a side-effect we let it change `cl_default_float_format` when it gets changed. The only other meaningful thing to do with it is converting it to an unsigned, for temporarily storing its value e.g. The user must not create an own working object of this class! Since C++ forces us to make the class definition visible in order to use an object we put in a flag which prevents other objects of that class to be created.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 _numeric_digits()

```
GiNaC::_numeric_digits::_numeric_digits ( )
```

[_numeric_digits](#) default ctor, checking for singleton invariance.

References [too_late](#).

6.2.3 Member Function Documentation

6.2.3.1 operator=()

```
\_numeric\_digits & GiNaC::_numeric_digits::operator= (
    long prec )
```

Assign a native long to global Digits object.

References [callbacklist](#), and [digits](#).

6.2.3.2 operator long()

```
GiNaC::_numeric_digits::operator long ( )
```

Convert global Digits object to native type long.

6.2.3.3 print()

```
void GiNaC::_numeric_digits::print (
    std::ostream & os ) const
```

Append global Digits object to ostream.

References [digits](#).

Referenced by [GiNaC::operator<<\(\)](#).

6.2.3.4 add_callback()

```
void GiNaC::_numeric_digits::add_callback (
    digits_changed_callback callback )
```

Add a new callback function.

References [callbacklist](#).

6.2.4 Member Data Documentation

6.2.4.1 digits

```
long GiNaC::_numeric_digits::digits [private]
```

Number of decimal digits.

Referenced by [operator=\(\)](#), and [print\(\)](#).

6.2.4.2 too_late

```
bool GiNaC::_numeric_digits::too_late = false [static], [private]
```

Already one object present.

Referenced by [_numeric_digits\(\)](#).

6.2.4.3 callbacklist

```
std::vector<digits_changed_callback> GiNaC::_numeric_digits::callbacklist [private]
```

Referenced by [add_callback\(\)](#), and [operator=\(\)](#).

The documentation for this class was generated from the following files:

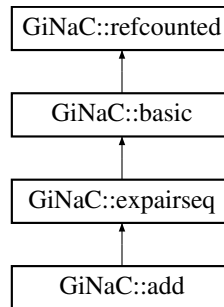
- [numeric.h](#)
- [numeric.cpp](#)

6.3 GiNaC::add Class Reference

Sum of expressions.

```
#include <add.h>
```

Inheritance diagram for GiNaC::add:



Public Member Functions

- [add](#) (const [ex](#) &lh, const [ex](#) &rh)
- [add](#) (const [exvector](#) &v)
- [add](#) (const [epvector](#) &v)
- [add](#) (const [epvector](#) &v, const [ex](#) &oc)
- [add](#) ([epvector](#) &&v)
- [add](#) ([epvector](#) &&v, const [ex](#) &oc)
- unsigned [precedence](#) () const override
Return relative operator precedence (for parenthezing output).
- bool [info](#) (unsigned inf) const override
Information about the object.
- bool [is_polynomial](#) (const [ex](#) &var) const override
Check whether this is a polynomial in the given variables.
- int [degree](#) (const [ex](#) &s) const override
Return degree of highest power in object s.
- int [ldegree](#) (const [ex](#) &s) const override
Return degree of lowest power in object s.
- [ex](#) [coeff](#) (const [ex](#) &s, int n=1) const override
Return coefficient of degree n in object s.
- [ex](#) [eval](#) () const override
Perform automatic term rewriting rules in this class.
- [ex](#) [evalm](#) () const override
Evaluate sums, products and integer powers of matrices.
- [ex](#) [series](#) (const [relational](#) &r, int [order](#), unsigned [options](#)=0) const override
Implementation of [ex::series\(\)](#) for sums.
- [ex](#) [normal](#) ([exmap](#) &repl, [exmap](#) &rev_lookup, [lst](#) &modifier) const override
Implementation of [ex::normal\(\)](#) for a sum.
- [numeric](#) [integer_content](#) () const override
- [ex](#) [smod](#) (const [numeric](#) &xi) const override
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- [numeric](#) [max_coefficient](#) () const override
Implementation [ex::max_coefficient\(\)](#).

- `ex conjugate ()` const override
 - `ex real_part ()` const override
 - `ex imag_part ()` const override
 - `exvector get_free_indices ()` const override
- Return a vector containing the free indices of an expression.*
- `ex eval_ncmul (const exvector &v)` const override

Public Member Functions inherited from `GiNaC::expairseq`

- `expairseq (const ex &lh, const ex &rh)`
 - `expairseq (const exvector &v)`
 - `expairseq (const epvector &v, const ex &oc, bool do_index_renaming=false)`
 - `expairseq (epvector &&vp, const ex &oc, bool do_index_renaming=false)`
 - unsigned `precedence ()` const override
- Return relative operator precedence (for parenthezing output).*
- bool `info (unsigned inf)` const override
- Information about the object.*
- size_t `nops ()` const override
- Number of operands/members.*
- `ex op (size_t i)` const override
- Return operand/member at position i.*
- `ex map (map_function &f)` const override
- Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).*
- `ex eval ()` const override
- Perform coefficient-wise automatic term rewriting rules in this class.*
- `ex to_rational (exmap &repl)` const override
- Implementation of `ex::to_rational()` for expairseqs.*
- `ex to_polynomial (exmap &repl)` const override
- Implementation of `ex::to_polynomial()` for expairseqs.*
- bool `match (const ex &pattern, exmap &repl_lst)` const override
- Check whether the expression matches a given pattern.*
- `ex subs (const exmap &m, unsigned options=0)` const override
- Substitute a set of objects by arbitrary expressions.*
- `ex conjugate ()` const override
 - void `archive (archive_node &n)` const override
- Save (serialize) the object into archive node.*
- void `read_archive (const archive_node &n, lst &syms)` override
- Load (deserialize) the object from an archive node.*

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic ()`
- basic destructor, virtual because class ex will delete objects of derived classes via a basic*.*
- `basic (const basic &other)`
 - const `basic & operator= (const basic &other)`
- basic assignment operator: the other object might be of a derived class.*
- virtual `basic * duplicate ()` const
- Create a clone of this object on the heap.*
- virtual `ex evalf ()` const
- Evaluate object numerically.*

- virtual `ex eval_integ ()` const
Evaluate integrals, if result is known.
- virtual `ex eval_indexed (const basic &i)` const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void `print (const print_context &c, unsigned level=0)` const
Output to stream.
- virtual void `dbgprint ()` const
Little wrapper around print to be called within a debugger.
- virtual void `dbgprinttree ()` const
Little wrapper around printtree to be called within a debugger.
- virtual `ex operator[] (const ex &index)` const
- virtual `ex operator[] (size_t i)` const
- virtual `ex &let_op (size_t i)`
Return modifiable operand/member at position i.
- virtual `ex &operator[] (const ex &index)`
- virtual `ex &operator[] (size_t i)`
- virtual bool `has (const ex &other, unsigned options=0)` const
Test for occurrence of a pattern.
- virtual void `accept (GiNaC::visitor &v)` const
- virtual `ex collect (const ex &s, bool distributed=false)` const
Sort expanded expression in terms of powers of some object(s).
- virtual `ex add_indexed (const ex &self, const ex &other)` const
Add two indexed expressions.
- virtual `ex scalar_mul_indexed (const ex &self, const numeric &other)` const
Multiply an indexed expression with a scalar.
- virtual bool `contract_with (exvector::iterator self, exvector::iterator other, exvector &v)` const
Try to contract two indexed expressions that appear in the same product.
- template<class T >
void `print_dispatch (const print_context &c, unsigned level)` const
Like print(), but dispatch to the specified class.
- void `print_dispatch (const registered_class_info &ri, const print_context &c, unsigned level)` const
Like print(), but dispatch to the specified class.
- `ex subs_one_level (const exmap &m, unsigned options)` const
Helper function for subs().
- `ex diff (const symbol &s, unsigned nth=1)` const
Default interface of nth derivative ex::diff(s, n).
- int `compare (const basic &other)` const
Compare objects syntactically to establish canonical ordering.
- bool `is_equal (const basic &other)` const
Test for syntactic equality.
- const `basic &hold ()` const
Stop further evaluation.
- unsigned `gethash ()` const
- const `basic &setflag (unsigned f)` const
Set some status_flags.
- const `basic &clearflag (unsigned f)` const
Clear some status_flags.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Protected Member Functions

- `ex derivative` (const `symbol` &s) const override
Implementation of `ex::diff()` for a sum.
- unsigned `return_type` () const override
- `return_type_t return_type_tinfo` () const override
- `ex thisexpairseq` (const `epvector` &v, const `ex` &oc, bool do_index_renaming=false) const override
Create an object of this type.
- `ex thisexpairseq` (`epvector` &&vp, const `ex` &oc, bool do_index_renaming=false) const override
- `expair split_ex_to_pair` (const `ex` &e) const override
Form an expair from an ex, using the corresponding semantics.
- `expair combine_ex_with_coeff_to_pair` (const `ex` &e, const `ex` &c) const override
- `expair combine_pair_with_coeff_to_pair` (const `expair` &p, const `ex` &c) const override
- `ex recombine_pair_to_ex` (const `expair` &p) const override
Form an ex out of an expair, using the corresponding semantics.
- `ex expand` (unsigned `options`=0) const override
Expand expression, i.e.
- void `print_add` (const `print_context` &c, const char *openbrace, const char *closebrace, const char *mul_sym, unsigned level) const
- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_latex` (const `print_latex` &c, unsigned level) const
- void `do_print_csrc` (const `print_csrc` &c, unsigned level) const
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const

Protected Member Functions inherited from `GiNaC::expairseq`

- bool `is_equal_same_type` (const `basic` &other) const override
Returns true if two objects of same type are equal.
- unsigned `return_type` () const override
- unsigned `calchash` () const override
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- `ex expand` (unsigned `options`=0) const override
Expand expression, i.e.
- virtual void `printseq` (const `print_context` &c, char delim, unsigned this_precedence, unsigned upper_precedence) const
- virtual void `printpair` (const `print_context` &c, const `expair` &p, unsigned upper_precedence) const
- virtual bool `expair_needs_further_processing` (`expair` it)
- virtual `ex default_overall_coeff` () const
- virtual void `combine_overall_coeff` (const `ex` &c)
- virtual void `combine_overall_coeff` (const `ex` &c1, const `ex` &c2)
- virtual bool `can_make_flat` (const `expair` &p) const
- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
- void `construct_from_2_ex` (const `ex` &lh, const `ex` &rh)

- void `construct_from_2_expairseq` (const `expairseq` &s1, const `expairseq` &s2)
- void `construct_from_expairseq_ex` (const `expairseq` &s, const `ex` &e)
- void `construct_from_exvector` (const `exvector` &v)
- void `construct_from_epvector` (const `epvector` &v, bool do_index_renaming=false)
- void `construct_from_epvector` (`epvector` &&v, bool do_index_renaming=false)
- void `make_flat` (const `exvector` &v)
Combine this expairseq with argument exvector.
- void `make_flat` (const `epvector` &v, bool do_index_renaming=false)
Combine this expairseq with argument epvector.
- void `canonicalize` ()
Brings this expairseq into a sorted (canonical) form.
- void `combine_same_terms_sorted_seq` ()
Compact a presorted expairseq by combining all matching expairs to one each.
- bool `is_canonical` () const
Check if this expairseq is in sorted (canonical) form.
- `epvector` `expandchildren` (unsigned `options`) const
Member-wise expand the expairs in this sequence.
- `epvector` `evalchildren` () const
Member-wise evaluate the expairs in this sequence.
- `epvector` `subschildren` (const `exmap` &m, unsigned `options`=0) const
Member-wise substitute in this sequence.

Protected Member Functions inherited from `GiNaC::basic`

- `basic` ()
- virtual bool `match_same_type` (const `basic` &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual int `compare_same_type` (const `basic` &other) const
Returns order relation between two objects of same type.
- void `ensure_if_modifiable` () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void `do_print` (const `print_context` &c, unsigned level) const
Default output to stream.
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
Tree output to stream.
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
Python parsable output to stream.

Friends

- class `mul`
- class `power`

Additional Inherited Members

Protected Attributes inherited from `GiNaC::expairseq`

- `epvector` `seq`
- `ex` `overall_coeff`

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.3.1 Detailed Description

Sum of expressions.

6.3.2 Constructor & Destructor Documentation

6.3.2.1 [add\(\)](#) [1/6]

```
GiNaC::add::add (
    const ex & lh,
    const ex & rh )
```

References [GiNaC::_ex0](#), [GiNaC::expairseq::construct_from_2_ex\(\)](#), [GINAC_ASSERT](#), [GiNaC::expairseq::is_canonical\(\)](#), and [GiNaC::expairseq::overall_coeff](#).

Referenced by [GiNaC::Gt::qExpand\(\)](#).

6.3.2.2 [add\(\)](#) [2/6]

```
GiNaC::add::add (
    const exvector & v )
```

References [GiNaC::_ex0](#), [GiNaC::expairseq::construct_from_exvector\(\)](#), [GINAC_ASSERT](#), [GiNaC::expairseq::is_canonical\(\)](#), and [GiNaC::expairseq::overall_coeff](#).

6.3.2.3 [add\(\)](#) [3/6]

```
GiNaC::add::add (
    const epvector & v )
```

References [GiNaC::_ex0](#), [GiNaC::expairseq::construct_from_epvector\(\)](#), [GINAC_ASSERT](#), [GiNaC::expairseq::is_canonical\(\)](#), and [GiNaC::expairseq::overall_coeff](#).

6.3.2.4 [add\(\)](#) [4/6]

```
GiNaC::add::add (
    const epvector & v,
    const ex & oc )
```

References [GiNaC::expairseq::construct_from_epvector\(\)](#), [GINAC_ASSERT](#), [GiNaC::expairseq::is_canonical\(\)](#), and [GiNaC::expairseq::overall_coeff](#).

6.3.2.5 add() [5/6]

```
GiNaC::add::add (
    epvector && v )
```

References [GiNaC::ex0](#), [GiNaC::expairseq::construct_from_epvector\(\)](#), [GINAC_ASSERT](#), [GiNaC::expairseq::is_canonical\(\)](#), and [GiNaC::expairseq::overall_coeff](#).

6.3.2.6 add() [6/6]

```
GiNaC::add::add (
    epvector && v,
    const ex & oc )
```

References [GiNaC::expairseq::construct_from_epvector\(\)](#), [GINAC_ASSERT](#), [GiNaC::expairseq::is_canonical\(\)](#), and [GiNaC::expairseq::overall_coeff](#).

6.3.3 Member Function Documentation

6.3.3.1 precedence()

```
unsigned GiNaC::add::precedence ( ) const [inline], [override], [virtual]
```

Return relative operator precedence (for parenthezing output).

Reimplemented from [GiNaC::basic](#).

Referenced by [do_print_csrc\(\)](#), and [print_add\(\)](#).

6.3.3.2 info()

```
bool GiNaC::add::info (
    unsigned inf ) const [override], [virtual]
```

Information about the object.

See also

class [info_flags](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::info_flags::integer](#), [GiNaC::info_flags::integer_polynomial](#), [GiNaC::info_flags::crational](#), [GiNaC::info_flags::crational_polynomial](#), [GiNaC::info_flags::even](#), [GiNaC::ex::info\(\)](#), [info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::info_flags::integer_polynomial](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::info_flags::nonnegative](#), [GiNaC::info_flags::nonnegint](#), [GiNaC::expairseq::overall_coeff](#), [GiNaC::info_flags::polynomial](#), [GiNaC::info_flags::posint](#), [GiNaC::info_flags::positive](#), [GiNaC::info_flags::rational](#), [GiNaC::info_flags::rational_function](#), [GiNaC::info_flags::rational_polynomial](#), [GiNaC::info_flags::real](#), [recombine_pair_to_ex\(\)](#), and [GiNaC::expairseq::seq](#).

Referenced by [info\(\)](#).

6.3.3.3 is_polynomial()

```
bool GiNaC::add::is_polynomial (
    const ex & var ) const [override], [virtual]
```

Check whether this is a polynomial in the given variables.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::expairseq::seq](#).

6.3.3.4 degree()

```
int GiNaC::add::degree (
    const ex & s ) const [override], [virtual]
```

Return degree of highest power in object s.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::is_zero\(\)](#), [GiNaC::expairseq::overall_coeff](#), and [GiNaC::expairseq::seq](#).

6.3.3.5 ldegree()

```
int GiNaC::add::ldegree (
    const ex & s ) const [override], [virtual]
```

Return degree of lowest power in object s.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::is_zero\(\)](#), [GiNaC::expairseq::overall_coeff](#), and [GiNaC::expairseq::seq](#).

6.3.3.6 coeff()

```
ex GiNaC::add::coeff (
    const ex & s,
    int n = 1 ) const [override], [virtual]
```

Return coefficient of degree n in object s.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex0](#), [GiNaC::clifford_max_label\(\)](#), [GiNaC::ex::coeff\(\)](#), [GiNaC::ncmul::coeff\(\)](#), [GiNaC::dirac_ONE\(\)](#), [GiNaC::ex::is_zero\(\)](#), n, [GiNaC::expairseq::overall_coeff](#), and [GiNaC::expairseq::seq](#).

Referenced by [print_add\(\)](#), and [smod\(\)](#).

6.3.3.7 eval()

```
ex GiNaC::add::eval ( ) const [override], [virtual]
```

Perform automatic term rewriting rules in this class.

In the following x stands for a symbolic variables of type `ex` and c stands for such an expression that contain a plain number.

- $+(;c) \rightarrow c$
- $+(x;0) \rightarrow x$

Reimplemented from [GiNaC::basic](#).

References [GiNaC::return_types::commutative](#), [GiNaC::expairseq::evalchildren\(\)](#), [GiNaC::status_flags::evaluated](#), [GiNaC::basic::flags](#), [GINAC_ASSERT](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::expairseq::overall_coeff](#), [recombine_pair_to_ex\(\)](#), [GiNaC::expairseq::seq](#), and [unlikely](#).

6.3.3.8 evalm()

```
ex GiNaC::add::evalm ( ) const [override], [virtual]
```

Evaluate sums, products and integer powers of matrices.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::matrix::add\(\)](#), [GiNaC::ex::evalm\(\)](#), [m](#), [GiNaC::expairseq::overall_coeff](#), [recombine_pair_to_ex\(\)](#), [GiNaC::expairseq::seq](#), and [split_ex_to_pair\(\)](#).

6.3.3.9 series()

```
ex GiNaC::add::series (
    const relational & r,
    int order,
    unsigned options = 0 ) const [override], [virtual]
```

Implementation of [ex::series\(\)](#) for sums.

This performs series addition when adding pseries objects.

See also

[ex::series](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex1](#), [GiNaC::expairseq::op\(\)](#), [options](#), [order](#), [GiNaC::expairseq::overall_coeff](#), [r](#), [GiNaC::expairseq::seq](#), and [GiNaC::ex::series\(\)](#).

6.3.3.10 normal()

```
ex GiNaC::add::normal (
    exmap & repl,
    exmap & rev_lookup,
    lst & modifier ) const [override], [virtual]
```

Implementation of [ex::normal\(\)](#) for a sum.

It expands terms and performs fractional addition.

See also

[ex::normal](#)

Reimplemented from [GiNaC::basic](#).

References [expand\(\)](#), [GiNaC::frac_cancel\(\)](#), [GiNaC::gcd\(\)](#), [GINAC_ASSERT](#), [n](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::container< C >::nops\(\)](#), [GiNaC::ex::normal\(\)](#), [GiNaC::container< C >::op\(\)](#), [GiNaC::expairseq::overall_coeff](#), [recombine_pair_to_ex\(\)](#), and [GiNaC::expairseq::seq](#).

6.3.3.11 integer_content()

```
numeric GiNaC::add::integer_content ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_num0_p](#), [GiNaC::_num1_p](#), [c](#), [GiNaC::denom\(\)](#), [GiNaC::gcd\(\)](#), [GINAC_ASSERT](#), [GiNaC::lcm\(\)](#), [GiNaC::numer\(\)](#), [GiNaC::expairseq::overall_coeff](#), and [GiNaC::expairseq::seq](#).

6.3.3.12 smod()

```
ex GiNaC::add::smod (
    const numeric & xi ) const [override], [virtual]
```

Apply symmetric modular homomorphism to an expanded multivariate polynomial.

This function is usually used internally by [heur_gcd\(\)](#).

Parameters

xi	modulus

Returns

mapped polynomial

See also

[heur_gcd](#)

Reimplemented from [GiNaC::basic](#).

References [coeff\(\)](#), [GINAC_ASSERT](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::expairseq::overall_coeff](#), [GiNaC::expairseq::seq](#), and [GiNaC::smod\(\)](#).

6.3.3.13 max_coefficient()

```
numeric GiNaC::add::max_coefficient ( ) const [override], [virtual]
```

Implementation [ex::max_coefficient\(\)](#).

See also

[heur_gcd](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::abs\(\)](#), [GINAC_ASSERT](#), [GiNaC::expairseq::overall_coeff](#), and [GiNaC::expairseq::seq](#).

6.3.3.14 conjugate()

```
ex GiNaC::add::conjugate ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::are_ex_trivially_equal\(\)](#), [GiNaC::ex::conjugate\(\)](#), [GiNaC::expairseq::nops\(\)](#), and [GiNaC::expairseq::op\(\)](#).

6.3.3.15 real_part()

```
ex GiNaC::add::real_part ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::coeff\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::expairseq::overall_coeff](#), [GiNaC::info_flags::real](#), [GiNaC::ex::real_part\(\)](#), [recombine_pair_to_ex\(\)](#), [GiNaC::expairseq::seq](#), and [split_ex_to_pair\(\)](#).

6.3.3.16 imag_part()

```
ex GiNaC::add::imag_part ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::coeff\(\)](#), [GiNaC::ex::imag_part\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::expairseq::overall_coeff](#), [GiNaC::info_flags::real](#), [recombine_pair_to_ex\(\)](#), [GiNaC::expairseq::seq](#), and [split_ex_to_pair\(\)](#).

6.3.3.17 get_free_indices()

```
exvector GiNaC::add::get_free_indices ( ) const [override], [virtual]
```

Return a vector containing the free indices of an expression.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::get_free_indices\(\)](#), [GiNaC::indices_consistent\(\)](#), [GiNaC::expairseq::nops\(\)](#), and [GiNaC::expairseq::op\(\)](#).

6.3.3.18 eval_ncmul()

```
ex GiNaC::add::eval_ncmul (
    const exvector & v ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::expairseq::seq](#).

6.3.3.19 derivative()

```
ex GiNaC::add::derivative (
    const symbol & y ) const [override], [protected], [virtual]
```

Implementation of [ex::diff\(\)](#) for a sum.

It differentiates each term.

See also

[ex::diff](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::expairseq::seq](#).

6.3.3.20 return_type()

```
unsigned GiNaC::add::return_type ( ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::return_types::commutative](#), and [GiNaC::expairseq::seq](#).

6.3.3.21 return_type_tinfo()

```
return_type_t GiNaC::add::return_type_tinfo ( ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::expairseq::seq](#).

6.3.3.22 thisexpairseq() [1/2]

```
ex GiNaC::add::thisexpairseq (
    const epvector & v,
    const ex & oc,
    bool do_index_renaming = false ) const [override], [protected], [virtual]
```

Create an object of this type.

This method works similar to a constructor. It is useful because expairseq has (at least) two possible different semantics but we want to inherit methods thus avoiding code duplication. Sometimes a method in expairseq has to create a new one of the same semantics, which cannot be done by a ctor because the name (add, mul,...) is unknown on the expairseq level. In order for this trick to work a derived class must of course override this definition.

Reimplemented from [GiNaC::expairseq](#).

6.3.3.23 thisexpairseq() [2/2]

```
ex GiNaC::add::thisexpairseq (
    epvector && vp,
    const ex & oc,
    bool do_index_renaming = false ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::expairseq](#).

6.3.3.24 split_ex_to_pair()

```
expair GiNaC::add::split_ex_to_pair (
    const ex & e ) const [override], [protected], [virtual]
```

Form an expair from an ex, using the corresponding semantics.

See also

[expairseq::recombine_pair_to_ex\(\)](#)

Reimplemented from [GiNaC::expairseq](#).

References [GiNaC::_ex1](#), [GiNaC::basic::clearflag\(\)](#), [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::hash_calculated](#), [GiNaC::ex::is_equal\(\)](#), and [GiNaC::expairseq::overall_coeff](#).

Referenced by [evalm\(\)](#), [imag_part\(\)](#), and [real_part\(\)](#).

6.3.3.25 combine_ex_with_coeff_to_pair()

```
expair GiNaC::add::combine_ex_with_coeff_to_pair (
    const ex & e,
    const ex & c ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::expairseq](#).

References [GiNaC::_ex1](#), [c](#), [GiNaC::basic::clearflag\(\)](#), [GiNaC::status_flags::evaluated](#), [GINAC_ASSERT](#), [GiNaC::status_flags::hash_calculated](#), [GiNaC::ex::is_equal\(\)](#), [likely](#), and [GiNaC::expairseq::overall_coeff](#).

6.3.3.26 combine_pair_with_coeff_to_pair()

```
expair GiNaC::add::combine_pair_with_coeff_to_pair (
    const expair & p,
    const ex & c ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::expairseq](#).

References [GiNaC::_ex1](#), [GiNaC::_num1_p](#), [c](#), [GiNaC::expair::coeff](#), [GINAC_ASSERT](#), and [GiNaC::expair::rest](#).

Referenced by [GiNaC::mul::eval\(\)](#), and [GiNaC::power::expand_add_2\(\)](#).

6.3.3.27 recombine_pair_to_ex()

```
ex GiNaC::add::recombine_pair_to_ex (
    const expair & p ) const [override], [protected], [virtual]
```

Form an ex out of an expair, using the corresponding semantics.

See also

[expairseq::split_ex_to_pair\(\)](#)

Reimplemented from [GiNaC::expairseq](#).

References [GiNaC::_num1_p](#), [GiNaC::expair::coeff](#), and [GiNaC::expair::rest](#).

Referenced by [eval\(\)](#), [evalm\(\)](#), [GiNaC::power::expand\(\)](#), [imag_part\(\)](#), [info\(\)](#), [normal\(\)](#), and [real_part\(\)](#).

6.3.3.28 expand()

```
ex GiNaC::add::expand (
    unsigned options = 0 ) const [override], [protected], [virtual]
```

Expand expression, i.e.

multiply it out and return the result as a new expression.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::expairseq::expandchildren\(\)](#), [GiNaC::status_flags::expanded](#), [options](#), [GiNaC::expairseq::overall_coeff](#), and [GiNaC::basic::setflag\(\)](#).

Referenced by [normal\(\)](#).

6.3.3.29 print_add()

```
void GiNaC::add::print_add (
    const print_context & c,
    const char * openbrace,
    const char * closebrace,
    const char * mul_sym,
    unsigned level ) const [protected]
```

References [GiNaC::_num1_p](#), [GiNaC::_num_1_p](#), [c](#), [coeff\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::expairseq::overall_coeff](#), [precedence\(\)](#), [GiNaC::basic::print\(\)](#), [GiNaC::ex::print\(\)](#), and [GiNaC::expairseq::seq](#).

Referenced by [do_print\(\)](#), and [do_print_latex\(\)](#).

6.3.3.30 do_print()

```
void GiNaC::add::do_print (
    const print\_context & c,
    unsigned level ) const [protected]
```

References [c](#), and [print_add\(\)](#).

6.3.3.31 do_print_latex()

```
void GiNaC::add::do_print_latex (
    const print\_latex & c,
    unsigned level ) const [protected]
```

References [c](#), and [print_add\(\)](#).

6.3.3.32 do_print_csrc()

```
void GiNaC::add::do_print_csrc (
    const print\_csrc & c,
    unsigned level ) const [protected]
```

References [GiNaC::_ex1](#), [GiNaC::_ex_1](#), [GiNaC::_num1_p](#), [GiNaC::_num_1_p](#), [c](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::expairseq::overall_coeff](#), [GiNaC::info_flags::positive](#), [precedence\(\)](#), [GiNaC::ex::print\(\)](#), [GiNaC::info_flags::real](#), and [GiNaC::expairseq::seq](#).

6.3.3.33 do_print_python_repr()

```
void GiNaC::add::do_print_python_repr (
    const print\_python\_repr & c,
    unsigned level ) const [protected]
```

References [c](#), [GiNaC::expairseq::nops\(\)](#), [GiNaC::expairseq::op\(\)](#), and [GiNaC::ex::print\(\)](#).

6.3.4 Friends And Related Symbol Documentation

6.3.4.1 mul

```
friend class mul [friend]
```

6.3.4.2 power

```
friend class power [friend]
```

The documentation for this class was generated from the following files:

- [add.h](#)
- [add.cpp](#)
- [indexed.cpp](#)
- [normal.cpp](#)
- [pseries.cpp](#)

6.4 GiNaC::archive Class Reference

This class holds archived versions of [GiNaC](#) expressions (class [ex](#)).

```
#include <archive.h>
```

Classes

- struct [archived_ex](#)
Archived expression descriptor.

Public Member Functions

- [archive](#) ()
Construct archive from expression using the default name "ex".
- [~archive](#) ()
- [archive](#) (const [ex](#) &e)
Construct archive from expression using the specified name.
- [archive_ex](#) (const [ex](#) &e, const char *n)
Archive an expression.
- [ex_unarchive_ex](#) (const [lst](#) &sym_lst, const char *name) const
Retrieve expression from archive by name.
- [ex_unarchive_ex](#) (const [lst](#) &sym_lst, unsigned index=0) const
Retrieve expression from archive by index.
- [ex_unarchive_ex](#) (const [lst](#) &sym_lst, std::string &name, unsigned index=0) const
Retrieve expression and its name from archive by index.
- unsigned [num_expressions](#) () const
Return number of archived expressions.
- const [archive_node](#) & [get_top_node](#) (unsigned index=0) const
Return reference to top node of an expression specified by index.
- void [clear](#) ()
Clear all archived expressions.
- [archive_node_id](#) [add_node](#) (const [archive_node](#) &n)
Add [archive_node](#) to archive if the corresponding expression is not already archived.
- [archive_node](#) & [get_node](#) ([archive_node_id](#) id)
Retrieve [archive_node](#) by ID.
- void [forget](#) ()
Delete cached unarchived expressions in all [archive_nodes](#) (mainly for debugging).
- void [prinraw](#) (std::ostream &os) const
Print archive to stream in ugly raw format (for debugging).
- [archive_atom](#) [atomize](#) (const std::string &s) const
Atomize a string (i.e.
- const std::string & [unatomize](#) ([archive_atom](#) id) const
Unatomize a string (i.e.

Private Types

- typedef std::map< std::string, [archive_atom](#) >::const_iterator [inv_at_cit](#)
The map of from strings to indices of the atoms vectors allows for faster archiving.

Private Attributes

- `std::vector< archive_node > nodes`
Vector of archived nodes.
- `std::vector< archived_ex > exprs`
Vector of archived expression descriptors.
- `std::vector< std::string > atoms`
Vector of atomized strings (using a vector allows faster unarchiving).
- `std::map< std::string, archive_atom > inverse_atoms`
- `std::map< ex, archive_node_id, ex_is_less > exprtable`
Map of stored expressions to nodes for faster archiving.

Friends

- `std::ostream & operator<< (std::ostream &os, const archive &ar)`
Write archive to binary data stream.
- `std::istream & operator>> (std::istream &is, archive &ar)`
Read archive from binary data stream.

6.4.1 Detailed Description

This class holds archived versions of [GiNaC](#) expressions (class `ex`).

An archive can be constructed from an expression and then written to a stream; or it can be read from a stream and then unarchived, yielding back the expression. Archives can hold multiple expressions which can be referred to by name or index number. The main component of the archive class is a vector of `archive_nodes` which each store one object of class `basic` (or a derived class).

6.4.2 Member Typedef Documentation

6.4.2.1 `inv_at_cit`

```
typedef std::map<std::string,archive_atom>::const_iterator GiNaC::archive::inv_at_cit [private]
```

The map of from strings to indices of the atoms vectors allows for faster archiving.

6.4.3 Constructor & Destructor Documentation

6.4.3.1 `archive()` [1/3]

```
GiNaC::archive::archive ( ) [inline]
```

6.4.3.2 `~archive()`

```
GiNaC::archive::~~archive ( ) [inline]
```

6.4.3.3 archive() [2/3]

```
GiNaC::archive::archive (
    const ex & e ) [inline]
```

Construct archive from expression using the default name "ex".

References [archive_ex\(\)](#).

6.4.3.4 archive() [3/3]

```
GiNaC::archive::archive (
    const ex & e,
    const char * n ) [inline]
```

Construct archive from expression using the specified name.

References [archive_ex\(\)](#), and [n](#).

6.4.4 Member Function Documentation

6.4.4.1 archive_ex()

```
void GiNaC::archive::archive_ex (
    const ex & e,
    const char * name )
```

Archive an expression.

Parameters

<i>e</i>	the expression to be archived
<i>name</i>	name under which the expression is stored

References [add_node\(\)](#), [atomize\(\)](#), and [exprs](#).

Referenced by [archive\(\)](#), and [archive\(\)](#).

6.4.4.2 unarchive_ex() [1/3]

```
ex GiNaC::archive::unarchive_ex (
    const lst & sym_lst,
    const char * name ) const
```

Retrieve expression from archive by name.

Parameters

<i>sym_lst</i>	list of pre-defined symbols
<i>name</i>	name of expression

References [atomize\(\)](#), [exprs](#), and [nodes](#).

6.4.4.3 unarchive_ex() [2/3]

```
ex GiNaC::archive::unarchive_ex (
    const lst & sym_lst,
    unsigned index = 0 ) const
```

Retrieve expression from archive by index.

Parameters

<i>sym_lst</i>	list of pre-defined symbols
<i>index</i>	index of expression

See also

[count_expressions](#)

References [exprs](#), and [nodes](#).

6.4.4.4 unarchive_ex() [3/3]

```
ex GiNaC::archive::unarchive_ex (
    const lst & sym_lst,
    std::string & name,
    unsigned index = 0 ) const
```

Retrieve expression and its name from archive by index.

Parameters

<i>sym_lst</i>	list of pre-defined symbols
<i>name</i>	receives the name of the expression
<i>index</i>	index of expression

See also

[count_expressions](#)

References [exprs](#), [nodes](#), and [unatomize\(\)](#).

6.4.4.5 num_expressions()

```
unsigned GiNaC::archive::num_expressions ( ) const
```

Return number of archived expressions.

References [exprs](#).

6.4.4.6 `get_top_node()`

```
const archive_node & GiNaC::archive::get_top_node (
    unsigned index = 0 ) const
```

Return reference to top node of an expression specified by index.

References [exprs](#), and [nodes](#).

6.4.4.7 `clear()`

```
void GiNaC::archive::clear ( )
```

Clear all archived expressions.

References [atoms](#), [exprs](#), [exprtable](#), [inverse_atoms](#), and [nodes](#).

6.4.4.8 `add_node()`

```
archive_node_id GiNaC::archive::add_node (
    const archive_node & n )
```

Add [archive_node](#) to archive if the corresponding expression is not already archived.

Returns

ID of archived node

References [exprtable](#), [n](#), and [nodes](#).

Referenced by [GiNaC::archive_node::add_ex\(\)](#), and [archive_ex\(\)](#).

6.4.4.9 `get_node()`

```
archive_node & GiNaC::archive::get_node (
    archive_node_id id )
```

Retrieve [archive_node](#) by ID.

References [nodes](#).

Referenced by [GiNaC::archive_node::find_ex\(\)](#), [GiNaC::archive_node::find_ex_by_loc\(\)](#), and [GiNaC::archive_node::find_ex_node\(\)](#).

6.4.4.10 `forget()`

```
void GiNaC::archive::forget ( )
```

Delete cached unarchived expressions in all [archive_nodes](#) (mainly for debugging).

References [GiNaC::archive_node::forget\(\)](#), and [nodes](#).

6.4.4.11 `printraw()`

```
void GiNaC::archive::printraw (
    std::ostream & os ) const
```

Print archive to stream in ugly raw format (for debugging).

References [atoms](#), [GiNaC::ex::begin\(\)](#), [exprs](#), [nodes](#), and [unatomize\(\)](#).

6.4.4.12 `atomize()`

```
archive_atom GiNaC::archive::atomize (
    const std::string & s ) const
```

Atomize a string (i.e.

convert it into an ID number that uniquely represents the string).

References [atoms](#), and [inverse_atoms](#).

Referenced by [GiNaC::archive_node::add_bool\(\)](#), [GiNaC::archive_node::add_ex\(\)](#), [GiNaC::archive_node::add_string\(\)](#), [GiNaC::archive_node::add_unsigned\(\)](#), [archive_ex\(\)](#), [GiNaC::archive_node::find_bool\(\)](#), [GiNaC::archive_node::find_ex\(\)](#), [GiNaC::archive_node::find_ex_node\(\)](#), [GiNaC::archive_node::find_first\(\)](#), [GiNaC::archive_node::find_last\(\)](#), [GiNaC::archive_node::find_property_range\(\)](#), [GiNaC::archive_node::find_string\(\)](#), [GiNaC::archive_node::find_unsigned\(\)](#), and [unarchive_ex\(\)](#).

6.4.4.13 `unatomize()`

```
const std::string & GiNaC::archive::unatomize (
    archive_atom id ) const
```

Unatomize a string (i.e.

convert the ID number back to the string).

References [atoms](#).

Referenced by [GiNaC::archive_node::find_string\(\)](#), [GiNaC::archive_node::get_properties\(\)](#), [GiNaC::archive_node::printraw\(\)](#), [printraw\(\)](#), and [unarchive_ex\(\)](#).

6.4.5 Friends And Related Symbol Documentation

6.4.5.1 `operator<<`

```
std::ostream & operator<< (
    std::ostream & os,
    const archive & ar ) [friend]
```

Write archive to binary data stream.

6.4.5.2 operator>>

```
std::istream & operator>> (
    std::istream & is,
    archive & ar ) [friend]
```

Read archive from binary data stream.

6.4.6 Member Data Documentation

6.4.6.1 nodes

```
std::vector<archive_node> GiNaC::archive::nodes [private]
```

Vector of archived nodes.

Referenced by [add_node\(\)](#), [clear\(\)](#), [forget\(\)](#), [get_node\(\)](#), [get_top_node\(\)](#), [printraw\(\)](#), [unarchive_ex\(\)](#), [unarchive_ex\(\)](#), and [unarchive_ex\(\)](#).

6.4.6.2 exprs

```
std::vector<archived_ex> GiNaC::archive::exprs [private]
```

Vector of archived expression descriptors.

Referenced by [archive_ex\(\)](#), [clear\(\)](#), [get_top_node\(\)](#), [num_expressions\(\)](#), [printraw\(\)](#), [unarchive_ex\(\)](#), [unarchive_ex\(\)](#), and [unarchive_ex\(\)](#).

6.4.6.3 atoms

```
std::vector<std::string> GiNaC::archive::atoms [mutable], [private]
```

Vector of atomized strings (using a vector allows faster unarchiving).

Referenced by [atomize\(\)](#), [clear\(\)](#), [printraw\(\)](#), and [unatomize\(\)](#).

6.4.6.4 inverse_atoms

```
std::map<std::string, archive_atom> GiNaC::archive::inverse_atoms [mutable], [private]
```

Referenced by [atomize\(\)](#), and [clear\(\)](#).

6.4.6.5 exprtable

```
std::map<ex, archive_node_id, ex_is_less> GiNaC::archive::exprtable [mutable], [private]
```

Map of stored expressions to nodes for faster archiving.

Referenced by [add_node\(\)](#), and [clear\(\)](#).

The documentation for this class was generated from the following files:

- [archive.h](#)
- [archive.cpp](#)

6.5 GiNaC::archive_node Class Reference

This class stores all properties needed to record/retrieve the state of one object of class basic (or a derived class).

```
#include <archive.h>
```

Classes

- struct [archive_node_cit_range](#)
- struct [property](#)
Archived property (data type, name and associated data)
- struct [property_info](#)
Information about a stored property.

Public Types

- enum [property_type](#) { [PTYPE_BOOL](#) , [PTYPE_UNSIGNED](#) , [PTYPE_STRING](#) , [PTYPE_NODE](#) }
Property data types.
- typedef std::vector< [property_info](#) > [propinfovector](#)
- typedef std::vector< [property](#) >::const_iterator [archive_node_cit](#)

Public Member Functions

- [archive_node](#) ([archive](#) &ar)
- [archive_node](#) ([archive](#) &ar, const [ex](#) &expr)
Recursively construct archive node from expression.
- const [archive_node](#) & [operator=](#) (const [archive_node](#) &other)
Assignment operator of [archive_node](#).
- void [add_bool](#) (const std::string &name, bool [value](#))
Add property of type "bool" to node.
- void [add_unsigned](#) (const std::string &name, unsigned [value](#))
Add property of type "unsigned int" to node.
- void [add_string](#) (const std::string &name, const std::string &[value](#))
Add property of type "string" to node.
- void [add_ex](#) (const std::string &name, const [ex](#) &[value](#))
Add property of type "ex" to node.
- bool [find_bool](#) (const std::string &name, bool &ret, unsigned index=0) const
Retrieve property of type "bool" from node.
- bool [find_unsigned](#) (const std::string &name, unsigned &ret, unsigned index=0) const
Retrieve property of type "unsigned" from node.
- bool [find_string](#) (const std::string &name, std::string &ret, unsigned index=0) const
Retrieve property of type "string" from node.
- [archive_node_cit](#) [find_first](#) (const std::string &name) const
Find the location in the vector of properties of the first/last property with a given name.
- [archive_node_cit](#) [find_last](#) (const std::string &name) const
- [archive_node_cit_range](#) [find_property_range](#) (const std::string &name1, const std::string &name2) const
Find a range of locations in the vector of properties.
- bool [find_ex](#) (const std::string &name, [ex](#) &ret, [lst](#) &sym_lst, unsigned index=0) const
Retrieve property of type "ex" from node.

- void `find_ex_by_loc` (`archive_node_cit` loc, `ex` &ret, `lst` &sym_lst) const
Retrieve property of type "ex" from the node if it is known that this node in fact contains such a property at the given location.
- const `archive_node` & `find_ex_node` (const std::string &name, unsigned index=0) const
Retrieve property of type "ex" from node, returning the node of the sub-expression.
- void `get_properties` (`propinfovector` &v) const
Return vector of properties stored in node.
- `ex unarchive` (`lst` &sym_lst) const
Convert archive node to GiNaC expression.
- bool `has_same_ex_as` (const `archive_node` &other) const
Check if the `archive_node` stores the same expression as another `archive_node`.
- bool `has_ex` () const
- `ex get_ex` () const
- void `forget` ()
Delete cached unarchived expressions from node (for debugging).
- void `printraw` (std::ostream &os) const
Output `archive_node` to stream in ugly raw format (for debugging).

Private Attributes

- `archive` & `a`
Reference to the archive to which this node belongs.
- std::vector< `property` > `props`
Vector of stored properties.
- bool `has_expression`
Flag indicating whether a cached unarchived representation of this node exists.
- `ex e`
The cached unarchived representation of this node (if any).

Friends

- std::ostream & `operator<<` (std::ostream &os, const `archive_node` &ar)
Write `archive_node` to binary data stream.
- std::istream & `operator>>` (std::istream &is, `archive_node` &ar)
Read `archive_node` from binary data stream.

6.5.1 Detailed Description

This class stores all properties needed to record/retrieve the state of one object of class basic (or a derived class).

Each property is addressed by its name and data type.

6.5.2 Member Typedef Documentation

6.5.2.1 propinfovector

```
typedef std::vector<property_info> GiNaC::archive_node::propinfovector
```

6.5.2.2 archive_node_cit

```
typedef std::vector<property>::const_iterator GiNaC::archive_node::archive_node_cit
```

6.5.3 Member Enumeration Documentation

6.5.3.1 property_type

```
enum GiNaC::archive_node::property_type
```

Property data types.

Enumerator

PTYPE_BOOL	
PTYPE_UNSIGNED	
PTYPE_STRING	
PTYPE_NODE	

6.5.4 Constructor & Destructor Documentation

6.5.4.1 archive_node() [1/2]

```
GiNaC::archive_node::archive_node (
    archive & ar ) [inline]
```

6.5.4.2 archive_node() [2/2]

```
GiNaC::archive_node::archive_node (
    archive & ar,
    const ex & expr )
```

Recursively construct archive node from expression.

References [GiNaC::ex::bp](#).

6.5.5 Member Function Documentation

6.5.5.1 operator=()

```
const archive_node & GiNaC::archive_node::operator= (
    const archive_node & other )
```

Assignment operator of [archive_node](#).

References [e](#), [has_expression](#), and [props](#).

6.5.5.2 add_bool()

```
void GiNaC::archive_node::add_bool (
    const std::string & name,
    bool value )
```

Add property of type "bool" to node.

References [a](#), [GiNaC::archive::atomize\(\)](#), [props](#), [PTYPE_BOOL](#), and [value](#).

6.5.5.3 add_unsigned()

```
void GiNaC::archive_node::add_unsigned (
    const std::string & name,
    unsigned value )
```

Add property of type "unsigned int" to node.

References [a](#), [GiNaC::archive::atomize\(\)](#), [props](#), [PTYPE_UNSIGNED](#), and [value](#).

6.5.5.4 add_string()

```
void GiNaC::archive_node::add_string (
    const std::string & name,
    const std::string & value )
```

Add property of type "string" to node.

References [a](#), [GiNaC::archive::atomize\(\)](#), [props](#), [PTYPE_STRING](#), and [value](#).

6.5.5.5 add_ex()

```
void GiNaC::archive_node::add_ex (
    const std::string & name,
    const ex & value )
```

Add property of type "ex" to node.

References [a](#), [GiNaC::archive::add_node\(\)](#), [GiNaC::archive::atomize\(\)](#), [props](#), [PTYPE_NODE](#), and [value](#).

Referenced by [GiNaC::container< C >::archive\(\)](#).

6.5.5.6 find_bool()

```
bool GiNaC::archive_node::find_bool (
    const std::string & name,
    bool & ret,
    unsigned index = 0 ) const
```

Retrieve property of type "bool" from node.

Returns

"true" if property was found, "false" otherwise

References [a](#), [GiNaC::archive::atomize\(\)](#), [props](#), and [PTYPE_BOOL](#).

6.5.5.7 find_unsigned()

```
bool GiNaC::archive_node::find_unsigned (
    const std::string & name,
    unsigned & ret,
    unsigned index = 0 ) const
```

Retrieve property of type "unsigned" from node.

Returns

"true" if property was found, "false" otherwise

References [a](#), [GiNaC::archive::atomize\(\)](#), [props](#), and [PTYPE_UNSIGNED](#).

6.5.5.8 find_string()

```
bool GiNaC::archive_node::find_string (
    const std::string & name,
    std::string & ret,
    unsigned index = 0 ) const
```

Retrieve property of type "string" from node.

Returns

"true" if property was found, "false" otherwise

References [a](#), [GiNaC::archive::atomize\(\)](#), [props](#), [PTYPE_STRING](#), and [GiNaC::archive::unatomize\(\)](#).

Referenced by [unarchive\(\)](#).

6.5.5.9 find_first()

```
archive_node::archive_node_cit GiNaC::archive_node::find_first (
    const std::string & name ) const
```

Find the location in the vector of properties of the first/last property with a given name.

References [a](#), [GiNaC::archive::atomize\(\)](#), and [props](#).

6.5.5.10 find_last()

```
archive_node::archive_node_cit GiNaC::archive_node::find_last (
    const std::string & name ) const
```

References [a](#), [GiNaC::archive::atomize\(\)](#), and [props](#).

6.5.5.11 find_property_range()

```
archive_node::archive_node_cit_range GiNaC::archive_node::find_property_range (
    const std::string & name1,
    const std::string & name2 ) const
```

Find a range of locations in the vector of properties.

The result begins at the first property with name1 and ends one past the last property with name2.

References [a](#), [GiNaC::archive::atomize\(\)](#), [GiNaC::archive_node::archive_node_cit_range::begin](#), [GiNaC::archive_node::archive_node](#) and [props](#).

6.5.5.12 find_ex()

```
bool GiNaC::archive_node::find_ex (
    const std::string & name,
    ex & ret,
    lst & sym_lst,
    unsigned index = 0 ) const
```

Retrieve property of type "ex" from node.

Returns

"true" if property was found, "false" otherwise

References [a](#), [GiNaC::archive::atomize\(\)](#), [GiNaC::archive::get_node\(\)](#), [props](#), [PTYPE_NODE](#), and [unarchive\(\)](#).

6.5.5.13 find_ex_by_loc()

```
void GiNaC::archive_node::find_ex_by_loc (
    archive_node_cit loc,
    ex & ret,
    lst & sym_lst ) const
```

Retrieve property of type "ex" from the node if it is known that this node in fact contains such a property at the given location.

This is much more efficient than the preceding function.

References [a](#), [GiNaC::archive::get_node\(\)](#), and [unarchive\(\)](#).

6.5.5.14 find_ex_node()

```
const archive_node & GiNaC::archive_node::find_ex_node (
    const std::string & name,
    unsigned index = 0 ) const
```

Retrieve property of type "ex" from node, returning the node of the sub-expression.

References [a](#), [GiNaC::archive::atomize\(\)](#), [GiNaC::archive::get_node\(\)](#), [props](#), and [PTYPE_NODE](#).

6.5.5.15 get_properties()

```
void GiNaC::archive_node::get_properties (
    propinfovector & v ) const
```

Return vector of properties stored in node.

References [a](#), [props](#), and [GiNaC::archive::unatomize\(\)](#).

6.5.5.16 unarchive()

```
ex GiNaC::archive_node::unarchive (
    lst & sym_lst ) const
```

Convert archive node to [GiNaC](#) expression.

References [GiNaC::status_flags::dynallocated](#), [e](#), [GiNaC::find_factory_fcn\(\)](#), [find_string\(\)](#), and [has_expression](#).

Referenced by [find_ex\(\)](#), and [find_ex_by_loc\(\)](#).

6.5.5.17 has_same_ex_as()

```
bool GiNaC::archive_node::has_same_ex_as (
    const archive_node & other ) const
```

Check if the [archive_node](#) stores the same expression as another [archive_node](#).

Returns

"true" if expressions are the same

References [GiNaC::ex::bp](#), [e](#), and [has_expression](#).

6.5.5.18 has_ex()

```
bool GiNaC::archive_node::has_ex ( ) const [inline]
```

References [has_expression](#).

6.5.5.19 get_ex()

```
ex GiNaC::archive_node::get_ex ( ) const [inline]
```

References [e](#).

6.5.5.20 forget()

```
void GiNaC::archive_node::forget ( )
```

Delete cached unarchived expressions from node (for debugging).

References [e](#), and [has_expression](#).

Referenced by [GiNaC::archive::forget\(\)](#).

6.5.5.21 printraw()

```
void GiNaC::archive_node::printraw (
    std::ostream & os ) const
```

Output [archive_node](#) to stream in ugly raw format (for debugging).

References [a](#), [GiNaC::ex::begin\(\)](#), [GiNaC::ex::bp](#), [e](#), [has_expression](#), [props](#), [PTYPE_BOOL](#), [PTYPE_NODE](#), [PTYPE_STRING](#), [PTYPE_UNSIGNED](#), and [GiNaC::archive::unatomize\(\)](#).

6.5.6 Friends And Related Symbol Documentation

6.5.6.1 operator<<

```
std::ostream & operator<< (
    std::ostream & os,
    const archive_node & ar ) [friend]
```

Write [archive_node](#) to binary data stream.

6.5.6.2 operator>>

```
std::istream & operator>> (
    std::istream & is,
    archive_node & ar ) [friend]
```

Read [archive_node](#) from binary data stream.

6.5.7 Member Data Documentation

6.5.7.1 a

```
archive& GiNaC::archive_node::a [private]
```

Reference to the archive to which this node belongs.

Referenced by [add_bool\(\)](#), [add_ex\(\)](#), [add_string\(\)](#), [add_unsigned\(\)](#), [find_bool\(\)](#), [find_ex\(\)](#), [find_ex_by_loc\(\)](#), [find_ex_node\(\)](#), [find_first\(\)](#), [find_last\(\)](#), [find_property_range\(\)](#), [find_string\(\)](#), [find_unsigned\(\)](#), [get_properties\(\)](#), and [printraw\(\)](#).

6.5.7.2 props

```
std::vector<property> GiNaC::archive_node::props [private]
```

Vector of stored properties.

Referenced by [add_bool\(\)](#), [add_ex\(\)](#), [add_string\(\)](#), [add_unsigned\(\)](#), [find_bool\(\)](#), [find_ex\(\)](#), [find_ex_node\(\)](#), [find_first\(\)](#), [find_last\(\)](#), [find_property_range\(\)](#), [find_string\(\)](#), [find_unsigned\(\)](#), [get_properties\(\)](#), [operator=\(\)](#), and [printraw\(\)](#).

6.5.7.3 has_expression

```
bool GiNaC::archive_node::has_expression [mutable], [private]
```

Flag indicating whether a cached unarchived representation of this node exists.

Referenced by [forget\(\)](#), [has_ex\(\)](#), [has_same_ex_as\(\)](#), [operator=\(\)](#), [printraw\(\)](#), and [unarchive\(\)](#).

6.5.7.4 e

```
ex GiNaC::archive_node::e [mutable], [private]
```

The cached unarchived representation of this node (if any).

Referenced by [forget\(\)](#), [get_ex\(\)](#), [has_same_ex_as\(\)](#), [operator=\(\)](#), [printraw\(\)](#), and [unarchive\(\)](#).

The documentation for this class was generated from the following files:

- [archive.h](#)
- [archive.cpp](#)

6.6 GiNaC::archive_node::archive_node_cit_range Struct Reference

```
#include <archive.h>
```

Public Attributes

- [archive_node_cit begin](#)
- [archive_node_cit end](#)

6.6.1 Member Data Documentation

6.6.1.1 begin

```
archive_node_cit GiNaC::archive_node::archive_node_cit_range::begin
```

Referenced by [GiNaC::archive_node::find_property_range\(\)](#).

6.6.1.2 end

`archive_node_cit` `GiNaC::archive_node::archive_node_cit_range::end`

Referenced by `GiNaC::archive_node::find_property_range()`.

The documentation for this struct was generated from the following file:

- `archive.h`

6.7 GiNaC::archive::archived_ex Struct Reference

Archived expression descriptor.

Public Member Functions

- `archived_ex` ()
- `archived_ex` (`archive_atom` `n`, `archive_node_id` `node`)

Public Attributes

- `archive_atom` `name`
Name of expression.
- `archive_node_id` `root`
ID of root node.

6.7.1 Detailed Description

Archived expression descriptor.

6.7.2 Constructor & Destructor Documentation

6.7.2.1 archived_ex() [1/2]

`GiNaC::archive::archived_ex::archived_ex ( )` [inline]

6.7.2.2 archived_ex() [2/2]

`GiNaC::archive::archived_ex::archived_ex (`
`archive_atom n,`
`archive_node_id node )` [inline]

6.7.3 Member Data Documentation

6.7.3.1 name

`archive_atom` GiNaC::archive::archived_ex::name

Name of expression.

6.7.3.2 root

`archive_node_id` GiNaC::archive::archived_ex::root

ID of root node.

The documentation for this struct was generated from the following file:

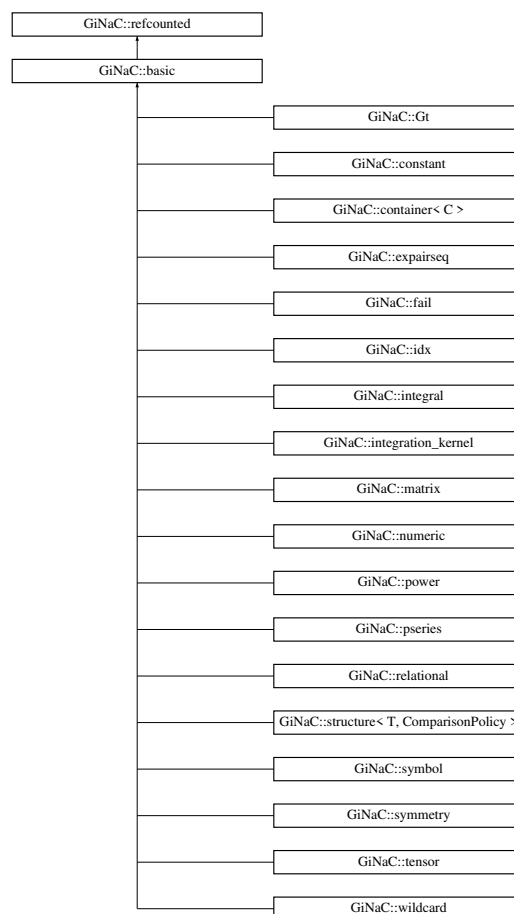
- [archive.h](#)

6.8 GiNaC::basic Class Reference

This class is the ABC (abstract base class) of [GiNaC](#)'s class hierarchy.

```
#include <basic.h>
```

Inheritance diagram for GiNaC::basic:



Public Member Functions

- virtual `~basic` ()
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic` (const `basic` &other)
- const `basic` & `operator=` (const `basic` &other)
basic assignment operator: the other object might be of a derived class.
- virtual `basic` * `duplicate` () const
Create a clone of this object on the heap.
- virtual `ex eval` () const
Perform automatic non-interruptive term rewriting rules.
- virtual `ex evalf` () const
Evaluate object numerically.
- virtual `ex evalm` () const
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ` () const
Evaluate integrals, if result is known.
- virtual `ex eval_indexed` (const `basic` &i) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void `print` (const `print_context` &c, unsigned level=0) const
Output to stream.
- virtual void `dbgprint` () const
Little wrapper around print to be called within a debugger.
- virtual void `dbgprinttree` () const
Little wrapper around printtree to be called within a debugger.
- virtual unsigned `precedence` () const
Return relative operator precedence (for parenthezing output).
- virtual bool `info` (unsigned inf) const
Information about the object.
- virtual size_t `nops` () const
Number of operands/members.
- virtual `ex op` (size_t i) const
Return operand/member at position i.
- virtual `ex operator[]` (const `ex` &index) const
- virtual `ex operator[]` (size_t i) const
- virtual `ex` & `let_op` (size_t i)
Return modifiable operand/member at position i.
- virtual `ex` & `operator[]` (const `ex` &index)
- virtual `ex` & `operator[]` (size_t i)
- virtual bool `has` (const `ex` &other, unsigned `options`=0) const
Test for occurrence of a pattern.
- virtual bool `match` (const `ex` &pattern, `exmap` &repls) const
Check whether the expression matches a given pattern.
- virtual `ex subs` (const `exmap` &m, unsigned `options`=0) const
Substitute a set of objects by arbitrary expressions.
- virtual `ex map` (`map_function` &f) const
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void `accept` (`GiNaC::visitor` &v) const
- virtual bool `is_polynomial` (const `ex` &var) const
Check whether this is a polynomial in the given variables.
- virtual int `degree` (const `ex` &s) const

- Return degree of highest power in object s.*

 - virtual int `ldegree` (const `ex` &s) const
- Return degree of lowest power in object s.*

 - virtual `ex coeff` (const `ex` &s, int `n`=1) const
- Return coefficient of degree n in object s.*

 - virtual `ex expand` (unsigned `options`=0) const
- Expand expression, i.e.*

 - virtual `ex collect` (const `ex` &s, bool `distributed`=false) const
- Sort expanded expression in terms of powers of some object(s).*

 - virtual `ex series` (const `relational` &r, int `order`, unsigned `options`=0) const
- Default implementation of `ex::series()`.*

 - virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const
- Default implementation of `ex::normal()`.*

 - virtual `ex to_rational` (`exmap` &repl) const
- Default implementation of `ex::to_rational()`.*

 - virtual `ex to_polynomial` (`exmap` &repl) const
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &xi) const
- Apply symmetric modular homomorphism to an expanded multivariate polynomial.*

 - virtual `numeric max_coefficient` () const
- Implementation `ex::max_coefficient()`.*

 - virtual `exvector get_free_indices` () const
- Return a vector containing the free indices of an expression.*

 - virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
- Add two indexed expressions.*

 - virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
- Multiply an indexed expression with a scalar.*

 - virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const
- Try to contract two indexed expressions that appear in the same product.*

 - virtual unsigned `return_type` () const
 - virtual `return_type_t return_type_tinfo` () const
 - virtual `ex conjugate` () const
 - virtual `ex real_part` () const
 - virtual `ex imag_part` () const
- template<class T >

 - void `print_dispatch` (const `print_context` &c, unsigned level) const

Like `print()`, but dispatch to the specified class.

 - void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const

Like `print()`, but dispatch to the specified class.

 - virtual void `archive` (`archive_node` &n) const

Save (serialize) the object into archive node.

 - virtual void `read_archive` (const `archive_node` &n, `lst` &syms)

Load (deserialize) the object from an archive node.

 - `ex subs_one_level` (const `exmap` &m, unsigned `options`) const

Helper function for `subs()`.

 - `ex diff` (const `symbol` &s, unsigned `nth`=1) const

Default interface of nth derivative `ex::diff(s, n)`.

 - int `compare` (const `basic` &other) const

Compare objects syntactically to establish canonical ordering.

 - bool `is_equal` (const `basic` &other) const

Test for syntactic equality.

- const [basic](#) & [hold](#) () const
Stop further evaluation.
- unsigned [gethash](#) () const
- const [basic](#) & [setflag](#) (unsigned f) const
Set some [status_flags](#).
- const [basic](#) & [clearflag](#) (unsigned f) const
Clear some [status_flags](#).

Public Member Functions inherited from [GiNaC::refcounted](#)

- [refcounted](#) () noexcept
- unsigned int [add_reference](#) () noexcept
- unsigned int [remove_reference](#) () noexcept
- unsigned int [get_refcount](#) () const noexcept
- void [set_refcount](#) (unsigned int r) noexcept

Protected Member Functions

- [basic](#) ()
- virtual [ex](#) [eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual [ex](#) [derivative](#) (const [symbol](#) &s) const
Default implementation of [ex::diff\(\)](#).
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Protected Attributes

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

Friends

- class [ex](#)

6.8.1 Detailed Description

This class is the ABC (abstract base class) of [GiNaC](#)'s class hierarchy.

6.8.2 Constructor & Destructor Documentation

6.8.2.1 `basic()` [1/2]

```
GiNaC::basic::basic ( ) [inline], [protected]
```

Referenced by [duplicate\(\)](#).

6.8.2.2 `~basic()`

```
virtual GiNaC::basic::~~basic ( ) [inline], [virtual]
```

basic destructor, virtual because class `ex` will delete objects of derived classes via a `basic*`.

References [GiNaC::status_flags::dynallocated](#), [flags](#), [GiNaC::refcounted::get_refcount\(\)](#), and [GINAC_ASSERT](#).

6.8.2.3 `basic()` [2/2]

```
GiNaC::basic::basic (
    const basic & other )
```

6.8.3 Member Function Documentation

6.8.3.1 `operator=()`

```
const basic & GiNaC::basic::operator= (
    const basic & other )
```

basic assignment operator: the other object might be of a derived class.

References [flags](#), and [hashvalue](#).

6.8.3.2 `duplicate()`

```
virtual basic * GiNaC::basic::duplicate ( ) const [inline], [virtual]
```

Create a clone of this object on the heap.

One can think of this as simulating a virtual copy constructor which is needed for instance by the `refcounted` construction of an `ex` from a `basic`.

Reimplemented in [GiNaC::realsymbol](#), and [GiNaC::possymbol](#).

References [basic\(\)](#), [GiNaC::status_flags::dynallocated](#), and [setflag\(\)](#).

Referenced by [GiNaC::ex::construct_from_basic\(\)](#), [GiNaC::idx::map\(\)](#), [GiNaC::idx::replace_dim\(\)](#), [GiNaC::idx::subs\(\)](#), [GiNaC::spinidx::toggle_dot\(\)](#), [GiNaC::varidx::toggle_variance\(\)](#), and [GiNaC::spinidx::toggle_variance_dot\(\)](#).

6.8.3.3 eval()

```
ex GiNaC::basic::eval ( ) const [virtual]
```

Perform automatic non-interruptive term rewriting rules.

Reimplemented in [GiNaC::add](#), [GiNaC::expairseq](#), [GiNaC::fderivative](#), [GiNaC::function](#), [GiNaC::Gt](#), [GiNaC::indexed](#), [GiNaC::integral](#), [GiNaC::mul](#), [GiNaC::ncmul](#), [GiNaC::numeric](#), [GiNaC::power](#), [GiNaC::pseries](#), [GiNaC::structure< T, ComparisonPolicy>](#), and [GiNaC::symbol](#).

Referenced by [GiNaC::ex::construct_from_basic\(\)](#).

6.8.3.4 evalf()

```
ex GiNaC::basic::evalf ( ) const [virtual]
```

Evaluate object numerically.

Reimplemented in [GiNaC::constant](#), [GiNaC::function](#), [GiNaC::Gt](#), [GiNaC::idx](#), [GiNaC::integral](#), [GiNaC::mul](#), [GiNaC::numeric](#), [GiNaC::power](#), [GiNaC::pseries](#), and [GiNaC::symbol](#).

References [GiNaC::nops\(\)](#).

Referenced by [GiNaC::Kronecker_dtau_kernel::get_numerical_value\(\)](#), [GiNaC::Kronecker_dtau_kernel::series_coeff_impl\(\)](#), and [GiNaC::Kronecker_dz_kernel::series_coeff_impl\(\)](#).

6.8.3.5 evalm()

```
ex GiNaC::basic::evalm ( ) const [virtual]
```

Evaluate sums, products and integer powers of matrices.

Reimplemented in [GiNaC::add](#), [GiNaC::matrix](#), [GiNaC::mul](#), [GiNaC::ncmul](#), [GiNaC::power](#), [GiNaC::pseries](#), and [GiNaC::structure< T, ComparisonPolicy>](#).

References [GiNaC::map_evalm](#), and [GiNaC::nops\(\)](#).

6.8.3.6 eval_integ()

```
ex GiNaC::basic::eval_integ ( ) const [virtual]
```

Evaluate integrals, if result is known.

Reimplemented in [GiNaC::integral](#), and [GiNaC::pseries](#).

References [GiNaC::map_eval_integ](#), and [GiNaC::nops\(\)](#).

6.8.3.7 eval_ncmul()

```
ex GiNaC::basic::eval_ncmul (
    const exvector & v ) const [protected], [virtual]
```

Reimplemented in [GiNaC::add](#), [GiNaC::clifford](#), [GiNaC::color](#), [GiNaC::function](#), [GiNaC::integral](#), [GiNaC::mul](#), [GiNaC::power](#), [GiNaC::relational](#), and [GiNaC::structure< T, ComparisonPolicy >](#).

References [GiNaC::hold_ncmul\(\)](#).

6.8.3.8 eval_indexed()

```
ex GiNaC::basic::eval_indexed (
    const basic & i ) const [virtual]
```

Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.

Reimplemented in [GiNaC::su3f](#), [GiNaC::su3d](#), [GiNaC::matrix](#), [GiNaC::structure< T, ComparisonPolicy >](#), [GiNaC::tensdelta](#), [GiNaC::tensmetric](#), [GiNaC::minkmetric](#), [GiNaC::spinmetric](#), and [GiNaC::tensepsilon](#).

6.8.3.9 print()

```
void GiNaC::basic::print (
    const print\_context & c,
    unsigned level = 0 ) const [virtual]
```

Output to stream.

This performs double dispatch on the dynamic type of *this and the dynamic type of the supplied print context.

Parameters

<i>c</i>	print context object that describes the output formatting
<i>level</i>	value that is used to identify the precedence or indentation level for placing parentheses and formatting

Reimplemented in [GiNaC::fderivative](#), [GiNaC::function](#), and [GiNaC::structure< T, ComparisonPolicy >](#).

References [c](#).

Referenced by [GiNaC::fderivative::print\(\)](#), [GiNaC::add::print_add\(\)](#), [GiNaC::mul::print_overall_coeff\(\)](#), and [GiNaC::pseries::print_series\(\)](#).

6.8.3.10 dbgprint()

```
void GiNaC::basic::dbgprint ( ) const [virtual]
```

Little wrapper around print to be called within a debugger.

This is needed because you cannot call `foo.print(cout)` from within the debugger because it might not know what `cout` is. This method can be invoked with no argument and it will simply print to `stdout`.

See also

[basic::print](#)
[basic::dbgprinttree](#)

6.8.3.11 dbgprinttree()

```
void GiNaC::basic::dbgprinttree ( ) const [virtual]
```

Little wrapper around printtree to be called within a debugger.

See also

[basic::dbgprint](#)

6.8.3.12 precedence()

```
unsigned GiNaC::basic::precedence ( ) const [virtual]
```

Return relative operator precedence (for parenthezing output).

Reimplemented in [GiNaC::add](#), [GiNaC::clifford](#), [GiNaC::container< C >](#), [GiNaC::expairseq](#), [GiNaC::function](#), [GiNaC::Gt](#), [GiNaC::indexed](#), [GiNaC::integral](#), [GiNaC::mul](#), [GiNaC::ncmul](#), [GiNaC::numeric](#), [GiNaC::power](#), [GiNaC::pseries](#), [GiNaC::relational](#), and [GiNaC::structure< T, ComparisonPolicy >](#).

6.8.3.13 info()

```
bool GiNaC::basic::info (
    unsigned inf ) const [virtual]
```

Information about the object.

See also

[class info_flags](#)

Reimplemented in [GiNaC::add](#), [GiNaC::constant](#), [GiNaC::container< C >](#), [GiNaC::expairseq](#), [GiNaC::function](#), [GiNaC::idx](#), [GiNaC::indexed](#), [GiNaC::mul](#), [GiNaC::ncmul](#), [GiNaC::numeric](#), [GiNaC::power](#), [GiNaC::relational](#), [GiNaC::structure< T, ComparisonPolicy >](#), [GiNaC::symbol](#), [GiNaC::tensdelta](#), [GiNaC::tensmetric](#), [GiNaC::minkmetric](#), [GiNaC::spinmetric](#), and [GiNaC::tensepsilon](#).

Referenced by [GiNaC::function::info\(\)](#), and [GiNaC::matrix::solve\(\)](#).

6.8.3.14 nops()

```
size_t GiNaC::basic::nops ( ) const [virtual]
```

Number of operands/members.

Reimplemented in [GiNaC::clifford](#), [GiNaC::container< C >](#), [GiNaC::expairseq](#), [GiNaC::Gt](#), [GiNaC::idx](#), [GiNaC::integral](#), [GiNaC::multiple_polylog_kernel](#), [GiNaC::ELi_kernel](#), [GiNaC::Ebar_kernel](#), [GiNaC::Kronecker_dtau_kernel](#), [GiNaC::Kronecker_dz_kernel](#), [GiNaC::Eisenstein_kernel](#), [GiNaC::Eisenstein_h_kernel](#), [GiNaC::modular_form_kernel](#), [GiNaC::user_defined_kernel](#), [GiNaC::matrix](#), [GiNaC::power](#), [GiNaC::pseries](#), [GiNaC::relational](#), and [GiNaC::structure< T, ComparisonPolicy >](#).

Referenced by [GiNaC::su3f::eval_indexed\(\)](#), [GiNaC::su3d::eval_indexed\(\)](#), [GiNaC::matrix::eval_indexed\(\)](#), [GiNaC::tensmetric::eval_indexed\(\)](#), [GiNaC::minkmetric::eval_indexed\(\)](#), [GiNaC::spinmetric::eval_indexed\(\)](#), [GiNaC::tensepsilon::eval_indexed\(\)](#), [GiNaC::H_eval\(\)](#), and [normal\(\)](#).

6.8.3.15 op()

```
ex GiNaC::basic::op (
    size_t i ) const [virtual]
```

Return operand/member at position i.

Reimplemented in [GiNaC::clifford](#), [GiNaC::container< C >](#), [GiNaC::expairseq](#), [GiNaC::idx](#), [GiNaC::integral](#), [GiNaC::multiple_polylog_kernel](#), [GiNaC::ELi_kernel](#), [GiNaC::Ebar_kernel](#), [GiNaC::Kronecker_dtau_kernel](#), [GiNaC::Kronecker_dz_kernel](#), [GiNaC::Eisenstein_kernel](#), [GiNaC::Eisenstein_h_kernel](#), [GiNaC::modular_form_kernel](#), [GiNaC::user_defined_kernel](#), [GiNaC::matrix](#), [GiNaC::power](#), [GiNaC::pseries](#), [GiNaC::relational](#), and [GiNaC::structure< T, ComparisonPolicy >](#).

Referenced by [GiNaC::su3f::eval_indexed\(\)](#), [GiNaC::su3d::eval_indexed\(\)](#), [GiNaC::matrix::eval_indexed\(\)](#), [GiNaC::tensmetric::eval_indexed\(\)](#), [GiNaC::minkmetric::eval_indexed\(\)](#), [GiNaC::spinmetric::eval_indexed\(\)](#), and [GiNaC::tensepsilon::eval_indexed\(\)](#).

6.8.3.16 operator[]() [1/4]

```
ex GiNaC::basic::operator[] (
    const ex & index ) const [virtual]
```

Reimplemented in [GiNaC::structure< T, ComparisonPolicy >](#).

References [GiNaC::op\(\)](#), and [GiNaC::to_int\(\)](#).

6.8.3.17 operator[]() [2/4]

```
ex GiNaC::basic::operator[] (
    size_t i ) const [virtual]
```

Reimplemented in [GiNaC::structure< T, ComparisonPolicy >](#).

References [GiNaC::op\(\)](#).

6.8.3.18 let_op()

```
ex & GiNaC::basic::let_op (
    size_t i ) [virtual]
```

Return modifiable operand/member at position i.

Reimplemented in [GiNaC::clifford](#), [GiNaC::container< C >](#), [GiNaC::integral](#), [GiNaC::multiple_polylog_kernel](#), [GiNaC::ELi_kernel](#), [GiNaC::Ebar_kernel](#), [GiNaC::Kronecker_dtau_kernel](#), [GiNaC::Kronecker_dz_kernel](#), [GiNaC::Eisenstein_kernel](#), [GiNaC::Eisenstein_h_kernel](#), [GiNaC::modular_form_kernel](#), [GiNaC::user_defined_kernel](#), [GiNaC::matrix](#), and [GiNaC::structure< T, ComparisonPolicy >](#).

Referenced by [map\(\)](#), and [subs\(\)](#).

6.8.3.19 operator[]() [3/4]

```
ex & GiNaC::basic::operator[] (
    const ex & index ) [virtual]
```

Reimplemented in [GiNaC::structure< T, ComparisonPolicy >](#).

References [GiNaC::to_int\(\)](#).

6.8.3.20 operator[]() [4/4]

```
ex & GiNaC::basic::operator[] (
    size_t i ) [virtual]
```

Reimplemented in [GiNaC::structure< T, ComparisonPolicy >](#).

6.8.3.21 has()

```
bool GiNaC::basic::has (
    const ex & pattern,
    unsigned options = 0 ) const [virtual]
```

Test for occurrence of a pattern.

An object 'has' a pattern if it matches the pattern itself or one of the children 'has' it. As a consequence (according to the definition of children) given $e=x+y+z$, $e.has(x)$ is true but $e.has(x+y)$ is false.

Reimplemented in [GiNaC::Gt](#), [GiNaC::mul](#), [GiNaC::numeric](#), [GiNaC::power](#), and [GiNaC::structure< T, ComparisonPolicy >](#).

References [GiNaC::ex::has\(\)](#), [GiNaC::match\(\)](#), [GiNaC::nops\(\)](#), [GiNaC::op\(\)](#), and [options](#).

Referenced by [GiNaC::mul::has\(\)](#), [GiNaC::power::has\(\)](#), and [series\(\)](#).

6.8.3.22 match()

```
bool GiNaC::basic::match (
    const ex & pattern,
    exmap & repl_lst ) const [virtual]
```

Check whether the expression matches a given pattern.

For every wildcard object in the pattern, a pair with the wildcard as a key and matching expression as a value is added to `repl_lst`.

Reimplemented in [GiNaC::expairseq](#), [GiNaC::Gt](#), [GiNaC::structure< T, ComparisonPolicy >](#), and [GiNaC::wildcard](#).

References [GiNaC::ex::match\(\)](#), [GiNaC::ex::nops\(\)](#), [GiNaC::nops\(\)](#), [GiNaC::op\(\)](#), and [GiNaC::ex::op\(\)](#).

6.8.3.23 match_same_type()

```
bool GiNaC::basic::match_same_type (
    const basic & other ) const [protected], [virtual]
```

Returns true if the attributes of two objects are similar enough for a match.

This function must not match subexpressions (this is already done by [basic::match\(\)](#)). Only attributes not accessible by [op\(\)](#) should be compared. This is also the reason why this function doesn't take the wildcard replacement list from [match\(\)](#) as an argument: only subexpressions are subject to wildcard matches. Also, this function only needs to be implemented for container classes because [is_equal_same_type\(\)](#) is automatically used instead of [match_same_type\(\)](#) if [nops\(\)](#) == 0.

See also

[basic::match](#)

Reimplemented in [GiNaC::clifford](#), [GiNaC::color](#), [GiNaC::fderivative](#), [GiNaC::function](#), [GiNaC::idx](#), [GiNaC::varidx](#), [GiNaC::spinidx](#), [GiNaC::matrix](#), [GiNaC::relational](#), and [GiNaC::structure< T, ComparisonPolicy >](#).

6.8.3.24 subs()

```
ex GiNaC::basic::subs (
    const exmap & m,
    unsigned options = 0 ) const [virtual]
```

Substitute a set of objects by arbitrary expressions.

The ex returned will already be evaluated.

Reimplemented in [GiNaC::clifford](#), [GiNaC::container< C >](#), [GiNaC::expairseq](#), [GiNaC::Gt](#), [GiNaC::idx](#), [GiNaC::matrix](#), [GiNaC::numeric](#), [GiNaC::power](#), [GiNaC::pseries](#), [GiNaC::relational](#), [GiNaC::structure< T, ComparisonPolicy >](#), and [GiNaC::symbol](#).

References [GiNaC::are_ex_trivially_equal\(\)](#), [clearflag\(\)](#), [let_op\(\)](#), [m](#), [GiNaC::nops\(\)](#), [GiNaC::op\(\)](#), [options](#), [GiNaC::ex::subs\(\)](#), and [subs_one_level\(\)](#).

Referenced by [GiNaC::tensmetric::eval_indexed\(\)](#).

6.8.3.25 map()

```
ex GiNaC::basic::map (
    map\_function & f ) const [virtual]
```

Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).

Reimplemented in [GiNaC::expairseq](#), [GiNaC::idx](#), [GiNaC::power](#), [GiNaC::relational](#), and [GiNaC::structure< T, ComparisonPolicy >](#).

References [GiNaC::are_ex_trivially_equal\(\)](#), [clearflag\(\)](#), [let_op\(\)](#), [n](#), [GiNaC::nops\(\)](#), and [GiNaC::op\(\)](#).

Referenced by [normal\(\)](#).

6.8.3.26 accept()

```
virtual void GiNaC::basic::accept (
    GiNaC::visitor & v ) const [inline], [virtual]
```

6.8.3.27 is_polynomial()

```
bool GiNaC::basic::is_polynomial (
    const ex & var ) const [virtual]
```

Check whether this is a polynomial in the given variables.

Reimplemented in [GiNaC::add](#), [GiNaC::constant](#), [GiNaC::mul](#), [GiNaC::numeric](#), [GiNaC::power](#), and [GiNaC::symbol](#).

References [GiNaC::has\(\)](#).

6.8.3.28 degree()

```
int GiNaC::basic::degree (
    const ex & s ) const [virtual]
```

Return degree of highest power in object s.

Reimplemented in [GiNaC::add](#), [GiNaC::integral](#), [GiNaC::mul](#), [GiNaC::ncmul](#), [GiNaC::numeric](#), [GiNaC::power](#), [GiNaC::pseries](#), and [GiNaC::structure< T, ComparisonPolicy >](#).

6.8.3.29 ldegree()

```
int GiNaC::basic::ldegree (
    const ex & s ) const [virtual]
```

Return degree of lowest power in object s.

Reimplemented in [GiNaC::add](#), [GiNaC::integral](#), [GiNaC::mul](#), [GiNaC::ncmul](#), [GiNaC::numeric](#), [GiNaC::power](#), [GiNaC::pseries](#), and [GiNaC::structure< T, ComparisonPolicy >](#).

6.8.3.30 coeff()

```
ex GiNaC::basic::coeff (
    const ex & s,
    int n = 1 ) const [virtual]
```

Return coefficient of degree n in object s.

Reimplemented in [GiNaC::add](#), [GiNaC::mul](#), [GiNaC::ncmul](#), [GiNaC::numeric](#), [GiNaC::power](#), [GiNaC::pseries](#), and [GiNaC::structure< T, ComparisonPolicy >](#).

References [GiNaC::_ex0](#), [GiNaC::_ex1](#), and [n](#).

Referenced by [GiNaC::expairseq::read_archive\(\)](#), [series\(\)](#), and [GiNaC::sqrfree_parfrac\(\)](#).

6.8.3.31 expand()

```
ex GiNaC::basic::expand (
    unsigned options = 0 ) const [virtual]
```

Expand expression, i.e.

multiply it out and return the result as a new expression.

Reimplemented in [GiNaC::add](#), [GiNaC::expairseq](#), [GiNaC::function](#), [GiNaC::indexed](#), [GiNaC::integral](#), [GiNaC::mul](#), [GiNaC::ncmul](#), [GiNaC::power](#), [GiNaC::pseries](#), and [GiNaC::structure< T, ComparisonPolicy >](#).

References [GiNaC::nops\(\)](#), and [options](#).

Referenced by [GiNaC::power::expand\(\)](#), [GiNaC::matrix::fraction_free_elimination\(\)](#), and [GiNaC::matrix::pivot\(\)](#).

6.8.3.32 collect()

```
ex GiNaC::basic::collect (
    const ex & s,
    bool distributed = false ) const [virtual]
```

Sort expanded expression in terms of powers of some object(s).

Parameters

<i>s</i>	object(s) to sort in
<i>distributed</i>	recursive or distributed form (only used when s is a list)

Reimplemented in [GiNaC::pseries](#), and [GiNaC::structure< T, ComparisonPolicy >](#).

References [GiNaC::_ex0](#), [GiNaC::_ex1](#), [GiNaC::ex::coeff\(\)](#), [GiNaC::coeff\(\)](#), [GiNaC::collect\(\)](#), [GiNaC::ex::degree\(\)](#), [GiNaC::degree\(\)](#), [GiNaC::expand\(\)](#), [GiNaC::ldegree\(\)](#), [n](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), [GiNaC::pow\(\)](#), and [x](#).

6.8.3.33 derivative()

```
ex GiNaC::basic::derivative (
    const symbol & s ) const [protected], [virtual]
```

Default implementation of [ex::diff\(\)](#).

It maps the operation on the operands (or returns 0 when the object has no operands).

See also

[ex::diff](#)

Reimplemented in [GiNaC::add](#), [GiNaC::constant](#), [GiNaC::fderivative](#), [GiNaC::function](#), [GiNaC::idx](#), [GiNaC::indexed](#), [GiNaC::integral](#), [GiNaC::mul](#), [GiNaC::ncmul](#), [GiNaC::numeric](#), [GiNaC::power](#), [GiNaC::pseries](#), [GiNaC::structure< T, ComparisonPolicy >](#) and [GiNaC::symbol](#).

References [GiNaC::_ex0](#), and [GiNaC::nops\(\)](#).

6.8.3.34 series()

```
ex GiNaC::basic::series (
    const relational & r,
    int order,
    unsigned options = 0 ) const [virtual]
```

Default implementation of [ex::series\(\)](#).

This performs Taylor expansion.

See also

[ex::series](#)

Reimplemented in [GiNaC::add](#), [GiNaC::fderivative](#), [GiNaC::function](#), [GiNaC::integral](#), [GiNaC::integration_kernel](#), [GiNaC::Eisenstein_kernel](#), [GiNaC::Eisenstein_h_kernel](#), [GiNaC::modular_form_kernel](#), [GiNaC::pseries](#), [GiNaC::structure< T, ComparisonPolicy>](#), [GiNaC::mul](#), [GiNaC::power](#), and [GiNaC::symbol](#).

References [GiNaC::_ex0](#), [GiNaC::_ex1](#), [coeff\(\)](#), [GiNaC::ex::diff\(\)](#), [GiNaC::numeric::div\(\)](#), [GiNaC::ex::expand\(\)](#), [has\(\)](#), [GiNaC::ex::is_zero\(\)](#), [n](#), [GiNaC::subs_options::no_pattern](#), [order](#), [r](#), and [GiNaC::ex::subs\(\)](#).

Referenced by [GiNaC::lgamma_series\(\)](#), [GiNaC::fderivative::series\(\)](#), [GiNaC::function::series\(\)](#), and [GiNaC::power::series\(\)](#).

6.8.3.35 normal()

```
ex GiNaC::basic::normal (
    exmap & repl,
    exmap & rev_lookup,
    lst & modifier ) const [virtual]
```

Default implementation of [ex::normal\(\)](#).

It normalizes the children and replaces the object with a temporary symbol.

See also

[ex::normal](#)

Reimplemented in [GiNaC::add](#), [GiNaC::mul](#), [GiNaC::numeric](#), [GiNaC::power](#), [GiNaC::pseries](#), [GiNaC::structure< T, ComparisonPolicy>](#), and [GiNaC::symbol](#).

References [GiNaC::_ex1](#), [GiNaC::ex::info\(\)](#), [map\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::subs_options::no_pattern](#), [nops\(\)](#), [GiNaC::container< C >::nops\(\)](#), [GiNaC::ex::op\(\)](#), [GiNaC::container< C >::op\(\)](#), [GiNaC::replace_with_symbol\(\)](#), and [GiNaC::ex::subs\(\)](#).

Referenced by [GiNaC::matrix::determinant\(\)](#), [GiNaC::matrix::gauss_elimination\(\)](#), [GiNaC::matrix::markowitz_elimination\(\)](#), [GiNaC::matrix::solve\(\)](#), and [GiNaC::Gt::tauToFundamental\(\)](#).

6.8.3.36 to_rational()

```
ex GiNaC::basic::to_rational (
    exmap & repl ) const [virtual]
```

Default implementation of [ex::to_rational\(\)](#).

This replaces the object with a temporary symbol.

Reimplemented in [GiNaC::expairseq](#), [GiNaC::numeric](#), [GiNaC::power](#), [GiNaC::structure< T, ComparisonPolicy >](#), and [GiNaC::symbol](#).

References [GiNaC::replace_with_symbol\(\)](#).

6.8.3.37 to_polynomial()

```
ex GiNaC::basic::to_polynomial (
    exmap & repl ) const [virtual]
```

Reimplemented in [GiNaC::expairseq](#), [GiNaC::numeric](#), [GiNaC::power](#), [GiNaC::structure< T, ComparisonPolicy >](#), and [GiNaC::symbol](#).

References [GiNaC::replace_with_symbol\(\)](#).

6.8.3.38 integer_content()

```
numeric GiNaC::basic::integer_content ( ) const [virtual]
```

Reimplemented in [GiNaC::add](#), [GiNaC::mul](#), [GiNaC::numeric](#), and [GiNaC::structure< T, ComparisonPolicy >](#).

References [GiNaC::_num1_p](#).

6.8.3.39 smod()

```
ex GiNaC::basic::smod (
    const numeric & xi ) const [virtual]
```

Apply symmetric modular homomorphism to an expanded multivariate polynomial.

This function is usually used internally by [heur_gcd\(\)](#).

Parameters

xi	modulus

Returns

mapped polynomial

See also

[heur_gcd](#)

Reimplemented in [GiNaC::add](#), [GiNaC::mul](#), [GiNaC::numeric](#), and [GiNaC::structure< T, ComparisonPolicy >](#).

6.8.3.40 max_coefficient()

```
numeric GiNaC::basic::max_coefficient ( ) const [virtual]
```

Implementation [ex::max_coefficient\(\)](#).

See also

[heur_gcd](#)

Reimplemented in [GiNaC::add](#), [GiNaC::mul](#), [GiNaC::numeric](#), and [GiNaC::structure< T, ComparisonPolicy >](#).

References [GiNaC::_num1_p](#).

6.8.3.41 get_free_indices()

```
exvector GiNaC::basic::get_free_indices ( ) const [virtual]
```

Return a vector containing the free indices of an expression.

Reimplemented in [GiNaC::add](#), [GiNaC::indexed](#), [GiNaC::integral](#), [GiNaC::mul](#), [GiNaC::ncmul](#), and [GiNaC::structure< T, ComparisonPolicy >](#).

6.8.3.42 add_indexed()

```
ex GiNaC::basic::add_indexed (
    const ex & self,
    const ex & other ) const [virtual]
```

Add two indexed expressions.

They are guaranteed to be of class `indexed` (or a subclass) and their indices are compatible. This function is used internally by [simplify_indexed\(\)](#).

Parameters

<i>self</i>	First indexed expression; its base object is <code>*this</code>
<i>other</i>	Second indexed expression

Returns

sum of `self` and `other`

See also

[ex::simplify_indexed\(\)](#)

Reimplemented in [GiNaC::matrix](#), and [GiNaC::structure< T, ComparisonPolicy >](#).

6.8.3.43 scalar_mul_indexed()

```
ex GiNaC::basic::scalar_mul_indexed (
    const ex & self,
    const numeric & other ) const [virtual]
```

Multiply an indexed expression with a scalar.

This function is used internally by [simplify_indexed\(\)](#).

Parameters

<i>self</i>	Indexed expression; its base object is *this
<i>other</i>	Numeric value

Returns

product of self and other

See also

[ex::simplify_indexed\(\)](#)

Reimplemented in [GiNaC::matrix](#), and [GiNaC::structure< T, ComparisonPolicy >](#).

6.8.3.44 contract_with()

```
bool GiNaC::basic::contract_with (
    exvector::iterator self,
    exvector::iterator other,
    exvector & v ) const [virtual]
```

Try to contract two indexed expressions that appear in the same product.

If a contraction exists, the function overwrites one or both of the expressions and returns true. Otherwise it returns false. It is guaranteed that both expressions are of class indexed (or a subclass) and that at least one dummy index has been found. This functions is used internally by [simplify_indexed\(\)](#).

Parameters

<i>self</i>	Pointer to first indexed expression; its base object is *this
<i>other</i>	Pointer to second indexed expression
<i>v</i>	The complete vector of factors

Returns

true if the contraction was successful, false otherwise

See also

[ex::simplify_indexed\(\)](#)

Reimplemented in [GiNaC::cliffordunit](#), [GiNaC::diracgamma](#), [GiNaC::su3t](#), [GiNaC::su3f](#), [GiNaC::su3d](#), [GiNaC::matrix](#), [GiNaC::structure< T, ComparisonPolicy >](#), [GiNaC::tensdelta](#), [GiNaC::tensmetric](#), [GiNaC::spinmetric](#), and [GiNaC::tensepsilon](#).

6.8.3.45 return_type()

```
unsigned GiNaC::basic::return_type ( ) const [virtual]
```

Reimplemented in [GiNaC::add](#), [GiNaC::clifford](#), [GiNaC::color](#), [GiNaC::su3f](#), [GiNaC::su3d](#), [GiNaC::expairseq](#), [GiNaC::fail](#), [GiNaC::function](#), [GiNaC::indexed](#), [GiNaC::integral](#), [GiNaC::matrix](#), [GiNaC::mul](#), [GiNaC::ncmul](#), [GiNaC::power](#), [GiNaC::relational](#), [GiNaC::structure< T, ComparisonPolicy >](#), [GiNaC::tensor](#), [GiNaC::tensdelta](#), [GiNaC::tensmetric](#), [GiNaC::minkmetric](#), and [GiNaC::tensepsilon](#).

6.8.3.46 return_type_tinfo()

```
return_type_t GiNaC::basic::return_type_tinfo ( ) const [virtual]
```

Reimplemented in [GiNaC::add](#), [GiNaC::clifford](#), [GiNaC::color](#), [GiNaC::function](#), [GiNaC::indexed](#), [GiNaC::integral](#), [GiNaC::mul](#), [GiNaC::ncmul](#), [GiNaC::power](#), [GiNaC::relational](#), and [GiNaC::structure< T, ComparisonPolicy >](#).

References [GiNaC::return_type_t::rl](#), and [GiNaC::return_type_t::tinfo](#).

6.8.3.47 conjugate()

```
ex GiNaC::basic::conjugate ( ) const [virtual]
```

Reimplemented in [GiNaC::add](#), [GiNaC::diracgamma5](#), [GiNaC::diracgammaL](#), [GiNaC::diracgammaR](#), [GiNaC::constant](#), [GiNaC::container< C >](#), [GiNaC::expairseq](#), [GiNaC::function](#), [GiNaC::spinidx](#), [GiNaC::integral](#), [GiNaC::matrix](#), [GiNaC::mul](#), [GiNaC::ncmul](#), [GiNaC::numeric](#), [GiNaC::power](#), [GiNaC::pseries](#), [GiNaC::symbol](#), and [GiNaC::realsymbol](#).

6.8.3.48 real_part()

```
ex GiNaC::basic::real_part ( ) const [virtual]
```

Reimplemented in [GiNaC::add](#), [GiNaC::constant](#), [GiNaC::container< C >](#), [GiNaC::function](#), [GiNaC::indexed](#), [GiNaC::matrix](#), [GiNaC::mul](#), [GiNaC::ncmul](#), [GiNaC::numeric](#), [GiNaC::power](#), [GiNaC::pseries](#), [GiNaC::symbol](#), and [GiNaC::realsymbol](#).

Referenced by [GiNaC::function::real_part\(\)](#), and [GiNaC::ncmul::real_part\(\)](#).

6.8.3.49 imag_part()

```
ex GiNaC::basic::imag_part ( ) const [virtual]
```

Reimplemented in [GiNaC::add](#), [GiNaC::constant](#), [GiNaC::container< C >](#), [GiNaC::function](#), [GiNaC::indexed](#), [GiNaC::matrix](#), [GiNaC::mul](#), [GiNaC::ncmul](#), [GiNaC::numeric](#), [GiNaC::power](#), [GiNaC::pseries](#), [GiNaC::symbol](#), and [GiNaC::realsymbol](#).

Referenced by [GiNaC::function::imag_part\(\)](#), and [GiNaC::ncmul::imag_part\(\)](#).

6.8.3.50 compare_same_type()

```
int GiNaC::basic::compare_same_type (
    const basic & other ) const [protected], [virtual]
```

Returns order relation between two objects of same type.

This needs to be implemented by each class. It may never return anything else than 0, signalling equality, or +1 and -1 signalling inequality and determining the canonical ordering. (Perl hackers will wonder why C++ doesn't feature the spaceship operator `<=>` for denoting just this.)

References [GiNaC::compare_pointers\(\)](#).

Referenced by [GiNaC::symbol::derivative\(\)](#).

6.8.3.51 is_equal_same_type()

```
bool GiNaC::basic::is_equal_same_type (
    const basic & other ) const [protected], [virtual]
```

Returns true if two objects of same type are equal.

Normally needs not be reimplemented as long as it wasn't overwritten by some parent class, since it just calls [compare_same_type\(\)](#). The reason why this function exists is that sometimes it is easier to determine equality than an order relation and then it can be overridden.

Reimplemented in [GiNaC::constant](#), [GiNaC::container< C >](#), [GiNaC::expairseq](#), [GiNaC::fderivative](#), [GiNaC::function](#), [GiNaC::numeric](#), [GiNaC::structure< T, ComparisonPolicy >](#), and [GiNaC::symbol](#).

6.8.3.52 calchash()

```
unsigned GiNaC::basic::calchash ( ) const [protected], [virtual]
```

Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.

The method inherited from class basic computes a hash value based on the type and hash values of possible members. For this reason it is well suited for container classes but atomic classes should override this implementation because otherwise they would all end up with the same hashvalue.

Reimplemented in [GiNaC::constant](#), [GiNaC::expairseq](#), [GiNaC::function](#), [GiNaC::Gt](#), [GiNaC::idx](#), [GiNaC::numeric](#), [GiNaC::relational](#), [GiNaC::structure< T, ComparisonPolicy >](#), [GiNaC::symbol](#), [GiNaC::symmetry](#), and [GiNaC::wildcard](#).

References [GiNaC::ex::gethash\(\)](#), [GiNaC::nops\(\)](#), [GiNaC::op\(\)](#), and [GiNaC::rotate_left\(\)](#).

Referenced by [gethash\(\)](#).

6.8.3.53 print_dispatch() [1/2]

```
template<class T >
void GiNaC::basic::print_dispatch (
    const print\_context & c,
    unsigned level ) const [inline]
```

Like [print\(\)](#), but dispatch to the specified class.

Can be used by implementations of print methods to dispatch to the method of the superclass.

See also

[basic::print](#)

References [c](#), and [print_dispatch\(\)](#).

Referenced by [print_dispatch\(\)](#).

6.8.3.54 print_dispatch() [2/2]

```
void GiNaC::basic::print_dispatch (
    const registered\_class\_info & ri,
    const print\_context & c,
    unsigned level ) const
```

Like [print\(\)](#), but dispatch to the specified class.

Can be used by implementations of print methods to dispatch to the method of the superclass.

See also

[basic::print](#)

References [c](#), [GiNaC::class_info< OPT >::get_parent\(\)](#), and [GiNaC::class_info< OPT >::options](#).

6.8.3.55 archive()

```
void GiNaC::basic::archive (
    archive\_node & n ) const [virtual]
```

Save (serialize) the object into archive node.

Archive the object.

Losely speaking, this method turns an expression into a byte stream (which can be saved and restored later on, or sent via network, etc.)

Reimplemented in [GiNaC::clifford](#), [GiNaC::color](#), [GiNaC::constant](#), [GiNaC::container< C >](#), [GiNaC::expairseq](#), [GiNaC::fderivative](#), [GiNaC::function](#), [GiNaC::Gt](#), [GiNaC::idx](#), [GiNaC::varidx](#), [GiNaC::spinidx](#), [GiNaC::indexed](#), [GiNaC::integral](#), [GiNaC::matrix](#), [GiNaC::numeric](#), [GiNaC::power](#), [GiNaC::pseries](#), [GiNaC::relational](#), [GiNaC::symbol](#), [GiNaC::symmetry](#), [GiNaC::minkmetric](#), [GiNaC::tensepsilon](#), and [GiNaC::wildcard](#).

References [n](#).

6.8.3.56 read_archive()

```
void GiNaC::basic::read_archive (
    const archive_node & n,
    lst & syms ) [virtual]
```

Load (deserialize) the object from an archive node.

Construct object from [archive_node](#).

Note

This method is essentially a constructor. However, constructors can't be virtual. So, if unarchiving routines are implemented as constructors one would need to define such a constructor in every class, even if all it does is simply calling constructor of a superclass.

Reimplemented in [GiNaC::clifford](#), [GiNaC::color](#), [GiNaC::container< C >](#), [GiNaC::constant](#), [GiNaC::expairseq](#), [GiNaC::fderivative](#), [GiNaC::function](#), [GiNaC::Gt](#), [GiNaC::idx](#), [GiNaC::varidx](#), [GiNaC::spinidx](#), [GiNaC::indexed](#), [GiNaC::integral](#), [GiNaC::matrix](#), [GiNaC::numeric](#), [GiNaC::power](#), [GiNaC::pseries](#), [GiNaC::relational](#), [GiNaC::symbol](#), [GiNaC::symmetry](#), [GiNaC::minkmetric](#), [GiNaC::tensepsilon](#), and [GiNaC::wildcard](#).

6.8.3.57 subs_one_level()

```
ex GiNaC::basic::subs_one_level (
    const exmap & m,
    unsigned options ) const
```

Helper function for [subs\(\)](#).

Does not recurse into subexpressions.

References [GiNaC::ex::find\(\)](#), [m](#), [GiNaC::match\(\)](#), [options](#), and [GiNaC::ex::subs\(\)](#).

Referenced by [GiNaC::mul::algebraic_subs_mul\(\)](#), [subs\(\)](#), [GiNaC::expairseq::subs\(\)](#), [GiNaC::matrix::subs\(\)](#), [GiNaC::numeric::subs\(\)](#), [GiNaC::power::subs\(\)](#), [GiNaC::relational::subs\(\)](#), and [GiNaC::symbol::subs\(\)](#).

6.8.3.58 diff()

```
ex GiNaC::basic::diff (
    const symbol & s,
    unsigned nth = 1 ) const
```

Default interface of nth derivative [ex::diff\(s, n\)](#).

It should be called instead of [::derivative\(s\)](#) for first derivatives and for nth derivatives it just recurses down.

Parameters

<i>s</i>	symbol to differentiate in
<i>nth</i>	order of differentiation

See also

[ex::diff](#)

References [GiNaC::ex::diff\(\)](#), and [GiNaC::ex::is_zero\(\)](#).

Referenced by [GiNaC::Gt::decomposeIntegrationPath\(\)](#), and [GiNaC::mul::derivative\(\)](#).

6.8.3.59 compare()

```
int GiNaC::basic::compare (
    const basic & other ) const
```

Compare objects syntactically to establish canonical ordering.

All compare functions return: -1 for *this less than other, 0 equal, 1 greater.

References [gethash\(\)](#).

6.8.3.60 is_equal()

```
bool GiNaC::basic::is_equal (
    const basic & other ) const
```

Test for syntactic equality.

This is only a quick test, meaning objects should be in the same domain. You might have to [.expand\(\)](#), [.normal\(\)](#) objects first, depending on the domain of your computation, to get a more reliable answer.

See also

[is_equal_same_type](#)

References [gethash\(\)](#).

Referenced by [GiNaC::ncmul::coeff\(\)](#), [GiNaC::power::coeff\(\)](#), [GiNaC::ncmul::degree\(\)](#), [GiNaC::power::degree\(\)](#), [GiNaC::ncmul::ldegree\(\)](#), [GiNaC::power::ldegree\(\)](#), and [GiNaC::wildcard::match\(\)](#).

6.8.3.61 hold()

```
const basic & GiNaC::basic::hold ( ) const
```

Stop further evaluation.

6.8.3.64 clearflag()

```
const basic & GiNaC::basic::clearflag (
    unsigned f ) const [inline]
```

Clear some [status_flags](#).

References [flags](#).

Referenced by [GiNaC::add::combine_ex_with_coeff_to_pair\(\)](#), [GiNaC::mul::eval\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::function::function\(\)](#), [GiNaC::expairseq::info\(\)](#), [GiNaC::power::info\(\)](#), [map\(\)](#), [GiNaC::idx::map\(\)](#), [GiNaC::idx::replace_dim\(\)](#), [GiNaC::mul::smod\(\)](#), [GiNaC::add::split_ex_to_pair\(\)](#), [subs\(\)](#), [GiNaC::idx::subs\(\)](#), [GiNaC::spinidx::toggle_dot\(\)](#), [GiNaC::varidx::toggle_variance\(\)](#), and [GiNaC::spinidx::toggle_variance_dot\(\)](#).

6.8.3.65 ensure_if_modifiable()

```
void GiNaC::basic::ensure_if_modifiable ( ) const [protected]
```

Ensure the object may be modified without hurting others, throws if this is not the case.

Referenced by [GiNaC::matrix::division_free_elimination\(\)](#), [GiNaC::matrix::fraction_free_elimination\(\)](#), [GiNaC::matrix::gauss_elimination\(\)](#), [GiNaC::clifford::let_op\(\)](#), [GiNaC::integral::let_op\(\)](#), [GiNaC::multiple_polylog_kernel::let_op\(\)](#), [GiNaC::ELi_kernel::let_op\(\)](#), [GiNaC::Ebar_kernel::let_op\(\)](#), [GiNaC::Kronecker_dtau_kernel::let_op\(\)](#), [GiNaC::Kronecker_dz_kernel::let_op\(\)](#), [GiNaC::Eisenstein_kernel::let_op\(\)](#), [GiNaC::Eisenstein_h_kernel::let_op\(\)](#), [GiNaC::modular_form_kernel::let_op\(\)](#), [GiNaC::user_defined_kernel::let_op\(\)](#), [GiNaC::matrix::let_op\(\)](#), [GiNaC::matrix::operator\(\)](#), and [GiNaC::matrix::pivot\(\)](#).

6.8.3.66 do_print()

```
void GiNaC::basic::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

Default output to stream.

References [c](#).

6.8.3.67 do_print_tree()

```
void GiNaC::basic::do_print_tree (
    const print_tree & c,
    unsigned level ) const [protected]
```

Tree output to stream.

References [c](#), [GiNaC::nops\(\)](#), [GiNaC::op\(\)](#), and [GiNaC::ex::print\(\)](#).

6.8.3.68 do_print_python_repr()

```
void GiNaC::basic::do_print_python_repr (
    const print_python_repr & c,
    unsigned level ) const [protected]
```

Python parsable output to stream.

References [c](#).

6.8.4 Friends And Related Symbol Documentation

6.8.4.1 ex

```
friend class ex [friend]
```

Referenced by [GiNaC::matrix::charpoly\(\)](#), [GiNaC::integral::expand\(\)](#), [GiNaC::power::expand_mul\(\)](#), and [GiNaC::structure< T, ComparisonPolicy >::scalar_mul_indexed\(\)](#).

6.8.5 Member Data Documentation

6.8.5.1 flags

```
unsigned GiNaC::basic::flags [mutable], [protected]
```

of type [status_flags](#)

Referenced by [GiNaC::expairseq::calchash\(\)](#), [GiNaC::function::calchash\(\)](#), [GiNaC::Gt::calchash\(\)](#), [GiNaC::idx::calchash\(\)](#), [GiNaC::relational::calchash\(\)](#), [GiNaC::symmetry::calchash\(\)](#), [clearflag\(\)](#), [GiNaC::ex::construct_from_basic\(\)](#), [GiNaC::clifford::do_print_tree\(\)](#), [GiNaC::constant::do_print_tree\(\)](#), [GiNaC::expairseq::do_print_tree\(\)](#), [GiNaC::fderivative::do_print_tree\(\)](#), [GiNaC::idx::do_print_tree\(\)](#), [GiNaC::varidx::do_print_tree\(\)](#), [GiNaC::spinidx::do_print_tree\(\)](#), [GiNaC::indexed::do_print_tree\(\)](#), [GiNaC::numeric::do_print_tree\(\)](#), [GiNaC::pseries::do_print_tree\(\)](#), [GiNaC::symbol::do_print_tree\(\)](#), [GiNaC::symmetry::do_print_tree\(\)](#), [GiNaC::wildcard::do_print_tree\(\)](#), [GiNaC::add::eval\(\)](#), [GiNaC::expairseq::eval\(\)](#), [GiNaC::function::eval\(\)](#), [GiNaC::Gt::eval\(\)](#), [GiNaC::integral::eval\(\)](#), [GiNaC::mul::eval\(\)](#), [GiNaC::ncmul::eval\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::integral::eval_integ\(\)](#), [GiNaC::integral::expand\(\)](#), [gethash\(\)](#), [GiNaC::expairseq::info\(\)](#), [GiNaC::mul::info\(\)](#), [GiNaC::power::info\(\)](#), [operator=\(\)](#), [GiNaC::function::print\(\)](#), [setflag\(\)](#), and [~basic\(\)](#).

6.8.5.2 hashvalue

```
unsigned GiNaC::basic::hashvalue [mutable], [protected]
```

hash value

Referenced by [GiNaC::constant::calchash\(\)](#), [GiNaC::expairseq::calchash\(\)](#), [GiNaC::function::calchash\(\)](#), [GiNaC::Gt::calchash\(\)](#), [GiNaC::idx::calchash\(\)](#), [GiNaC::numeric::calchash\(\)](#), [GiNaC::relational::calchash\(\)](#), [GiNaC::symbol::calchash\(\)](#), [GiNaC::symmetry::calchash\(\)](#), [GiNaC::wildcard::calchash\(\)](#), [GiNaC::clifford::do_print_tree\(\)](#), [GiNaC::constant::do_print_tree\(\)](#), [GiNaC::expairseq::do_print_tree\(\)](#), [GiNaC::fderivative::do_print_tree\(\)](#), [GiNaC::idx::do_print_tree\(\)](#), [GiNaC::varidx::do_print_tree\(\)](#), [GiNaC::spinidx::do_print_tree\(\)](#), [GiNaC::indexed::do_print_tree\(\)](#), [GiNaC::numeric::do_print_tree\(\)](#), [GiNaC::pseries::do_print_tree\(\)](#), [GiNaC::symbol::do_print_tree\(\)](#), [GiNaC::symmetry::do_print_tree\(\)](#), [GiNaC::wildcard::do_print_tree\(\)](#), [gethash\(\)](#), [operator=\(\)](#), and [GiNaC::function::print\(\)](#).

The documentation for this class was generated from the following files:

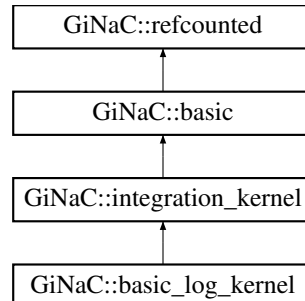
- [basic.h](#)
- [basic.cpp](#)
- [normal.cpp](#)
- [pseries.cpp](#)

6.9 GiNaC::basic_log_kernel Class Reference

The basic integration kernel with a logarithmic singularity at the origin.

```
#include <integration_kernel.h>
```

Inheritance diagram for GiNaC::basic_log_kernel:



Protected Member Functions

- `cl::cl_N series_coeff_impl` (int i) const override
For $\omega = d\lambda$ only the coefficient of λ^0 is non-zero.
- void `do_print` (const `print_context` &c, unsigned level) const

Protected Member Functions inherited from GiNaC::integration_kernel

- virtual bool `uses_Laurent_series` () const
Returns true, if the coefficients are computed from the Laurent series (in which case the method `Laurent_series` needs to be implemented).
- `ex get_numerical_value_impl` (const `ex` &lambda, const `ex` &pre, int shift, int N_trunc) const
The actual implementation for computing a numerical value for the integrand.
- void `do_print` (const `print_context` &c, unsigned level) const

Protected Member Functions inherited from GiNaC::basic

- `basic` ()
- virtual `ex eval_ncmul` (const `exvector` &v) const
- virtual bool `match_same_type` (const `basic` &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual `ex derivative` (const `symbol` &s) const
Default implementation of `ex::diff()`.
- virtual int `compare_same_type` (const `basic` &other) const
Returns order relation between two objects of same type.
- virtual bool `is_equal_same_type` (const `basic` &other) const
Returns true if two objects of same type are equal.
- virtual unsigned `calchash` () const
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- void `ensure_if_modifiable` () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void `do_print` (const `print_context` &c, unsigned level) const
Default output to stream.
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
Tree output to stream.
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
Python parsable output to stream.

Additional Inherited Members

Public Member Functions inherited from [GiNaC::integration_kernel](#)

- [ex series](#) (const [relational](#) &r, int [order](#), unsigned [options](#)=0) const override
Default implementation of [ex::series\(\)](#).
- virtual bool [has_trailing_zero](#) (void) const
This routine returns true, if the integration kernel has a trailing zero.
- virtual bool [is_numeric](#) (void) const
This routine returns true, if the integration kernel can be evaluated numerically.
- virtual [ex Laurent_series](#) (const [ex](#) &x, int [order](#)) const
Returns the Laurent series, starting possibly with the pole term.
- virtual [ex get_numerical_value](#) (const [ex](#) &lambda, int N_trunc=0) const
Evaluates the integrand at lambda.
- size_t [get_cache_size](#) (void) const
Returns the current size of the cache.
- void [set_cache_step](#) (int cache_steps) const
Sets the step size by which the cache is increased.
- [ex get_series_coeff](#) (int i) const
Wrapper around [series_coeff\(i\)](#), converts [cl_N](#) to numeric.
- [cln::cl_N series_coeff](#) (int i) const
Subclasses have either to implement [series_coeff_impl](#) or the two methods [Laurent_series](#) and [uses_Laurent_series](#).

Public Member Functions inherited from [GiNaC::basic](#)

- virtual [~basic](#) ()
basic destructor, virtual because class [ex](#) will delete objects of derived classes via a [basic](#).*
- [basic](#) (const [basic](#) &other)
- const [basic](#) & [operator=](#) (const [basic](#) &other)
basic assignment operator: the other object might be of a derived class.
- virtual [basic * duplicate](#) () const
Create a clone of this object on the heap.
- virtual [ex eval](#) () const
Perform automatic non-interruptive term rewriting rules.
- virtual [ex evalf](#) () const
Evaluate object numerically.
- virtual [ex evalm](#) () const
Evaluate sums, products and integer powers of matrices.
- virtual [ex eval_integ](#) () const
Evaluate integrals, if result is known.
- virtual [ex eval_indexed](#) (const [basic](#) &i) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void [print](#) (const [print_context](#) &c, unsigned level=0) const
Output to stream.
- virtual void [dbgprint](#) () const
Little wrapper around [print](#) to be called within a debugger.
- virtual void [dbgprinttree](#) () const
Little wrapper around [printtree](#) to be called within a debugger.
- virtual unsigned [precedence](#) () const
Return relative operator precedence (for parenthezing output).

- virtual bool `info` (unsigned inf) const
Information about the object.
- virtual size_t `nops` () const
Number of operands/members.
- virtual `ex op` (size_t i) const
Return operand/member at position i.
- virtual `ex operator[]` (const `ex` &index) const
- virtual `ex operator[]` (size_t i) const
- virtual `ex & let_op` (size_t i)
Return modifiable operand/member at position i.
- virtual `ex & operator[]` (const `ex` &index)
- virtual `ex & operator[]` (size_t i)
- virtual bool `has` (const `ex` &other, unsigned `options`=0) const
Test for occurrence of a pattern.
- virtual bool `match` (const `ex` &pattern, `exmap` &repls) const
Check whether the expression matches a given pattern.
- virtual `ex subs` (const `exmap` &m, unsigned `options`=0) const
Substitute a set of objects by arbitrary expressions.
- virtual `ex map` (`map_function` &f) const
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void `accept` (`GiNaC::visitor` &v) const
- virtual bool `is_polynomial` (const `ex` &var) const
Check whether this is a polynomial in the given variables.
- virtual int `degree` (const `ex` &s) const
Return degree of highest power in object s.
- virtual int `ldegree` (const `ex` &s) const
Return degree of lowest power in object s.
- virtual `ex coeff` (const `ex` &s, int n=1) const
Return coefficient of degree n in object s.
- virtual `ex expand` (unsigned `options`=0) const
Expand expression, i.e.
- virtual `ex collect` (const `ex` &s, bool distributed=false) const
Sort expanded expression in terms of powers of some object(s).
- virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const
Default implementation of `ex::normal()`.
- virtual `ex to_rational` (`exmap` &repl) const
Default implementation of `ex::to_rational()`.
- virtual `ex to_polynomial` (`exmap` &repl) const
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &xi) const
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient` () const
Implementation `ex::max_coefficient()`.
- virtual `exvector get_free_indices` () const
Return a vector containing the free indices of an expression.
- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
Add two indexed expressions.
- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
Multiply an indexed expression with a scalar.
- virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const
Try to contract two indexed expressions that appear in the same product.

- virtual unsigned [return_type](#) () const
- virtual [return_type_t](#) [return_type_tinfo](#) () const
- virtual [ex conjugate](#) () const
- virtual [ex real_part](#) () const
- virtual [ex imag_part](#) () const
- template<class T >
void [print_dispatch](#) (const [print_context](#) &c, unsigned level) const
Like [print\(\)](#), but dispatch to the specified class.
- void [print_dispatch](#) (const [registered_class_info](#) &ri, const [print_context](#) &c, unsigned level) const
Like [print\(\)](#), but dispatch to the specified class.
- virtual void [archive](#) ([archive_node](#) &n) const
Save (serialize) the object into archive node.
- virtual void [read_archive](#) (const [archive_node](#) &n, [lst](#) &syms)
Load (deserialize) the object from an archive node.
- [ex subs_one_level](#) (const [exmap](#) &m, unsigned [options](#)) const
Helper function for [subs\(\)](#).
- [ex diff](#) (const [symbol](#) &s, unsigned nth=1) const
Default interface of nth derivative [ex::diff\(s, n\)](#).
- int [compare](#) (const [basic](#) &other) const
Compare objects syntactically to establish canonical ordering.
- bool [is_equal](#) (const [basic](#) &other) const
Test for syntactic equality.
- const [basic](#) & [hold](#) () const
Stop further evaluation.
- unsigned [gethash](#) () const
- const [basic](#) & [setflag](#) (unsigned f) const
Set some [status_flags](#).
- const [basic](#) & [clearflag](#) (unsigned f) const
Clear some [status_flags](#).

Public Member Functions inherited from [GiNaC::refcounted](#)

- [refcounted](#) () noexcept
- unsigned int [add_reference](#) () noexcept
- unsigned int [remove_reference](#) () noexcept
- unsigned int [get_refcount](#) () const noexcept
- void [set_refcount](#) (unsigned int r) noexcept

Protected Attributes inherited from [GiNaC::integration_kernel](#)

- int [cache_step_size](#)
- std::vector< [cln::cl_N](#) > [series_vec](#)

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.9.1 Detailed Description

The basic integration kernel with a logarithmic singularity at the origin.

This class represents the differential one-form

$$L_0 = \frac{d\lambda}{\lambda}$$

6.9.2 Member Function Documentation

6.9.2.1 `series_coeff_impl()`

```
cln::cl_N GiNaC::basic_log_kernel::series_coeff_impl (
    int i ) const [override], [protected], [virtual]
```

For $\omega = d\lambda$ only the coefficient of λ^0 is non-zero.

The i-th coefficient corresponds to the power λ^{i-1} .

Reimplemented from [GiNaC::integration_kernel](#).

6.9.2.2 `do_print()`

```
void GiNaC::basic_log_kernel::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

References [c](#).

The documentation for this class was generated from the following files:

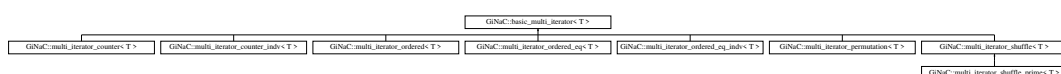
- [integration_kernel.h](#)
- [integration_kernel.cpp](#)

6.10 GiNaC::basic_multi_iterator< T > Class Template Reference

[basic_multi_iterator](#) is a base class.

```
#include <utils_multi_iterator.h>
```

Inheritance diagram for GiNaC::basic_multi_iterator< T >:



Public Member Functions

- [basic_multi_iterator](#) (void)
Default constructor.
- [basic_multi_iterator](#) (T [B](#), T [N](#), size_t [k](#))
Construct a multi_iterator with upper limit N, lower limit B and size k.
- [basic_multi_iterator](#) (T [B](#), T [N](#), const std::vector< T > &vv)
Construct from a vector.
- virtual [~basic_multi_iterator](#) ()
Destructor.
- size_t [size](#) (void) const
Returns the size of a multi_iterator.
- bool [overflow](#) (void) const
Return the overflow flag.
- const std::vector< T > & [get_vector](#) (void) const
Returns a reference to the vector v.
- T [operator\[\]](#) (size_t i) const
Subscription via [].
- T & [operator\[\]](#) (size_t i)
Subscription via [].
- T [operator\(\)](#) (size_t i) const
Subscription via ().
- T & [operator\(\)](#) (size_t i)
Subscription via ().
- virtual [basic_multi_iterator](#)< T > & [init](#) (void)
Initialize the multi-index to.
- virtual [basic_multi_iterator](#)< T > & [operator++](#) (int)
No effect for basic_multi_iterator.

Protected Attributes

- T [N](#)
- T [B](#)
- std::vector< T > [v](#)
- bool [flag_overflow](#)

Friends

- template<class TT >
std::ostream & [operator<<](#) (std::ostream &os, const [basic_multi_iterator](#)< TT > &v)

6.10.1 Detailed Description

```
template<class T>
class GiNaC::basic_multi_iterator< T >
```

[basic_multi_iterator](#) is a base class.

The base class itself does not do anything useful. A typical use of a class derived from [basic_multi_iterator](#) is

```
multi_iterator_ordered<int> k(0,4,2);

for( k.init(); !k.overflow(); k++) { std::cout << k << std::endl; }
```

which prints out

```
multi_iterator_ordered(0,1) multi_iterator_ordered(0,2) multi_iterator_ordered(0,3) multi_iterator_ordered(1,2)
multi_iterator_ordered(1,3) multi_iterator_ordered(2,3)
```

Individual components of k can be accessed with k[i] or k(i).

All classes derived from [basic_multi_iterator](#) follow the same syntax.

6.10.2 Constructor & Destructor Documentation

6.10.2.1 basic_multi_iterator() [1/3]

```
template<class T >
GiNaC::basic_multi_iterator< T >::basic_multi_iterator (
    void ) [inline]
```

Default constructor.

6.10.2.2 basic_multi_iterator() [2/3]

```
template<class T >
GiNaC::basic_multi_iterator< T >::basic_multi_iterator (
    T B,
    T N,
    size_t k ) [inline], [explicit]
```

Construct a multi_iterator with upper limit N, lower limit B and size k .

6.10.2.3 basic_multi_iterator() [3/3]

```
template<class T >
GiNaC::basic_multi_iterator< T >::basic_multi_iterator (
    T B,
    T N,
    const std::vector< T > & vv ) [inline], [explicit]
```

Construct from a vector.

6.10.2.4 ~basic_multi_iterator()

```
template<class T >
GiNaC::basic_multi_iterator< T >::~~basic_multi_iterator ( ) [inline], [virtual]
```

Destructor.

6.10.3 Member Function Documentation

6.10.3.1 size()

```
template<class T >
size_t GiNaC::basic_multi_iterator< T >::size (
    void ) const [inline]
```

Returns the size of a multi_iterator.

Referenced by [GiNaC::multi_iterator_shuffle< T >::multi_iterator_shuffle\(\)](#), [GiNaC::multi_iterator_shuffle< T >::operator++\(\)](#), [GiNaC::operator<<\(\)](#), [GiNaC::operator<<\(\)](#), [GiNaC::operator<<\(\)](#), [GiNaC::operator<<\(\)](#), [GiNaC::operator<<\(\)](#), [GiNaC::operator<<\(\)](#), [GiNaC::operator<<\(\)](#), [GiNaC::operator<<\(\)](#), and [GiNaC::operator<<\(\)](#).

6.10.3.2 overflow()

```
template<class T >
bool GiNaC::basic_multi_iterator< T >::overflow (
    void ) const [inline]
```

Return the overflow flag.

6.10.3.3 get_vector()

```
template<class T >
const std::vector< T > & GiNaC::basic_multi_iterator< T >::get_vector (
    void ) const [inline]
```

Returns a reference to the vector v.

6.10.3.4 operator[]() [1/2]

```
template<class T >
T GiNaC::basic_multi_iterator< T >::operator[] (
    size_t i ) const [inline]
```

Subscription via [].

6.10.3.5 operator[]() [2/2]

```
template<class T >
T & GiNaC::basic_multi_iterator< T >::operator[] (
    size_t i ) [inline]
```

Subscription via [].

6.10.3.6 operator()() [1/2]

```
template<class T >
T GiNaC::basic_multi_iterator< T >::operator() (
    size_t i ) const [inline]
```

Subscription via ()

6.10.3.7 operator()() [2/2]

```
template<class T >
T & GiNaC::basic_multi_iterator< T >::operator() (
    size_t i ) [inline]
```

Subscription via ()

6.10.3.8 init()

```
template<class T >
basic_multi_iterator< T > & GiNaC::basic_multi_iterator< T >::init (
    void ) [inline], [virtual]
```

Initialize the multi-index to.

$$(n_1, n_2, n_3, \dots, n_k) = (B, B, \dots, B)$$

Reimplemented in [GiNaC::multi_iterator_ordered< T >](#), [GiNaC::multi_iterator_ordered_eq< T >](#), [GiNaC::multi_iterator_ordered_eq< T >](#), [GiNaC::multi_iterator_counter< T >](#), [GiNaC::multi_iterator_counter_indv< T >](#), [GiNaC::multi_iterator_permutation< T >](#), [GiNaC::multi_iterator_shuffle< T >](#), and [GiNaC::multi_iterator_shuffle_prime< T >](#).

6.10.3.9 operator++()

```
template<class T >
basic_multi_iterator< T > & GiNaC::basic_multi_iterator< T >::operator++ (
    int ) [inline], [virtual]
```

No effect for [basic_multi_iterator](#).

Reimplemented in [GiNaC::multi_iterator_ordered< T >](#), [GiNaC::multi_iterator_ordered_eq< T >](#), [GiNaC::multi_iterator_ordered_eq< T >](#), [GiNaC::multi_iterator_counter< T >](#), [GiNaC::multi_iterator_counter_indv< T >](#), [GiNaC::multi_iterator_permutation< T >](#), and [GiNaC::multi_iterator_shuffle< T >](#).

6.10.4 Friends And Related Symbol Documentation

6.10.4.1 operator<<

```
template<class T >
template<class TT >
std::ostream & operator<< (
    std::ostream & os,
    const basic_multi_iterator< TT > & v ) [friend]
```

6.10.5 Member Data Documentation

6.10.5.1 N

```
template<class T >
T GiNaC::basic_multi_iterator< T >::N [protected]
```

6.10.5.2 B

```
template<class T >
T GiNaC::basic_multi_iterator< T >::B [protected]
```

Referenced by [GiNaC::multi_iterator_shuffle< T >::multi_iterator_shuffle\(\)](#), [GiNaC::operator<<\(\)](#), [GiNaC::operator<<\(\)](#), [GiNaC::operator<<\(\)](#), [GiNaC::operator<<\(\)](#), [GiNaC::operator<<\(\)](#), [GiNaC::operator<<\(\)](#), [GiNaC::operator<<\(\)](#), [GiNaC::operator<<\(\)](#), and [GiNaC::operator<<\(\)](#).

6.10.5.3 v

```
template<class T >
std::vector<T> GiNaC::basic_multi_iterator< T >::v [protected]
```

Referenced by [GiNaC::multi_iterator_shuffle< T >::multi_iterator_shuffle\(\)](#).

6.10.5.4 flag_overflow

```
template<class T >
bool GiNaC::basic_multi_iterator< T >::flag_overflow [protected]
```

The documentation for this class was generated from the following file:

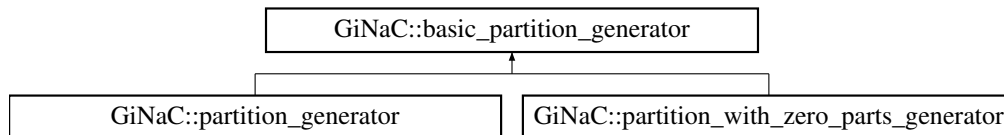
- [utils_multi_iterator.h](#)

6.11 GiNaC::basic_partition_generator Class Reference

Base class for generating all bounded combinatorial partitions of an integer n with exactly m parts in non-decreasing order.

```
#include <utils.h>
```

Inheritance diagram for GiNaC::basic_partition_generator:



Classes

- struct [mpartition2](#)

Protected Member Functions

- [basic_partition_generator](#) (unsigned n _, unsigned m _)

Protected Attributes

- [mpartition2](#) `mpgen`

6.11.1 Detailed Description

Base class for generating all bounded combinatorial partitions of an integer n with exactly m parts in non-decreasing order.

6.11.2 Constructor & Destructor Documentation

6.11.2.1 basic_partition_generator()

```
GiNaC::basic_partition_generator::basic_partition_generator (
    unsigned  $n$ _,
    unsigned  $m$ _ ) [inline], [protected]
```

6.11.3 Member Data Documentation

6.11.3.1 mpmgen

```
mpartition2 GiNaC::basic_partition_generator::mpgen [protected]
```

Referenced by [GiNaC::partition_with_zero_parts_generator::get\(\)](#), [GiNaC::partition_generator::get\(\)](#), [GiNaC::partition_with_zero_parts_generator::next\(\)](#), and [GiNaC::partition_generator::next\(\)](#).

The documentation for this class was generated from the following file:

- [utils.h](#)

6.12 GiNaC::class_info< OPT > Class Template Reference

```
#include <class_info.h>
```

Classes

- struct [tree_node](#)

Public Member Functions

- [class_info](#) (const OPT &o)
- [class_info](#) * [get_parent](#) () const
Get pointer to [class_info](#) of parent class (or nullptr).

Static Public Member Functions

- static const [class_info](#) * [find](#) (const std::string &class_name)
Find [class_info](#) by name.
- static void [dump_hierarchy](#) (bool verbose=false)
Dump class hierarchy to std::cout.

Public Attributes

- OPT [options](#)

Static Private Member Functions

- static void [dump_tree](#) ([tree_node](#) *n, const std::string &prefix, bool verbose)
- static void [identify_parents](#) ()

Private Attributes

- [class_info](#) * [next](#)
- [class_info](#) * [parent](#)

Static Private Attributes

- static [class_info](#) * [first](#) = nullptr
- static bool [parents_identified](#) = false

6.12.1 Constructor & Destructor Documentation

6.12.1.1 class_info()

```
template<class OPT >
GiNaC::class_info< OPT >::class_info (
    const OPT & o ) [inline]
```

References [GiNaC::class_info< OPT >::first](#), and [GiNaC::class_info< OPT >::parents_identified](#).

6.12.2 Member Function Documentation

6.12.2.1 `get_parent()`

```
template<class OPT >
class_info * GiNaC::class_info< OPT >::get_parent ( ) const [inline]
```

Get pointer to `class_info` of parent class (or nullptr).

References `GiNaC::class_info< OPT >::identify_parents()`, and `GiNaC::class_info< OPT >::parent`.

Referenced by `GiNaC::class_info< OPT >::dump_hierarchy()`, `GiNaC::function::print()`, and `GiNaC::basic::print_dispatch()`.

6.12.2.2 `find()`

```
template<class OPT >
const class_info< OPT > * GiNaC::class_info< OPT >::find (
    const std::string & class_name ) [static]
```

Find `class_info` by name.

References `GiNaC::class_info< OPT >::find()`, `GiNaC::class_info< OPT >::next`, and `GiNaC::class_info< OPT >::options`.

Referenced by `GiNaC::class_info< OPT >::find()`.

6.12.2.3 `dump_hierarchy()`

```
template<class OPT >
void GiNaC::class_info< OPT >::dump_hierarchy (
    bool verbose = false ) [static]
```

Dump class hierarchy to `std::cout`.

References `GiNaC::class_info< OPT >::get_parent()`, `GiNaC::class_info< OPT >::next`, and `GiNaC::tree()`.

6.12.2.4 `dump_tree()`

```
template<class OPT >
void GiNaC::class_info< OPT >::dump_tree (
    tree_node * n,
    const std::string & prefix,
    bool verbose ) [static], [private]
```

References `n`, and `GiNaC::class_info< OPT >::options`.

6.12.2.5 `identify_parents()`

```
template<class OPT >
void GiNaC::class_info< OPT >::identify_parents ( ) [static], [private]
```

References `GiNaC::class_info< OPT >::next`.

Referenced by `GiNaC::class_info< OPT >::get_parent()`.

6.12.3 Member Data Documentation

6.12.3.1 options

```
template<class OPT >
OPT GiNaC::class_info< OPT >::options
```

Referenced by [GiNaC::class_info< OPT >::dump_tree\(\)](#), [GiNaC::class_info< OPT >::find\(\)](#), [GiNaC::function::print\(\)](#), [GiNaC::basic::print_dispatch\(\)](#), and [GiNaC::set_print_func\(\)](#).

6.12.3.2 first

```
template<class OPT >
class_info< OPT > * GiNaC::class_info< OPT >::first = nullptr [static], [private]
```

Referenced by [GiNaC::class_info< OPT >::class_info\(\)](#).

6.12.3.3 next

```
template<class OPT >
class_info* GiNaC::class_info< OPT >::next [private]
```

Referenced by [GiNaC::class_info< OPT >::dump_hierarchy\(\)](#), [GiNaC::class_info< OPT >::find\(\)](#), and [GiNaC::class_info< OPT >::i](#).

6.12.3.4 parent

```
template<class OPT >
class_info* GiNaC::class_info< OPT >::parent [mutable], [private]
```

Referenced by [GiNaC::class_info< OPT >::get_parent\(\)](#).

6.12.3.5 parents_identified

```
template<class OPT >
bool GiNaC::class_info< OPT >::parents_identified = false [static], [private]
```

Referenced by [GiNaC::class_info< OPT >::class_info\(\)](#).

The documentation for this class was generated from the following file:

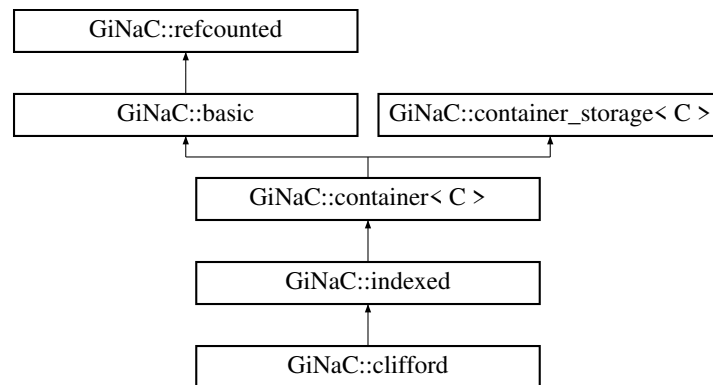
- [class_info.h](#)

6.13 GiNaC::clifford Class Reference

This class holds an object representing an element of the Clifford algebra (the Dirac gamma matrices).

```
#include <clifford.h>
```

Inheritance diagram for GiNaC::clifford:



Public Member Functions

- `clifford` (const `ex` &b, unsigned char rl=0)
Construct object without any indices.
- `clifford` (const `ex` &b, const `ex` &mu, const `ex` &metr, unsigned char rl=0, int comm_sign=-1)
Construct object with one Lorentz index.
- `clifford` (unsigned char rl, const `ex` &metr, int comm_sign, const `exvector` &v)
- `clifford` (unsigned char rl, const `ex` &metr, int comm_sign, `exvector` &&v)
- unsigned `precedence` () const override
Return relative operator precedence (for parenthezing output).
- void `archive` (`archive_node` &n) const override
Save (serialize) the object into archive node.
- void `read_archive` (const `archive_node` &n, `lst` &sym_lst) override
Load (deserialize) the object from an archive node.
- unsigned char `get_representation_label` () const
- `ex` `get_metric` () const
- virtual `ex` `get_metric` (const `ex` &i, const `ex` &j, bool symmetrised=false) const
- bool `same_metric` (const `ex` &other) const
- int `get_commutator_sign` () const
- size_t `nops` () const override
Number of operands/members.
- `ex` `op` (size_t i) const override
Return operand/member at position i.
- `ex` & `let_op` (size_t i) override
Return modifiable operand/member at position i.
- `ex` `subs` (const `exmap` &m, unsigned `options`=0) const override
Substitute a set of objects by arbitrary expressions.

Public Member Functions inherited from GiNaC::indexed

- `indexed` (const `ex` &b)
Construct indexed object with no index.
- `indexed` (const `ex` &b, const `ex` &i1)
Construct indexed object with one index.
- `indexed` (const `ex` &b, const `ex` &i1, const `ex` &i2)
Construct indexed object with two indices.
- `indexed` (const `ex` &b, const `ex` &i1, const `ex` &i2, const `ex` &i3)
Construct indexed object with three indices.
- `indexed` (const `ex` &b, const `ex` &i1, const `ex` &i2, const `ex` &i3, const `ex` &i4)
Construct indexed object with four indices.
- `indexed` (const `ex` &b, const `symmetry` &symm, const `ex` &i1, const `ex` &i2)
Construct indexed object with two indices and a specified symmetry.
- `indexed` (const `ex` &b, const `symmetry` &symm, const `ex` &i1, const `ex` &i2, const `ex` &i3)
Construct indexed object with three indices and a specified symmetry.
- `indexed` (const `ex` &b, const `symmetry` &symm, const `ex` &i1, const `ex` &i2, const `ex` &i3, const `ex` &i4)
Construct indexed object with four indices and a specified symmetry.
- `indexed` (const `ex` &b, const `exvector` &iv)
Construct indexed object with a specified vector of indices.
- `indexed` (const `ex` &b, const `symmetry` &symm, const `exvector` &iv)
Construct indexed object with a specified vector of indices and symmetry.
- `indexed` (const `symmetry` &symm, const `exprseq` &es)
- `indexed` (const `symmetry` &symm, const `exvector` &v)
- `indexed` (const `symmetry` &symm, `exvector` &&v)
- unsigned `precedence` () const override
Return relative operator precedence (for parenthezing output).
- bool `info` (unsigned inf) const override
Information about the object.
- `ex eval` () const override
Perform automatic non-interruptive term rewriting rules.
- `ex real_part` () const override
- `ex imag_part` () const override
- `exvector get_free_indices` () const override
Return a vector containing the free indices of an expression.
- void `archive` (`archive_node` &n) const override
Save (a.k.a.
- void `read_archive` (const `archive_node` &n, `lst` &symbols) override
Read (a.k.a.
- bool `all_index_values_are` (unsigned inf) const
Check whether all index values have a certain property.
- `exvector get_indices` () const
Return a vector containing the object's indices.
- `exvector get_dummy_indices` () const
Return a vector containing the dummy indices of the object, if any.
- `exvector get_dummy_indices` (const `indexed` &other) const
Return a vector containing the dummy indices in the contraction with another indexed object.
- bool `has_dummy_index_for` (const `ex` &i) const
Check whether the object has an index that forms a dummy index pair with a given index.
- `ex get_symmetry` () const
Return symmetry properties.

Public Member Functions inherited from `GiNaC::container< C >`

- `container` (STLT const &s)
- `container` (STLT &&v)
- `container` (exvector::const_iterator b, exvector::const_iterator e)
- `container` (std::initializer_list< ex > il)
- `size_t nops` () const override
Number of operands/members.
- `ex op` (size_t i) const override
Return operand/member at position i.
- `ex & let_op` (size_t i) override
Return modifiable operand/member at position i.
- `ex subs` (const exmap &m, unsigned options=0) const override
Substitute a set of objects by arbitrary expressions.
- `container & prepend` (const ex &b)
Add element at front.
- `container & append` (const ex &b)
Add element at back.
- `container & remove_first` ()
Remove first element.
- `container & remove_last` ()
Remove last element.
- `container & remove_all` ()
Remove all elements.
- `container & sort` ()
Sort elements.
- `container & unique` ()
Remove adjacent duplicate elements.
- `const_iterator begin` () const
- `const_iterator end` () const
- `const_reverse_iterator rbegin` () const
- `const_reverse_iterator rend` () const

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic` ()
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic` (const basic &other)
- `const basic & operator=` (const basic &other)
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate` () const
Create a clone of this object on the heap.
- virtual `ex evalf` () const
Evaluate object numerically.
- virtual `ex evalm` () const
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ` () const
Evaluate integrals, if result is known.
- virtual `ex eval_indexed` (const basic &i) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void `print` (const print_context &c, unsigned level=0) const

- *Output to stream.*
virtual void `dbgprint` () const
- *Little wrapper around print to be called within a debugger.*
virtual void `dbgprinttree` () const
- *Little wrapper around printtree to be called within a debugger.*
virtual `ex operator[]` (const `ex` &index) const
- virtual `ex operator[]` (size_t i) const
- virtual `ex & operator[]` (const `ex` &index)
- virtual `ex & operator[]` (size_t i)
- virtual bool `has` (const `ex` &other, unsigned `options`=0) const
Test for occurrence of a pattern.
- virtual bool `match` (const `ex` &pattern, `exmap` &repls) const
Check whether the expression matches a given pattern.
- virtual `ex map` (`map_function` &f) const
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void `accept` (GiNaC::visitor &v) const
- virtual bool `is_polynomial` (const `ex` &var) const
Check whether this is a polynomial in the given variables.
- virtual int `degree` (const `ex` &s) const
Return degree of highest power in object s.
- virtual int `ldegree` (const `ex` &s) const
Return degree of lowest power in object s.
- virtual `ex coeff` (const `ex` &s, int `n`=1) const
Return coefficient of degree n in object s.
- virtual `ex collect` (const `ex` &s, bool `distributed`=false) const
Sort expanded expression in terms of powers of some object(s).
- virtual `ex series` (const `relational` &r, int `order`, unsigned `options`=0) const
Default implementation of `ex::series()`.
- virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const
Default implementation of `ex::normal()`.
- virtual `ex to_rational` (`exmap` &repl) const
Default implementation of `ex::to_rational()`.
- virtual `ex to_polynomial` (`exmap` &repl) const
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &xi) const
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient` () const
Implementation `ex::max_coefficient()`.
- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
Add two indexed expressions.
- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
Multiply an indexed expression with a scalar.
- virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const
Try to contract two indexed expressions that appear in the same product.
- template<class T >
void `print_dispatch` (const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- `ex subs_one_level` (const `exmap` &m, unsigned `options`) const
Helper function for `subs()`.

- `ex diff` (const `symbol` &s, unsigned nth=1) const
Default interface of nth derivative `ex::diff(s, n)`.
- int `compare` (const `basic` &other) const
Compare objects syntactically to establish canonical ordering.
- bool `is_equal` (const `basic` &other) const
Test for syntactic equality.
- const `basic` & `hold` () const
Stop further evaluation.
- unsigned `gethash` () const
- const `basic` & `setflag` (unsigned f) const
Set some `status_flags`.
- const `basic` & `clearflag` (unsigned f) const
Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Protected Member Functions

- `ex eval_ncmul` (const `exvector` &v) const override
Perform automatic simplification on noncommutative product of clifford objects.
- bool `match_same_type` (const `basic` &other) const override
Returns true if the attributes of two objects are similar enough for a match.
- `ex thiscontainer` (const `exvector` &v) const override
- `ex thiscontainer` (`exvector` &&v) const override
- unsigned `return_type` () const override
- `return_type_t` `return_type_tinfo` () const override
- void `do_print_dflit` (const `print_dflit` &c, unsigned level) const
- void `do_print_latex` (const `print_latex` &c, unsigned level) const
- void `do_print_tree` (const `print_tree` &c, unsigned level) const

Protected Member Functions inherited from `GiNaC::indexed`

- `ex derivative` (const `symbol` &s) const override
Implementation of `ex::diff()` for an indexed object always returns 0.
- `ex thiscontainer` (const `exvector` &v) const override
- `ex thiscontainer` (`exvector` &&v) const override
- unsigned `return_type` () const override
- `return_type_t` `return_type_tinfo` () const override
- `ex expand` (unsigned `options`=0) const override
Expand expression, i.e.
- void `printindices` (const `print_context` &c, unsigned level) const
- void `print_indexed` (const `print_context` &c, const char *openbrace, const char *closebrace, unsigned level) const
- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_latex` (const `print_latex` &c, unsigned level) const
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
- void `validate` () const
Check whether all indices are of class `idx` and validate the symmetry tree.

Protected Member Functions inherited from [GiNaC::container< C >](#)

- [ex conjugate](#) () const override
- bool [is_equal_same_type](#) (const [basic](#) &other) const override
Returns true if two objects of same type are equal.
- virtual [ex thiscontainer](#) (const [STLT](#) &v) const
Similar to [duplicate\(\)](#), but with a preset sequence.
- virtual [ex thiscontainer](#) ([STLT](#) &&v) const
Similar to [duplicate\(\)](#), but with a preset sequence (which gets pilfered).
- virtual void [printseq](#) (const [print_context](#) &c, char openbracket, char delim, char closebracket, unsigned this←_precedence, unsigned upper_precedence=0) const
Print sequence of contained elements.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
- void [do_print_python](#) (const [print_python](#) &c, unsigned level) const
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
- [STLT](#) [subchildren](#) (const [exmap](#) &m, unsigned [options](#)=0) const

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Protected Member Functions inherited from [GiNaC::container_storage< C >](#)

- [container_storage](#) ()
- [container_storage](#) (size_t n, const [ex](#) &e)
- [container_storage](#) (std::initializer_list< [ex](#) > il)
- template<class In >
 [container_storage](#) (In b, In e)
- void [reserve](#) (size_t)
- [~container_storage](#) ()
- void [reserve](#) (size_t n)
- void [reserve](#) (std::vector< [ex](#) > &v, size_t n)

Protected Attributes

- unsigned char [representation_label](#)
Representation label to distinguish independent spin lines.
- [ex metric](#)
Metric of the space, all constructors make it an indexed object.
- int [commutator_sign](#)
It is the sign in the definition $e_{\sim i} e_{\sim j} \pm e_{\sim j} e_{\sim i} = B(i, j) + B(j, i)$

Protected Attributes inherited from [GiNaC::indexed](#)

- [ex symtree](#)
Index symmetry (tree of symmetry objects)

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

Protected Attributes inherited from [GiNaC::container_storage< C >](#)

- [STLT seq](#)

Additional Inherited Members

Public Types inherited from [GiNaC::container< C >](#)

- typedef [STLT::const_iterator](#) [const_iterator](#)
- typedef [STLT::const_reverse_iterator](#) [const_reverse_iterator](#)

Protected Types inherited from [GiNaC::container< C >](#)

- typedef [container_storage< C >::STLT](#) [STLT](#)

Protected Types inherited from [GiNaC::container_storage< C >](#)

- typedef [C< ex >](#) [STLT](#)

Static Protected Member Functions inherited from [GiNaC::container< C >](#)

- static unsigned [get_default_flags](#) ()
Specialization of [container::get_default_flags\(\)](#) for [lst](#).
- static char [get_open_delim](#) ()
Specialization of [container::get_open_delim\(\)](#) for [lst](#).
- static char [get_close_delim](#) ()
Specialization of [container::get_close_delim\(\)](#) for [lst](#).

Static Protected Member Functions inherited from [GiNaC::container_storage< C >](#)

- static void [reserve](#) ([STLT](#) &, [size_t](#))

6.13.1 Detailed Description

This class holds an object representing an element of the Clifford algebra (the Dirac gamma matrices).

These objects only carry Lorentz indices. Spinor indices are hidden. A representation label (an unsigned 8-bit integer) is used to distinguish elements from different Clifford algebras (objects with different labels commute).

6.13.2 Constructor & Destructor Documentation

6.13.2.1 clifford() [1/4]

```
GiNaC::clifford::clifford (
    const ex & b,
    unsigned char rl = 0 )
```

Construct object without any indices.

This constructor is for internal use only. Use the [dirac_ONE\(\)](#) function instead.

See also

[dirac_ONE](#)

6.13.2.2 clifford() [2/4]

```
GiNaC::clifford::clifford (
    const ex & b,
    const ex & mu,
    const ex & metr,
    unsigned char rl = 0,
    int comm_sign = -1 )
```

Construct object with one Lorentz index.

This constructor is for internal use only. Use the [clifford_unit\(\)](#) or [dirac_gamma\(\)](#) functions instead.

See also

[clifford_unit](#)

[dirac_gamma](#)

References [GINAC_ASSERT](#).

6.13.2.3 clifford() [3/4]

```
GiNaC::clifford::clifford (
    unsigned char rl,
    const ex & metr,
    int comm_sign,
    const exvector & v )
```

6.13.2.4 clifford() [4/4]

```
GiNaC::clifford::clifford (
    unsigned char rl,
    const ex & metr,
    int comm_sign,
    exvector && v )
```

6.13.3 Member Function Documentation

6.13.3.1 precedence()

```
unsigned GiNaC::clifford::precedence ( ) const [inline], [override], [virtual]
```

Return relative operator precedence (for parenthezing output).

Reimplemented from [GiNaC::basic](#).

Referenced by [do_print_dflt\(\)](#), and [do_print_latex\(\)](#).

6.13.3.2 archive()

```
void GiNaC::clifford::archive (
    archive_node & n ) const [override], [virtual]
```

Save (serialize) the object into archive node.

Archive the object.

Losely speaking, this method turns an expression into a byte stream (which can be saved and restored later on, or sent via network, etc.)

Reimplemented from [GiNaC::basic](#).

References [commutator_sign](#), [metric](#), [n](#), and [representation_label](#).

6.13.3.3 read_archive()

```
void GiNaC::clifford::read_archive (
    const archive_node & n,
    lst & syms ) [override], [virtual]
```

Load (deserialize) the object from an archive node.

Construct object from [archive_node](#).

Note

This method is essentially a constructor. However, constructors can't be virtual. So, if unarchiving routines are implemented as constructors one would need to define such a constructor in every class, even if all it does is simply calling constructor of a superclass.

Reimplemented from [GiNaC::basic](#).

References [commutator_sign](#), [metric](#), [n](#), and [representation_label](#).

6.13.3.4 eval_ncmul()

```
ex GiNaC::clifford::eval_ncmul (
    const exvector & v ) const [override], [protected], [virtual]
```

Perform automatic simplification on noncommutative product of clifford objects.

This removes superfluous ONEs, permutes gamma5/L/R's to the front and removes squares of gamma objects.

Reimplemented from [GiNaC::basic](#).

6.13.3.5 match_same_type()

```
bool GiNaC::clifford::match_same_type (
    const basic & other ) const [override], [protected], [virtual]
```

Returns true if the attributes of two objects are similar enough for a match.

This function must not match subexpressions (this is already done by [basic::match\(\)](#)). Only attributes not accessible by [op\(\)](#) should be compared. This is also the reason why this function doesn't take the wildcard replacement list from [match\(\)](#) as an argument: only subexpressions are subject to wildcard matches. Also, this function only needs to be implemented for container classes because [is_equal_same_type\(\)](#) is automatically used instead of [match_same_type\(\)](#) if [nops\(\)](#) == 0.

See also

[basic::match](#)

Reimplemented from [GiNaC::basic](#).

References [commutator_sign](#), [get_commutator_sign\(\)](#), [GINAC_ASSERT](#), [representation_label](#), and [same_metric\(\)](#).

6.13.3.6 thiscontainer() [1/2]

```
ex GiNaC::clifford::thiscontainer (
    const exvector & v ) const [override], [protected]
```

6.13.3.7 thiscontainer() [2/2]

```
ex GiNaC::clifford::thiscontainer (
    exvector && v ) const [override], [protected]
```

6.13.3.8 return_type()

```
unsigned GiNaC::clifford::return_type ( ) const [inline], [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::return_types::noncommutative](#).

6.13.3.9 return_type_tinfo()

```
return_type_t GiNaC::clifford::return_type_tinfo ( ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [representation_label](#).

6.13.3.10 get_representation_label()

```
unsigned char GiNaC::clifford::get_representation_label ( ) const [inline]
```

References [representation_label](#).

6.13.3.11 get_metric() [1/2]

```
ex GiNaC::clifford::get_metric ( ) const [inline]
```

References [metric](#).

Referenced by [same_metric\(\)](#).

6.13.3.12 get_metric() [2/2]

```
ex GiNaC::clifford::get_metric (
    const ex & i,
    const ex & j,
    bool symmetrised = false ) const [virtual]
```

References [GiNaC::_ex1_2](#), [GiNaC::ex::get_free_indices\(\)](#), [GiNaC::indexed::get_symmetry\(\)](#), [metric](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::ex::op\(\)](#), [GiNaC::indexed::simplify_indexed](#), [GiNaC::ex::subs\(\)](#), and [GiNaC::symmetric2\(\)](#).

6.13.3.13 same_metric()

```
bool GiNaC::clifford::same_metric (
    const ex & other ) const
```

References [GiNaC::ex::get_free_indices\(\)](#), [get_metric\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::ex::op\(\)](#), [op\(\)](#), and [GiNaC::indexed::simplify_indexed](#).

Referenced by [match_same_type\(\)](#).

6.13.3.14 get_commutator_sign()

```
int GiNaC::clifford::get_commutator_sign ( ) const [inline]
```

References [commutator_sign](#).

Referenced by [match_same_type\(\)](#).

6.13.3.15 nops()

```
size_t GiNaC::clifford::nops ( ) const [inline], [override], [virtual]
```

Number of operands/members.

Reimplemented from [GiNaC::basic](#).

Referenced by [let_op\(\)](#), and [op\(\)](#).

6.13.3.16 op()

```
ex GiNaC::clifford::op (
    size_t i ) const [override], [virtual]
```

Return operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [GINAC_ASSERT](#), [nops\(\)](#), and [representation_label](#).

Referenced by [same_metric\(\)](#).

6.13.3.17 let_op()

```
ex & GiNaC::clifford::let_op (
    size_t i ) [override], [virtual]
```

Return modifiable operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::basic::ensure_if_modifiable\(\)](#), [GINAC_ASSERT](#), [nops\(\)](#), and [representation_label](#).

6.13.3.18 subs()

```
ex GiNaC::clifford::subs (
    const exmap & m,
    unsigned options = 0 ) const [override], [virtual]
```

Substitute a set of objects by arbitrary expressions.

The ex returned will already be evaluated.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::are_ex_trivially_equal\(\)](#), [c](#), [m](#), [options](#), and [GiNaC::ex::subs\(\)](#).

6.13.3.19 do_print_dflt()

```
void GiNaC::clifford::do_print_dflt (
    const print\_dflt & c,
    unsigned level ) const [protected]
```

References [c](#), [GiNaC::is_dirac_slash\(\)](#), [precedence\(\)](#), [GiNaC::indexed::printindices\(\)](#), [representation_label](#), and [GiNaC::container_storage< C >::seq](#).

6.13.3.20 do_print_latex()

```
void GiNaC::clifford::do_print_latex (
    const print\_latex & c,
    unsigned level ) const [protected]
```

References [c](#), [GiNaC::is_dirac_slash\(\)](#), [precedence\(\)](#), [representation_label](#), and [GiNaC::container_storage< C >::seq](#).

6.13.3.21 do_print_tree()

```
void GiNaC::clifford::do_print_tree (
    const print\_tree & c,
    unsigned level ) const [protected]
```

References [c](#), [GiNaC::basic::flags](#), [GiNaC::basic::hashvalue](#), [metric](#), [GiNaC::ex::print\(\)](#), [GiNaC::indexed::printindices\(\)](#), [GiNaC::container_storage< C >::seq](#), and [GiNaC::indexed::symtree](#).

6.13.4 Member Data Documentation

6.13.4.1 representation_label

```
unsigned char GiNaC::clifford::representation_label [protected]
```

Representation label to distinguish independent spin lines.

Referenced by [archive\(\)](#), [do_print_dflt\(\)](#), [do_print_latex\(\)](#), [get_representation_label\(\)](#), [let_op\(\)](#), [match_same_type\(\)](#), [op\(\)](#), [read_archive\(\)](#), and [return_type_tinfo\(\)](#).

6.13.4.2 metric

```
ex GiNaC::clifford::metric [protected]
```

Metric of the space, all constructors make it an indexed object.

Referenced by [archive\(\)](#), [do_print_tree\(\)](#), [get_metric\(\)](#), [get_metric\(\)](#), and [read_archive\(\)](#).

6.13.4.3 commutator_sign

```
int GiNaC::clifford::commutator_sign [protected]
```

It is the sign in the definition $e_i e_j \pm e_j e_i = B(i, j) + B(j, i)$

Referenced by [archive\(\)](#), [get_commutator_sign\(\)](#), [match_same_type\(\)](#), and [read_archive\(\)](#).

The documentation for this class was generated from the following files:

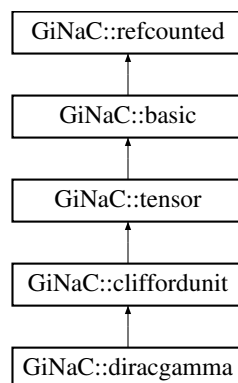
- [clifford.h](#)
- [clifford.cpp](#)

6.14 GiNaC::cliffordunit Class Reference

This class represents the Clifford algebra generators (units).

```
#include <clifford.h>
```

Inheritance diagram for GiNaC::cliffordunit:



Public Member Functions

- bool [contract_with](#) (exvector::iterator self, exvector::iterator other, [exvector](#) &v) const override
Contraction of a Clifford unit with something else.

Public Member Functions inherited from [GiNaC::tensor](#)

- bool [replace_contr_index](#) (exvector::iterator self, exvector::iterator other) const
Replace dummy index in contracted-with object by the contracting object's second index (used internally for delta and metric tensor contractions).

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic ()`
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic (const basic &other)`
- `const basic & operator= (const basic &other)`
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate () const`
Create a clone of this object on the heap.
- virtual `ex eval () const`
Perform automatic non-interruptive term rewriting rules.
- virtual `ex evalf () const`
Evaluate object numerically.
- virtual `ex evalm () const`
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ () const`
Evaluate integrals, if result is known.
- virtual `ex eval_indexed (const basic &i) const`
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual `void print (const print_context &c, unsigned level=0) const`
Output to stream.
- virtual `void dbgprint () const`
Little wrapper around print to be called within a debugger.
- virtual `void dbgprinttree () const`
Little wrapper around printtree to be called within a debugger.
- virtual `unsigned precedence () const`
Return relative operator precedence (for parenthezing output).
- virtual `bool info (unsigned inf) const`
Information about the object.
- virtual `size_t nops () const`
Number of operands/members.
- virtual `ex op (size_t i) const`
Return operand/member at position i.
- virtual `ex operator[] (const ex &index) const`
- virtual `ex operator[] (size_t i) const`
- virtual `ex & let_op (size_t i)`
Return modifiable operand/member at position i.
- virtual `ex & operator[] (const ex &index)`
- virtual `ex & operator[] (size_t i)`
- virtual `bool has (const ex &other, unsigned options=0) const`
Test for occurrence of a pattern.
- virtual `bool match (const ex &pattern, exmap &repls) const`
Check whether the expression matches a given pattern.
- virtual `ex subs (const exmap &m, unsigned options=0) const`
Substitute a set of objects by arbitrary expressions.
- virtual `ex map (map_function &f) const`
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual `void accept (GiNaC::visitor &v) const`
- virtual `bool is_polynomial (const ex &var) const`
Check whether this is a polynomial in the given variables.
- virtual `int degree (const ex &s) const`

- Return degree of highest power in object s.*

 - virtual int [ldegree](#) (const [ex](#) &s) const
- Return degree of lowest power in object s.*

 - virtual [ex coeff](#) (const [ex](#) &s, int n=1) const
- Return coefficient of degree n in object s.*

 - virtual [ex expand](#) (unsigned [options](#)=0) const
- Expand expression, i.e.*

 - virtual [ex collect](#) (const [ex](#) &s, bool distributed=false) const
- Sort expanded expression in terms of powers of some object(s).*

 - virtual [ex series](#) (const [relational](#) &r, int [order](#), unsigned [options](#)=0) const
- Default implementation of [ex::series\(\)](#).*

 - virtual [ex normal](#) ([exmap](#) &repl, [exmap](#) &rev_lookup, [lst](#) &modifier) const
- Default implementation of [ex::normal\(\)](#).*

 - virtual [ex to_rational](#) ([exmap](#) &repl) const
- Default implementation of [ex::to_rational\(\)](#).*

 - virtual [ex to_polynomial](#) ([exmap](#) &repl) const
- virtual [numeric integer_content](#) () const
- virtual [ex smod](#) (const [numeric](#) &xi) const
- Apply symmetric modular homomorphism to an expanded multivariate polynomial.*

 - virtual [numeric max_coefficient](#) () const
- Implementation [ex::max_coefficient\(\)](#).*

 - virtual [exvector get_free_indices](#) () const
- Return a vector containing the free indices of an expression.*

 - virtual [ex add_indexed](#) (const [ex](#) &self, const [ex](#) &other) const
- Add two indexed expressions.*

 - virtual [ex scalar_mul_indexed](#) (const [ex](#) &self, const [numeric](#) &other) const
- Multiply an indexed expression with a scalar.*

 - virtual [return_type_t return_type_tinfo](#) () const
- virtual [ex conjugate](#) () const
- virtual [ex real_part](#) () const
- virtual [ex imag_part](#) () const
- template<class T >
- void [print_dispatch](#) (const [print_context](#) &c, unsigned level) const

Like [print\(\)](#), but dispatch to the specified class.

 - void [print_dispatch](#) (const [registered_class_info](#) &ri, const [print_context](#) &c, unsigned level) const

Like [print\(\)](#), but dispatch to the specified class.

 - virtual void [archive](#) ([archive_node](#) &n) const
- Save (serialize) the object into archive node.*

 - virtual void [read_archive](#) (const [archive_node](#) &n, [lst](#) &syms)
- Load (deserialize) the object from an archive node.*

 - [ex subs_one_level](#) (const [exmap](#) &m, unsigned [options](#)) const
- Helper function for [subs\(\)](#).*

 - [ex diff](#) (const [symbol](#) &s, unsigned nth=1) const
- Default interface of nth derivative [ex::diff\(s, n\)](#).*

 - int [compare](#) (const [basic](#) &other) const
- Compare objects syntactically to establish canonical ordering.*

 - bool [is_equal](#) (const [basic](#) &other) const
- Test for syntactic equality.*

 - const [basic](#) & [hold](#) () const
- Stop further evaluation.*

 - unsigned [gethash](#) () const

- const [basic](#) & [setflag](#) (unsigned f) const
Set some [status_flags](#).
- const [basic](#) & [clearflag](#) (unsigned f) const
Clear some [status_flags](#).

Public Member Functions inherited from [GiNaC::refcounted](#)

- [refcounted](#) () noexcept
- unsigned int [add_reference](#) () noexcept
- unsigned int [remove_reference](#) () noexcept
- unsigned int [get_refcount](#) () const noexcept
- void [set_refcount](#) (unsigned int r) noexcept

Protected Member Functions

- void [do_print](#) (const [print_context](#) &c, unsigned level) const
- void [do_print_latex](#) (const [print_latex](#) &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::tensor](#)

- unsigned [return_type](#) () const override

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual [ex derivative](#) (const [symbol](#) &s) const
Default implementation of [ex::diff\(\)](#).
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Additional Inherited Members

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.14.1 Detailed Description

This class represents the Clifford algebra generators (units).

6.14.2 Member Function Documentation

6.14.2.1 `contract_with()`

```
bool GiNaC::cliffordunit::contract_with (
    exvector::iterator self,
    exvector::iterator other,
    exvector & v ) const [override], [virtual]
```

Contraction of a Clifford unit with something else.

Reimplemented from [GiNaC::basic](#).

Reimplemented in [GiNaC::diracgamma](#).

6.14.2.2 `do_print()`

```
void GiNaC::cliffordunit::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

6.14.2.3 `do_print_latex()`

```
void GiNaC::cliffordunit::do_print_latex (
    const print_latex & c,
    unsigned level ) const [protected]
```

The documentation for this class was generated from the following files:

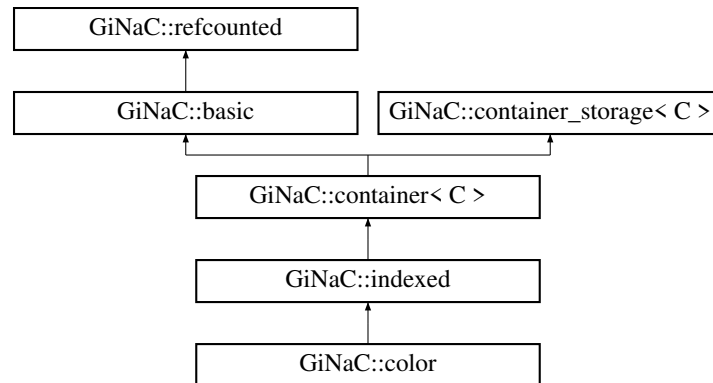
- [clifford.h](#)
- [clifford.cpp](#)

6.15 GiNaC::color Class Reference

This class holds a generator T_a or the unity element of the Lie algebra of $SU(3)$, as used for calculations in quantum chromodynamics.

```
#include <color.h>
```

Inheritance diagram for GiNaC::color:



Public Member Functions

- `color` (const `ex` &b, unsigned char rl=0)
Construct object without any color index.
- `color` (const `ex` &b, const `ex` &i1, unsigned char rl=0)
Construct object with one color index.
- `color` (unsigned char rl, const `exvector` &v)
- `color` (unsigned char rl, `exvector` &&v)
- void `archive` (`archive_node` &n) const override
Save (serialize) the object into archive node.
- void `read_archive` (const `archive_node` &n, `lst` &sym_lst) override
Load (deserialize) the object from an archive node.
- unsigned char `get_representation_label` () const

Public Member Functions inherited from GiNaC::indexed

- `indexed` (const `ex` &b)
Construct indexed object with no index.
- `indexed` (const `ex` &b, const `ex` &i1)
Construct indexed object with one index.
- `indexed` (const `ex` &b, const `ex` &i1, const `ex` &i2)
Construct indexed object with two indices.
- `indexed` (const `ex` &b, const `ex` &i1, const `ex` &i2, const `ex` &i3)
Construct indexed object with three indices.
- `indexed` (const `ex` &b, const `ex` &i1, const `ex` &i2, const `ex` &i3, const `ex` &i4)
Construct indexed object with four indices.
- `indexed` (const `ex` &b, const `symmetry` &symm, const `ex` &i1, const `ex` &i2)
Construct indexed object with two indices and a specified symmetry.
- `indexed` (const `ex` &b, const `symmetry` &symm, const `ex` &i1, const `ex` &i2, const `ex` &i3)

- Construct indexed object with three indices and a specified symmetry.*

 - `indexed` (const `ex` &b, const `symmetry` &`symm`, const `ex` &i1, const `ex` &i2, const `ex` &i3, const `ex` &i4)
- Construct indexed object with four indices and a specified symmetry.*

 - `indexed` (const `ex` &b, const `exvector` &iv)
- Construct indexed object with a specified vector of indices.*

 - `indexed` (const `ex` &b, const `symmetry` &`symm`, const `exvector` &iv)
- Construct indexed object with a specified vector of indices and symmetry.*

 - `indexed` (const `symmetry` &`symm`, const `exprseq` &es)
 - `indexed` (const `symmetry` &`symm`, const `exvector` &v)
 - `indexed` (const `symmetry` &`symm`, `exvector` &&v)
 - unsigned `precedence` () const override
- Return relative operator precedence (for parenthezing output).*

 - bool `info` (unsigned inf) const override
- Information about the object.*

 - `ex eval` () const override
- Perform automatic non-interruptive term rewriting rules.*

 - `ex real_part` () const override
 - `ex imag_part` () const override
 - `exvector get_free_indices` () const override
- Return a vector containing the free indices of an expression.*

 - void `archive` (`archive_node` &n) const override
- Save (a.k.a.*

 - void `read_archive` (const `archive_node` &n, `lst` &`syms`) override
- Read (a.k.a.*

 - bool `all_index_values_are` (unsigned inf) const
- Check whether all index values have a certain property.*

 - `exvector get_indices` () const
- Return a vector containing the object's indices.*

 - `exvector get_dummy_indices` () const
- Return a vector containing the dummy indices of the object, if any.*

 - `exvector get_dummy_indices` (const `indexed` &other) const
- Return a vector containing the dummy indices in the contraction with another indexed object.*

 - bool `has_dummy_index_for` (const `ex` &i) const
- Check whether the object has an index that forms a dummy index pair with a given index.*

 - `ex get_symmetry` () const
- Return symmetry properties.*

Public Member Functions inherited from `GiNaC::container< C >`

- `container` (STLT const &s)
- `container` (STLT &&v)
- `container` (exvector::const_iterator b, exvector::const_iterator e)
- `container` (std::initializer_list< `ex` > il)
- size_t `nops` () const override
- Number of operands/members.*
- `ex op` (size_t i) const override
- Return operand/member at position i.*
- `ex & let_op` (size_t i) override
- Return modifiable operand/member at position i.*
- `ex subs` (const `exmap` &m, unsigned `options`=0) const override
- Substitute a set of objects by arbitrary expressions.*

- `container & prepend (const ex &b)`
Add element at front.
- `container & append (const ex &b)`
Add element at back.
- `container & remove_first ()`
Remove first element.
- `container & remove_last ()`
Remove last element.
- `container & remove_all ()`
Remove all elements.
- `container & sort ()`
Sort elements.
- `container & unique ()`
Remove adjacent duplicate elements.
- `const_iterator begin () const`
- `const_iterator end () const`
- `const_reverse_iterator rbegin () const`
- `const_reverse_iterator rend () const`

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic ()`
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic (const basic &other)`
- `const basic & operator= (const basic &other)`
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate () const`
Create a clone of this object on the heap.
- virtual `ex evalf () const`
Evaluate object numerically.
- virtual `ex evalm () const`
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ () const`
Evaluate integrals, if result is known.
- virtual `ex eval_indexed (const basic &i) const`
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void `print (const print_context &c, unsigned level=0) const`
Output to stream.
- virtual void `dbgprint () const`
Little wrapper around print to be called within a debugger.
- virtual void `dbgprinttree () const`
Little wrapper around printtree to be called within a debugger.
- virtual `ex operator[] (const ex &index) const`
- virtual `ex operator[] (size_t i) const`
- virtual `ex & operator[] (const ex &index)`
- virtual `ex & operator[] (size_t i)`
- virtual bool `has (const ex &other, unsigned options=0) const`
Test for occurrence of a pattern.
- virtual bool `match (const ex &pattern, exmap &repls) const`
Check whether the expression matches a given pattern.
- virtual `ex map (map_function &f) const`

- Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).*

 - virtual void `accept` (GiNaC::visitor &v) const
 - virtual bool `is_polynomial` (const ex &var) const

Check whether this is a polynomial in the given variables.

 - virtual int `degree` (const ex &s) const

Return degree of highest power in object s.

 - virtual int `ldegree` (const ex &s) const

Return degree of lowest power in object s.

 - virtual ex `coeff` (const ex &s, int n=1) const

Return coefficient of degree n in object s.

 - virtual ex `collect` (const ex &s, bool distributed=false) const

Sort expanded expression in terms of powers of some object(s).

 - virtual ex `series` (const relational &r, int order, unsigned options=0) const

Default implementation of ex::series().

 - virtual ex `normal` (exmap &repl, exmap &rev_lookup, lst &modifier) const

Default implementation of ex::normal().

 - virtual ex `to_rational` (exmap &repl) const

Default implementation of ex::to_rational().

 - virtual ex `to_polynomial` (exmap &repl) const
 - virtual numeric `integer_content` () const
 - virtual ex `smod` (const numeric &xi) const

Apply symmetric modular homomorphism to an expanded multivariate polynomial.

 - virtual numeric `max_coefficient` () const

Implementation ex::max_coefficient().

 - virtual ex `add_indexed` (const ex &self, const ex &other) const

Add two indexed expressions.

 - virtual ex `scalar_mul_indexed` (const ex &self, const numeric &other) const

Multiply an indexed expression with a scalar.

 - virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, exvector &v) const

Try to contract two indexed expressions that appear in the same product.

 - template<class T>
 - void `print_dispatch` (const print_context &c, unsigned level) const

Like print(), but dispatch to the specified class.
 - void `print_dispatch` (const registered_class_info &ri, const print_context &c, unsigned level) const

Like print(), but dispatch to the specified class.

 - ex `subs_one_level` (const exmap &m, unsigned options) const

Helper function for subs().

 - ex `diff` (const symbol &s, unsigned nth=1) const

Default interface of nth derivative ex::diff(s, n).

 - int `compare` (const basic &other) const

Compare objects syntactically to establish canonical ordering.

 - bool `is_equal` (const basic &other) const

Test for syntactic equality.

 - const basic & `hold` () const

Stop further evaluation.

 - unsigned `gethash` () const
 - const basic & `setflag` (unsigned f) const

Set some status_flags.

 - const basic & `clearflag` (unsigned f) const

Clear some status_flags.

Public Member Functions inherited from [GiNaC::refcounted](#)

- [refcounted](#) () noexcept
- unsigned int [add_reference](#) () noexcept
- unsigned int [remove_reference](#) () noexcept
- unsigned int [get_refcount](#) () const noexcept
- void [set_refcount](#) (unsigned int r) noexcept

Protected Member Functions

- [ex eval_ncmul](#) (const [exvector](#) &v) const override
Perform automatic simplification on noncommutative product of color objects.
- bool [match_same_type](#) (const [basic](#) &other) const override
Returns true if the attributes of two objects are similar enough for a match.
- [ex thiscontainer](#) (const [exvector](#) &v) const override
- [ex thiscontainer](#) ([exvector](#) &&v) const override
- unsigned [return_type](#) () const override
- [return_type_t](#) [return_type_tinfo](#) () const override

Protected Member Functions inherited from [GiNaC::indexed](#)

- [ex derivative](#) (const [symbol](#) &s) const override
Implementation of [ex::diff\(\)](#) for an indexed object always returns 0.
- [ex thiscontainer](#) (const [exvector](#) &v) const override
- [ex thiscontainer](#) ([exvector](#) &&v) const override
- unsigned [return_type](#) () const override
- [return_type_t](#) [return_type_tinfo](#) () const override
- [ex expand](#) (unsigned [options](#)=0) const override
Expand expression, i.e.
- void [printindices](#) (const [print_context](#) &c, unsigned level) const
- void [print_indexed](#) (const [print_context](#) &c, const char *openbrace, const char *closebrace, unsigned level) const
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
- void [do_print_latex](#) (const [print_latex](#) &c, unsigned level) const
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
- void [validate](#) () const
Check whether all indices are of class idx and validate the symmetry tree.

Protected Member Functions inherited from [GiNaC::container< C >](#)

- [ex conjugate](#) () const override
- bool [is_equal_same_type](#) (const [basic](#) &other) const override
Returns true if two objects of same type are equal.
- virtual [ex thiscontainer](#) (const [STLT](#) &v) const
Similar to [duplicate\(\)](#), but with a preset sequence.
- virtual [ex thiscontainer](#) ([STLT](#) &&v) const
Similar to [duplicate\(\)](#), but with a preset sequence (which gets pilfered).
- virtual void [printseq](#) (const [print_context](#) &c, char openbracket, char delim, char closebracket, unsigned this←_precedence, unsigned upper_precedence=0) const
Print sequence of contained elements.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
- void [do_print_python](#) (const [print_python](#) &c, unsigned level) const
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
- [STLT](#) [subchildren](#) (const [exmap](#) &m, unsigned [options](#)=0) const

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Protected Member Functions inherited from [GiNaC::container_storage< C >](#)

- [container_storage](#) ()
- [container_storage](#) (size_t n, const [ex](#) &e)
- [container_storage](#) (std::initializer_list< [ex](#) > il)
- template<class In >
 [container_storage](#) (In b, In e)
- void [reserve](#) (size_t)
- [~container_storage](#) ()
- void [reserve](#) (size_t n)
- void [reserve](#) (std::vector< [ex](#) > &v, size_t n)

Private Attributes

- unsigned char [representation_label](#)
Representation label to distinguish independent color matrices coming from separated fermion lines.

Additional Inherited Members

Public Types inherited from [GiNaC::container< C >](#)

- typedef STLT::const_iterator [const_iterator](#)
- typedef STLT::const_reverse_iterator [const_reverse_iterator](#)

Protected Types inherited from [GiNaC::container< C >](#)

- typedef [container_storage](#)< C >::STLT [STLT](#)

Protected Types inherited from [GiNaC::container_storage< C >](#)

- typedef C< [ex](#) > [STLT](#)

Static Protected Member Functions inherited from [GiNaC::container< C >](#)

- static unsigned [get_default_flags](#) ()
*Specialization of [container::get_default_flags\(\)](#) for *lst*.*
- static char [get_open_delim](#) ()
*Specialization of [container::get_open_delim\(\)](#) for *lst*.*
- static char [get_close_delim](#) ()
*Specialization of [container::get_close_delim\(\)](#) for *lst*.*

Static Protected Member Functions inherited from [GiNaC::container_storage< C >](#)

- static void [reserve](#) (STLT &, size_t)

Protected Attributes inherited from [GiNaC::indexed](#)

- [ex symtree](#)
Index symmetry (tree of symmetry objects)

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

Protected Attributes inherited from [GiNaC::container_storage< C >](#)

- [STLT seq](#)

6.15.1 Detailed Description

This class holds a generator `T_a` or the unity element of the Lie algebra of $SU(3)$, as used for calculations in quantum chromodynamics.

A representation label (an unsigned 8-bit integer) is used to distinguish elements from different Lie algebras (objects with different labels commute). These objects implement an abstract representation of the group, not a specific matrix representation. The indices used for color objects should not have a variance.

6.15.2 Constructor & Destructor Documentation

6.15.2.1 [color\(\)](#) [1/4]

```
GiNaC::color::color (
    const ex & b,
    unsigned char r1 = 0 )
```

Construct object without any color index.

This constructor is for internal use only. Use the [color_ONE\(\)](#) function instead.

See also

[color_ONE](#)

6.15.2.2 color() [2/4]

```
GiNaC::color::color (
    const ex & b,
    const ex & i1,
    unsigned char r1 = 0 )
```

Construct object with one color index.

This constructor is for internal use only. Use the [color_T\(\)](#) function instead.

See also

[color_T](#)

6.15.2.3 color() [3/4]

```
GiNaC::color::color (
    unsigned char r1,
    const exvector & v )
```

6.15.2.4 color() [4/4]

```
GiNaC::color::color (
    unsigned char r1,
    exvector && v )
```

6.15.3 Member Function Documentation

6.15.3.1 archive()

```
void GiNaC::color::archive (
    archive\_node & n ) const [override], [virtual]
```

Save (serialize) the object into archive node.

Archive the object.

Losely speaking, this method turns an expression into a byte stream (which can be saved and restored later on, or sent via network, etc.)

Reimplemented from [GiNaC::basic](#).

References [n](#), and [representation_label](#).

6.15.3.2 read_archive()

```
void GiNaC::color::read_archive (
    const archive_node & n,
    lst & syms ) [override], [virtual]
```

Load (deserialize) the object from an archive node.

Construct object from [archive_node](#).

Note

This method is essentially a constructor. However, constructors can't be virtual. So, if unarchiving routines are implemented as constructors one would need to define such a constructor in every class, even if all it does is simply calling constructor of a superclass.

Reimplemented from [GiNaC::basic](#).

References [n](#), and [representation_label](#).

6.15.3.3 eval_ncmul()

```
ex GiNaC::color::eval_ncmul (
    const exvector & v ) const [override], [protected], [virtual]
```

Perform automatic simplification on noncommutative product of color objects.

This removes superfluous ONes.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::hold_ncmul\(\)](#), and [GiNaC::container_storage< C >::reserve\(\)](#).

6.15.3.4 match_same_type()

```
bool GiNaC::color::match_same_type (
    const basic & other ) const [override], [protected], [virtual]
```

Returns true if the attributes of two objects are similar enough for a match.

This function must not match subexpressions (this is already done by [basic::match\(\)](#)). Only attributes not accessible by [op\(\)](#) should be compared. This is also the reason why this function doesn't take the wildcard replacement list from [match\(\)](#) as an argument: only subexpressions are subject to wildcard matches. Also, this function only needs to be implemented for container classes because [is_equal_same_type\(\)](#) is automatically used instead of [match_same_type\(\)](#) if [nops\(\)](#) == 0.

See also

[basic::match](#)

Reimplemented from [GiNaC::basic](#).

References [GINAC_ASSERT](#), and [representation_label](#).

6.15.3.5 thiscontainer() [1/2]

```
ex GiNaC::color::thiscontainer (
    const exvector & v ) const [override], [protected]
```

References [representation_label](#).

6.15.3.6 thiscontainer() [2/2]

```
ex GiNaC::color::thiscontainer (
    exvector && v ) const [override], [protected]
```

References [representation_label](#).

6.15.3.7 return_type()

```
unsigned GiNaC::color::return_type ( ) const [inline], [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::return_types::noncommutative](#).

6.15.3.8 return_type_tinfo()

```
return\_type\_t GiNaC::color::return_type_tinfo ( ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [representation_label](#).

6.15.3.9 get_representation_label()

```
unsigned char GiNaC::color::get_representation_label ( ) const [inline]
```

References [representation_label](#).

6.15.4 Member Data Documentation

6.15.4.1 representation_label

```
unsigned char GiNaC::color::representation_label [private]
```

Representation label to distinguish independent color matrices coming from separated fermion lines.

Referenced by [archive\(\)](#), [get_representation_label\(\)](#), [match_same_type\(\)](#), [read_archive\(\)](#), [return_type_tinfo\(\)](#), [thiscontainer\(\)](#), and [thiscontainer\(\)](#).

The documentation for this class was generated from the following files:

- [color.h](#)
- [color.cpp](#)

6.16 GiNaC::compare_all_equal< T > Class Template Reference

Comparison policy: all structures of one type are equal.

```
#include <structure.h>
```

Protected Member Functions

- [~compare_all_equal\(\)](#)

Static Protected Member Functions

- static bool [struct_is_equal](#) (const T *t1, const T *t2)
- static int [struct_compare](#) (const T *t1, const T *t2)

6.16.1 Detailed Description

```
template<class T>
class GiNaC::compare_all_equal< T >
```

Comparison policy: all structures of one type are equal.

6.16.2 Constructor & Destructor Documentation

6.16.2.1 ~compare_all_equal()

```
template<class T >
GiNaC::compare_all_equal< T >::~~compare_all_equal ( ) [inline], [protected]
```

6.16.3 Member Function Documentation

6.16.3.1 struct_is_equal()

```
template<class T >
static bool GiNaC::compare_all_equal< T >::struct_is_equal (
    const T * t1,
    const T * t2 ) [inline], [static], [protected]
```

6.16.3.2 struct_compare()

```
template<class T >
static int GiNaC::compare_all_equal< T >::struct_compare (
    const T * t1,
    const T * t2 ) [inline], [static], [protected]
```

The documentation for this class was generated from the following file:

- [structure.h](#)

6.17 GiNaC::compare_bitwise< T > Class Template Reference

Comparison policy: use bit-wise comparison to compare structures.

```
#include <structure.h>
```

Protected Member Functions

- [~compare_bitwise](#) ()

Static Protected Member Functions

- static bool [struct_is_equal](#) (const T *t1, const T *t2)
- static int [struct_compare](#) (const T *t1, const T *t2)

6.17.1 Detailed Description

```
template<class T>
class GiNaC::compare_bitwise< T >
```

Comparison policy: use bit-wise comparison to compare structures.

6.17.2 Constructor & Destructor Documentation

6.17.2.1 ~compare_bitwise()

```
template<class T >
GiNaC::compare_bitwise< T >::~~compare_bitwise ( ) [inline], [protected]
```

6.17.3 Member Function Documentation

6.17.3.1 struct_is_equal()

```
template<class T >
static bool GiNaC::compare_bitwise< T >::struct_is_equal (
    const T * t1,
    const T * t2 ) [inline], [static], [protected]
```

6.17.3.2 struct_compare()

```
template<class T >
static int GiNaC::compare_bitwise< T >::struct_compare (
    const T * t1,
    const T * t2 ) [inline], [static], [protected]
```

The documentation for this class was generated from the following file:

- [structure.h](#)

6.18 GiNaC::compare_std_less< T > Class Template Reference

Comparison policy: use std::equal_to/std::less (defaults to operators == and <) to compare structures.

```
#include <structure.h>
```

Protected Member Functions

- [~compare_std_less\(\)](#)

Static Protected Member Functions

- static bool [struct_is_equal](#) (const T *t1, const T *t2)
- static int [struct_compare](#) (const T *t1, const T *t2)

6.18.1 Detailed Description

```
template<class T>
class GiNaC::compare_std_less< T >
```

Comparison policy: use std::equal_to/std::less (defaults to operators == and <) to compare structures.

6.18.2 Constructor & Destructor Documentation

6.18.2.1 ~compare_std_less()

```
template<class T >
GiNaC::compare_std_less< T >::~~compare_std_less ( ) [inline], [protected]
```

6.18.3 Member Function Documentation

6.18.3.1 struct_is_equal()

```
template<class T >
static bool GiNaC::compare_std_less< T >::struct_is_equal (
    const T * t1,
    const T * t2 ) [inline], [static], [protected]
```

6.18.3.2 struct_compare()

```
template<class T >
static int GiNaC::compare_std_less< T >::struct_compare (
    const T * t1,
    const T * t2 ) [inline], [static], [protected]
```

The documentation for this class was generated from the following file:

- [structure.h](#)

6.19 GiNaC::composition_generator Class Reference

Generate all compositions of a partition of an integer n , starting with the compositions which has non-decreasing order.

```
#include <utils.h>
```

Classes

- struct [coolmulti](#)

Public Member Functions

- [composition_generator](#) (const std::vector< unsigned > &partition)
- const std::vector< unsigned > & [get](#) () const
- bool [next](#) ()

Private Attributes

- struct [GiNaC::composition_generator::coolmulti](#) cmgen
- bool [atend](#)
- bool [trivial](#)
- std::vector< unsigned > [composition](#)
- bool [current_updated](#)

6.19.1 Detailed Description

Generate all compositions of a partition of an integer n , starting with the compositions which has non-decreasing order.

6.19.2 Constructor & Destructor Documentation

6.19.2.1 composition_generator()

```
GiNaC::composition_generator::composition_generator (
    const std::vector< unsigned > & partition ) [inline], [explicit]
```

References [trivial](#).

6.19.3 Member Function Documentation

6.19.3.1 get()

```
const std::vector< unsigned > & GiNaC::composition_generator::get ( ) const [inline]
```

References [cmgen](#), [composition](#), [current_updated](#), [GiNaC::composition_generator::coolmulti::head](#), [GiNaC::composition_generator::coolmulti::element::value](#).

Referenced by [GiNaC::power::expand_add\(\)](#).

6.19.3.2 next()

```
bool GiNaC::composition_generator::next ( ) [inline]
```

References [atend](#), [cmgen](#), [current_updated](#), [GiNaC::composition_generator::coolmulti::finished\(\)](#), [GiNaC::composition_generator::coolmulti::next\(\)](#) and [trivial](#).

Referenced by [GiNaC::power::expand_add\(\)](#).

6.19.4 Member Data Documentation

6.19.4.1 cmgen

```
struct GiNaC::composition_generator::coolmulti GiNaC::composition_generator::cmgen [private]
```

Referenced by [get\(\)](#), and [next\(\)](#).

6.19.4.2 atend

```
bool GiNaC::composition_generator::atend [private]
```

Referenced by [next\(\)](#).

6.19.4.3 trivial

```
bool GiNaC::composition_generator::trivial [private]
```

Referenced by [composition_generator\(\)](#), and [next\(\)](#).

6.19.4.4 composition

```
std::vector<unsigned> GiNaC::composition_generator::composition [mutable], [private]
```

Referenced by [get\(\)](#).

6.19.4.5 current_updated

```
bool GiNaC::composition_generator::current_updated [mutable], [private]
```

Referenced by [get\(\)](#), and [next\(\)](#).

The documentation for this class was generated from the following file:

- [utils.h](#)

6.20 GiNaC::const_iterator Class Reference

```
#include <ex.h>
```

Public Types

- using `iterator_category` = `std::random_access_iterator_tag`
- using `value_type` = `ex`
- using `difference_type` = `ptrdiff_t`
- using `pointer` = `const ex *`
- using `reference` = `const ex &`

Public Member Functions

- `const_iterator` () noexcept
- `ex operator*` () const
- `std::unique_ptr< ex > operator->` () const
- `ex operator[]` (`difference_type n`) const
- `const_iterator & operator++` () noexcept
- `const_iterator operator++` (`int`) noexcept
- `const_iterator & operator+=` (`difference_type n`) noexcept
- `const_iterator operator+` (`difference_type n`) const noexcept
- `const_iterator & operator--` () noexcept
- `const_iterator operator--` (`int`) noexcept
- `const_iterator & operator-=` (`difference_type n`) noexcept
- `const_iterator operator-` (`difference_type n`) const noexcept
- `bool operator==` (`const const_iterator &other`) const noexcept
- `bool operator!=` (`const const_iterator &other`) const noexcept
- `bool operator<` (`const const_iterator &other`) const noexcept
- `bool operator>` (`const const_iterator &other`) const noexcept
- `bool operator<=` (`const const_iterator &other`) const noexcept
- `bool operator>=` (`const const_iterator &other`) const noexcept

Protected Attributes

- `ex e`
- `size_t i`

Private Member Functions

- `const_iterator` (`const ex &e_`, `size_t i_`) noexcept

Friends

- class `ex`
- class `const_preorder_iterator`
- class `const_postorder_iterator`
- `const_iterator operator+` (`difference_type n`, `const const_iterator &it`) noexcept
- `difference_type operator-` (`const const_iterator &lhs`, `const const_iterator &rhs`) noexcept

6.20.1 Member Typedef Documentation

6.20.1.1 `iterator_category`

```
using GiNaC::const_iterator::iterator_category = std::random_access_iterator_tag
```

6.20.1.2 `value_type`

```
using GiNaC::const_iterator::value_type = ex
```

6.20.1.3 `difference_type`

```
using GiNaC::const_iterator::difference_type = ptrdiff_t
```

6.20.1.4 `pointer`

```
using GiNaC::const_iterator::pointer = const ex *
```

6.20.1.5 `reference`

```
using GiNaC::const_iterator::reference = const ex &
```

6.20.2 Constructor & Destructor Documentation

6.20.2.1 `const_iterator()` [1/2]

```
GiNaC::const_iterator::const_iterator ( ) [inline], [noexcept]
```

Referenced by [operator+\(\)](#), and [operator-\(\)](#).

6.20.2.2 `const_iterator()` [2/2]

```
GiNaC::const_iterator::const_iterator (
    const ex & e_,
    size_t i_ ) [inline], [private], [noexcept]
```

6.20.3 Member Function Documentation

6.20.3.1 `operator*()`

```
ex GiNaC::const_iterator::operator* ( ) const [inline]
```

References [e](#), [i](#), and [GiNaC::ex::op\(\)](#).

6.20.3.2 operator->()

```
std::unique_ptr< ex > GiNaC::const_iterator::operator-> ( ) const [inline]
```

References [ex](#).

6.20.3.3 operator[]()

```
ex GiNaC::const_iterator::operator[] (   
    difference\_type n ) const [inline]
```

References [e](#), [i](#), [n](#), and [GiNaC::ex::op\(\)](#).

6.20.3.4 operator++() [1/2]

```
const\_iterator & GiNaC::const_iterator::operator++ ( ) [inline], [noexcept]
```

References [i](#).

6.20.3.5 operator++() [2/2]

```
const\_iterator GiNaC::const_iterator::operator++ (   
    int ) [inline], [noexcept]
```

References [i](#).

6.20.3.6 operator+=()

```
const\_iterator & GiNaC::const_iterator::operator+= (   
    difference\_type n ) [inline], [noexcept]
```

References [i](#), and [n](#).

6.20.3.7 operator+()

```
const\_iterator GiNaC::const_iterator::operator+ (   
    difference\_type n ) const [inline], [noexcept]
```

References [const_iterator\(\)](#), [e](#), [i](#), and [n](#).

6.20.3.8 operator--() [1/2]

```
const\_iterator & GiNaC::const_iterator::operator-- ( ) [inline], [noexcept]
```

References [i](#).

6.20.3.9 operator--() [2/2]

```
const_iterator GiNaC::const_iterator::operator-- (
    int ) [inline], [noexcept]
```

References [i](#).

6.20.3.10 operator-=()

```
const_iterator & GiNaC::const_iterator::operator-= (
    difference_type n ) [inline], [noexcept]
```

References [i](#), and [n](#).

6.20.3.11 operator-()

```
const_iterator GiNaC::const_iterator::operator- (
    difference_type n ) const [inline], [noexcept]
```

References [const_iterator\(\)](#), [e](#), [i](#), and [n](#).

6.20.3.12 operator==()

```
bool GiNaC::const_iterator::operator== (
    const const_iterator & other ) const [inline], [noexcept]
```

References [GiNaC::are_ex_trivially_equal\(\)](#), [e](#), and [i](#).

6.20.3.13 operator!=(())

```
bool GiNaC::const_iterator::operator!= (
    const const_iterator & other ) const [inline], [noexcept]
```

6.20.3.14 operator<()

```
bool GiNaC::const_iterator::operator< (
    const const_iterator & other ) const [inline], [noexcept]
```

References [i](#).

6.20.3.15 operator>()

```
bool GiNaC::const_iterator::operator> (
    const const_iterator & other ) const [inline], [noexcept]
```

6.20.3.16 operator<=()

```
bool GiNaC::const_iterator::operator<= (
    const const\_iterator & other ) const [inline], [noexcept]
```

6.20.3.17 operator>=()

```
bool GiNaC::const_iterator::operator>= (
    const const\_iterator & other ) const [inline], [noexcept]
```

6.20.4 Friends And Related Symbol Documentation

6.20.4.1 ex

```
friend class ex [friend]
```

Referenced by [operator->\(\)](#).

6.20.4.2 const_preorder_iterator

```
friend class const\_preorder\_iterator [friend]
```

6.20.4.3 const_postorder_iterator

```
friend class const\_postorder\_iterator [friend]
```

6.20.4.4 operator+

```
const\_iterator operator+ (
    difference\_type n,
    const const\_iterator & it ) [friend]
```

6.20.4.5 operator-

```
difference\_type operator- (
    const const\_iterator & lhs,
    const const\_iterator & rhs ) [friend]
```

6.20.5 Member Data Documentation

6.20.5.1 e

```
ex GiNaC::const_iterator::e [protected]
```

Referenced by [operator*\(\)](#), [operator+\(\)](#), [operator-\(\)](#), [operator==\(\)](#), and [operator\[\]\(\)](#).

6.20.5.2 i

```
size_t GiNaC::const_iterator::i [protected]
```

Referenced by [operator*\(\)](#), [operator+\(\)](#), [operator++\(\)](#), [operator++\(\)](#), [operator+=\(\)](#), [operator-\(\)](#), [operator--\(\)](#), [operator--\(\)](#), [operator-=\(\)](#), [operator<\(\)](#), [operator==\(\)](#), and [operator\[\]\(\)](#).

The documentation for this class was generated from the following file:

- [ex.h](#)

6.21 GiNaC::const_postorder_iterator Class Reference

```
#include <ex.h>
```

Public Types

- using [iterator_category](#) = std::forward_iterator_tag
- using [value_type](#) = [ex](#)
- using [difference_type](#) = ptrdiff_t
- using [pointer](#) = const [ex](#) *
- using [reference](#) = const [ex](#) &

Public Member Functions

- [const_postorder_iterator](#) () noexcept
- [const_postorder_iterator](#) (const [ex](#) &e, size_t n)
- [reference operator*](#) () const
- [pointer operator->](#) () const
- [const_postorder_iterator](#) & [operator++](#) ()
- [const_postorder_iterator](#) [operator++](#) (int)
- bool [operator==](#) (const [const_postorder_iterator](#) &other) const noexcept
- bool [operator!=](#) (const [const_postorder_iterator](#) &other) const noexcept

Private Member Functions

- void [descend](#) ()
- void [increment](#) ()

Private Attributes

- std::stack< [internal::_iter_rep](#), std::vector< [internal::_iter_rep](#) > > s

6.21.1 Member Typedef Documentation

6.21.1.1 iterator_category

```
using GiNaC::const_postorder_iterator::iterator_category = std::forward_iterator_tag
```

6.21.1.2 value_type

```
using GiNaC::const_postorder_iterator::value_type = ex
```

6.21.1.3 difference_type

```
using GiNaC::const_postorder_iterator::difference_type = ptrdiff_t
```

6.21.1.4 pointer

```
using GiNaC::const_postorder_iterator::pointer = const ex *
```

6.21.1.5 reference

```
using GiNaC::const_postorder_iterator::reference = const ex &
```

6.21.2 Constructor & Destructor Documentation

6.21.2.1 const_postorder_iterator() [1/2]

```
GiNaC::const_postorder_iterator::const_postorder_iterator ( ) [inline], [noexcept]
```

6.21.2.2 const_postorder_iterator() [2/2]

```
GiNaC::const_postorder_iterator::const_postorder_iterator (
    const ex & e,
    size_t n ) [inline]
```

References [descend\(\)](#), [n](#), and [s](#).

6.21.3 Member Function Documentation

6.21.3.1 operator*()

```
reference GiNaC::const_postorder_iterator::operator* ( ) const [inline]
```

References [s](#).

6.21.3.2 operator->()

```
pointer GiNaC::const_postorder_iterator::operator-> ( ) const [inline]
```

References [s](#).

6.21.3.3 `operator++()` [1/2]

```
const_postorder_iterator & GiNaC::const_postorder_iterator::operator++ ( ) [inline]
```

References [increment\(\)](#).

6.21.3.4 `operator++()` [2/2]

```
const_postorder_iterator GiNaC::const_postorder_iterator::operator++ (
    int ) [inline]
```

References [increment\(\)](#).

6.21.3.5 `operator==()`

```
bool GiNaC::const_postorder_iterator::operator== (
    const const_postorder_iterator & other ) const [inline], [noexcept]
```

References [s](#).

6.21.3.6 `operator!=(())`

```
bool GiNaC::const_postorder_iterator::operator!= (
    const const_postorder_iterator & other ) const [inline], [noexcept]
```

6.21.3.7 `descend()`

```
void GiNaC::const_postorder_iterator::descend ( ) [inline], [private]
```

References [GiNaC::internal::_iter_rep::e](#), [GiNaC::internal::_iter_rep::i](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), and [s](#).

Referenced by [const_postorder_iterator\(\)](#), and [increment\(\)](#).

6.21.3.8 `increment()`

```
void GiNaC::const_postorder_iterator::increment ( ) [inline], [private]
```

References [descend\(\)](#), and [s](#).

Referenced by [operator++\(\)](#), and [operator++\(\)](#).

6.21.4 Member Data Documentation

6.21.4.1 s

```
std::stack<internal::_iter_rep, std::vector<internal::_iter_rep> > GiNaC::const_postorder_iterator::s [private]
```

Referenced by [const_postorder_iterator\(\)](#), [descend\(\)](#), [increment\(\)](#), [operator*\(\)](#), [operator->\(\)](#), and [operator==\(\)](#).

The documentation for this class was generated from the following file:

- [ex.h](#)

6.22 GiNaC::const_preorder_iterator Class Reference

```
#include <ex.h>
```

Public Types

- using [iterator_category](#) = std::forward_iterator_tag
- using [value_type](#) = [ex](#)
- using [difference_type](#) = ptrdiff_t
- using [pointer](#) = const [ex](#) *
- using [reference](#) = const [ex](#) &

Public Member Functions

- [const_preorder_iterator](#) () noexcept
- [const_preorder_iterator](#) (const [ex](#) &e, size_t n)
- [reference operator*](#) () const
- [pointer operator->](#) () const
- [const_preorder_iterator](#) & [operator++](#) ()
- [const_preorder_iterator](#) [operator++](#) (int)
- bool [operator==](#) (const [const_preorder_iterator](#) &other) const noexcept
- bool [operator!=](#) (const [const_preorder_iterator](#) &other) const noexcept

Private Member Functions

- void [increment](#) ()

Private Attributes

- std::stack< [internal::_iter_rep](#), std::vector< [internal::_iter_rep](#) > > [s](#)

6.22.1 Member Typedef Documentation

6.22.1.1 iterator_category

```
using GiNaC::const_preorder_iterator::iterator_category = std::forward_iterator_tag
```

6.22.1.2 value_type

```
using GiNaC::const_preorder_iterator::value_type = ex
```

6.22.1.3 difference_type

```
using GiNaC::const_preorder_iterator::difference_type = ptrdiff_t
```

6.22.1.4 pointer

```
using GiNaC::const_preorder_iterator::pointer = const ex *
```

6.22.1.5 reference

```
using GiNaC::const_preorder_iterator::reference = const ex &
```

6.22.2 Constructor & Destructor Documentation

6.22.2.1 const_preorder_iterator() [1/2]

```
GiNaC::const_preorder_iterator::const_preorder_iterator ( ) [inline], [noexcept]
```

6.22.2.2 const_preorder_iterator() [2/2]

```
GiNaC::const_preorder_iterator::const_preorder_iterator (
    const ex & e,
    size_t n ) [inline]
```

References [n](#), and [s](#).

6.22.3 Member Function Documentation

6.22.3.1 operator*()

```
reference GiNaC::const_preorder_iterator::operator* ( ) const [inline]
```

References [s](#).

6.22.3.2 operator->()

```
pointer GiNaC::const_preorder_iterator::operator-> ( ) const [inline]
```

References [s](#).

6.22.3.3 operator++() [1/2]

```
const_preorder_iterator & GiNaC::const_preorder_iterator::operator++ ( ) [inline]
```

References [increment\(\)](#).

6.22.3.4 operator++() [2/2]

```
const_preorder_iterator GiNaC::const_preorder_iterator::operator++ (
    int ) [inline]
```

References [increment\(\)](#).

6.22.3.5 operator==()

```
bool GiNaC::const_preorder_iterator::operator== (
    const const_preorder_iterator & other ) const [inline], [noexcept]
```

References [s](#).

6.22.3.6 operator!=(())

```
bool GiNaC::const_preorder_iterator::operator!= (
    const const_preorder_iterator & other ) const [inline], [noexcept]
```

6.22.3.7 increment()

```
void GiNaC::const_preorder_iterator::increment ( ) [inline], [private]
```

References [GiNaC::internal::_iter_rep::e](#), [GiNaC::internal::_iter_rep::i](#), [GiNaC::internal::_iter_rep::i_end](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), and [s](#).

Referenced by [operator++\(\)](#), and [operator++\(\)](#).

6.22.4 Member Data Documentation

6.22.4.1 s

```
std::stack<internal::_iter_rep, std::vector<internal::_iter_rep> > GiNaC::const_preorder_↵
iterator::s [private]
```

Referenced by [const_preorder_iterator\(\)](#), [increment\(\)](#), [operator*\(\)](#), [operator->\(\)](#), and [operator==\(\)](#).

The documentation for this class was generated from the following file:

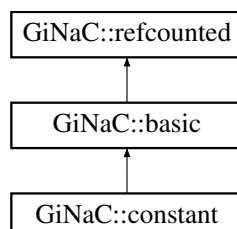
- [ex.h](#)

6.23 GiNaC::constant Class Reference

This class holds constants, symbols with specific numerical value.

```
#include <constant.h>
```

Inheritance diagram for GiNaC::constant:



Public Member Functions

- [constant](#) (const std::string &initname, [evalfunctype](#) efun=nullptr, const std::string &texname=std::string(), unsigned [domain=domain::complex](#))
- [constant](#) (const std::string &initname, const [numeric](#) &initnumber, const std::string &texname=std::string(), unsigned [domain=domain::complex](#))
- bool [info](#) (unsigned inf) const override
Information about the object.
- [ex evalf](#) () const override
Evaluate object numerically.
- bool [is_polynomial](#) (const [ex](#) &var) const override
Check whether this is a polynomial in the given variables.
- [ex conjugate](#) () const override
- [ex real_part](#) () const override
- [ex imag_part](#) () const override
- void [archive](#) ([archive_node](#) &n) const override
Save (serialize) the object into archive node.
- void [read_archive](#) (const [archive_node](#) &n, [lst](#) &syms) override
Load (deserialize) the object from an archive node.
- std::string [get_name](#) () const
- std::string [get_TeX_name](#) () const

Public Member Functions inherited from [GiNaC::basic](#)

- virtual [~basic](#) ()
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- [basic](#) (const [basic](#) &other)
- const [basic](#) & [operator=](#) (const [basic](#) &other)
basic assignment operator: the other object might be of a derived class.
- virtual [basic](#) * [duplicate](#) () const
Create a clone of this object on the heap.
- virtual [ex eval](#) () const
Perform automatic non-interruptive term rewriting rules.
- virtual [ex evalm](#) () const

- Evaluate sums, products and integer powers of matrices.*

 - virtual `ex eval_integ ()` const
- Evaluate integrals, if result is known.*

 - virtual `ex eval_indexed (const basic &i)` const
- Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.*

 - virtual void `print (const print_context &c, unsigned level=0)` const
- Output to stream.*

 - virtual void `dbgprint ()` const
- Little wrapper around print to be called within a debugger.*

 - virtual void `dbgprinttree ()` const
- Return relative operator precedence (for parenthezing output).*

 - virtual unsigned `precedence ()` const
- Number of operands/members.*

 - virtual `size_t nops ()` const
- Return operand/member at position i.*

 - virtual `ex operator[] (const ex &index)` const
 - virtual `ex operator[] (size_t i)` const
 - virtual `ex &let_op (size_t i)`
- Return modifiable operand/member at position i.*

 - virtual `ex &operator[] (const ex &index)`
 - virtual `ex &operator[] (size_t i)`
- Test for occurrence of a pattern.*

 - virtual bool `has (const ex &other, unsigned options=0)` const
- Check whether the expression matches a given pattern.*

 - virtual bool `match (const ex &pattern, exmap &repls)` const
- Substitute a set of objects by arbitrary expressions.*

 - virtual `ex subs (const exmap &m, unsigned options=0)` const
- Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).*

 - virtual void `accept (GiNaC::visitor &v)` const
 - virtual int `degree (const ex &s)` const
- Return degree of highest power in object s.*

 - virtual int `ldegree (const ex &s)` const
- Return degree of lowest power in object s.*

 - virtual `ex coeff (const ex &s, int n=1)` const
- Return coefficient of degree n in object s.*

 - virtual `ex expand (unsigned options=0)` const
- Expand expression, i.e.*

 - virtual `ex collect (const ex &s, bool distributed=false)` const
- Sort expanded expression in terms of powers of some object(s).*

 - virtual `ex series (const relational &r, int order, unsigned options=0)` const
- Default implementation of `ex::series()`.*

 - virtual `ex normal (exmap &repl, exmap &rev_lookup, lst &modifier)` const
- Default implementation of `ex::normal()`.*

 - virtual `ex to_rational (exmap &repl)` const
- Default implementation of `ex::to_rational()`.*

 - virtual `ex to_polynomial (exmap &repl)` const
 - virtual `numeric integer_content ()` const
 - virtual `ex smod (const numeric &xi)` const

- *Apply symmetric modular homomorphism to an expanded multivariate polynomial.*
- virtual `numeric max_coefficient () const`
Implementation `ex::max_coefficient()`.
- virtual `exvector get_free_indices () const`
Return a vector containing the free indices of an expression.
- virtual `ex add_indexed (const ex &self, const ex &other) const`
Add two indexed expressions.
- virtual `ex scalar_mul_indexed (const ex &self, const numeric &other) const`
Multiply an indexed expression with a scalar.
- virtual `bool contract_with (exvector::iterator self, exvector::iterator other, exvector &v) const`
Try to contract two indexed expressions that appear in the same product.
- virtual `unsigned return_type () const`
- virtual `return_type_t return_type_tinfo () const`
- `template<class T >`
`void print_dispatch (const print_context &c, unsigned level) const`
Like `print()`, but dispatch to the specified class.
- `void print_dispatch (const registered_class_info &ri, const print_context &c, unsigned level) const`
Like `print()`, but dispatch to the specified class.
- `ex subs_one_level (const exmap &m, unsigned options) const`
Helper function for `subs()`.
- `ex diff (const symbol &s, unsigned nth=1) const`
Default interface of `nth` derivative `ex::diff(s, n)`.
- `int compare (const basic &other) const`
Compare objects syntactically to establish canonical ordering.
- `bool is_equal (const basic &other) const`
Test for syntactic equality.
- `const basic & hold () const`
Stop further evaluation.
- `unsigned gethash () const`
- `const basic & setflag (unsigned f) const`
Set some `status_flags`.
- `const basic & clearflag (unsigned f) const`
Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted () noexcept`
- `unsigned int add_reference () noexcept`
- `unsigned int remove_reference () noexcept`
- `unsigned int get_refcount () const noexcept`
- `void set_refcount (unsigned int r) noexcept`

Protected Member Functions

- `ex derivative (const symbol &s) const override`
Implementation of `ex::diff()` for a constant always returns 0.
- `bool is_equal_same_type (const basic &other) const override`
Returns true if two objects of same type are equal.
- `unsigned calchash () const override`
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- `void do_print (const print_context &c, unsigned level) const`
- `void do_print_tree (const print_tree &c, unsigned level) const`
- `void do_print_latex (const print_latex &c, unsigned level) const`
- `void do_print_python_repr (const print_python_repr &c, unsigned level) const`

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Private Attributes

- std::string [name](#)
printname of this constant
- std::string [TeX_name](#)
LaTeX name.
- [evalffunc](#)type ef
- [ex](#) number
numerical value this constant [evalf\(\)](#)s to
- unsigned [serial](#)
unique serial number for comparison
- unsigned [domain](#)
numerical value this constant [evalf\(\)](#)s to

Static Private Attributes

- static unsigned [next_serial](#) = 0

Additional Inherited Members

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.23.1 Detailed Description

This class holds constants, symbols with specific numerical value.

Each object of this class must either provide their own function to evaluate it to class numeric or provide the constant as a numeric (if it's an exact number).

6.23.2 Constructor & Destructor Documentation

6.23.2.1 `constant()` [1/2]

```
GiNaC::constant::constant (
    const std::string & initname,
    evalffunctype efun = nullptr,
    const std::string & texname = std::string(),
    unsigned domain = domain::complex )
```

References [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::expanded](#), [name](#), [GiNaC::basic::setflag\(\)](#), and [TeX_name](#).

6.23.2.2 `constant()` [2/2]

```
GiNaC::constant::constant (
    const std::string & initname,
    const numeric & initnumber,
    const std::string & texname = std::string(),
    unsigned domain = domain::complex )
```

References [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::expanded](#), [name](#), [GiNaC::basic::setflag\(\)](#), and [TeX_name](#).

6.23.3 Member Function Documentation

6.23.3.1 `info()`

```
bool GiNaC::constant::info (
    unsigned inf ) const [override], [virtual]
```

Information about the object.

See also

class [info_flags](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::info_flags::nonnegative](#), [GiNaC::info_flags::polynomial](#), [GiNaC::domain::positive](#), [GiNaC::info_flags::positive](#), [GiNaC::domain::real](#), and [GiNaC::info_flags::real](#).

6.23.3.2 `evalf()`

```
ex GiNaC::constant::evalf ( ) const [override], [virtual]
```

Evaluate object numerically.

Reimplemented from [GiNaC::basic](#).

References [ef](#), [GiNaC::ex::evalf\(\)](#), and [number](#).

Referenced by [GiNaC::EllipticE_eval\(\)](#), and [GiNaC::EllipticK_eval\(\)](#).

6.23.3.3 is_polynomial()

```
bool GiNaC::constant::is_polynomial (
    const ex & var ) const [override], [virtual]
```

Check whether this is a polynomial in the given variables.

Reimplemented from [GiNaC::basic](#).

6.23.3.4 conjugate()

```
ex GiNaC::constant::conjugate ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::domain::positive](#), and [GiNaC::domain::real](#).

6.23.3.5 real_part()

```
ex GiNaC::constant::real_part ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::domain::positive](#), and [GiNaC::domain::real](#).

6.23.3.6 imag_part()

```
ex GiNaC::constant::imag_part ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::domain::positive](#), and [GiNaC::domain::real](#).

6.23.3.7 archive()

```
void GiNaC::constant::archive (
    archive\_node & n ) const [override], [virtual]
```

Save (serialize) the object into archive node.

Archive the object.

Losely speaking, this method turns an expression into a byte stream (which can be saved and restored later on, or sent via network, etc.)

Reimplemented from [GiNaC::basic](#).

References [n](#), and [name](#).

6.23.3.8 read_archive()

```
void GiNaC::constant::read_archive (
    const archive_node & n,
    lst & syms ) [override], [virtual]
```

Load (deserialize) the object from an archive node.

Construct object from [archive_node](#).

Note

This method is essentially a constructor. However, constructors can't be virtual. So, if unarchiving routines are implemented as constructors one would need to define such a constructor in every class, even if all it does is simply calling constructor of a superclass.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::Catalan](#), [GiNaC::Euler](#), [n](#), [name](#), and [GiNaC::Pi](#).

6.23.3.9 derivative()

```
ex GiNaC::constant::derivative (
    const symbol & s ) const [override], [protected], [virtual]
```

Implementation of [ex::diff\(\)](#) for a constant always returns 0.

See also

[ex::diff](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex0](#).

6.23.3.10 is_equal_same_type()

```
bool GiNaC::constant::is_equal_same_type (
    const basic & other ) const [override], [protected], [virtual]
```

Returns true if two objects of same type are equal.

Normally needs not be reimplemented as long as it wasn't overwritten by some parent class, since it just calls [compare_same_type\(\)](#). The reason why this function exists is that sometimes it is easier to determine equality than an order relation and then it can be overridden.

Reimplemented from [GiNaC::basic](#).

References [GINAC_ASSERT](#), and [serial](#).

6.23.3.11 calchash()

```
unsigned GiNaC::constant::calchash ( ) const [override], [protected], [virtual]
```

Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.

The method inherited from class `basic` computes a hash value based on the type and hash values of possible members. For this reason it is well suited for container classes but atomic classes should override this implementation because otherwise they would all end up with the same hashvalue.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::golden_ratio_hash\(\)](#), [GiNaC::status_flags::hash_calculated](#), [GiNaC::basic::hashvalue](#), [serial](#), and [GiNaC::basic::setflag\(\)](#).

6.23.3.12 get_name()

```
std::string GiNaC::constant::get_name ( ) const [inline]
```

References [name](#).

6.23.3.13 get_TeX_name()

```
std::string GiNaC::constant::get_TeX_name ( ) const [inline]
```

References [TeX_name](#).

6.23.3.14 do_print()

```
void GiNaC::constant::do_print (
    const print\_context & c,
    unsigned level ) const [protected]
```

References [c](#), and [name](#).

6.23.3.15 do_print_tree()

```
void GiNaC::constant::do_print_tree (
    const print\_tree & c,
    unsigned level ) const [protected]
```

References [c](#), [GiNaC::basic::flags](#), [GiNaC::basic::hashvalue](#), and [name](#).

6.23.3.16 do_print_latex()

```
void GiNaC::constant::do_print_latex (
    const print\_latex & c,
    unsigned level ) const [protected]
```

References [c](#), and [TeX_name](#).

6.23.3.17 do_print_python_repr()

```
void GiNaC::constant::do_print_python_repr (
    const print\_python\_repr & c,
    unsigned level ) const [protected]
```

References [c](#), [name](#), and [TeX_name](#).

6.23.4 Member Data Documentation

6.23.4.1 name

```
std::string GiNaC::constant::name [private]
```

printname of this constant

Referenced by [archive\(\)](#), [constant\(\)](#), [constant\(\)](#), [do_print\(\)](#), [do_print_python_repr\(\)](#), [do_print_tree\(\)](#), [get_name\(\)](#), and [read_archive\(\)](#).

6.23.4.2 TeX_name

```
std::string GiNaC::constant::TeX_name [private]
```

LaTeX name.

Referenced by [constant\(\)](#), [constant\(\)](#), [do_print_latex\(\)](#), [do_print_python_repr\(\)](#), and [get_TeX_name\(\)](#).

6.23.4.3 ef

```
evalffuncType GiNaC::constant::ef [private]
```

Referenced by [evalf\(\)](#).

6.23.4.4 number

```
ex GiNaC::constant::number [private]
```

numerical value this constant [evalf\(\)](#)s to

Referenced by [evalf\(\)](#).

6.23.4.5 serial

```
unsigned GiNaC::constant::serial [private]
```

unique serial number for comparison

Referenced by [calchash\(\)](#), and [is_equal_same_type\(\)](#).

6.23.4.6 next_serial

```
unsigned GiNaC::constant::next_serial = 0 [static], [private]
```

6.23.4.7 domain

```
unsigned GiNaC::constant::domain [private]
```

numerical value this constant [evalf\(\)](#)s to

The documentation for this class was generated from the following files:

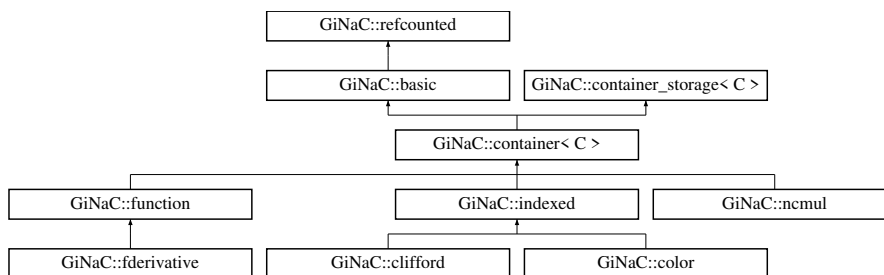
- [constant.h](#)
- [constant.cpp](#)

6.24 GiNaC::container< C > Class Template Reference

Wrapper template for making [GiNaC](#) classes out of STL containers.

```
#include <container.h>
```

Inheritance diagram for GiNaC::container< C >:



Public Types

- typedef `STL::const_iterator` [const_iterator](#)
- typedef `STL::const_reverse_iterator` [const_reverse_iterator](#)

Public Member Functions

- [container](#) (STLT const &s)
- [container](#) (STLT &&v)
- [container](#) (exvector::const_iterator b, exvector::const_iterator e)
- [container](#) (std::initializer_list< [ex](#) > il)
- bool [info](#) (unsigned inf) const override
Information about the object.
- unsigned [precedence](#) () const override
Return relative operator precedence (for parenthezing output).
- size_t [nops](#) () const override
Number of operands/members.
- [ex op](#) (size_t i) const override
Return operand/member at position i.
- [ex](#) & [let_op](#) (size_t i) override
Return modifiable operand/member at position i.
- [ex subs](#) (const [exmap](#) &m, unsigned [options](#)=0) const override
Substitute a set of objects by arbitrary expressions.
- void [read_archive](#) (const [archive_node](#) &n, [lst](#) &sym_lst) override
Load (deserialize) the object from an archive node.
- void [archive](#) ([archive_node](#) &n) const override
Archive the object.
- [container](#) & [prepend](#) (const [ex](#) &b)
Add element at front.
- [container](#) & [append](#) (const [ex](#) &b)
Add element at back.
- [container](#) & [remove_first](#) ()
Remove first element.
- [container](#) & [remove_last](#) ()
Remove last element.
- [container](#) & [remove_all](#) ()
Remove all elements.
- [container](#) & [sort](#) ()
Sort elements.
- [container](#) & [unique](#) ()
Remove adjacent duplicate elements.
- [const_iterator](#) [begin](#) () const
- [const_iterator](#) [end](#) () const
- [const_reverse_iterator](#) [rbegin](#) () const
- [const_reverse_iterator](#) [rend](#) () const

Public Member Functions inherited from [GiNaC::basic](#)

- virtual [~basic](#) ()
basic destructor, virtual because class [ex](#) will delete objects of derived classes via a [basic](#).*
- [basic](#) (const [basic](#) &other)
- const [basic](#) & [operator=](#) (const [basic](#) &other)
basic assignment operator: the other object might be of a derived class.
- virtual [basic](#) * [duplicate](#) () const
Create a clone of this object on the heap.
- virtual [ex eval](#) () const

- Perform automatic non-interruptive term rewriting rules.*
- virtual `ex evalf () const`
Evaluate object numerically.
- virtual `ex evalm () const`
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ () const`
Evaluate integrals, if result is known.
- virtual `ex eval_indexed (const basic &i) const`
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void `print (const print_context &c, unsigned level=0) const`
Output to stream.
- virtual void `dbgprint () const`
Little wrapper around print to be called within a debugger.
- virtual void `dbgprinntree () const`
Little wrapper around printtree to be called within a debugger.
- virtual `ex operator[] (const ex &index) const`
- virtual `ex operator[] (size_t i) const`
- virtual `ex & operator[] (const ex &index)`
- virtual `ex & operator[] (size_t i)`
- virtual bool `has (const ex &other, unsigned options=0) const`
Test for occurrence of a pattern.
- virtual bool `match (const ex &pattern, exmap &repls) const`
Check whether the expression matches a given pattern.
- virtual `ex map (map_function &f) const`
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void `accept (GiNaC::visitor &v) const`
- virtual bool `is_polynomial (const ex &var) const`
Check whether this is a polynomial in the given variables.
- virtual int `degree (const ex &s) const`
Return degree of highest power in object s.
- virtual int `ldegree (const ex &s) const`
Return degree of lowest power in object s.
- virtual `ex coeff (const ex &s, int n=1) const`
Return coefficient of degree n in object s.
- virtual `ex expand (unsigned options=0) const`
Expand expression, i.e.
- virtual `ex collect (const ex &s, bool distributed=false) const`
Sort expanded expression in terms of powers of some object(s).
- virtual `ex series (const relational &r, int order, unsigned options=0) const`
Default implementation of `ex::series()`.
- virtual `ex normal (exmap &repl, exmap &rev_lookup, lst &modifier) const`
Default implementation of `ex::normal()`.
- virtual `ex to_rational (exmap &repl) const`
Default implementation of `ex::to_rational()`.
- virtual `ex to_polynomial (exmap &repl) const`
- virtual `numeric integer_content () const`
- virtual `ex smod (const numeric &xi) const`
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient () const`
Implementation `ex::max_coefficient()`.
- virtual `exvector get_free_indices () const`

- *Return a vector containing the free indices of an expression.*
- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
- *Add two indexed expressions.*
- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
- *Multiply an indexed expression with a scalar.*
- virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const
- *Try to contract two indexed expressions that appear in the same product.*
- virtual unsigned `return_type` () const
- virtual `return_type_t return_type_tinfo` () const
- template<class T >
void `print_dispatch` (const `print_context` &c, unsigned level) const
- *Like `print()`, but dispatch to the specified class.*
- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const
- *Like `print()`, but dispatch to the specified class.*
- `ex subs_one_level` (const `exmap` &m, unsigned `options`) const
- *Helper function for `subs()`.*
- `ex diff` (const `symbol` &s, unsigned nth=1) const
- *Default interface of nth derivative `ex::diff(s, n)`.*
- int `compare` (const `basic` &other) const
- *Compare objects syntactically to establish canonical ordering.*
- bool `is_equal` (const `basic` &other) const
- *Test for syntactic equality.*
- const `basic` & `hold` () const
- *Stop further evaluation.*
- unsigned `gethash` () const
- const `basic` & `setflag` (unsigned f) const
- *Set some `status_flags`.*
- const `basic` & `clearflag` (unsigned f) const
- *Clear some `status_flags`.*

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Protected Types

- typedef `container_storage`< C >::STLT `STLT`

Protected Types inherited from `GiNaC::container_storage< C >`

- typedef C< `ex` > `STLT`

Protected Member Functions

- `ex conjugate ()` const override
- `ex real_part ()` const override
- `ex imag_part ()` const override
- `bool is_equal_same_type` (const `basic` &other) const override
Returns true if two objects of same type are equal.
- virtual `ex thiscontainer` (const `STLT` &v) const
Similar to `duplicate()`, but with a preset sequence.
- virtual `ex thiscontainer` (`STLT` &&v) const
Similar to `duplicate()`, but with a preset sequence (which gets pilfered).
- virtual void `printseq` (const `print_context` &c, char openbracket, char delim, char closebracket, unsigned this←_precedence, unsigned upper_precedence=0) const
Print sequence of contained elements.
- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
- void `do_print_python` (const `print_python` &c, unsigned level) const
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
- `STLT` `subchildren` (const `exmap` &m, unsigned `options`=0) const

Protected Member Functions inherited from `GiNaC::basic`

- `basic ()`
- virtual `ex eval_ncmul` (const `exvector` &v) const
- virtual `bool match_same_type` (const `basic` &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual `ex derivative` (const `symbol` &s) const
Default implementation of `ex::diff()`.
- virtual `int compare_same_type` (const `basic` &other) const
Returns order relation between two objects of same type.
- virtual unsigned `calchash` () const
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- void `ensure_if_modifiable` () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void `do_print` (const `print_context` &c, unsigned level) const
Default output to stream.
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
Tree output to stream.
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
Python parsable output to stream.

Protected Member Functions inherited from `GiNaC::container_storage< C >`

- `container_storage ()`
- `container_storage` (size_t n, const `ex` &e)
- `container_storage` (std::initializer_list< `ex` > il)
- template<class In >
 `container_storage` (In b, In e)
- void `reserve` (size_t)
- `~container_storage ()`
- void `reserve` (size_t n)
- void `reserve` (std::vector< `ex` > &v, size_t n)

Static Protected Member Functions

- static unsigned [get_default_flags](#) ()
*Specialization of [container::get_default_flags\(\)](#) for *lst*.*
- static char [get_open_delim](#) ()
*Specialization of [container::get_open_delim\(\)](#) for *lst*.*
- static char [get_close_delim](#) ()
*Specialization of [container::get_close_delim\(\)](#) for *lst*.*

Static Protected Member Functions inherited from [GiNaC::container_storage< C >](#)

- static void [reserve](#) (STLT &, size_t)

Private Member Functions

- void [sort_](#) (std::random_access_iterator_tag)
- void [sort_](#) (std::input_iterator_tag)
- void [unique_](#) ()
- void [unique_](#) ()
*Specialization of [container::unique_\(\)](#) for *std::list*.*

Additional Inherited Members

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

Protected Attributes inherited from [GiNaC::container_storage< C >](#)

- [STLT seq](#)

6.24.1 Detailed Description

```
template<template< class T, class=std::allocator< T > > class C>
class GiNaC::container< C >
```

Wrapper template for making [GiNaC](#) classes out of STL containers.

6.24.2 Member Typedef Documentation

6.24.2.1 STLT

```
template<template< class T, class=std::allocator< T > > class C>
typedef container\_storage<C>::STLT GiNaC::container< C >::STLT [protected]
```


6.24.2.2 const_iterator

```
template<template< class T, class=std::allocator< T > > class C>
typedef STLT::const_iterator GiNaC::container< C >::const\_iterator
```

6.24.2.3 const_reverse_iterator

```
template<template< class T, class=std::allocator< T > > class C>
typedef STLT::const_reverse_iterator GiNaC::container< C >::const\_reverse\_iterator
```

6.24.3 Constructor & Destructor Documentation

6.24.3.1 container() [1/4]

```
template<template< class T, class=std::allocator< T > > class C>
GiNaC::container< C >::container (
    STLT const & s ) [inline]
```

References [GiNaC::container< C >::get_default_flags\(\)](#), [GiNaC::container_storage< C >::seq](#), and [GiNaC::basic::setflag\(\)](#).

6.24.3.2 container() [2/4]

```
template<template< class T, class=std::allocator< T > > class C>
GiNaC::container< C >::container (
    STLT && v ) [inline], [explicit]
```

References [GiNaC::container< C >::get_default_flags\(\)](#), [GiNaC::container_storage< C >::seq](#), and [GiNaC::basic::setflag\(\)](#).

6.24.3.3 container() [3/4]

```
template<template< class T, class=std::allocator< T > > class C>
GiNaC::container< C >::container (
    exvector::const_iterator b,
    exvector::const_iterator e ) [inline]
```

References [GiNaC::container< C >::get_default_flags\(\)](#), and [GiNaC::basic::setflag\(\)](#).

6.24.3.4 container() [4/4]

```
template<template< class T, class=std::allocator< T > > class C>
GiNaC::container< C >::container (
    std::initializer_list< ex > il ) [inline]
```

References [GiNaC::container< C >::get_default_flags\(\)](#), and [GiNaC::basic::setflag\(\)](#).

6.24.4 Member Function Documentation

6.24.4.1 `get_default_flags()`

```
unsigned GiNaC::lst::get_default_flags ( ) [inline], [static], [protected]
```

Specialization of [container::get_default_flags\(\)](#) for `lst`.

Referenced by [GiNaC::container< C >::container\(\)](#), [GiNaC::container< C >::container\(\)](#), [GiNaC::container< C >::container\(\)](#), [GiNaC::container< C >::container\(\)](#), and [GiNaC::container< C >::read_archive\(\)](#).

6.24.4.2 `get_open_delim()`

```
char GiNaC::lst::get_open_delim ( ) [inline], [static], [protected]
```

Specialization of [container::get_open_delim\(\)](#) for `lst`.

6.24.4.3 `get_close_delim()`

```
char GiNaC::lst::get_close_delim ( ) [inline], [static], [protected]
```

Specialization of [container::get_close_delim\(\)](#) for `lst`.

6.24.4.4 `info()`

```
template bool GiNaC::container< C >::info (
    unsigned inf ) const [inline], [override], [virtual]
```

Information about the object.

See also

class [info_flags](#)

Reimplemented from [GiNaC::basic](#).

Reimplemented in [GiNaC::function](#), [GiNaC::indexed](#), and [GiNaC::ncmul](#).

6.24.4.5 `precedence()`

```
template<template< class T, class=std::allocator< T > > class C>
unsigned GiNaC::container< C >::precedence ( ) const [inline], [override], [virtual]
```

Return relative operator precedence (for parenthezing output).

Reimplemented from [GiNaC::basic](#).

Reimplemented in [GiNaC::function](#), [GiNaC::indexed](#), and [GiNaC::ncmul](#).

Referenced by [GiNaC::fderivative::do_print\(\)](#), [GiNaC::fderivative::do_print_csrf\(\)](#), [GiNaC::fderivative::do_print_latex\(\)](#), and [GiNaC::function::print\(\)](#).

6.24.4.6 nops()

```
template<template< class T, class=std::allocator< T > > class C>
size_t GiNaC::container< C >::nops ( ) const [inline], [override], [virtual]
```

Number of operands/members.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::container_storage< C >::seq](#).

Referenced by [GiNaC::function::calchash\(\)](#), [GiNaC::ncmul::conjugate\(\)](#), [GiNaC::ex::denom\(\)](#), [GiNaC::diag_matrix\(\)](#), [GiNaC::fderivative::do_print_tree\(\)](#), [GiNaC::G3_eval\(\)](#), [GiNaC::G3_evalf\(\)](#), [GiNaC::ncmul::get_free_indices\(\)](#), [GiNaC::H_evalf\(\)](#), [GiNaC::container< C >::imag_part\(\)](#), [GiNaC::is_discriminant_of_quadratic_number_field\(\)](#), [GiNaC::iterated_integral_evalf_impl\(\)](#), [GiNaC::Gt::kernel::kernel\(\)](#), [GiNaC::Gt::lst_evaluate\(\)](#), [GiNaC::lst_to_matrix\(\)](#), [GiNaC::ex::normal\(\)](#), [GiNaC::basic::normal\(\)](#), [GiNaC::add::normal\(\)](#), [GiNaC::mul::normal\(\)](#), [GiNaC::power::normal\(\)](#), [GiNaC::ex::numer\(\)](#), [GiNaC::ex::numer_denom\(\)](#), [GiNaC::function::print\(\)](#), [GiNaC::container< C >::real_part\(\)](#), [GiNaC::sqrfree\(\)](#), and [GiNaC::ex::subs\(\)](#).

6.24.4.7 op()

```
template<template< class T, class=std::allocator< T > > class C>
ex GiNaC::container< C >::op (
    size_t i ) const [override], [virtual]
```

Return operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [GINAC_ASSERT](#), and [GiNaC::nops\(\)](#).

Referenced by [GiNaC::function::calchash\(\)](#), [GiNaC::ex::denom\(\)](#), [GiNaC::mul::expand\(\)](#), [GiNaC::ncmul::expand\(\)](#), [GiNaC::ncmul::get_free_indices\(\)](#), [GiNaC::H_evalf\(\)](#), [GiNaC::indexed::imag_part\(\)](#), [GiNaC::is_discriminant_of_quadratic_number_field\(\)](#), [GiNaC::Gt::kernel::kernel\(\)](#), [GiNaC::kronecker_symbol\(\)](#), [GiNaC::Gt::lst_evaluate\(\)](#), [GiNaC::ex::normal\(\)](#), [GiNaC::basic::normal\(\)](#), [GiNaC::add::normal\(\)](#), [GiNaC::mul::normal\(\)](#), [GiNaC::power::normal\(\)](#), [GiNaC::ex::numer\(\)](#), [GiNaC::ex::numer_denom\(\)](#), [GiNaC::indexed::real_part\(\)](#), [GiNaC::rename_dummy_indices_uniquely\(\)](#), [GiNaC::rename_dummy_indices_uniquely\(\)](#), [GiNaC::rename_dummy_indices_uniquely\(\)](#), [GiNaC::indexed::return_type\(\)](#), [GiNaC::indexed::return_type_tinfo\(\)](#), [GiNaC::sqrfree\(\)](#), [GiNaC::symm\(\)](#), and [GiNaC::symmetrize_cyclic\(\)](#).

6.24.4.8 let_op()

```
template<template< class T, class=std::allocator< T > > class C>
ex & GiNaC::container< C >::let_op (
    size_t i ) [override], [virtual]
```

Return modifiable operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [GINAC_ASSERT](#), and [GiNaC::nops\(\)](#).

6.24.4.9 subs()

```
template<template< class T, class=std::allocator< T > > class C>
ex GiNaC::container< C >::subs (
    const exmap & m,
    unsigned options = 0 ) const [override], [virtual]
```

Substitute a set of objects by arbitrary expressions.

The ex returned will already be evaluated.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::is_a\(\)](#), [m](#), and [options](#).

6.24.4.10 read_archive()

```
template<template< class T, class=std::allocator< T > > class C>
void GiNaC::container< C >::read_archive (
    const archive_node & n,
    lst & syms ) [inline], [override], [virtual]
```

Load (deserialize) the object from an archive node.

Construct object from [archive_node](#).

Note

This method is essentially a constructor. However, constructors can't be virtual. So, if unarchiving routines are implemented as constructors one would need to define such a constructor in every class, even if all it does is simply calling constructor of a superclass.

Reimplemented from [GiNaC::basic](#).

Reimplemented in [GiNaC::fderivative](#), [GiNaC::function](#), and [GiNaC::indexed](#).

References [GiNaC::container< C >::get_default_flags\(\)](#), [n](#), [GiNaC::container_storage< C >::reserve\(\)](#), [GiNaC::container_storage< C >::seq\(\)](#), and [GiNaC::basic::setflag\(\)](#).

6.24.4.11 archive()

```
template<template< class T, class=std::allocator< T > > class C>
void GiNaC::container< C >::archive (
    archive_node & n ) const [inline], [override], [virtual]
```

Archive the object.

Reimplemented from [GiNaC::basic](#).

Reimplemented in [GiNaC::fderivative](#), [GiNaC::function](#), and [GiNaC::indexed](#).

References [GiNaC::archive_node::add_ex\(\)](#), [n](#), and [GiNaC::container_storage< C >::seq\(\)](#).

6.24.4.12 conjugate()

```
template<template< class T, class=std::allocator< T > > class C>
ex GiNaC::container< C >::conjugate ( ) const [inline], [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

Reimplemented in [GiNaC::function](#), and [GiNaC::ncmul](#).

References [GiNaC::are_ex_trivially_equal\(\)](#), [GiNaC::ex::conjugate\(\)](#), [GiNaC::container_storage< C >::reserve\(\)](#), [GiNaC::container_storage< C >::seq](#), [GiNaC::container< C >::thiscontainer\(\)](#), and [x](#).

Referenced by [GiNaC::ncmul::conjugate\(\)](#).

6.24.4.13 real_part()

```
template<template< class T, class=std::allocator< T > > class C>
ex GiNaC::container< C >::real_part ( ) const [inline], [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

Reimplemented in [GiNaC::function](#), [GiNaC::indexed](#), and [GiNaC::ncmul](#).

References [GiNaC::container< C >::begin\(\)](#), [cont](#), [GiNaC::container< C >::end\(\)](#), [GiNaC::container< C >::nops\(\)](#), [GiNaC::container_storage< C >::reserve\(\)](#), and [GiNaC::container< C >::thiscontainer\(\)](#).

6.24.4.14 imag_part()

```
template<template< class T, class=std::allocator< T > > class C>
ex GiNaC::container< C >::imag_part ( ) const [inline], [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

Reimplemented in [GiNaC::function](#), [GiNaC::indexed](#), and [GiNaC::ncmul](#).

References [GiNaC::container< C >::begin\(\)](#), [cont](#), [GiNaC::container< C >::end\(\)](#), [GiNaC::container< C >::nops\(\)](#), [GiNaC::container_storage< C >::reserve\(\)](#), and [GiNaC::container< C >::thiscontainer\(\)](#).

6.24.4.15 is_equal_same_type()

```
template<template< class T, class=std::allocator< T > > class C>
bool GiNaC::container< C >::is_equal_same_type (
    const basic & other ) const [override], [protected], [virtual]
```

Returns true if two objects of same type are equal.

Normally needs not be reimplemented as long as it wasn't overwritten by some parent class, since it just calls [compare_same_type\(\)](#). The reason why this function exists is that sometimes it is easier to determine equality than an order relation and then it can be overridden.

Reimplemented from [GiNaC::basic](#).

Reimplemented in [GiNaC::fderivative](#), and [GiNaC::function](#).

References [GINAC_ASSERT](#), and [GiNaC::container_storage< C >::seq](#).

Referenced by [GiNaC::function::is_equal_same_type\(\)](#).

6.24.4.16 thiscontainer() [1/2]

```
template<template< class T, class=std::allocator< T > > class C>
virtual ex GiNaC::container< C >::thiscontainer (
    const STLT & v ) const [inline], [protected], [virtual]
```

Similar to [duplicate\(\)](#), but with a preset sequence.

Must be overridden by derived classes.

Referenced by [GiNaC::container< C >::conjugate\(\)](#), [GiNaC::container< C >::imag_part\(\)](#), and [GiNaC::container< C >::real_part\(\)](#).

6.24.4.17 thiscontainer() [2/2]

```
template<template< class T, class=std::allocator< T > > class C>
virtual ex GiNaC::container< C >::thiscontainer (
    STLT && v ) const [inline], [protected], [virtual]
```

Similar to [duplicate\(\)](#), but with a preset sequence (which gets pilfered).

Must be overridden by derived classes.

6.24.4.18 printseq()

```
template<template< class T, class=std::allocator< T > > class C>
void GiNaC::container< C >::printseq (
    const print\_context & c,
    char openbracket,
    char delim,
    char closebracket,
    unsigned this_precedence,
    unsigned upper_precedence = 0 ) const [protected], [virtual]
```

Print sequence of contained elements.

References [c](#).

Referenced by [GiNaC::fderivative::do_print\(\)](#), [GiNaC::ncmul::do_print\(\)](#), [GiNaC::ncmul::do_print_csrc\(\)](#), [GiNaC::fderivative::do_print_latex\(\)](#), and [GiNaC::function::print\(\)](#).

6.24.4.19 sort_() [1/2]

```
template<template< class T, class=std::allocator< T > > class C>
void GiNaC::container< C >::sort_ (
    std::random_access_iterator_tag ) [inline], [private]
```

References [GiNaC::container_storage< C >::seq](#).

6.24.4.20 sort_() [2/2]

```
template<template< class T, class=std::allocator< T > > class C>
void GiNaC::container< C >::sort_ (
    std::input_iterator_tag ) [inline], [private]
```

References [GiNaC::container_storage< C >::seq](#).

6.24.4.21 unique_() [1/2]

```
template<template< class T, class=std::allocator< T > > class C>
void GiNaC::container< C >::unique_ ( ) [inline], [private]
```

References [GiNaC::container_storage< C >::seq](#).

6.24.4.22 prepend()

```
template<template< class T, class=std::allocator< T > > class C>
container< C > & GiNaC::container< C >::prepend (
    const ex & b )
```

Add element at front.

6.24.4.23 append()

```
template<template< class T, class=std::allocator< T > > class C>
container< C > & GiNaC::container< C >::append (
    const ex & b )
```

Add element at back.

Referenced by [GiNaC::Gt::archive\(\)](#), [GiNaC::Gt_detail::pathintegral_term::G_path\(\)](#), [GiNaC::H_evalf\(\)](#), [GiNaC::ifactor\(\)](#), [GiNaC::Gt_detail::TransformExpressionWithCache< Type >::impl\(\)](#), [GiNaC::Li_eval\(\)](#), [GiNaC::Isolve\(\)](#), [GiNaC::Gt::lst_evaluate\(\)](#), [GiNaC::symbol::read_archive\(\)](#), [GiNaC::replace_with_symbol\(\)](#), [GiNaC::S_eval\(\)](#), [GiNaC::sqrfree\(\)](#), [GiNaC::symm\(\)](#), and [GiNaC::symmetrize_cyclic\(\)](#).

6.24.4.24 remove_first()

```
template<template< class T, class=std::allocator< T > > class C>
container< C > & GiNaC::container< C >::remove_first ( )
```

Remove first element.

Referenced by [GiNaC::sqrfree\(\)](#), and [GiNaC::symmetrize_cyclic\(\)](#).

6.24.4.25 remove_last()

```
template<template< class T, class=std::allocator< T > > class C>
container< C > & GiNaC::container< C >::remove_last ( )
```

Remove last element.

6.24.4.26 remove_all()

```
template<template< class T, class=std::allocator< T > > class C>
container< C > & GiNaC::container< C >::remove_all ( )
```

Remove all elements.

6.24.4.27 sort()

```
template<template< class T, class=std::allocator< T > > class C>
container< C > & GiNaC::container< C >::sort ( )
```

Sort elements.

6.24.4.28 unique()

```
template<template< class T, class=std::allocator< T > > class C>
container< C > & GiNaC::container< C >::unique ( )
```

Remove adjacent duplicate elements.

6.24.4.29 begin()

```
template<template< class T, class=std::allocator< T > > class C>
const_iterator GiNaC::container< C >::begin ( ) const [inline]
```

References [GiNaC::container_storage< C >::seq](#).

Referenced by [GiNaC::ex::antisymmetrize\(\)](#), [GiNaC::ncmul::conjugate\(\)](#), [GiNaC::G3_eval\(\)](#), [GiNaC::G3_evalf\(\)](#), [GiNaC::H_evalf\(\)](#), [GiNaC::container< C >::imag_part\(\)](#), [GiNaC::kronecker_symbol\(\)](#), [GiNaC::container< C >::real_part\(\)](#), [GiNaC::ex::subs\(\)](#), [GiNaC::ex::symmetrize\(\)](#), [GiNaC::ex::symmetrize_cyclic\(\)](#), [GiNaC::zeta1_evalf\(\)](#), [GiNaC::zeta2_evalf\(\)](#), and [GiNaC::zeta2_print_latex\(\)](#).

6.24.4.30 end()

```
template<template< class T, class=std::allocator< T > > class C>
const_iterator GiNaC::container< C >::end ( ) const [inline]
```

References [GiNaC::container_storage< C >::seq](#).

Referenced by [GiNaC::ex::antisymmetrize\(\)](#), [GiNaC::fderivative::archive\(\)](#), [GiNaC::ncmul::conjugate\(\)](#), [GiNaC::fderivative::do_print\(\)](#), [GiNaC::fderivative::do_print_csrc\(\)](#), [GiNaC::fderivative::do_print_latex\(\)](#), [GiNaC::fderivative::do_print_tree\(\)](#), [GiNaC::ncmul::expandchildren\(\)](#), [GiNaC::H_evalf\(\)](#), [GiNaC::container< C >::imag_part\(\)](#), [GiNaC::kronecker_symbol\(\)](#), [GiNaC::container< C >::real_part\(\)](#), [GiNaC::ncmul::return_type\(\)](#), [GiNaC::ex::subs\(\)](#), [GiNaC::ex::symmetrize\(\)](#), and [GiNaC::ex::symmetrize_cyclic\(\)](#).

6.24.4.31 rbegin()

```
template<template< class T, class=std::allocator< T > > class C>
const_reverse_iterator GiNaC::container< C >::rbegin ( ) const [inline]
```

References [GiNaC::container_storage< C >::seq](#).

6.24.4.32 `rend()`

```
template<template< class T, class=std::allocator< T > > class C>
const_reverse_iterator GiNaC::container< C >::rend ( ) const [inline]
```

References [GiNaC::container_storage< C >::seq](#).

6.24.4.33 `do_print()`

```
template<template< class T, class=std::allocator< T > > class C>
void GiNaC::container< C >::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

References [c](#).

6.24.4.34 `do_print_tree()`

```
template<template< class T, class=std::allocator< T > > class C>
void GiNaC::container< C >::do_print_tree (
    const print_tree & c,
    unsigned level ) const [protected]
```

References [c](#), and [GiNaC::nops\(\)](#).

6.24.4.35 `do_print_python()`

```
template<template< class T, class=std::allocator< T > > class C>
void GiNaC::container< C >::do_print_python (
    const print_python & c,
    unsigned level ) const [protected]
```

References [c](#).

6.24.4.36 `do_print_python_repr()`

```
template<template< class T, class=std::allocator< T > > class C>
void GiNaC::container< C >::do_print_python_repr (
    const print_python_repr & c,
    unsigned level ) const [protected]
```

References [c](#).

6.24.4.37 `subchildren()`

```
template<template< class T, class=std::allocator< T > > class C>
const STLT GiNaC::container< C >::subchildren (
    const exmap & m,
    unsigned options = 0 ) const [protected]
```

References [GiNaC::are_ex_trivially_equal\(\)](#), [m](#), [options](#), and [GiNaC::ex::subs\(\)](#).

6.24.4.38 unique_() [2/2]

```
void GiNaC::container< std::list >::unique_ ( ) [inline], [private]
```

Specialization of `container::unique_()` for `std::list`.

The documentation for this class was generated from the following files:

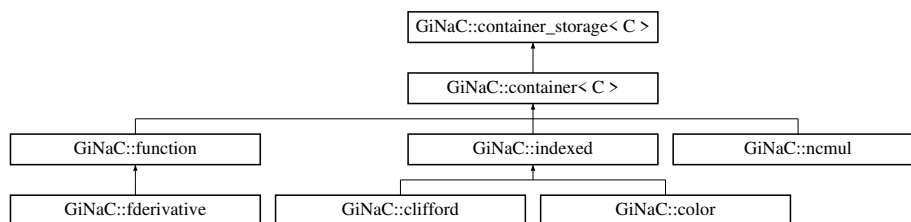
- [container.h](#)
- [exprseq.h](#)
- [lst.h](#)

6.25 GiNaC::container_storage< C > Class Template Reference

Helper template for encapsulating the `reserve()` mechanics of STL containers.

```
#include <container.h>
```

Inheritance diagram for `GiNaC::container_storage< C >`:



Protected Types

- typedef `C< ex >` [STLT](#)

Protected Member Functions

- [container_storage](#) ()
- [container_storage](#) (size_t n, const [ex](#) &e)
- [container_storage](#) (std::initializer_list< [ex](#) > il)
- template<class In >
 [container_storage](#) (In b, In e)
- void [reserve](#) (size_t)
- [~container_storage](#) ()
- void [reserve](#) (size_t n)
- void [reserve](#) (std::vector< [ex](#) > &v, size_t n)

Static Protected Member Functions

- static void [reserve](#) ([STLT](#) &, size_t)

Protected Attributes

- [STLT seq](#)

6.25.1 Detailed Description

```
template<template< class T, class=std::allocator< T > > class C>
class GiNaC::container_storage< C >
```

Helper template for encapsulating the [reserve\(\)](#) mechanics of STL containers.

6.25.2 Member Typedef Documentation

6.25.2.1 STLT

```
template<template< class T, class=std::allocator< T > > class C>
typedef C<ex> GiNaC::container_storage< C >::STLT [protected]
```

6.25.3 Constructor & Destructor Documentation

6.25.3.1 container_storage() [1/4]

```
template<template< class T, class=std::allocator< T > > class C>
GiNaC::container_storage< C >::container_storage ( ) [inline], [protected]
```

6.25.3.2 container_storage() [2/4]

```
template<template< class T, class=std::allocator< T > > class C>
GiNaC::container_storage< C >::container_storage (
    size_t n,
    const ex & e ) [inline], [protected]
```

6.25.3.3 container_storage() [3/4]

```
template<template< class T, class=std::allocator< T > > class C>
GiNaC::container_storage< C >::container_storage (
    std::initializer_list< ex > il ) [inline], [protected]
```

6.25.3.4 container_storage() [4/4]

```
template<template< class T, class=std::allocator< T > > class C>
template<class In >
GiNaC::container_storage< C >::container_storage (
    In b,
    In e ) [inline], [protected]
```

6.25.3.5 ~container_storage()

```
template<template< class T, class=std::allocator< T > > class C>
GiNaC::container_storage< C >::~~container_storage ( ) [inline], [protected]
```

6.25.4 Member Function Documentation

6.25.4.1 reserve() [1/4]

```
template<template< class T, class=std::allocator< T > > class C>
void GiNaC::container_storage< C >::reserve (
    size_t ) [inline], [protected]
```

Referenced by [GiNaC::container< C >::conjugate\(\)](#), [GiNaC::ncmul::eval\(\)](#), [GiNaC::color::eval_ncmul\(\)](#), [GiNaC::container< C >::image\(\)](#), [GiNaC::container< C >::read_archive\(\)](#), [GiNaC::container< C >::real_part\(\)](#), and [GiNaC::rename_dummy_indices_uniquely\(\)](#).

6.25.4.2 reserve() [2/4]

```
template<template< class T, class=std::allocator< T > > class C>
static void GiNaC::container_storage< C >::reserve (
    STLT & ,
    size_t ) [inline], [static], [protected]
```

6.25.4.3 reserve() [3/4]

```
void GiNaC::container_storage< std::vector >::reserve (
    size_t n ) [inline], [protected]
```

References [n](#).

6.25.4.4 reserve() [4/4]

```
void GiNaC::container_storage< std::vector >::reserve (
    std::vector< ex > & v,
    size_t n ) [inline], [protected]
```

References [n](#).

6.25.5 Member Data Documentation

6.25.5.1 seq

```
template<template< class T, class=std::allocator< T > > class C>
STLT GiNaC::container_storage< C >::seq [protected]
```

Referenced by [GiNaC::indexed::all_index_values_are\(\)](#), [GiNaC::container< C >::archive\(\)](#), [GiNaC::container< C >::begin\(\)](#), [GiNaC::ncmul::coeff\(\)](#), [GiNaC::container< C >::conjugate\(\)](#), [GiNaC::function::conjugate\(\)](#), [GiNaC::container< C >::container\(\)](#), [GiNaC::container< C >::container\(\)](#), [GiNaC::ncmul::degree\(\)](#), [GiNaC::fderivative::derivative\(\)](#), [GiNaC::function::derivative\(\)](#), [GiNaC::ncmul::derivative\(\)](#), [GiNaC::clifford::do_print_dflt\(\)](#), [GiNaC::clifford::do_print_latex\(\)](#), [GiNaC::clifford::do_print_tree\(\)](#), [GiNaC::fderivative::do_print_tree\(\)](#), [GiNaC::indexed::do_print_tree\(\)](#), [GiNaC::container< C >::end\(\)](#), [GiNaC::fderivative::eval\(\)](#), [GiNaC::function::eval\(\)](#), [GiNaC::indexed::eval\(\)](#), [GiNaC::ncmul::eval\(\)](#), [GiNaC::function::eval_ncmul\(\)](#), [GiNaC::function::evalf\(\)](#), [GiNaC::ncmul::evalm\(\)](#), [GiNaC::function::expand\(\)](#), [GiNaC::indexed::expand\(\)](#), [GiNaC::ncmul::expand\(\)](#), [GiNaC::ncmul::expandchildren\(\)](#), [GiNaC::indexed::get_dummy_indices\(\)](#), [GiNaC::ncmul::get_factors\(\)](#), [GiNaC::indexed::get_free_indices\(\)](#), [GiNaC::indexed::get_indices\(\)](#), [GiNaC::indexed::has_dummy_index_for\(\)](#), [GiNaC::function::imag_part\(\)](#), [GiNaC::indexed::indexed\(\)](#), [GiNaC::indexed::indexed\(\)](#), [GiNaC::function::info\(\)](#), [GiNaC::indexed::info\(\)](#), [GiNaC::remember_table_entry::is_equal\(\)](#), [GiNaC::container< C >::is_equal_same_type\(\)](#), [GiNaC::ncmul::ldegree\(\)](#), [GiNaC::container< C >::nops\(\)](#), [GiNaC::function::print\(\)](#), [GiNaC::indexed::print_indexed\(\)](#), [GiNaC::indexed::printindices\(\)](#), [GiNaC::container< C >::rbegin\(\)](#), [GiNaC::container< C >::read_archive\(\)](#), [GiNaC::function::read_archive\(\)](#), [GiNaC::indexed::read_archive\(\)](#), [GiNaC::function::real_part\(\)](#), [GiNaC::container< C >::rend\(\)](#), [GiNaC::function::return_type\(\)](#), [GiNaC::ncmul::return_type\(\)](#), [GiNaC::function::return_type_tinfo\(\)](#), [GiNaC::ncmul::return_type_tinfo\(\)](#), [GiNaC::function::series\(\)](#), [GiNaC::container< C >::sort_\(\)](#), [GiNaC::container< C >::sort_\(\)](#), [GiNaC::container< C >::unique_\(\)](#), and [GiNaC::indexed::validate\(\)](#).

The documentation for this class was generated from the following file:

- [container.h](#)

6.26 GiNaC::composition_generator::coolmulti Struct Reference

Classes

- struct [element](#)

Public Member Functions

- [coolmulti](#) (const std::vector< unsigned > &partition)
- [~coolmulti](#) ()
- void [next_permutation](#) ()
- bool [finished](#) () const

Public Attributes

- [element](#) * [head](#)
- [element](#) * [i](#)
- [element](#) * [after_i](#)

6.26.1 Constructor & Destructor Documentation

6.26.1.1 coolmulti()

```
GiNaC::composition_generator::coolmulti::coolmulti (
    const std::vector< unsigned > & partition ) [inline], [explicit]
```

References [after_i](#), [head](#), [i](#), [n](#), and [GiNaC::composition_generator::coolmulti::element::next](#).

6.26.1.2 ~coolmulti()

```
GiNaC::composition_generator::coolmulti::~~coolmulti ( ) [inline]
```

References [head](#).

6.26.2 Member Function Documentation

6.26.2.1 next_permutation()

```
void GiNaC::composition_generator::coolmulti::next_permutation ( ) [inline]
```

References [after_i](#), [head](#), [i](#), [k](#), [GiNaC::composition_generator::coolmulti::element::next](#), and [GiNaC::composition_generator::coolmulti::element::next](#).

Referenced by [GiNaC::composition_generator::next\(\)](#).

6.26.2.2 finished()

```
bool GiNaC::composition_generator::coolmulti::finished ( ) const [inline]
```

References [after_i](#), [head](#), [GiNaC::composition_generator::coolmulti::element::next](#), and [GiNaC::composition_generator::coolmulti::element::next](#).

Referenced by [GiNaC::composition_generator::next\(\)](#).

6.26.3 Member Data Documentation

6.26.3.1 head

```
element* GiNaC::composition_generator::coolmulti::head
```

Referenced by [coolmulti\(\)](#), [finished\(\)](#), [GiNaC::composition_generator::get\(\)](#), [next_permutation\(\)](#), and [~coolmulti\(\)](#).

6.26.3.2 i

```
element * GiNaC::composition_generator::coolmulti::i
```

Referenced by [coolmulti\(\)](#), and [next_permutation\(\)](#).

6.26.3.3 after_i

`element` * `GiNaC::composition_generator::coolmulti::after_i`

Referenced by `coolmulti()`, `finished()`, and `next_permutation()`.

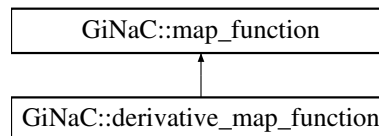
The documentation for this struct was generated from the following file:

- `utils.h`

6.27 GiNaC::derivative_map_function Struct Reference

Function object to be applied by `basic::derivative()`.

Inheritance diagram for `GiNaC::derivative_map_function`:



Public Member Functions

- `derivative_map_function` (const `symbol` &sym)
- `ex operator()` (const `ex` &e) override

Public Member Functions inherited from `GiNaC::map_function`

- virtual `~map_function` ()

Public Attributes

- const `symbol` & `s`

Additional Inherited Members

Public Types inherited from `GiNaC::map_function`

- typedef const `ex` & `argument_type`
- typedef `ex` `result_type`

6.27.1 Detailed Description

Function object to be applied by `basic::derivative()`.

6.27.2 Constructor & Destructor Documentation

6.27.2.1 derivative_map_function()

```
GiNaC::derivative_map_function::derivative_map_function (
    const symbol & sym ) [inline]
```

6.27.3 Member Function Documentation

6.27.3.1 operator()()

```
ex GiNaC::derivative_map_function::operator() (
    const ex & e ) [inline], [override], [virtual]
```

Implements [GiNaC::map_function](#).

References [GiNaC::diff\(\)](#).

6.27.4 Member Data Documentation

6.27.4.1 s

```
const symbol& GiNaC::derivative_map_function::s
```

The documentation for this struct was generated from the following file:

- [basic.cpp](#)

6.28 GiNaC::determinant_algo Class Reference

Switch to control algorithm for determinant computation.

```
#include <flags.h>
```

Public Types

- enum {
[automatic](#) , [gauss](#) , [divfree](#) , [laplace](#) ,
[bareiss](#) }

6.28.1 Detailed Description

Switch to control algorithm for determinant computation.

6.28.2 Member Enumeration Documentation

6.28.2.1 anonymous enum

```
anonymous enum
```


Enumerator

automatic	Let the system choose. A heuristics is applied for automatic determination of a suitable algorithm.
gauss	Gauss elimination. If $m_{i,j}^{(0)}$ are the entries of the original matrix, then the matrix is transformed into triangular form by applying the rules $m_{i,j}^{(k+1)} = m_{i,j}^{(k)} - m_{i,k}^{(k)} m_{k,j}^{(k)} / m_{k,k}^{(k)}$ The determinant is then just the product of diagonal elements. Choose this algorithm only for purely numerical matrices.
divfree	Division-free elimination. This is a modification of Gauss elimination where the division by the pivot element is not carried out. If $m_{i,j}^{(0)}$ are the entries of the original matrix, then the matrix is transformed into triangular form by applying the rules $m_{i,j}^{(k+1)} = m_{i,j}^{(k)} m_{k,k}^{(k)} - m_{i,k}^{(k)} m_{k,j}^{(k)}$ The determinant can later be computed by inspecting the diagonal elements only. This algorithm is only there for the purpose of cross-checks. It is never fast.
laplace	Laplace elimination. This is plain recursive elimination along minors although multiple minors are avoided by the algorithm. Although the algorithm is exponential in complexity it is frequently the fastest one when the matrix is populated by complicated symbolic expressions.
bareiss	Bareiss fraction-free elimination. This is a modification of Gauss elimination where the division by the pivot element is <i>delayed</i> until it can be carried out without computing GCDs. If $m_{i,j}^{(0)}$ are the entries of the original matrix, then the matrix is transformed into triangular form by applying the rules $m_{i,j}^{(k+1)} = (m_{i,j}^{(k)} m_{k,k}^{(k)} - m_{i,k}^{(k)} m_{k,j}^{(k)}) / m_{k-1,k-1}^{(k-1)}$ (We have set $m_{-1,-1}^{(-1)} = 1$ in order to avoid a case distinction in above formula.) It can be shown that nothing more than polynomial long division is needed for carrying out the division. The determinant can then be read of from the lower right entry. This algorithm is rarely fast for computing determinants.

The documentation for this class was generated from the following file:

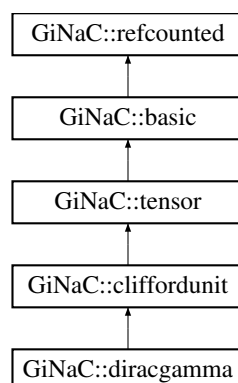
- [flags.h](#)

6.29 GiNaC::diracgamma Class Reference

This class represents the Dirac gamma Lorentz vector.

```
#include <clifford.h>
```

Inheritance diagram for GiNaC::diracgamma:



Public Member Functions

- bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const override
Contraction of a gamma matrix with something else.

Public Member Functions inherited from `GiNaC::tensor`

- bool `replace_contr_index` (exvector::iterator self, exvector::iterator other) const
Replace dummy index in contracted-with object by the contracting object's second index (used internally for delta and metric tensor contractions).

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic` ()
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic` (const `basic` &other)
- const `basic` & `operator=` (const `basic` &other)
basic assignment operator: the other object might be of a derived class.
- virtual `basic` * `duplicate` () const
Create a clone of this object on the heap.
- virtual `ex eval` () const
Perform automatic non-interruptive term rewriting rules.
- virtual `ex evalf` () const
Evaluate object numerically.
- virtual `ex evalm` () const
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ` () const
Evaluate integrals, if result is known.
- virtual `ex eval_indexed` (const `basic` &i) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void `print` (const `print_context` &c, unsigned level=0) const
Output to stream.
- virtual void `dbgprint` () const
Little wrapper around print to be called within a debugger.
- virtual void `dbgprinttree` () const
Little wrapper around printtree to be called within a debugger.
- virtual unsigned `precedence` () const
Return relative operator precedence (for parenthezing output).
- virtual bool `info` (unsigned inf) const
Information about the object.
- virtual size_t `nops` () const
Number of operands/members.
- virtual `ex op` (size_t i) const
Return operand/member at position i.
- virtual `ex operator[]` (const `ex` &index) const
- virtual `ex operator[]` (size_t i) const
- virtual `ex` & `let_op` (size_t i)
Return modifiable operand/member at position i.
- virtual `ex` & `operator[]` (const `ex` &index)
- virtual `ex` & `operator[]` (size_t i)

- virtual bool [has](#) (const [ex](#) &other, unsigned [options](#)=0) const
Test for occurrence of a pattern.
- virtual bool [match](#) (const [ex](#) &pattern, [exmap](#) &repls) const
Check whether the expression matches a given pattern.
- virtual [ex subs](#) (const [exmap](#) &m, unsigned [options](#)=0) const
Substitute a set of objects by arbitrary expressions.
- virtual [ex map](#) ([map_function](#) &f) const
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void [accept](#) ([GiNaC::visitor](#) &v) const
- virtual bool [is_polynomial](#) (const [ex](#) &var) const
Check whether this is a polynomial in the given variables.
- virtual int [degree](#) (const [ex](#) &s) const
Return degree of highest power in object s.
- virtual int [ldegree](#) (const [ex](#) &s) const
Return degree of lowest power in object s.
- virtual [ex coeff](#) (const [ex](#) &s, int [n](#)=1) const
Return coefficient of degree n in object s.
- virtual [ex expand](#) (unsigned [options](#)=0) const
Expand expression, i.e.
- virtual [ex collect](#) (const [ex](#) &s, bool distributed=false) const
Sort expanded expression in terms of powers of some object(s).
- virtual [ex series](#) (const [relational](#) &r, int [order](#), unsigned [options](#)=0) const
Default implementation of [ex::series\(\)](#).
- virtual [ex normal](#) ([exmap](#) &repl, [exmap](#) &rev_lookup, [lst](#) &modifier) const
Default implementation of [ex::normal\(\)](#).
- virtual [ex to_rational](#) ([exmap](#) &repl) const
Default implementation of [ex::to_rational\(\)](#).
- virtual [ex to_polynomial](#) ([exmap](#) &repl) const
- virtual [numeric integer_content](#) () const
- virtual [ex smod](#) (const [numeric](#) &xi) const
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual [numeric max_coefficient](#) () const
Implementation [ex::max_coefficient\(\)](#).
- virtual [exvector get_free_indices](#) () const
Return a vector containing the free indices of an expression.
- virtual [ex add_indexed](#) (const [ex](#) &self, const [ex](#) &other) const
Add two indexed expressions.
- virtual [ex scalar_mul_indexed](#) (const [ex](#) &self, const [numeric](#) &other) const
Multiply an indexed expression with a scalar.
- virtual [return_type_t return_type_tinfo](#) () const
- virtual [ex conjugate](#) () const
- virtual [ex real_part](#) () const
- virtual [ex imag_part](#) () const
- template<class T >
void [print_dispatch](#) (const [print_context](#) &c, unsigned level) const
Like [print\(\)](#), but dispatch to the specified class.
- void [print_dispatch](#) (const [registered_class_info](#) &ri, const [print_context](#) &c, unsigned level) const
Like [print\(\)](#), but dispatch to the specified class.
- virtual void [archive](#) ([archive_node](#) &n) const
Save (serialize) the object into archive node.
- virtual void [read_archive](#) (const [archive_node](#) &n, [lst](#) &syms)

- *Load (deserialize) the object from an archive node.*
- `ex subs_one_level` (const `exmap` &`m`, unsigned `options`) const
- *Helper function for `subs()`.*
- `ex diff` (const `symbol` &`s`, unsigned `nth=1`) const
- *Default interface of `nth` derivative `ex::diff(s, n)`.*
- `int compare` (const `basic` &`other`) const
- *Compare objects syntactically to establish canonical ordering.*
- `bool is_equal` (const `basic` &`other`) const
- *Test for syntactic equality.*
- `const basic & hold` () const
- *Stop further evaluation.*
- `unsigned gethash` () const
- `const basic & setflag` (unsigned `f`) const
- *Set some `status_flags`.*
- `const basic & clearflag` (unsigned `f`) const
- *Clear some `status_flags`.*

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- `unsigned int add_reference` () noexcept
- `unsigned int remove_reference` () noexcept
- `unsigned int get_refcount` () const noexcept
- `void set_refcount` (unsigned int `r`) noexcept

Protected Member Functions

- `void do_print` (const `print_context` &`c`, unsigned `level`) const
- `void do_print_latex` (const `print_latex` &`c`, unsigned `level`) const

Protected Member Functions inherited from `GiNaC::cliffordunit`

- `void do_print` (const `print_context` &`c`, unsigned `level`) const
- `void do_print_latex` (const `print_latex` &`c`, unsigned `level`) const

Protected Member Functions inherited from `GiNaC::tensor`

- `unsigned return_type` () const override

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual [ex derivative](#) (const [symbol](#) &s) const
Default implementation of [ex::diff\(\)](#).
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Additional Inherited Members

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.29.1 Detailed Description

This class represents the Dirac gamma Lorentz vector.

6.29.2 Member Function Documentation

6.29.2.1 [contract_with\(\)](#)

```
bool GiNaC::diracgamma::contract_with (
    exvector::iterator self,
    exvector::iterator other,
    exvector & v ) const [override], [virtual]
```

Contraction of a gamma matrix with something else.

Reimplemented from [GiNaC::cliffordunit](#).

6.29.2.2 do_print()

```
void GiNaC::diracgamma::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

6.29.2.3 do_print_latex()

```
void GiNaC::diracgamma::do_print_latex (
    const print_latex & c,
    unsigned level ) const [protected]
```

The documentation for this class was generated from the following files:

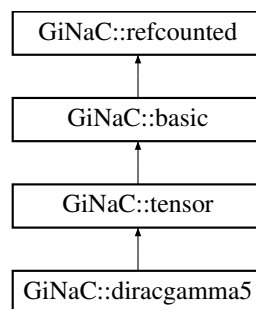
- [clifford.h](#)
- [clifford.cpp](#)

6.30 GiNaC::diracgamma5 Class Reference

This class represents the Dirac gamma5 object which anticommutes with all other gammas.

```
#include <clifford.h>
```

Inheritance diagram for GiNaC::diracgamma5:



Protected Member Functions

- void [do_print](#) (const [print_context](#) &c, unsigned level) const
- void [do_print_latex](#) (const [print_latex](#) &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::tensor](#)

- unsigned [return_type](#) () const override

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual [ex derivative](#) (const [symbol](#) &s) const
Default implementation of [ex::diff\(\)](#).
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Private Member Functions

- [ex conjugate](#) () const override

Additional Inherited Members

Public Member Functions inherited from [GiNaC::tensor](#)

- bool [replace_contr_index](#) ([exvector::iterator](#) self, [exvector::iterator](#) other) const
Replace dummy index in contracted-with object by the contracting object's second index (used internally for delta and metric tensor contractions).

Public Member Functions inherited from [GiNaC::basic](#)

- virtual [~basic](#) ()
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- [basic](#) (const [basic](#) &other)
- const [basic](#) & [operator=](#) (const [basic](#) &other)
basic assignment operator: the other object might be of a derived class.
- virtual [basic](#) * [duplicate](#) () const
Create a clone of this object on the heap.
- virtual [ex eval](#) () const
Perform automatic non-interruptive term rewriting rules.
- virtual [ex evalf](#) () const
Evaluate object numerically.
- virtual [ex evalm](#) () const

- Evaluate sums, products and integer powers of matrices.*

 - virtual `ex eval_integ ()` const
- Evaluate integrals, if result is known.*

 - virtual `ex eval_indexed (const basic &i)` const
- Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.*

 - virtual void `print (const print_context &c, unsigned level=0)` const
- Output to stream.*

 - virtual void `dbgprint ()` const
- Little wrapper around print to be called within a debugger.*

 - virtual void `dbgprinttree ()` const
- Little wrapper around printtree to be called within a debugger.*

 - virtual unsigned `precedence ()` const
- Return relative operator precedence (for parenthezing output).*

 - virtual bool `info (unsigned inf)` const
- Information about the object.*

 - virtual size_t `nops ()` const
- Number of operands/members.*

 - virtual `ex op (size_t i)` const
- Return operand/member at position i.*

 - virtual `ex operator[] (const ex &index)` const
 - virtual `ex operator[] (size_t i)` const
 - virtual `ex & let_op (size_t i)`
- Return modifiable operand/member at position i.*

 - virtual `ex & operator[] (const ex &index)`
 - virtual `ex & operator[] (size_t i)`
- Test for occurrence of a pattern.*

 - virtual bool `has (const ex &other, unsigned options=0)` const
- Test for occurrence of a pattern.*

 - virtual bool `match (const ex &pattern, exmap &repls)` const
- Check whether the expression matches a given pattern.*

 - virtual `ex subs (const exmap &m, unsigned options=0)` const
- Substitute a set of objects by arbitrary expressions.*

 - virtual `ex map (map_function &f)` const
- Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).*

 - virtual void `accept (GiNaC::visitor &v)` const
- Check whether this is a polynomial in the given variables.*

 - virtual bool `is_polynomial (const ex &var)` const
- Return degree of highest power in object s.*

 - virtual int `degree (const ex &s)` const
- Return degree of lowest power in object s.*

 - virtual int `ldegree (const ex &s)` const
- Return coefficient of degree n in object s.*

 - virtual `ex coeff (const ex &s, int n=1)` const
- Expand expression, i.e.*

 - virtual `ex expand (unsigned options=0)` const
- Sort expanded expression in terms of powers of some object(s).*

 - virtual `ex collect (const ex &s, bool distributed=false)` const
- Default implementation of ex::series().*

 - virtual `ex series (const relational &r, int order, unsigned options=0)` const
- Default implementation of ex::normal().*

 - virtual `ex normal (exmap &repl, exmap &rev_lookup, lst &modifier)` const
- Default implementation of ex::normal().*

 - virtual `ex to_rational (exmap &repl)` const

Default implementation of `ex::to_rational()`.

- virtual `ex to_polynomial (exmap &repl) const`
- virtual `numeric integer_content () const`
- virtual `ex smod (const numeric &xi) const`

Apply symmetric modular homomorphism to an expanded multivariate polynomial.

- virtual `numeric max_coefficient () const`

Implementation `ex::max_coefficient()`.

- virtual `exvector get_free_indices () const`

Return a vector containing the free indices of an expression.

- virtual `ex add_indexed (const ex &self, const ex &other) const`

Add two indexed expressions.

- virtual `ex scalar_mul_indexed (const ex &self, const numeric &other) const`

Multiply an indexed expression with a scalar.

- virtual `bool contract_with (exvector::iterator self, exvector::iterator other, exvector &v) const`

Try to contract two indexed expressions that appear in the same product.

- virtual `return_type_t return_type_tinfo () const`
- virtual `ex real_part () const`
- virtual `ex imag_part () const`

- template<class T >

void `print_dispatch (const print_context &c, unsigned level) const`

Like `print()`, but dispatch to the specified class.

- void `print_dispatch (const registered_class_info &ri, const print_context &c, unsigned level) const`

Like `print()`, but dispatch to the specified class.

- virtual void `archive (archive_node &n) const`

Save (serialize) the object into archive node.

- virtual void `read_archive (const archive_node &n, lst &syms)`

Load (deserialize) the object from an archive node.

- `ex subs_one_level (const exmap &m, unsigned options) const`

Helper function for `subs()`.

- `ex diff (const symbol &s, unsigned nth=1) const`

Default interface of `nth` derivative `ex::diff(s, n)`.

- int `compare (const basic &other) const`

Compare objects syntactically to establish canonical ordering.

- bool `is_equal (const basic &other) const`

Test for syntactic equality.

- const `basic & hold () const`

Stop further evaluation.

- unsigned `gethash () const`

- const `basic & setflag (unsigned f) const`

Set some `status_flags`.

- const `basic & clearflag (unsigned f) const`

Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted () noexcept`
- unsigned int `add_reference () noexcept`
- unsigned int `remove_reference () noexcept`
- unsigned int `get_refcount () const noexcept`
- void `set_refcount (unsigned int r) noexcept`

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.30.1 Detailed Description

This class represents the Dirac gamma5 object which anticommutes with all other gammas.

6.30.2 Member Function Documentation

6.30.2.1 conjugate()

```
ex GiNaC::diracgamma5::conjugate ( ) const [override], [private], [virtual]
```

Reimplemented from [GiNaC::basic](#).

6.30.2.2 do_print()

```
void GiNaC::diracgamma5::do_print (
    const print\_context & c,
    unsigned level ) const [protected]
```

6.30.2.3 do_print_latex()

```
void GiNaC::diracgamma5::do_print_latex (
    const print\_latex & c,
    unsigned level ) const [protected]
```

The documentation for this class was generated from the following files:

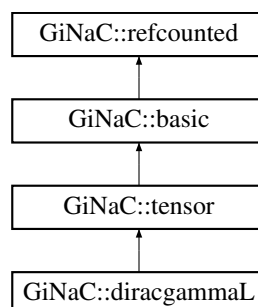
- [clifford.h](#)
- [clifford.cpp](#)

6.31 GiNaC::diracgammaL Class Reference

This class represents the Dirac gammaL object which behaves like $1/2 (1-\text{gamma}5)$.

```
#include <clifford.h>
```

Inheritance diagram for GiNaC::diracgammaL:



Protected Member Functions

- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_latex` (const `print_latex` &c, unsigned level) const

Protected Member Functions inherited from `GiNaC::tensor`

- unsigned `return_type` () const override

Protected Member Functions inherited from `GiNaC::basic`

- `basic` ()
- virtual `ex eval_ncmul` (const `exvector` &v) const
- virtual bool `match_same_type` (const `basic` &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual `ex derivative` (const `symbol` &s) const
Default implementation of `ex::diff()`.
- virtual int `compare_same_type` (const `basic` &other) const
Returns order relation between two objects of same type.
- virtual bool `is_equal_same_type` (const `basic` &other) const
Returns true if two objects of same type are equal.
- virtual unsigned `calchash` () const
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- void `ensure_if_modifiable` () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void `do_print` (const `print_context` &c, unsigned level) const
Default output to stream.
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
Tree output to stream.
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
Python parsable output to stream.

Private Member Functions

- `ex conjugate` () const override

Additional Inherited Members**Public Member Functions inherited from `GiNaC::tensor`**

- bool `replace_contr_index` (`exvector::iterator` self, `exvector::iterator` other) const
Replace dummy index in contracted-with object by the contracting object's second index (used internally for delta and metric tensor contractions).

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic ()`
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic (const basic &other)`
- `const basic & operator= (const basic &other)`
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate () const`
Create a clone of this object on the heap.
- virtual `ex eval () const`
Perform automatic non-interruptive term rewriting rules.
- virtual `ex evalf () const`
Evaluate object numerically.
- virtual `ex evalm () const`
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ () const`
Evaluate integrals, if result is known.
- virtual `ex eval_indexed (const basic &i) const`
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual `void print (const print_context &c, unsigned level=0) const`
Output to stream.
- virtual `void dbgprint () const`
Little wrapper around print to be called within a debugger.
- virtual `void dbgprinttree () const`
Little wrapper around printtree to be called within a debugger.
- virtual `unsigned precedence () const`
Return relative operator precedence (for parenthezing output).
- virtual `bool info (unsigned inf) const`
Information about the object.
- virtual `size_t nops () const`
Number of operands/members.
- virtual `ex op (size_t i) const`
Return operand/member at position i.
- virtual `ex operator[] (const ex &index) const`
- virtual `ex operator[] (size_t i) const`
- virtual `ex & let_op (size_t i)`
Return modifiable operand/member at position i.
- virtual `ex & operator[] (const ex &index)`
- virtual `ex & operator[] (size_t i)`
- virtual `bool has (const ex &other, unsigned options=0) const`
Test for occurrence of a pattern.
- virtual `bool match (const ex &pattern, exmap &repls) const`
Check whether the expression matches a given pattern.
- virtual `ex subs (const exmap &m, unsigned options=0) const`
Substitute a set of objects by arbitrary expressions.
- virtual `ex map (map_function &f) const`
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual `void accept (GiNaC::visitor &v) const`
- virtual `bool is_polynomial (const ex &var) const`
Check whether this is a polynomial in the given variables.
- virtual `int degree (const ex &s) const`

- Return degree of highest power in object s.*

 - virtual int `ldegree` (const `ex` &s) const
- Return degree of lowest power in object s.*

 - virtual `ex coeff` (const `ex` &s, int `n`=1) const
- Return coefficient of degree n in object s.*

 - virtual `ex expand` (unsigned `options`=0) const
- Expand expression, i.e.*

 - virtual `ex collect` (const `ex` &s, bool `distributed`=false) const
- Sort expanded expression in terms of powers of some object(s).*

 - virtual `ex series` (const `relational` &r, int `order`, unsigned `options`=0) const
- Default implementation of `ex::series()`.*

 - virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const
- Default implementation of `ex::normal()`.*

 - virtual `ex to_rational` (`exmap` &repl) const
- Default implementation of `ex::to_rational()`.*

 - virtual `ex to_polynomial` (`exmap` &repl) const
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &xi) const
- Apply symmetric modular homomorphism to an expanded multivariate polynomial.*

 - virtual `numeric max_coefficient` () const
- Implementation `ex::max_coefficient()`.*

 - virtual `exvector get_free_indices` () const
- Return a vector containing the free indices of an expression.*

 - virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
- Add two indexed expressions.*

 - virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
- Multiply an indexed expression with a scalar.*

 - virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const
- Try to contract two indexed expressions that appear in the same product.*

 - virtual `return_type_t return_type_tinfo` () const
- virtual `ex real_part` () const
- virtual `ex imag_part` () const
- template<class T >
 - void `print_dispatch` (const `print_context` &c, unsigned `level`) const
- Like `print()`, but dispatch to the specified class.*

 - void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned `level`) const
- Like `print()`, but dispatch to the specified class.*

 - virtual void `archive` (`archive_node` &n) const
- Save (serialize) the object into archive node.*

 - virtual void `read_archive` (const `archive_node` &n, `lst` &syms)
- Load (deserialize) the object from an archive node.*

 - `ex subs_one_level` (const `exmap` &m, unsigned `options`) const
- Helper function for `subs()`.*

 - `ex diff` (const `symbol` &s, unsigned `nth`=1) const
- Default interface of nth derivative `ex::diff(s, n)`.*

 - int `compare` (const `basic` &other) const
- Compare objects syntactically to establish canonical ordering.*

 - bool `is_equal` (const `basic` &other) const
- Test for syntactic equality.*

 - const `basic` & `hold` () const
- Stop further evaluation.*

- unsigned [gethash](#) () const
- const [basic](#) & [setflag](#) (unsigned f) const
Set some [status_flags](#).
- const [basic](#) & [clearflag](#) (unsigned f) const
Clear some [status_flags](#).

Public Member Functions inherited from [GiNaC::refcounted](#)

- [refcounted](#) () noexcept
- unsigned int [add_reference](#) () noexcept
- unsigned int [remove_reference](#) () noexcept
- unsigned int [get_refcount](#) () const noexcept
- void [set_refcount](#) (unsigned int r) noexcept

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.31.1 Detailed Description

This class represents the Dirac gammaL object which behaves like 1/2 (1-gamma5).

6.31.2 Member Function Documentation

6.31.2.1 conjugate()

```
ex GiNaC::diracgammaL::conjugate ( ) const [override], [private], [virtual]
```

Reimplemented from [GiNaC::basic](#).

6.31.2.2 do_print()

```
void GiNaC::diracgammaL::do_print (
    const print\_context & c,
    unsigned level ) const [protected]
```

6.31.2.3 do_print_latex()

```
void GiNaC::diracgammaL::do_print_latex (
    const print\_latex & c,
    unsigned level ) const [protected]
```

The documentation for this class was generated from the following files:

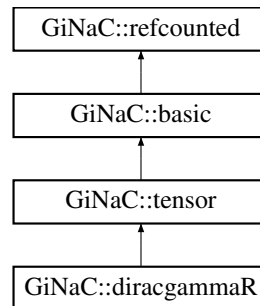
- [clifford.h](#)
- [clifford.cpp](#)

6.32 GiNaC::diracgammaR Class Reference

This class represents the Dirac gammaL object which behaves like $\frac{1}{2} (1+\gamma_5)$.

```
#include <clifford.h>
```

Inheritance diagram for GiNaC::diracgammaR:



Protected Member Functions

- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_latex` (const `print_latex` &c, unsigned level) const

Protected Member Functions inherited from `GiNaC::tensor`

- unsigned `return_type` () const override

Protected Member Functions inherited from `GiNaC::basic`

- `basic` ()
- virtual `ex eval_ncmul` (const `exvector` &v) const
- virtual bool `match_same_type` (const `basic` &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual `ex derivative` (const `symbol` &s) const
Default implementation of `ex::diff()`.
- virtual int `compare_same_type` (const `basic` &other) const
Returns order relation between two objects of same type.
- virtual bool `is_equal_same_type` (const `basic` &other) const
Returns true if two objects of same type are equal.
- virtual unsigned `calchash` () const
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- void `ensure_if_modifiable` () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void `do_print` (const `print_context` &c, unsigned level) const
Default output to stream.
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
Tree output to stream.
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
Python parsable output to stream.

Private Member Functions

- `ex conjugate ()` const override

Additional Inherited Members

Public Member Functions inherited from `GiNaC::tensor`

- `bool replace_contr_index (exvector::iterator self, exvector::iterator other) const`
Replace dummy index in contracted-with object by the contracting object's second index (used internally for delta and metric tensor contractions).

Public Member Functions inherited from `GiNaC::basic`

- `virtual ~basic ()`
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic (const basic &other)`
- `const basic & operator= (const basic &other)`
basic assignment operator: the other object might be of a derived class.
- `virtual basic * duplicate () const`
Create a clone of this object on the heap.
- `virtual ex eval () const`
Perform automatic non-interruptive term rewriting rules.
- `virtual ex evalf () const`
Evaluate object numerically.
- `virtual ex evalm () const`
Evaluate sums, products and integer powers of matrices.
- `virtual ex eval_integ () const`
Evaluate integrals, if result is known.
- `virtual ex eval_indexed (const basic &i) const`
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- `virtual void print (const print_context &c, unsigned level=0) const`
Output to stream.
- `virtual void dbgprint () const`
Little wrapper around print to be called within a debugger.
- `virtual void dbgprinttree () const`
Little wrapper around printtree to be called within a debugger.
- `virtual unsigned precedence () const`
Return relative operator precedence (for parenthezing output).
- `virtual bool info (unsigned inf) const`
Information about the object.
- `virtual size_t nops () const`
Number of operands/members.
- `virtual ex op (size_t i) const`
Return operand/member at position i.
- `virtual ex operator[] (const ex &index) const`
- `virtual ex operator[] (size_t i) const`
- `virtual ex & let_op (size_t i)`
Return modifiable operand/member at position i.
- `virtual ex & operator[] (const ex &index)`

- virtual `ex & operator[]` (size_t i)
- virtual bool `has` (const `ex` &other, unsigned `options`=0) const
Test for occurrence of a pattern.
- virtual bool `match` (const `ex` &pattern, `exmap` &repls) const
Check whether the expression matches a given pattern.
- virtual `ex subs` (const `exmap` &m, unsigned `options`=0) const
Substitute a set of objects by arbitrary expressions.
- virtual `ex map` (`map_function` &f) const
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void `accept` (GiNaC::visitor &v) const
- virtual bool `is_polynomial` (const `ex` &var) const
Check whether this is a polynomial in the given variables.
- virtual int `degree` (const `ex` &s) const
Return degree of highest power in object s.
- virtual int `ldegree` (const `ex` &s) const
Return degree of lowest power in object s.
- virtual `ex coeff` (const `ex` &s, int `n`=1) const
Return coefficient of degree n in object s.
- virtual `ex expand` (unsigned `options`=0) const
Expand expression, i.e.
- virtual `ex collect` (const `ex` &s, bool `distributed`=false) const
Sort expanded expression in terms of powers of some object(s).
- virtual `ex series` (const `relational` &r, int `order`, unsigned `options`=0) const
Default implementation of `ex::series()`.
- virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const
Default implementation of `ex::normal()`.
- virtual `ex to_rational` (`exmap` &repl) const
Default implementation of `ex::to_rational()`.
- virtual `ex to_polynomial` (`exmap` &repl) const
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &xi) const
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient` () const
Implementation `ex::max_coefficient()`.
- virtual `exvector get_free_indices` () const
Return a vector containing the free indices of an expression.
- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
Add two indexed expressions.
- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
Multiply an indexed expression with a scalar.
- virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const
Try to contract two indexed expressions that appear in the same product.
- virtual `return_type_t return_type_info` () const
- virtual `ex real_part` () const
- virtual `ex imag_part` () const
- template<class T >
void `print_dispatch` (const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- virtual void `archive` (`archive_node` &n) const

- *Save (serialize) the object into archive node.*
- virtual void `read_archive` (const `archive_node` &n, `lst` &syms)
- *Load (deserialize) the object from an archive node.*
- `ex subs_one_level` (const `exmap` &m, unsigned `options`) const
- *Helper function for `subs()`.*
- `ex diff` (const `symbol` &s, unsigned `nth`=1) const
- *Default interface of `nth` derivative `ex::diff(s, n)`.*
- int `compare` (const `basic` &other) const
- *Compare objects syntactically to establish canonical ordering.*
- bool `is_equal` (const `basic` &other) const
- *Test for syntactic equality.*
- const `basic` & `hold` () const
- *Stop further evaluation.*
- unsigned `gethash` () const
- const `basic` & `setflag` (unsigned `f`) const
- *Set some `status_flags`.*
- const `basic` & `clearflag` (unsigned `f`) const
- *Clear some `status_flags`.*

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int `r`) noexcept

Protected Attributes inherited from `GiNaC::basic`

- unsigned `flags`
of type `status_flags`
- unsigned `hashvalue`
hash value

6.32.1 Detailed Description

This class represents the Dirac gammaL object which behaves like $1/2 (1+\gamma_5)$.

6.32.2 Member Function Documentation

6.32.2.1 `conjugate()`

```
ex GiNaC::diracgammaR::conjugate ( ) const [override], [private], [virtual]
```

Reimplemented from `GiNaC::basic`.

6.32.2.2 do_print()

```
void GiNaC::diracgammaR::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

6.32.2.3 do_print_latex()

```
void GiNaC::diracgammaR::do_print_latex (
    const print_latex & c,
    unsigned level ) const [protected]
```

The documentation for this class was generated from the following files:

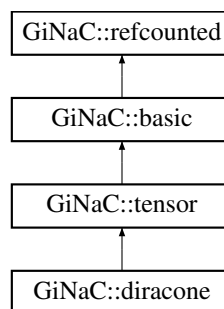
- [clifford.h](#)
- [clifford.cpp](#)

6.33 GiNaC::diracone Class Reference

This class represents the Clifford algebra unity element.

```
#include <clifford.h>
```

Inheritance diagram for GiNaC::diracone:



Protected Member Functions

- void [do_print](#) (const [print_context](#) &c, unsigned level) const
- void [do_print_latex](#) (const [print_latex](#) &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::tensor](#)

- unsigned [return_type](#) () const override

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual [ex derivative](#) (const [symbol](#) &s) const
Default implementation of [ex::diff\(\)](#).
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Additional Inherited Members

Public Member Functions inherited from [GiNaC::tensor](#)

- bool [replace_contr_index](#) (exvector::iterator self, exvector::iterator other) const
Replace dummy index in contracted-with object by the contracting object's second index (used internally for delta and metric tensor contractions).

Public Member Functions inherited from [GiNaC::basic](#)

- virtual [~basic](#) ()
basic destructor, virtual because class [ex](#) will delete objects of derived classes via a [basic](#).*
- [basic](#) (const [basic](#) &other)
- const [basic](#) & [operator=](#) (const [basic](#) &other)
basic assignment operator: the other object might be of a derived class.
- virtual [basic](#) * [duplicate](#) () const
Create a clone of this object on the heap.
- virtual [ex eval](#) () const
Perform automatic non-interruptive term rewriting rules.
- virtual [ex evalf](#) () const
Evaluate object numerically.
- virtual [ex evalm](#) () const
Evaluate sums, products and integer powers of matrices.
- virtual [ex eval_integ](#) () const
Evaluate integrals, if result is known.
- virtual [ex eval_indexed](#) (const [basic](#) &i) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.

- virtual void `print` (const `print_context` &c, unsigned level=0) const
Output to stream.
- virtual void `dbgprint` () const
Little wrapper around print to be called within a debugger.
- virtual void `dbgprnttree` () const
Little wrapper around prnttree to be called within a debugger.
- virtual unsigned `precedence` () const
Return relative operator precedence (for parenthezing output).
- virtual bool `info` (unsigned inf) const
Information about the object.
- virtual size_t `nops` () const
Number of operands/members.
- virtual `ex op` (size_t i) const
Return operand/member at position i.
- virtual `ex operator[]` (const `ex` &index) const
- virtual `ex operator[]` (size_t i) const
- virtual `ex & let_op` (size_t i)
Return modifiable operand/member at position i.
- virtual `ex & operator[]` (const `ex` &index)
- virtual `ex & operator[]` (size_t i)
- virtual bool `has` (const `ex` &other, unsigned `options`=0) const
Test for occurrence of a pattern.
- virtual bool `match` (const `ex` &pattern, `exmap` &repls) const
Check whether the expression matches a given pattern.
- virtual `ex subs` (const `exmap` &m, unsigned `options`=0) const
Substitute a set of objects by arbitrary expressions.
- virtual `ex map` (`map_function` &f) const
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void `accept` (GiNaC::visitor &v) const
- virtual bool `is_polynomial` (const `ex` &var) const
Check whether this is a polynomial in the given variables.
- virtual int `degree` (const `ex` &s) const
Return degree of highest power in object s.
- virtual int `ldegree` (const `ex` &s) const
Return degree of lowest power in object s.
- virtual `ex coeff` (const `ex` &s, int n=1) const
Return coefficient of degree n in object s.
- virtual `ex expand` (unsigned `options`=0) const
Expand expression, i.e.
- virtual `ex collect` (const `ex` &s, bool distributed=false) const
Sort expanded expression in terms of powers of some object(s).
- virtual `ex series` (const `relational` &r, int `order`, unsigned `options`=0) const
Default implementation of `ex::series()`.
- virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const
Default implementation of `ex::normal()`.
- virtual `ex to_rational` (`exmap` &repl) const
Default implementation of `ex::to_rational()`.
- virtual `ex to_polynomial` (`exmap` &repl) const
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &xi) const
Apply symmetric modular homomorphism to an expanded multivariate polynomial.

- virtual [numeric max_coefficient](#) () const
Implementation [ex::max_coefficient\(\)](#).
- virtual [exvector get_free_indices](#) () const
Return a vector containing the free indices of an expression.
- virtual [ex add_indexed](#) (const [ex](#) &self, const [ex](#) &other) const
Add two indexed expressions.
- virtual [ex scalar_mul_indexed](#) (const [ex](#) &self, const [numeric](#) &other) const
Multiply an indexed expression with a scalar.
- virtual bool [contract_with](#) (exvector::iterator self, exvector::iterator other, [exvector](#) &v) const
Try to contract two indexed expressions that appear in the same product.
- virtual [return_type_t return_type_tinfo](#) () const
- virtual [ex conjugate](#) () const
- virtual [ex real_part](#) () const
- virtual [ex imag_part](#) () const
- template<class T >
void [print_dispatch](#) (const [print_context](#) &c, unsigned level) const
Like [print\(\)](#), but dispatch to the specified class.
- void [print_dispatch](#) (const [registered_class_info](#) &ri, const [print_context](#) &c, unsigned level) const
Like [print\(\)](#), but dispatch to the specified class.
- virtual void [archive](#) ([archive_node](#) &n) const
Save (serialize) the object into archive node.
- virtual void [read_archive](#) (const [archive_node](#) &n, [lst](#) &syms)
Load (deserialize) the object from an archive node.
- [ex subs_one_level](#) (const [exmap](#) &m, unsigned [options](#)) const
Helper function for [subs\(\)](#).
- [ex diff](#) (const [symbol](#) &s, unsigned nth=1) const
Default interface of nth derivative [ex::diff\(s, n\)](#).
- int [compare](#) (const [basic](#) &other) const
Compare objects syntactically to establish canonical ordering.
- bool [is_equal](#) (const [basic](#) &other) const
Test for syntactic equality.
- const [basic](#) & [hold](#) () const
Stop further evaluation.
- unsigned [gethash](#) () const
- const [basic](#) & [setflag](#) (unsigned f) const
Set some [status_flags](#).
- const [basic](#) & [clearflag](#) (unsigned f) const
Clear some [status_flags](#).

Public Member Functions inherited from [GiNaC::refcounted](#)

- [refcounted](#) () noexcept
- unsigned int [add_reference](#) () noexcept
- unsigned int [remove_reference](#) () noexcept
- unsigned int [get_refcount](#) () const noexcept
- void [set_refcount](#) (unsigned int r) noexcept

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.33.1 Detailed Description

This class represents the Clifford algebra unity element.

6.33.2 Member Function Documentation

6.33.2.1 do_print()

```
void GiNaC::diracone::do_print (
    const print\_context & c,
    unsigned level ) const [protected]
```

6.33.2.2 do_print_latex()

```
void GiNaC::diracone::do_print_latex (
    const print\_latex & c,
    unsigned level ) const [protected]
```

The documentation for this class was generated from the following file:

- [clifford.h](#)

6.34 GiNaC::do_taylor Class Reference

Exception class thrown by classes which provide their own series expansion to signal that ordinary Taylor expansion is safe.

```
#include <function.h>
```

6.34.1 Detailed Description

Exception class thrown by classes which provide their own series expansion to signal that ordinary Taylor expansion is safe.

The documentation for this class was generated from the following file:

- [function.h](#)

6.35 GiNaC::domain Class Reference

Domain of an object.

```
#include <flags.h>
```

Public Types

- enum { [complex](#) , [real](#) , [positive](#) }

6.35.1 Detailed Description

Domain of an object.

6.35.2 Member Enumeration Documentation

6.35.2.1 anonymous enum

```
anonymous enum
```

Enumerator

complex	
real	
positive	

The documentation for this class was generated from the following file:

- [flags.h](#)

6.36 GiNaC::dunno Class Reference

Exception class thrown by functions to signal unimplemented functionality so the expression may just be .hold()

```
#include <utils.h>
```

6.36.1 Detailed Description

Exception class thrown by functions to signal unimplemented functionality so the expression may just be .hold()

The documentation for this class was generated from the following file:

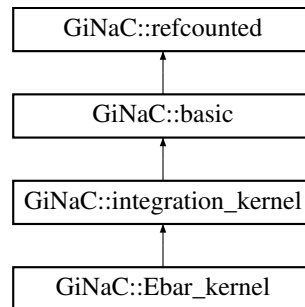
- [utils.h](#)

6.37 GiNaC::Ebar_kernel Class Reference

The Ebar-kernel.

```
#include <integration_kernel.h>
```

Inheritance diagram for GiNaC::Ebar_kernel:



Public Member Functions

- [Ebar_kernel](#) (const [ex](#) &n, const [ex](#) &m, const [ex](#) &x, const [ex](#) &y)
- [size_t nops](#) () const override
Number of operands/members.
- [ex op](#) (size_t i) const override
Return operand/member at position i.
- [ex & let_op](#) (size_t i) override
Return modifiable operand/member at position i.
- [bool is_numeric](#) (void) const override
This routine returns true, if the integration kernel can be evaluated numerically.
- [ex get_numerical_value](#) (const [ex](#) &qbar, int N_trunc=0) const override
Returns the value of Ebar_{n,m}(x,y,qbar)

Public Member Functions inherited from [GiNaC::integration_kernel](#)

- [ex series](#) (const [relational](#) &r, int [order](#), unsigned [options](#)=0) const override
Default implementation of [ex::series\(\)](#).
- virtual [bool has_trailing_zero](#) (void) const
This routine returns true, if the integration kernel has a trailing zero.
- virtual [ex Laurent_series](#) (const [ex](#) &x, int [order](#)) const
Returns the Laurent series, starting possibly with the pole term.
- [size_t get_cache_size](#) (void) const
Returns the current size of the cache.
- void [set_cache_step](#) (int cache_steps) const
Sets the step size by which the cache is increased.
- [ex get_series_coeff](#) (int i) const
Wrapper around [series_coeff\(i\)](#), converts [cl_N](#) to numeric.
- [cln::cl_N series_coeff](#) (int i) const
Subclasses have either to implement [series_coeff_impl](#) or the two methods [Laurent_series](#) and [uses_Laurent_series](#).

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic ()`
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic (const basic &other)`
- `const basic & operator= (const basic &other)`
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate () const`
Create a clone of this object on the heap.
- virtual `ex eval () const`
Perform automatic non-interruptive term rewriting rules.
- virtual `ex evalf () const`
Evaluate object numerically.
- virtual `ex evalm () const`
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ () const`
Evaluate integrals, if result is known.
- virtual `ex eval_indexed (const basic &i) const`
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual `void print (const print_context &c, unsigned level=0) const`
Output to stream.
- virtual `void dbgprint () const`
Little wrapper around print to be called within a debugger.
- virtual `void dbgprinttree () const`
Little wrapper around printtree to be called within a debugger.
- virtual `unsigned precedence () const`
Return relative operator precedence (for parenthezing output).
- virtual `bool info (unsigned inf) const`
Information about the object.
- virtual `ex operator[] (const ex &index) const`
- virtual `ex operator[] (size_t i) const`
- virtual `ex & operator[] (const ex &index)`
- virtual `ex & operator[] (size_t i)`
- virtual `bool has (const ex &other, unsigned options=0) const`
Test for occurrence of a pattern.
- virtual `bool match (const ex &pattern, exmap &repls) const`
Check whether the expression matches a given pattern.
- virtual `ex subs (const exmap &m, unsigned options=0) const`
Substitute a set of objects by arbitrary expressions.
- virtual `ex map (map_function &f) const`
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual `void accept (GiNaC::visitor &v) const`
- virtual `bool is_polynomial (const ex &var) const`
Check whether this is a polynomial in the given variables.
- virtual `int degree (const ex &s) const`
Return degree of highest power in object s.
- virtual `int ldegree (const ex &s) const`
Return degree of lowest power in object s.
- virtual `ex coeff (const ex &s, int n=1) const`
Return coefficient of degree n in object s.
- virtual `ex expand (unsigned options=0) const`

- Expand expression, i.e.*

 - virtual `ex collect` (const `ex` &s, bool distributed=false) const

Sort expanded expression in terms of powers of some object(s).
- virtual `ex normal` (exmap &repl, exmap &rev_lookup, lst &modifier) const

Default implementation of `ex::normal()`.
- virtual `ex to_rational` (exmap &repl) const

Default implementation of `ex::to_rational()`.
- virtual `ex to_polynomial` (exmap &repl) const
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &xi) const

Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient` () const

Implementation `ex::max_coefficient()`.
- virtual `exvector get_free_indices` () const

Return a vector containing the free indices of an expression.
- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const

Add two indexed expressions.
- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const

Multiply an indexed expression with a scalar.
- virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const

Try to contract two indexed expressions that appear in the same product.
- virtual unsigned `return_type` () const
- virtual `return_type_t return_type_tinfo` () const
- virtual `ex conjugate` () const
- virtual `ex real_part` () const
- virtual `ex imag_part` () const
- template<class T >
 - void `print_dispatch` (const `print_context` &c, unsigned level) const

Like `print()`, but dispatch to the specified class.
- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const

Like `print()`, but dispatch to the specified class.
- virtual void `archive` (`archive_node` &n) const

Save (serialize) the object into archive node.
- virtual void `read_archive` (const `archive_node` &n, lst &syms)

Load (deserialize) the object from an archive node.
- `ex subs_one_level` (const `exmap` &m, unsigned options) const

Helper function for `subs()`.
- `ex diff` (const `symbol` &s, unsigned nth=1) const

Default interface of nth derivative `ex::diff(s, n)`.
- int `compare` (const `basic` &other) const

Compare objects syntactically to establish canonical ordering.
- bool `is_equal` (const `basic` &other) const

Test for syntactic equality.
- const `basic` & `hold` () const

Stop further evaluation.
- unsigned `gethash` () const
- const `basic` & `setflag` (unsigned f) const

Set some `status_flags`.
- const `basic` & `clearflag` (unsigned f) const

Clear some `status_flags`.

Public Member Functions inherited from [GiNaC::refcounted](#)

- [refcounted](#) () noexcept
- unsigned int [add_reference](#) () noexcept
- unsigned int [remove_reference](#) () noexcept
- unsigned int [get_refcount](#) () const noexcept
- void [set_refcount](#) (unsigned int r) noexcept

Protected Member Functions

- `cl::cl_N` [series_coeff_impl](#) (int i) const override
For $\omega = d\lambda$ only the coefficient of λ^0 is non-zero.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::integration_kernel](#)

- virtual bool [uses_Laurent_series](#) () const
Returns true, if the coefficients are computed from the Laurent series (in which case the method `Laurent_series` needs to be implemented).
- `ex` [get_numerical_value_impl](#) (const `ex` &lambda, const `ex` &pre, int shift, int N_trunc) const
The actual implementation for computing a numerical value for the integrand.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual `ex` [eval_ncmul](#) (const `exvector` &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual `ex` [derivative](#) (const [symbol](#) &s) const
Default implementation of `ex::diff()`.
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Protected Attributes

- `ex` n
- `ex` m
- `ex` x
- `ex` y

Protected Attributes inherited from [GiNaC::integration_kernel](#)

- int [cache_step_size](#)
- std::vector< cln::cl_N > [series_vec](#)

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.37.1 Detailed Description

The Ebar-kernel.

This class represents the differential one-form

$$\omega_{n;m}^{\bar{E}}(x; y) = \bar{E}_{n;m}(x; y; \bar{q}) \frac{d\bar{q}}{\bar{q}}$$

6.37.2 Constructor & Destructor Documentation

6.37.2.1 Ebar_kernel()

```
GiNaC::Ebar_kernel::Ebar_kernel (
    const ex & n,
    const ex & m,
    const ex & x,
    const ex & y )
```

6.37.3 Member Function Documentation

6.37.3.1 nops()

```
size_t GiNaC::Ebar_kernel::nops ( ) const [override], [virtual]
```

Number of operands/members.

Reimplemented from [GiNaC::basic](#).

6.37.3.2 op()

```
ex GiNaC::Ebar_kernel::op (
    size_t i ) const [override], [virtual]
```

Return operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [m](#), [n](#), [x](#), and [y](#).

6.37.3.3 let_op()

```
ex & GiNaC::Ebar_kernel::let_op (
    size_t i ) [override], [virtual]
```

Return modifiable operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::basic::ensure_if_modifiable\(\)](#), [m](#), [n](#), [x](#), and [y](#).

6.37.3.4 is_numeric()

```
bool GiNaC::Ebar_kernel::is_numeric (
    void ) const [override], [virtual]
```

This routine returns true, if the integration kernel can be evaluated numerically.

Reimplemented from [GiNaC::integration_kernel](#).

References [GiNaC::ex::evalf\(\)](#), [GiNaC::ex::info\(\)](#), [m](#), [n](#), [GiNaC::info_flags::nonnegint](#), [GiNaC::info_flags::numeric](#), [x](#), and [y](#).

6.37.3.5 get_numerical_value()

```
ex GiNaC::Ebar_kernel::get_numerical_value (
    const ex & qbar,
    int N_trunc = 0 ) const [override], [virtual]
```

Returns the value of $Ebar_{\{n,m\}}(x,y,qbar)$

Reimplemented from [GiNaC::integration_kernel](#).

References [GiNaC::integration_kernel::get_numerical_value_impl\(\)](#), and [qbar](#).

Referenced by [GiNaC::Kronecker_dtau_kernel::get_numerical_value\(\)](#), and [GiNaC::Kronecker_dz_kernel::series_coeff_impl\(\)](#).

6.37.3.6 series_coeff_impl()

```
cln::cl_N GiNaC::Ebar_kernel::series_coeff_impl (
    int i ) const [override], [protected], [virtual]
```

For $\omega = d\lambda$ only the coefficient of λ^0 is non-zero.

The i-th coefficient corresponds to the power λ^{i-1} .

Reimplemented from [GiNaC::integration_kernel](#).

References [GiNaC::ex::evalf\(\)](#), [k](#), [m](#), [n](#), [x](#), and [y](#).

6.37.3.7 do_print()

```
void GiNaC::Ebar_kernel::do_print (
    const print\_context & c,
    unsigned level ) const [protected]
```

References [c](#), [m](#), [n](#), [GiNaC::ex::print\(\)](#), [x](#), and [y](#).

6.37.4 Member Data Documentation

6.37.4.1 n

```
ex GiNaC::Ebar_kernel::n [protected]
```

Referenced by [do_print\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [series_coeff_impl\(\)](#).

6.37.4.2 m

```
ex GiNaC::Ebar_kernel::m [protected]
```

Referenced by [do_print\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [series_coeff_impl\(\)](#).

6.37.4.3 x

```
ex GiNaC::Ebar_kernel::x [protected]
```

Referenced by [do_print\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [series_coeff_impl\(\)](#).

6.37.4.4 y

```
ex GiNaC::Ebar_kernel::y [protected]
```

Referenced by [do_print\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [series_coeff_impl\(\)](#).

The documentation for this class was generated from the following files:

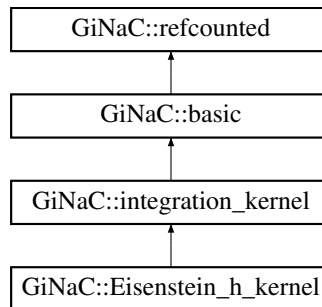
- [integration_kernel.h](#)
- [integration_kernel.cpp](#)

6.38 GiNaC::Eisenstein_h_kernel Class Reference

The kernel corresponding to the Eisenstein series $h_{k,N,r,s}(\tau)$.

```
#include <integration_kernel.h>
```

Inheritance diagram for GiNaC::Eisenstein_h_kernel:



Public Member Functions

- [Eisenstein_h_kernel](#) (const [ex](#) &k, const [ex](#) &N, const [ex](#) &r, const [ex](#) &s, const [ex](#) &C_norm=numeric(1))
- [ex series](#) (const [relational](#) &r, int [order](#), unsigned [options](#)=0) const override
The series method for this class returns the qbar-expansion of the modular form, without an additional factor of C_norm/qbar.
- [size_t nops](#) () const override
Number of operands/members.
- [ex op](#) (size_t i) const override
Return operand/member at position i.
- [ex & let_op](#) (size_t i) override
Return modifiable operand/member at position i.
- [bool is_numeric](#) (void) const override
This routine returns true, if the integration kernel can be evaluated numerically.
- [ex Laurent_series](#) (const [ex](#) &x, int [order](#)) const override
Returns the Laurent series, starting possibly with the pole term.
- [ex get_numerical_value](#) (const [ex](#) &qbar, int N_trunc=0) const override
Returns the value of the modular form.
- [ex coefficient_a0](#) (const [numeric](#) &k, const [numeric](#) &r, const [numeric](#) &s, const [numeric](#) &N) const
The constant coefficient in the Fourier expansion.
- [ex coefficient_an](#) (const [numeric](#) &n, const [numeric](#) &k, const [numeric](#) &r, const [numeric](#) &s, const [numeric](#) &N) const
The higher coefficients in the Fourier expansion.
- [ex q_expansion_modular_form](#) (const [ex](#) &q, int [order](#)) const

Public Member Functions inherited from GiNaC::integration_kernel

- **ex series** (const relational &r, int order, unsigned options=0) const override
Default implementation of ex::series().
- virtual bool **has_trailing_zero** (void) const
This routine returns true, if the integration kernel has a trailing zero.
- size_t **get_cache_size** (void) const
Returns the current size of the cache.
- void **set_cache_step** (int cache_steps) const
Sets the step size by which the cache is increased.
- **ex get_series_coeff** (int i) const
Wrapper around series_coeff(i), converts cl_N to numeric.
- cln::cl_N **series_coeff** (int i) const
Subclasses have either to implement series_coeff_impl or the two methods Laurent_series and uses_Laurent_series.

Public Member Functions inherited from GiNaC::basic

- virtual **~basic** ()
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- **basic** (const basic &other)
- const **basic & operator=** (const basic &other)
basic assignment operator: the other object might be of a derived class.
- virtual **basic * duplicate** () const
Create a clone of this object on the heap.
- virtual **ex eval** () const
Perform automatic non-interruptive term rewriting rules.
- virtual **ex evalf** () const
Evaluate object numerically.
- virtual **ex evalm** () const
Evaluate sums, products and integer powers of matrices.
- virtual **ex eval_integ** () const
Evaluate integrals, if result is known.
- virtual **ex eval_indexed** (const basic &i) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void **print** (const print_context &c, unsigned level=0) const
Output to stream.
- virtual void **dbgprint** () const
Little wrapper around print to be called within a debugger.
- virtual void **dbgprinttree** () const
Little wrapper around printtree to be called within a debugger.
- virtual unsigned **precedence** () const
Return relative operator precedence (for parenthezing output).
- virtual bool **info** (unsigned inf) const
Information about the object.
- virtual **ex operator[]** (const ex &index) const
- virtual **ex operator[]** (size_t i) const
- virtual **ex & operator[]** (const ex &index)
- virtual **ex & operator[]** (size_t i)
- virtual bool **has** (const ex &other, unsigned options=0) const
Test for occurrence of a pattern.
- virtual bool **match** (const ex &pattern, exmap &repls) const

- Check whether the expression matches a given pattern.*

 - virtual `ex subs` (const `exmap` &m, unsigned `options`=0) const

Substitute a set of objects by arbitrary expressions.
- virtual `ex map` (`map_function` &f) const

Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void `accept` (`GiNaC::visitor` &v) const
- virtual bool `is_polynomial` (const `ex` &var) const

Check whether this is a polynomial in the given variables.
- virtual int `degree` (const `ex` &s) const

Return degree of highest power in object s.
- virtual int `ldegree` (const `ex` &s) const

Return degree of lowest power in object s.
- virtual `ex coeff` (const `ex` &s, int `n`=1) const

Return coefficient of degree n in object s.
- virtual `ex expand` (unsigned `options`=0) const

Expand expression, i.e.
- virtual `ex collect` (const `ex` &s, bool `distributed`=false) const

Sort expanded expression in terms of powers of some object(s).
- virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const

Default implementation of `ex::normal()`.
- virtual `ex to_rational` (`exmap` &repl) const

Default implementation of `ex::to_rational()`.
- virtual `ex to_polynomial` (`exmap` &repl) const
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &xi) const

Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient` () const

Implementation `ex::max_coefficient()`.
- virtual `exvector get_free_indices` () const

Return a vector containing the free indices of an expression.
- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const

Add two indexed expressions.
- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const

Multiply an indexed expression with a scalar.
- virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const

Try to contract two indexed expressions that appear in the same product.
- virtual unsigned `return_type` () const
- virtual `return_type_t return_type_tinfo` () const
- virtual `ex conjugate` () const
- virtual `ex real_part` () const
- virtual `ex imag_part` () const
- template<class T >
 - void `print_dispatch` (const `print_context` &c, unsigned level) const

Like `print()`, but dispatch to the specified class.
- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const

Like `print()`, but dispatch to the specified class.
- virtual void `archive` (`archive_node` &n) const

Save (serialize) the object into archive node.
- virtual void `read_archive` (const `archive_node` &n, `lst` &syms)

Load (deserialize) the object from an archive node.
- `ex subs_one_level` (const `exmap` &m, unsigned `options`) const

- *Helper function for [subs\(\)](#).*
- [ex diff](#) (const [symbol](#) &s, unsigned nth=1) const
Default interface of nth derivative [ex::diff\(s, n\)](#).
- int [compare](#) (const [basic](#) &other) const
Compare objects syntactically to establish canonical ordering.
- bool [is_equal](#) (const [basic](#) &other) const
Test for syntactic equality.
- const [basic](#) & [hold](#) () const
Stop further evaluation.
- unsigned [gethash](#) () const
- const [basic](#) & [setflag](#) (unsigned f) const
Set some [status_flags](#).
- const [basic](#) & [clearflag](#) (unsigned f) const
Clear some [status_flags](#).

Public Member Functions inherited from [GiNaC::refcounted](#)

- [refcounted](#) () noexcept
- unsigned int [add_reference](#) () noexcept
- unsigned int [remove_reference](#) () noexcept
- unsigned int [get_refcount](#) () const noexcept
- void [set_refcount](#) (unsigned int r) noexcept

Protected Member Functions

- bool [uses_Laurent_series](#) () const override
Returns true, if the coefficients are computed from the Laurent series (in which case the method [Laurent_series](#) needs to be implemented).
- void [do_print](#) (const [print_context](#) &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::integration_kernel](#)

- virtual [cln::cl_N series_coeff_impl](#) (int i) const
For $\omega = d\lambda$ only the coefficient of λ^0 is non-zero.
- [ex get_numerical_value_impl](#) (const [ex](#) &lambda, const [ex](#) &pre, int shift, int N_trunc) const
The actual implementation for computing a numerical value for the integrand.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual [ex derivative](#) (const [symbol](#) &s) const
Default implementation of [ex::diff\(\)](#).
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const

- Returns true if two objects of same type are equal.*
- virtual unsigned `calchash` () const
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- void `ensure_if_modifiable` () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void `do_print` (const `print_context` &c, unsigned level) const
Default output to stream.
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
Tree output to stream.
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
Python parsable output to stream.

Protected Attributes

- `ex k`
- `ex N`
- `ex r`
- `ex s`
- `ex C_norm`

Protected Attributes inherited from `GiNaC::integration_kernel`

- int `cache_step_size`
- `std::vector< cln::cl_N >` `series_vec`

Protected Attributes inherited from `GiNaC::basic`

- unsigned `flags`
of type `status_flags`
- unsigned `hashvalue`
hash value

6.38.1 Detailed Description

The kernel corresponding to the Eisenstein series $h_{k,N,r,s}(\tau)$.

This class represents the differential one-form

$$\omega_{k,N,r,s}^{\text{Eisenstein,h}} = C_k h_{k,N,r,s}(\tau) \frac{d\bar{q}_N}{\bar{q}_N}$$

6.38.2 Constructor & Destructor Documentation

6.38.2.1 `Eisenstein_h_kernel()`

```
GiNaC::Eisenstein_h_kernel::Eisenstein_h_kernel (
    const ex & k,
    const ex & N,
    const ex & r,
    const ex & s,
    const ex & C_norm = numeric(1) )
```

6.38.3 Member Function Documentation

6.38.3.1 series()

```
ex GiNaC::Eisenstein_h_kernel::series (
    const relational & r,
    int order,
    unsigned options = 0 ) const [override], [virtual]
```

The series method for this class returns the qbar-expansion of the modular form, without an additional factor of $C\_norm/qbar$.

This allows for easy use in the class [modular_form_kernel](#).

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::lhs\(\)](#), [order](#), [q_expansion_modular_form\(\)](#), [qbar](#), [r](#), [GiNaC::ex::rhs\(\)](#), and [GiNaC::ex::series\(\)](#).

6.38.3.2 nops()

```
size_t GiNaC::Eisenstein_h_kernel::nops ( ) const [override], [virtual]
```

Number of operands/members.

Reimplemented from [GiNaC::basic](#).

6.38.3.3 op()

```
ex & GiNaC::Eisenstein_h_kernel::op (
    size_t i ) const [override], [virtual]
```

Return operand/member at position i .

Reimplemented from [GiNaC::basic](#).

References [C_norm](#), [k](#), [N](#), [r](#), and [s](#).

6.38.3.4 let_op()

```
ex & GiNaC::Eisenstein_h_kernel::let_op (
    size_t i ) [override], [virtual]
```

Return modifiable operand/member at position i .

Reimplemented from [GiNaC::basic](#).

References [C_norm](#), [GiNaC::basic::ensure_if_modifiable\(\)](#), [k](#), [N](#), [r](#), and [s](#).

6.38.3.5 is_numeric()

```
bool GiNaC::Eisenstein_h_kernel::is_numeric (
    void ) const [override], [virtual]
```

This routine returns true, if the integration kernel can be evaluated numerically.

Reimplemented from [GiNaC::integration_kernel](#).

References [C_norm](#), [GiNaC::ex::evalf\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [k](#), [N](#), [GiNaC::info_flags::nonnegint](#), [GiNaC::info_flags::numeric](#), [GiNaC::info_flags::posint](#), [r](#), and [s](#).

6.38.3.6 Laurent_series()

```
ex GiNaC::Eisenstein_h_kernel::Laurent_series (
    const ex & x,
    int order ) const [override], [virtual]
```

Returns the Laurent series, starting possibly with the pole term.

Neglected terms are of order $O(x^{order})$.

Reimplemented from [GiNaC::integration_kernel](#).

References [C_norm](#), [order](#), [q_expansion_modular_form\(\)](#), [GiNaC::ex::series\(\)](#), and [x](#).

6.38.3.7 get_numerical_value()

```
ex GiNaC::Eisenstein_h_kernel::get_numerical_value (
    const ex & qbar,
    int N_trunc = 0 ) const [override], [virtual]
```

Returns the value of the modular form.

Reimplemented from [GiNaC::integration_kernel](#).

References [C_norm](#), [GiNaC::integration_kernel::get_numerical_value_impl\(\)](#), and [qbar](#).

6.38.3.8 uses_Laurent_series()

```
bool GiNaC::Eisenstein_h_kernel::uses_Laurent_series ( ) const [override], [protected], [virtual]
```

Returns true, if the coefficients are computed from the Laurent series (in which case the method `Laurent_series` needs to be implemented).

Returns false if this is not the case (and the class has an implementation of `series_coeff_impl`).

Reimplemented from [GiNaC::integration_kernel](#).

6.38.3.9 coefficient_a0()

```
ex GiNaC::Eisenstein_h_kernel::coefficient_a0 (
    const numeric & k,
    const numeric & r,
    const numeric & s,
    const numeric & N ) const
```

The constant coefficient in the Fourier expansion.

References [GiNaC::Bernoulli_polynomial\(\)](#), [GiNaC::cos\(\)](#), [GiNaC::I](#), [GiNaC::irem\(\)](#), [k](#), [GiNaC::mod\(\)](#), [N](#), [GiNaC::Pi](#), [r](#), [s](#), and [GiNaC::sin\(\)](#).

Referenced by [q_expansion_modular_form\(\)](#).

6.38.3.10 coefficient_an()

```
ex GiNaC::Eisenstein_h_kernel::coefficient_an (
    const numeric & n,
    const numeric & k,
    const numeric & r,
    const numeric & s,
    const numeric & N ) const
```

The higher coefficients in the Fourier expansion.

References [GiNaC::exp\(\)](#), [GiNaC::I](#), [GiNaC::irem\(\)](#), [k](#), [m](#), [GiNaC::mod\(\)](#), [n](#), [N](#), [GiNaC::Pi](#), [GiNaC::pow\(\)](#), [r](#), and [s](#).

Referenced by [q_expansion_modular_form\(\)](#).

6.38.3.11 q_expansion_modular_form()

```
ex GiNaC::Eisenstein_h_kernel::q_expansion_modular_form (
    const ex & q,
    int order ) const
```

References [coefficient_a0\(\)](#), [coefficient_an\(\)](#), [k](#), [N](#), [GiNaC::pow\(\)](#), [r](#), [s](#), and [GiNaC::ex::series\(\)](#).

Referenced by [Laurent_series\(\)](#), and [series\(\)](#).

6.38.3.12 do_print()

```
void GiNaC::Eisenstein_h_kernel::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

References [c](#), [C_norm](#), [k](#), [N](#), [GiNaC::ex::print\(\)](#), [r](#), and [s](#).

6.38.4 Member Data Documentation

6.38.4.1 k

`ex GiNaC::Eisenstein_h_kernel::k [protected]`

Referenced by [coefficient_a0\(\)](#), [coefficient_an\(\)](#), [do_print\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [q_expansion_modular_form\(\)](#).

6.38.4.2 N

`ex GiNaC::Eisenstein_h_kernel::N [protected]`

Referenced by [coefficient_a0\(\)](#), [coefficient_an\(\)](#), [do_print\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [q_expansion_modular_form\(\)](#).

6.38.4.3 r

`ex GiNaC::Eisenstein_h_kernel::r [protected]`

Referenced by [coefficient_a0\(\)](#), [coefficient_an\(\)](#), [do_print\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), [q_expansion_modular_form\(\)](#), and [series\(\)](#).

6.38.4.4 s

`ex GiNaC::Eisenstein_h_kernel::s [protected]`

Referenced by [coefficient_a0\(\)](#), [coefficient_an\(\)](#), [do_print\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [q_expansion_modular_form\(\)](#).

6.38.4.5 C_norm

`ex GiNaC::Eisenstein_h_kernel::C_norm [protected]`

Referenced by [do_print\(\)](#), [get_numerical_value\(\)](#), [is_numeric\(\)](#), [Laurent_series\(\)](#), [let_op\(\)](#), and [op\(\)](#).

The documentation for this class was generated from the following files:

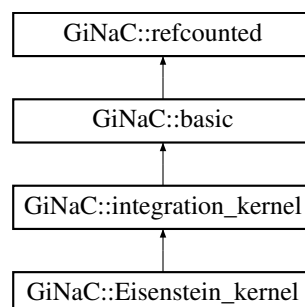
- [integration_kernel.h](#)
- [integration_kernel.cpp](#)

6.39 GiNaC::Eisenstein_kernel Class Reference

The kernel corresponding to the Eisenstein series $E_{k,N,a,b,K}(\tau)$.

```
#include <integration_kernel.h>
```

Inheritance diagram for GiNaC::Eisenstein_kernel:



Public Member Functions

- **Eisenstein_kernel** (const **ex** &k, const **ex** &N, const **ex** &a, const **ex** &b, const **ex** &K, const **ex** &C_norm=numeric(1))
- **ex series** (const **relational** &r, int **order**, unsigned **options**=0) const override
The series method for this class returns the qbar-expansion of the modular form, without an additional factor of C_norm/qbar.
- **size_t nops** () const override
Number of operands/members.
- **ex op** (size_t i) const override
Return operand/member at position i.
- **ex & let_op** (size_t i) override
Return modifiable operand/member at position i.
- **bool is_numeric** (void) const override
This routine returns true, if the integration kernel can be evaluated numerically.
- **ex Laurent_series** (const **ex** &x, int **order**) const override
Returns the Laurent series, starting possibly with the pole term.
- **ex get_numerical_value** (const **ex** &qbar, int N_trunc=0) const override
Returns the value of the modular form.
- **ex q_expansion_modular_form** (const **ex** &q, int **order**) const

Public Member Functions inherited from **GiNaC::integration_kernel**

- **ex series** (const **relational** &r, int **order**, unsigned **options**=0) const override
*Default implementation of **ex::series()**.*
- **virtual bool has_trailing_zero** (void) const
This routine returns true, if the integration kernel has a trailing zero.
- **size_t get_cache_size** (void) const
Returns the current size of the cache.
- **void set_cache_step** (int cache_steps) const
Sets the step size by which the cache is increased.
- **ex get_series_coeff** (int i) const
*Wrapper around **series_coeff(i)**, converts **cl_N** to numeric.*
- **cln::cl_N series_coeff** (int i) const
*Subclasses have either to implement **series_coeff_impl** or the two methods **Laurent_series** and **uses_Laurent_series**.*

Public Member Functions inherited from **GiNaC::basic**

- **virtual ~basic** ()
*basic destructor, virtual because class **ex** will delete objects of derived classes via a **basic***.*
- **basic** (const **basic** &other)
- **const basic & operator=** (const **basic** &other)
basic assignment operator: the other object might be of a derived class.
- **virtual basic * duplicate** () const
Create a clone of this object on the heap.
- **virtual ex eval** () const
Perform automatic non-interruptive term rewriting rules.
- **virtual ex evalf** () const
Evaluate object numerically.
- **virtual ex evalm** () const

- Evaluate sums, products and integer powers of matrices.*

 - virtual `ex eval_integ ()` const
- Evaluate integrals, if result is known.*

 - virtual `ex eval_indexed (const basic &i)` const
- Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.*

 - virtual void `print (const print_context &c, unsigned level=0)` const
- Output to stream.*

 - virtual void `dbgprint ()` const
- Little wrapper around print to be called within a debugger.*

 - virtual void `dbgprinttree ()` const
- Little wrapper around printtree to be called within a debugger.*

 - virtual unsigned `precedence ()` const
- Return relative operator precedence (for parenthesizing output).*

 - virtual bool `info (unsigned inf)` const
- Information about the object.*

 - virtual `ex operator[] (const ex &index)` const
 - virtual `ex operator[] (size_t i)` const
 - virtual `ex & operator[] (const ex &index)`
 - virtual `ex & operator[] (size_t i)`
 - virtual bool `has (const ex &other, unsigned options=0)` const
- Test for occurrence of a pattern.*

 - virtual bool `match (const ex &pattern, exmap &repls)` const
- Check whether the expression matches a given pattern.*

 - virtual `ex subs (const exmap &m, unsigned options=0)` const
- Substitute a set of objects by arbitrary expressions.*

 - virtual `ex map (map_function &f)` const
- Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).*

 - virtual void `accept (GiNaC::visitor &v)` const
 - virtual bool `is_polynomial (const ex &var)` const
- Check whether this is a polynomial in the given variables.*

 - virtual int `degree (const ex &s)` const
- Return degree of highest power in object s.*

 - virtual int `ldegree (const ex &s)` const
- Return degree of lowest power in object s.*

 - virtual `ex coeff (const ex &s, int n=1)` const
- Return coefficient of degree n in object s.*

 - virtual `ex expand (unsigned options=0)` const
- Expand expression, i.e.*

 - virtual `ex collect (const ex &s, bool distributed=false)` const
- Sort expanded expression in terms of powers of some object(s).*

 - virtual `ex normal (exmap &repl, exmap &rev_lookup, lst &modifier)` const
- Default implementation of `ex::normal()`.*

 - virtual `ex to_rational (exmap &repl)` const
- Default implementation of `ex::to_rational()`.*

 - virtual `ex to_polynomial (exmap &repl)` const
 - virtual `numeric integer_content ()` const
 - virtual `ex smod (const numeric &xi)` const
- Apply symmetric modular homomorphism to an expanded multivariate polynomial.*

 - virtual `numeric max_coefficient ()` const
- Implementation `ex::max_coefficient()`.*

 - virtual `exvector get_free_indices ()` const

- *Return a vector containing the free indices of an expression.*
- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
- *Add two indexed expressions.*
- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
- *Multiply an indexed expression with a scalar.*
- virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const
- *Try to contract two indexed expressions that appear in the same product.*
- virtual unsigned `return_type` () const
- virtual `return_type_t return_type_tinfo` () const
- virtual `ex conjugate` () const
- virtual `ex real_part` () const
- virtual `ex imag_part` () const
- template<class T >
- void `print_dispatch` (const `print_context` &c, unsigned level) const
- *Like `print()`, but dispatch to the specified class.*
- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const
- *Like `print()`, but dispatch to the specified class.*
- virtual void `archive` (`archive_node` &n) const
- *Save (serialize) the object into archive node.*
- virtual void `read_archive` (const `archive_node` &n, `lst` &syms)
- *Load (deserialize) the object from an archive node.*
- `ex subs_one_level` (const `exmap` &m, unsigned `options`) const
- *Helper function for `subs()`.*
- `ex diff` (const `symbol` &s, unsigned nth=1) const
- *Default interface of nth derivative `ex::diff(s, n)`.*
- int `compare` (const `basic` &other) const
- *Compare objects syntactically to establish canonical ordering.*
- bool `is_equal` (const `basic` &other) const
- *Test for syntactic equality.*
- const `basic` & `hold` () const
- *Stop further evaluation.*
- unsigned `gethash` () const
- const `basic` & `setflag` (unsigned f) const
- *Set some `status_flags`.*
- const `basic` & `clearflag` (unsigned f) const
- *Clear some `status_flags`.*

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Protected Member Functions

- bool `uses_Laurent_series` () const override
- *Returns true, if the coefficients are computed from the Laurent series (in which case the method `Laurent_series` needs to be implemented).*
- void `do_print` (const `print_context` &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::integration_kernel](#)

- virtual `cln::cl_N series_coeff_impl` (int i) const
For $\omega = d\lambda$ only the coefficient of λ^0 is non-zero.
- `ex get_numerical_value_impl` (const `ex` &lambda, const `ex` &pre, int shift, int N_trunc) const
The actual implementation for computing a numerical value for the integrand.
- void `do_print` (const `print_context` &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::basic](#)

- `basic` ()
- virtual `ex eval_ncmul` (const `exvector` &v) const
- virtual bool `match_same_type` (const `basic` &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual `ex derivative` (const `symbol` &s) const
Default implementation of `ex::diff()`.
- virtual int `compare_same_type` (const `basic` &other) const
Returns order relation between two objects of same type.
- virtual bool `is_equal_same_type` (const `basic` &other) const
Returns true if two objects of same type are equal.
- virtual unsigned `calchash` () const
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- void `ensure_if_modifiable` () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void `do_print` (const `print_context` &c, unsigned level) const
Default output to stream.
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
Tree output to stream.
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
Python parsable output to stream.

Protected Attributes

- `ex k`
- `ex N`
- `ex a`
- `ex b`
- `ex K`
- `ex C_norm`

Protected Attributes inherited from [GiNaC::integration_kernel](#)

- int `cache_step_size`
- `std::vector< cln::cl_N >` `series_vec`

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned `flags`
of type `status_flags`
- unsigned `hashvalue`
hash value

6.39.1 Detailed Description

The kernel corresponding to the Eisenstein series $E_{k,N,a,b,K}(\tau)$.

This class represents the differential one-form

$$\omega_{k,N,a,b,K}^{\text{Eisenstein}} = C_k E_{k,N,a,b,K}(\tau) \frac{d\bar{q}_N}{\bar{q}_N}$$

The integers a and b are either one or the discriminant of a quadratic number field. This class represents Eisenstein series, which can be defined by primitive Dirichlet characters from the Kronecker symbol. This implies that the characters take the values -1,0,1, i.e. no higher roots of unity occur. The

$$\bar{q}$$

-expansion has then rational coefficients.

Ref.: W. Stein, Modular Forms: A Computational Approach, Chapter 5

6.39.2 Constructor & Destructor Documentation

6.39.2.1 Eisenstein_kernel()

```
GiNaC::Eisenstein_kernel::Eisenstein_kernel (
    const ex & k,
    const ex & N,
    const ex & a,
    const ex & b,
    const ex & K,
    const ex & C_norm = numeric(1) )
```

6.39.3 Member Function Documentation

6.39.3.1 series()

```
ex GiNaC::Eisenstein_kernel::series (
    const relational & r,
    int order,
    unsigned options = 0 ) const [override], [virtual]
```

The series method for this class returns the qbar-expansion of the modular form, without an additional factor of C_norm/qbar.

This allows for easy use in the class [modular_form_kernel](#).

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::lhs\(\)](#), [order](#), [q_expansion_modular_form\(\)](#), [qbar](#), [r](#), and [GiNaC::ex::series\(\)](#).

6.39.3.2 nops()

```
size_t GiNaC::Eisenstein_kernel::nops ( ) const [override], [virtual]
```

Number of operands/members.

Reimplemented from [GiNaC::basic](#).

6.39.3.3 op()

```
ex GiNaC::Eisenstein_kernel::op (
    size_t i ) const [override], [virtual]
```

Return operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [a](#), [b](#), [C_norm](#), [k](#), [K](#), and [N](#).

6.39.3.4 let_op()

```
ex & GiNaC::Eisenstein_kernel::let_op (
    size_t i ) [override], [virtual]
```

Return modifiable operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [a](#), [b](#), [C_norm](#), [GiNaC::basic::ensure_if_modifiable\(\)](#), [k](#), [K](#), and [N](#).

6.39.3.5 is_numeric()

```
bool GiNaC::Eisenstein_kernel::is_numeric (
    void ) const [override], [virtual]
```

This routine returns true, if the integration kernel can be evaluated numerically.

Reimplemented from [GiNaC::integration_kernel](#).

References [a](#), [b](#), [C_norm](#), [GiNaC::ex::evalf\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [k](#), [K](#), [N](#), [GiNaC::info_flags::nonnegint](#), [GiNaC::info_flags::numeric](#), and [GiNaC::info_flags::posint](#).

6.39.3.6 Laurent_series()

```
ex GiNaC::Eisenstein_kernel::Laurent_series (
    const ex & x,
    int order ) const [override], [virtual]
```

Returns the Laurent series, starting possibly with the pole term.

Neglected terms are of order $O(x^{order})$.

Reimplemented from [GiNaC::integration_kernel](#).

References [C_norm](#), [order](#), [q_expansion_modular_form\(\)](#), [GiNaC::ex::series\(\)](#), and [x](#).

6.39.3.7 get_numerical_value()

```
ex GiNaC::Eisenstein_kernel::get_numerical_value (
    const ex & qbar,
    int N_trunc = 0 ) const [override], [virtual]
```

Returns the value of the modular form.

Reimplemented from [GiNaC::integration_kernel](#).

References [C_norm](#), [GiNaC::integration_kernel::get_numerical_value_impl\(\)](#), and [qbar](#).

6.39.3.8 uses_Laurent_series()

```
bool GiNaC::Eisenstein_kernel::uses_Laurent_series ( ) const [override], [protected], [virtual]
```

Returns true, if the coefficients are computed from the Laurent series (in which case the method `Laurent_series` needs to be implemented).

Returns false if this is not the case (and the class has an implementation of `series_coeff_impl`).

Reimplemented from [GiNaC::integration_kernel](#).

6.39.3.9 q_expansion_modular_form()

```
ex GiNaC::Eisenstein_kernel::q_expansion_modular_form (
    const ex & q,
    int order ) const
```

References [a](#), [b](#), [k](#), [K](#), [N](#), and [order](#).

Referenced by [Laurent_series\(\)](#), and [series\(\)](#).

6.39.3.10 do_print()

```
void GiNaC::Eisenstein_kernel::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

References [a](#), [b](#), [c](#), [C_norm](#), [k](#), [K](#), [N](#), and [GiNaC::ex::print\(\)](#).

6.39.4 Member Data Documentation

6.39.4.1 k

```
ex GiNaC::Eisenstein_kernel::k [protected]
```

Referenced by [do_print\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [q_expansion_modular_form\(\)](#).

6.39.4.2 N

```
ex GiNaC::Eisenstein_kernel::N [protected]
```

Referenced by [do_print\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [q_expansion_modular_form\(\)](#).

6.39.4.3 a

```
ex GiNaC::Eisenstein_kernel::a [protected]
```

Referenced by [do_print\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [q_expansion_modular_form\(\)](#).

6.39.4.4 b

```
ex GiNaC::Eisenstein_kernel::b [protected]
```

Referenced by [do_print\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [q_expansion_modular_form\(\)](#).

6.39.4.5 K

```
ex GiNaC::Eisenstein_kernel::K [protected]
```

Referenced by [do_print\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [q_expansion_modular_form\(\)](#).

6.39.4.6 C_norm

```
ex GiNaC::Eisenstein_kernel::C_norm [protected]
```

Referenced by [do_print\(\)](#), [get_numerical_value\(\)](#), [is_numeric\(\)](#), [Laurent_series\(\)](#), [let_op\(\)](#), and [op\(\)](#).

The documentation for this class was generated from the following files:

- [integration_kernel.h](#)
- [integration_kernel.cpp](#)

6.40 GiNaC::composition_generator::coolmulti::element Struct Reference

```
#include <utils.h>
```

Public Member Functions

- [element](#) (unsigned val, [element](#) *n)
- [~element](#) ()

Public Attributes

- unsigned [value](#)
- [element](#) * [next](#)

6.40.1 Constructor & Destructor Documentation

6.40.1.1 [element\(\)](#)

```
GiNaC::composition_generator::coolmulti::element::element (
    unsigned val,
    element * n ) [inline]
```

6.40.1.2 [~element\(\)](#)

```
GiNaC::composition_generator::coolmulti::element::~~element ( ) [inline]
```

References [next](#).

6.40.2 Member Data Documentation

6.40.2.1 [value](#)

```
unsigned GiNaC::composition_generator::coolmulti::element::value
```

Referenced by [GiNaC::composition_generator::coolmulti::finished\(\)](#), [GiNaC::composition_generator::get\(\)](#), and [GiNaC::composition_generator::coolmulti::next_permutation\(\)](#).

6.40.2.2 [next](#)

```
element* GiNaC::composition_generator::coolmulti::element::next
```

Referenced by [GiNaC::composition_generator::coolmulti::coolmulti\(\)](#), [GiNaC::composition_generator::coolmulti::finished\(\)](#), [GiNaC::composition_generator::get\(\)](#), [GiNaC::composition_generator::coolmulti::next_permutation\(\)](#), and [~element\(\)](#).

The documentation for this struct was generated from the following file:

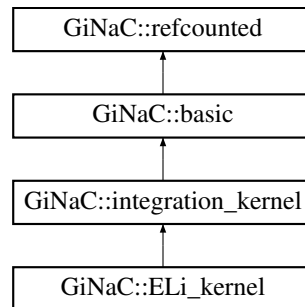
- [utils.h](#)

6.41 GiNaC::ELi_kernel Class Reference

The ELi-kernel.

```
#include <integration_kernel.h>
```

Inheritance diagram for GiNaC::ELi_kernel:



Public Member Functions

- [ELi_kernel](#) (const [ex](#) &n, const [ex](#) &m, const [ex](#) &x, const [ex](#) &y)
- [size_t nops](#) () const override
Number of operands/members.
- [ex op](#) (size_t i) const override
Return operand/member at position i.
- [ex & let_op](#) (size_t i) override
Return modifiable operand/member at position i.
- [bool is_numeric](#) (void) const override
This routine returns true, if the integration kernel can be evaluated numerically.
- [ex get_numerical_value](#) (const [ex](#) &qbar, int N_trunc=0) const override
Returns the value of $ELi_{\{n,m\}}(x,y,qbar)$

Public Member Functions inherited from [GiNaC::integration_kernel](#)

- [ex series](#) (const [relational](#) &r, int [order](#), unsigned [options](#)=0) const override
Default implementation of [ex::series\(\)](#).
- virtual [bool has_trailing_zero](#) (void) const
This routine returns true, if the integration kernel has a trailing zero.
- virtual [ex Laurent_series](#) (const [ex](#) &x, int [order](#)) const
Returns the Laurent series, starting possibly with the pole term.
- [size_t get_cache_size](#) (void) const
Returns the current size of the cache.
- void [set_cache_step](#) (int cache_steps) const
Sets the step size by which the cache is increased.
- [ex get_series_coeff](#) (int i) const
Wrapper around [series_coeff\(i\)](#), converts cl_N to numeric.
- [cln::cl_N series_coeff](#) (int i) const
Subclasses have either to implement [series_coeff_impl](#) or the two methods [Laurent_series](#) and [uses_Laurent_series](#).

Public Member Functions inherited from **GiNaC::basic**

- virtual `~basic ()`
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic (const basic &other)`
- `const basic & operator= (const basic &other)`
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate () const`
Create a clone of this object on the heap.
- virtual `ex eval () const`
Perform automatic non-interruptive term rewriting rules.
- virtual `ex evalf () const`
Evaluate object numerically.
- virtual `ex evalm () const`
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ () const`
Evaluate integrals, if result is known.
- virtual `ex eval_indexed (const basic &i) const`
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual `void print (const print_context &c, unsigned level=0) const`
Output to stream.
- virtual `void dbgprint () const`
Little wrapper around print to be called within a debugger.
- virtual `void dbgprinttree () const`
Little wrapper around printtree to be called within a debugger.
- virtual `unsigned precedence () const`
Return relative operator precedence (for parenthesizing output).
- virtual `bool info (unsigned inf) const`
Information about the object.
- virtual `ex operator[] (const ex &index) const`
- virtual `ex operator[] (size_t i) const`
- virtual `ex & operator[] (const ex &index)`
- virtual `ex & operator[] (size_t i)`
- virtual `bool has (const ex &other, unsigned options=0) const`
Test for occurrence of a pattern.
- virtual `bool match (const ex &pattern, exmap &repls) const`
Check whether the expression matches a given pattern.
- virtual `ex subs (const exmap &m, unsigned options=0) const`
Substitute a set of objects by arbitrary expressions.
- virtual `ex map (map_function &f) const`
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual `void accept (GiNaC::visitor &v) const`
- virtual `bool is_polynomial (const ex &var) const`
Check whether this is a polynomial in the given variables.
- virtual `int degree (const ex &s) const`
Return degree of highest power in object s.
- virtual `int ldegree (const ex &s) const`
Return degree of lowest power in object s.
- virtual `ex coeff (const ex &s, int n=1) const`
Return coefficient of degree n in object s.
- virtual `ex expand (unsigned options=0) const`

- Expand expression, i.e.*

 - virtual `ex collect` (const `ex` &s, bool distributed=false) const

Sort expanded expression in terms of powers of some object(s).
- virtual `ex normal` (exmap &repl, exmap &rev_lookup, lst &modifier) const

Default implementation of `ex::normal()`.
- virtual `ex to_rational` (exmap &repl) const

Default implementation of `ex::to_rational()`.
- virtual `ex to_polynomial` (exmap &repl) const
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &xi) const

Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient` () const

Implementation `ex::max_coefficient()`.
- virtual `exvector get_free_indices` () const

Return a vector containing the free indices of an expression.
- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const

Add two indexed expressions.
- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const

Multiply an indexed expression with a scalar.
- virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const

Try to contract two indexed expressions that appear in the same product.
- virtual unsigned `return_type` () const
- virtual `return_type_t return_type_tinfo` () const
- virtual `ex conjugate` () const
- virtual `ex real_part` () const
- virtual `ex imag_part` () const
- template<class T >
 - void `print_dispatch` (const `print_context` &c, unsigned level) const

Like `print()`, but dispatch to the specified class.
- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const

Like `print()`, but dispatch to the specified class.
- virtual void `archive` (`archive_node` &n) const

Save (serialize) the object into archive node.
- virtual void `read_archive` (const `archive_node` &n, lst &syms)

Load (deserialize) the object from an archive node.
- `ex subs_one_level` (const exmap &m, unsigned options) const

Helper function for `subs()`.
- `ex diff` (const `symbol` &s, unsigned nth=1) const

Default interface of nth derivative `ex::diff(s, n)`.
- int `compare` (const `basic` &other) const

Compare objects syntactically to establish canonical ordering.
- bool `is_equal` (const `basic` &other) const

Test for syntactic equality.
- const `basic` & `hold` () const

Stop further evaluation.
- unsigned `gethash` () const
- const `basic` & `setflag` (unsigned f) const

Set some `status_flags`.
- const `basic` & `clearflag` (unsigned f) const

Clear some `status_flags`.

Public Member Functions inherited from GiNaC::refcounted

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Protected Member Functions

- `cl::cl_N_series_coeff_impl` (int i) const override
For $\omega = d\lambda$ only the coefficient of λ^0 is non-zero.
- void `do_print` (const `print_context` &c, unsigned level) const

Protected Member Functions inherited from GiNaC::integration_kernel

- virtual bool `uses_Laurent_series` () const
Returns true, if the coefficients are computed from the Laurent series (in which case the method `Laurent_series` needs to be implemented).
- `ex` `get_numerical_value_impl` (const `ex` &lambda, const `ex` &pre, int shift, int N_trunc) const
The actual implementation for computing a numerical value for the integrand.
- void `do_print` (const `print_context` &c, unsigned level) const

Protected Member Functions inherited from GiNaC::basic

- `basic` ()
- virtual `ex` `eval_ncmul` (const `exvector` &v) const
- virtual bool `match_same_type` (const `basic` &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual `ex` `derivative` (const `symbol` &s) const
Default implementation of `ex::diff()`.
- virtual int `compare_same_type` (const `basic` &other) const
Returns order relation between two objects of same type.
- virtual bool `is_equal_same_type` (const `basic` &other) const
Returns true if two objects of same type are equal.
- virtual unsigned `calchash` () const
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- void `ensure_if_modifiable` () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void `do_print` (const `print_context` &c, unsigned level) const
Default output to stream.
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
Tree output to stream.
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
Python parsable output to stream.

Protected Attributes

- `ex` n
- `ex` m
- `ex` x
- `ex` y

Protected Attributes inherited from [GiNaC::integration_kernel](#)

- int [cache_step_size](#)
- std::vector< cln::cl_N > [series_vec](#)

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.41.1 Detailed Description

The ELi-kernel.

This class represents the differential one-form

$$\omega_{n;m}^{\text{ELi}}(x; y) = \text{ELi}_{n;m}(x; y; \bar{q}) \frac{d\bar{q}}{\bar{q}}$$

6.41.2 Constructor & Destructor Documentation

6.41.2.1 [ELi_kernel\(\)](#)

```
GiNaC::ELi_kernel::ELi_kernel (
    const ex & n,
    const ex & m,
    const ex & x,
    const ex & y )
```

6.41.3 Member Function Documentation

6.41.3.1 [nops\(\)](#)

```
size_t GiNaC::ELi_kernel::nops ( ) const [override], [virtual]
```

Number of operands/members.

Reimplemented from [GiNaC::basic](#).

6.41.3.2 [op\(\)](#)

```
ex GiNaC::ELi_kernel::op (
    size_t i ) const [override], [virtual]
```

Return operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [m](#), [n](#), [x](#), and [y](#).

6.41.3.3 let_op()

```
ex & GiNaC::ELi_kernel::let_op (
    size_t i ) [override], [virtual]
```

Return modifiable operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::basic::ensure_if_modifiable\(\)](#), [m](#), [n](#), [x](#), and [y](#).

6.41.3.4 is_numeric()

```
bool GiNaC::ELi_kernel::is_numeric (
    void ) const [override], [virtual]
```

This routine returns true, if the integration kernel can be evaluated numerically.

Reimplemented from [GiNaC::integration_kernel](#).

References [GiNaC::ex::evalf\(\)](#), [GiNaC::ex::info\(\)](#), [m](#), [n](#), [GiNaC::info_flags::nonnegint](#), [GiNaC::info_flags::numeric](#), [x](#), and [y](#).

6.41.3.5 get_numerical_value()

```
ex GiNaC::ELi_kernel::get_numerical_value (
    const ex & qbar,
    int N_trunc = 0 ) const [override], [virtual]
```

Returns the value of $\text{ELi}_{\{n,m\}}(x,y,qbar)$

Reimplemented from [GiNaC::integration_kernel](#).

References [GiNaC::integration_kernel::get_numerical_value_impl\(\)](#), and [qbar](#).

6.41.3.6 series_coeff_impl()

```
cln::cl_N GiNaC::ELi_kernel::series_coeff_impl (
    int i ) const [override], [protected], [virtual]
```

For $\omega = d\lambda$ only the coefficient of λ^0 is non-zero.

The i-th coefficient corresponds to the power λ^{i-1} .

Reimplemented from [GiNaC::integration_kernel](#).

References [GiNaC::ex::evalf\(\)](#), [k](#), [m](#), [n](#), [x](#), and [y](#).

6.41.3.7 do_print()

```
void GiNaC::ELi_kernel::do_print (
    const print\_context & c,
    unsigned level ) const [protected]
```

References [c](#), [m](#), [n](#), [GiNaC::ex::print\(\)](#), [x](#), and [y](#).

6.41.4 Member Data Documentation

6.41.4.1 n

```
ex GiNaC::ELi_kernel::n [protected]
```

Referenced by [do_print\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [series_coeff_impl\(\)](#).

6.41.4.2 m

```
ex GiNaC::ELi_kernel::m [protected]
```

Referenced by [do_print\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [series_coeff_impl\(\)](#).

6.41.4.3 x

```
ex GiNaC::ELi_kernel::x [protected]
```

Referenced by [do_print\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [series_coeff_impl\(\)](#).

6.41.4.4 y

```
ex GiNaC::ELi_kernel::y [protected]
```

Referenced by [do_print\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [series_coeff_impl\(\)](#).

The documentation for this class was generated from the following files:

- [integration_kernel.h](#)
- [integration_kernel.cpp](#)

6.42 std::equal_to< GiNaC::ex > Struct Reference

Specialization of `std::equal_to()` for `ex` objects.

```
#include <ex.h>
```


Public Member Functions

- `bool operator()` (const `GiNaC::ex` &e1, const `GiNaC::ex` &e2) const noexcept

6.42.1 Detailed Description

Specialization of `std::equal_to()` for `ex` objects.

6.42.2 Member Function Documentation

6.42.2.1 operator()

```
bool std::equal_to< GiNaC::ex >::operator() (
    const GiNaC::ex & e1,
    const GiNaC::ex & e2 ) const [inline], [noexcept]
```

The documentation for this struct was generated from the following file:

- `ex.h`

6.43 GiNaC::error_and_integral Struct Reference

Public Member Functions

- `error_and_integral` (const `ex` &err, const `ex` &integ)

Public Attributes

- `ex error`
- `ex integral`

6.43.1 Constructor & Destructor Documentation

6.43.1.1 error_and_integral()

```
GiNaC::error_and_integral::error_and_integral (
    const ex & err,
    const ex & integ ) [inline]
```

6.43.2 Member Data Documentation

6.43.2.1 error

`ex` `GiNaC::error_and_integral::error`

Referenced by `GiNaC::error_and_integral_is_less::operator()()`.

6.43.2.2 integral

`ex` `GiNaC::error_and_integral::integral`

Referenced by `GiNaC::error_and_integral_is_less::operator()`.

The documentation for this struct was generated from the following file:

- `integral.cpp`

6.44 GiNaC::error_and_integral_is_less Struct Reference

Public Member Functions

- `bool operator()` (const `error_and_integral` &`e1`, const `error_and_integral` &`e2`) const

6.44.1 Member Function Documentation

6.44.1.1 operator()

```
bool GiNaC::error_and_integral_is_less::operator() (
    const error_and_integral & e1,
    const error_and_integral & e2 ) const    [inline]
```

References `c`, `GiNaC::ex::compare()`, `GiNaC::error_and_integral::error`, and `GiNaC::error_and_integral::integral`.

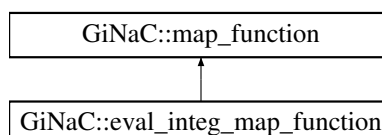
The documentation for this struct was generated from the following file:

- `integral.cpp`

6.45 GiNaC::eval_integ_map_function Struct Reference

Function object to be applied by `basic::eval_integ()`.

Inheritance diagram for `GiNaC::eval_integ_map_function`:



Public Member Functions

- `ex operator()` (const `ex` &`e`) override

Public Member Functions inherited from [GiNaC::map_function](#)

- virtual [~map_function](#) ()

Additional Inherited Members

Public Types inherited from [GiNaC::map_function](#)

- typedef const [ex](#) & [argument_type](#)
- typedef [ex](#) [result_type](#)

6.45.1 Detailed Description

Function object to be applied by [basic::eval_integ\(\)](#).

6.45.2 Member Function Documentation

6.45.2.1 operator>()

```
ex GiNaC::eval_integ_map_function::operator() (
    const ex & e ) [inline], [override], [virtual]
```

Implements [GiNaC::map_function](#).

References [GiNaC::eval_integ\(\)](#).

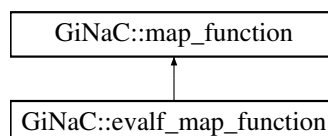
The documentation for this struct was generated from the following file:

- [basic.cpp](#)

6.46 GiNaC::evalf_map_function Struct Reference

Function object to be applied by [basic::evalf\(\)](#).

Inheritance diagram for [GiNaC::evalf_map_function](#):



Public Member Functions

- [ex operator\(\)](#) (const [ex](#) &e) override

Public Member Functions inherited from [GiNaC::map_function](#)

- virtual [~map_function\(\)](#)

Additional Inherited Members

Public Types inherited from [GiNaC::map_function](#)

- typedef const [ex](#) & [argument_type](#)
- typedef [ex](#) [result_type](#)

6.46.1 Detailed Description

Function object to be applied by [basic::evalf\(\)](#).

6.46.2 Member Function Documentation

6.46.2.1 operator>()

```
ex GiNaC::evalf_map_function::operator() (
    const ex & e ) [inline], [override], [virtual]
```

Implements [GiNaC::map_function](#).

References [GiNaC::evalf\(\)](#).

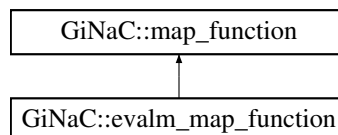
The documentation for this struct was generated from the following file:

- [basic.cpp](#)

6.47 GiNaC::evalm_map_function Struct Reference

Function object to be applied by [basic::evalm\(\)](#).

Inheritance diagram for [GiNaC::evalm_map_function](#):



Public Member Functions

- [ex operator\(\)](#) (const [ex](#) &e) override

Public Member Functions inherited from [GiNaC::map_function](#)

- virtual [~map_function](#) ()

Additional Inherited Members

Public Types inherited from [GiNaC::map_function](#)

- typedef const [ex](#) & [argument_type](#)
- typedef [ex](#) [result_type](#)

6.47.1 Detailed Description

Function object to be applied by [basic::evalm\(\)](#).

6.47.2 Member Function Documentation

6.47.2.1 operator>()

```
ex GiNaC::evalm_map_function::operator() (  
    const ex & e ) [inline], [override], [virtual]
```

Implements [GiNaC::map_function](#).

References [GiNaC::evalm\(\)](#).

The documentation for this struct was generated from the following file:

- [basic.cpp](#)

6.48 GiNaC::ex Class Reference

Lightweight wrapper for [GiNaC](#)'s symbolic objects.

```
#include <ex.h>
```

Public Member Functions

- `ex ()` noexcept
- `ex (const basic &other)`
- `ex (int i)`
- `ex (unsigned int i)`
- `ex (long i)`
- `ex (unsigned long i)`
- `ex (long long i)`
- `ex (unsigned long long i)`
- `ex (double const d)`
- `ex (const std::string &s, const ex &l)`
Construct ex from string and a list of symbols.
- `void swap (ex &other)` noexcept
Efficiently swap the contents of two expressions.
- `const_iterator begin ()` const noexcept
- `const_iterator end ()` const noexcept
- `const_preorder_iterator preorder_begin ()` const
- `const_preorder_iterator preorder_end ()` const noexcept
- `const_postorder_iterator postorder_begin ()` const
- `const_postorder_iterator postorder_end ()` const noexcept
- `ex eval ()` const
- `ex evalf ()` const
- `ex evalm ()` const
- `ex eval_ncmul (const exvector &v)` const
- `ex eval_integ ()` const
- `void print (const print_context &c, unsigned level=0)` const
Print expression to stream.
- `void dbgprint ()` const
Little wrapper around print to be called within a debugger.
- `void dbgprinttree ()` const
Little wrapper around printtree to be called within a debugger.
- `bool info (unsigned inf)` const
- `size_t nops ()` const
- `ex op (size_t i)` const
- `ex operator[] (const ex &index)` const
- `ex operator[] (size_t i)` const
- `ex & let_op (size_t i)`
Return modifiable operand/member at position i.
- `ex & operator[] (const ex &index)`
- `ex & operator[] (size_t i)`
- `ex lhs ()` const
Left hand side of relational expression.
- `ex rhs ()` const
Right hand side of relational expression.
- `ex conjugate ()` const
- `ex real_part ()` const
- `ex imag_part ()` const
- `bool has (const ex &pattern, unsigned options=0)` const
- `bool find (const ex &pattern, exset &found)` const
Find all occurrences of a pattern.
- `bool match (const ex &pattern)` const
Check whether expression matches a specified pattern.

- `bool match` (const `ex` &pattern, `exmap` &repls) const
- `ex subs` (const `exmap` &m, unsigned `options`=0) const
- `ex subs` (const `lst` &ls, const `lst` &lr, unsigned `options`=0) const
Substitute objects in an expression (syntactic substitution) and return the result as a new expression.
- `ex subs` (const `ex` &e, unsigned `options`=0) const
Substitute objects in an expression (syntactic substitution) and return the result as a new expression.
- `ex map` (`map_function` &f) const
- `ex map` (`ex`(*f)(const `ex` &e)) const
- `void accept` (`visitor` &v) const
- `void traverse_preorder` (`visitor` &v) const
Traverse expression tree with given visitor, preorder traversal.
- `void traverse_postorder` (`visitor` &v) const
Traverse expression tree with given visitor, postorder traversal.
- `void traverse` (`visitor` &v) const
- `bool is_polynomial` (const `ex` &vars) const
Check whether expression is a polynomial.
- `int degree` (const `ex` &s) const
- `int ldegree` (const `ex` &s) const
- `ex coeff` (const `ex` &s, int `n`=1) const
- `ex lcoeff` (const `ex` &s) const
- `ex tcoeff` (const `ex` &s) const
- `ex expand` (unsigned `options`=0) const
Expand an expression.
- `ex collect` (const `ex` &s, bool `distributed`=false) const
- `ex diff` (const `symbol` &s, unsigned `nth`=1) const
Compute partial derivative of an expression.
- `ex series` (const `ex` &r, int `order`, unsigned `options`=0) const
Compute the truncated series expansion of an expression.
- `ex normal` () const
Normalization of rational functions.
- `ex to_rational` (`exmap` &repl) const
Rationalization of non-rational functions.
- `ex to_polynomial` (`exmap` &repl) const
- `ex numer` () const
Get numerator of an expression.
- `ex denom` () const
Get denominator of an expression.
- `ex numer_denom` () const
Get numerator and denominator of an expression.
- `ex unit` (const `ex` &x) const
Compute unit part (= sign of leading coefficient) of a multivariate polynomial in $Q[x]$.
- `ex content` (const `ex` &x) const
Compute content part (= unit normal GCD of all coefficients) of a multivariate polynomial in $Q[x]$.
- `numeric_integer_content` () const
Compute the integer content (= GCD of all numeric coefficients) of an expanded polynomial.
- `ex primpart` (const `ex` &x) const
Compute primitive part of a multivariate polynomial in $Q[x]$.
- `ex primpart` (const `ex` &x, const `ex` &cont) const
Compute primitive part of a multivariate polynomial in $Q[x]$ when the content part is already known.
- `void unitcontprim` (const `ex` &x, `ex` &u, `ex` &c, `ex` &p) const
Compute unit part, content part, and primitive part of a multivariate polynomial in $Q[x]$.
- `ex smod` (const `numeric` &xi) const

- `numeric max_coefficient () const`
Return maximum (absolute value) coefficient of a polynomial.
- `exvector get_free_indices () const`
- `ex simplify_indexed (unsigned options=0) const`
Simplify/canonicalize expression containing indexed objects.
- `ex simplify_indexed (const scalar_products &sp, unsigned options=0) const`
Simplify/canonicalize expression containing indexed objects.
- `int compare (const ex &other) const`
- `bool is_equal (const ex &other) const`
- `bool is_zero () const`
- `bool is_zero_matrix () const`
Check whether expression is zero or zero matrix.
- `ex symmetrize () const`
Symmetrize expression over its free indices.
- `ex symmetrize (const lst &l) const`
Symmetrize expression over a list of objects (symbols, indices).
- `ex antisymmetrize () const`
Antisymmetrize expression over its free indices.
- `ex antisymmetrize (const lst &l) const`
Antisymmetrize expression over a list of objects (symbols, indices).
- `ex symmetrize_cyclic () const`
Symmetrize expression by cyclic permutation over its free indices.
- `ex symmetrize_cyclic (const lst &l) const`
Symmetrize expression by cyclic permutation over a list of objects (symbols, indices).
- `unsigned return_type () const`
- `return_type_t return_type_tinfo () const`
- `unsigned gethash () const`

Private Member Functions

- `void makewritable ()`
Make this ex writable (if more than one ex handle the same basic) by unlinking the object and creating an unshared copy of it.
- `void share (const ex &other) const`
Share equal objects between expressions.

Static Private Member Functions

- `static ptr< basic > construct_from_basic (const basic &other)`
Helper function for the ex-from-basic constructor.
- `static basic & construct_from_int (int i)`
- `static basic & construct_from_uint (unsigned int i)`
- `static basic & construct_from_long (long i)`
- `static basic & construct_from_ulong (unsigned long i)`
- `static basic & construct_from_longlong (long long i)`
- `static basic & construct_from_ulonglong (unsigned long long i)`
- `static basic & construct_from_double (double d)`
- `static ptr< basic > construct_from_string_and_lst (const std::string &s, const ex &l)`

Private Attributes

- `ptr< basic > bp`
pointer to basic object managed by this

Friends

- class `archive_node`
- bool `are_ex_trivially_equal` (const `ex` &, const `ex` &)
Compare two objects of class quickly without doing a deep tree traversal.
- template<class T >
const T & `ex_to` (const `ex` &)
Return a reference to the basic-derived class T object embedded in an expression.
- template<class T >
bool `is_a` (const `ex` &)
Check if ex is a handle to a T, including base classes.
- template<class T >
bool `is_exactly_a` (const `ex` &)
Check if ex is a handle to a T, not including base classes.

6.48.1 Detailed Description

Lightweight wrapper for GiNaC's symbolic objects.

It holds a pointer to the other object in order to do garbage collection by the method of reference counting. I.e., it is a smart pointer. Also, the constructor `ex::ex(const basic & other)` calls the methods that do automatic evaluation. E.g., `x-x` turns automatically into 0.

6.48.2 Constructor & Destructor Documentation

6.48.2.1 `ex()` [1/10]

```
GiNaC::ex::ex ( ) [inline], [noexcept]
```

References `bp`, `GiNaC::status_flags::dynallocated`, and `GINAC_ASSERT`.

6.48.2.2 `ex()` [2/10]

```
GiNaC::ex::ex (
    const basic & other ) [inline]
```

References `bp`, `GiNaC::status_flags::dynallocated`, and `GINAC_ASSERT`.

6.48.2.3 `ex()` [3/10]

```
GiNaC::ex::ex (
    int i ) [inline]
```

References `bp`, `GiNaC::status_flags::dynallocated`, and `GINAC_ASSERT`.

6.48.2.4 ex() [4/10]

```
GiNaC::ex::ex (
    unsigned int i ) [inline]
```

References [bp](#), [GiNaC::status_flags::dynallocated](#), and [GINAC_ASSERT](#).

6.48.2.5 ex() [5/10]

```
GiNaC::ex::ex (
    long i ) [inline]
```

References [bp](#), [GiNaC::status_flags::dynallocated](#), and [GINAC_ASSERT](#).

6.48.2.6 ex() [6/10]

```
GiNaC::ex::ex (
    unsigned long i ) [inline]
```

References [bp](#), [GiNaC::status_flags::dynallocated](#), and [GINAC_ASSERT](#).

6.48.2.7 ex() [7/10]

```
GiNaC::ex::ex (
    long long i ) [inline]
```

References [bp](#), [GiNaC::status_flags::dynallocated](#), and [GINAC_ASSERT](#).

6.48.2.8 ex() [8/10]

```
GiNaC::ex::ex (
    unsigned long long i ) [inline]
```

References [bp](#), [GiNaC::status_flags::dynallocated](#), and [GINAC_ASSERT](#).

6.48.2.9 ex() [9/10]

```
GiNaC::ex::ex (
    double const d ) [inline]
```

References [bp](#), [GiNaC::status_flags::dynallocated](#), and [GINAC_ASSERT](#).

6.48.2.10 ex() [10/10]

```
GiNaC::ex::ex (
    const std::string & s,
    const ex & l ) [inline]
```

Construct ex from string and a list of symbols.

The input grammar is similar to the [GiNaC](#) output format. All symbols and indices to be used in the expression must be specified in a list in the second argument. Undefined symbols and other parser errors will throw an exception.

References [bp](#), [GiNaC::status_flags::dynamically_allocated](#), and [GINAC_ASSERT](#).

6.48.3 Member Function Documentation

6.48.3.1 swap()

```
void GiNaC::ex::swap (
    ex & other ) [inline], [noexcept]
```

Efficiently swap the contents of two expressions.

References [bp](#), [GiNaC::status_flags::dynamically_allocated](#), and [GINAC_ASSERT](#).

Referenced by [GiNaC::ncmul::derivative\(\)](#), [GiNaC::ex_swap::operator\(\)\(\)](#), [GiNaC::swap\(\)](#), [GiNaC::expair::swap\(\)](#), and [std::swap\(\)](#).

6.48.3.2 begin()

```
const_iterator GiNaC::ex::begin ( ) const [inline], [noexcept]
```

Referenced by [GiNaC::abs_expand\(\)](#), [GiNaC::antisymmetrize\(\)](#), [GiNaC::mul::derivative\(\)](#), [GiNaC::divide_in_z\(\)](#), [GiNaC::function::eval_ncmul\(\)](#), [GiNaC::ncmul::evalm\(\)](#), [GiNaC::exp_expand\(\)](#), [GiNaC::mul::expandchildren\(\)](#), [GiNaC::G3_eval\(\)](#), [GiNaC::G3_evalf\(\)](#), [GiNaC::H_deriv\(\)](#), [GiNaC::H_eval\(\)](#), [GiNaC::H_print_latex\(\)](#), [GiNaC::Li_eval\(\)](#), [GiNaC::Li_evalf\(\)](#), [GiNaC::Li_print_latex\(\)](#), [GiNaC::log_expand\(\)](#), [GiNaC::expairseq::match\(\)](#), [GiNaC::archive_node::prinraw\(\)](#), [GiNaC::archive::prinraw\(\)](#), [GiNaC::rename_dummy_indices\(\)](#), [GiNaC::rename_dummy_indices_uniquely\(\)](#), [GiNaC::expairseq::subschildren\(\)](#), [GiNaC::symmetrize\(\)](#), [GiNaC::symmetrize_cyclic\(\)](#), and [GiNaC::zeta2_print_latex\(\)](#).

6.48.3.3 end()

```
const_iterator GiNaC::ex::end ( ) const [inline], [noexcept]
```

References [nops\(\)](#).

Referenced by [GiNaC::abs_expand\(\)](#), [GiNaC::mul::evalm\(\)](#), [GiNaC::exp_expand\(\)](#), [GiNaC::G3_eval\(\)](#), [GiNaC::G3_evalf\(\)](#), [GiNaC::Li_print_latex\(\)](#), [GiNaC::log_expand\(\)](#), and [GiNaC::rename_dummy_indices_uniquely\(\)](#).

6.48.3.4 preorder_begin()

```
const_preorder_iterator GiNaC::ex::preorder_begin ( ) const [inline]
```

References [nops\(\)](#).

6.48.3.5 preorder_end()

```
const_preorder_iterator GiNaC::ex::preorder_end ( ) const [inline], [noexcept]
```

6.48.3.6 postorder_begin()

```
const_postorder_iterator GiNaC::ex::postorder_begin ( ) const [inline]
```

References [nops\(\)](#).

6.48.3.7 postorder_end()

```
const_postorder_iterator GiNaC::ex::postorder_end ( ) const [inline], [noexcept]
```

6.48.3.8 eval()

```
ex GiNaC::ex::eval ( ) const [inline]
```

References [bp](#), and [eval\(\)](#).

Referenced by [eval\(\)](#), and [GiNaC::eval\(\)](#).

6.48.3.9 evalf()

```
ex GiNaC::ex::evalf ( ) const [inline]
```

References [bp](#), and [evalf\(\)](#).

Referenced by [GiNaC::adaptivesimpson\(\)](#), [GiNaC::EllipticE_evalf\(\)](#), [GiNaC::EllipticK_evalf\(\)](#), [evalf\(\)](#), [GiNaC::constant::evalf\(\)](#), [GiNaC::Gt::evalf\(\)](#), [GiNaC::integral::evalf\(\)](#), [GiNaC::mul::evalf\(\)](#), [GiNaC::power::evalf\(\)](#), [GiNaC::evalf\(\)](#), [GiNaC::evalf\(\)](#), [GiNaC::fsolve\(\)](#), [GiNaC::Kronecker_dtau_kernel::get_numerical_value\(\)](#), [GiNaC::integration_kernel::get_numerical_value_impl\(\)](#), [GiNaC::Gt::Gt\(\)](#), [GiNaC::Gt::Gt\(\)](#), [GiNaC::H_eval\(\)](#), [GiNaC::H_evalf\(\)](#), [GiNaC::multiple_polylog_kernel::is_numeric\(\)](#), [GiNaC::ELi_kernel::is_numeric\(\)](#), [GiNaC::Ebar_kernel::is_numeric\(\)](#), [GiNaC::Kronecker_dtau_kernel::is_numeric\(\)](#), [GiNaC::Kronecker_dz_kernel::is_numeric\(\)](#), [GiNaC::Eisenstein_kernel::is_numeric\(\)](#), [GiNaC::Eisenstein_h_kernel::is_numeric\(\)](#), [GiNaC::modular_form_kernel::is_numeric\(\)](#), [GiNaC::user_defined_kernel::is_numeric\(\)](#), [GiNaC::iterated_integral_evalf_impl\(\)](#), [GiNaC::Li_evalf\(\)](#), [GiNaC::Gt::qExpand\(\)](#), [GiNaC::S_evalf\(\)](#), [GiNaC::integration_kernel::series_coeff\(\)](#), [GiNaC::multiple_polylog_kernel::series_coeff_impl\(\)](#), [GiNaC::ELi_kernel::series_coeff_impl\(\)](#), [GiNaC::Ebar_kernel::series_coeff_impl\(\)](#), [GiNaC::Kronecker_dtau_kernel::series_coeff_impl\(\)](#), [GiNaC::Kronecker_dz_kernel::series_coeff_impl\(\)](#), and [GiNaC::subsvalue\(\)](#).

6.48.3.10 evalm()

```
ex GiNaC::ex::evalm ( ) const [inline]
```

References [bp](#), and [evalm\(\)](#).

Referenced by [evalm\(\)](#), [GiNaC::add::evalm\(\)](#), [GiNaC::mul::evalm\(\)](#), [GiNaC::power::evalm\(\)](#), [GiNaC::pseries::evalm\(\)](#), [GiNaC::evalm\(\)](#), and [is_zero_matrix\(\)](#).

6.48.3.11 eval_ncmul()

```
ex GiNaC::ex::eval_ncmul (
    const exvector & v ) const [inline]
```

References [bp](#), and [eval_ncmul\(\)](#).

Referenced by [eval_ncmul\(\)](#), [GiNaC::integral::eval_ncmul\(\)](#), and [GiNaC::relational::eval_ncmul\(\)](#).

6.48.3.12 eval_integ()

```
ex GiNaC::ex::eval_integ ( ) const [inline]
```

References [bp](#), and [eval_integ\(\)](#).

Referenced by [eval_integ\(\)](#), [GiNaC::integral::eval_integ\(\)](#), [GiNaC::pseries::eval_integ\(\)](#), and [GiNaC::eval_integ\(\)](#).

6.48.3.13 print()

```
void GiNaC::ex::print (
    const print\_context & c,
    unsigned level = 0 ) const
```

Print expression to stream.

The formatting of the output is determined by the kind of [print_context](#) object that is passed. Possible formattings include ginsh-parsable output (the default), tree-like output for debugging, and C++ source.

See also

[print_context](#)

References [bp](#), and [c](#).

Referenced by [GiNaC::abs_print_csrc_float\(\)](#), [GiNaC::abs_print_latex\(\)](#), [GiNaC::conjugate_print_latex\(\)](#), [GiNaC::integral::do_print\(\)](#), [GiNaC::multiple_polylog_kernel::do_print\(\)](#), [GiNaC::ELi_kernel::do_print\(\)](#), [GiNaC::Ebar_kernel::do_print\(\)](#), [GiNaC::Kronecker_dtau_kernel::do_print\(\)](#), [GiNaC::Kronecker_dz_kernel::do_print\(\)](#), [GiNaC::Eisenstein_kernel::do_print\(\)](#), [GiNaC::Eisenstein_h_kernel::do_print\(\)](#), [GiNaC::modular_form_kernel::do_print\(\)](#), [GiNaC::user_defined_kernel::do_print\(\)](#), [GiNaC::mul::do_print\(\)](#), [GiNaC::relational::do_print\(\)](#), [GiNaC::add::do_print_csrc\(\)](#), [GiNaC::idx::do_print_csrc\(\)](#), [GiNaC::mul::do_print_csrc\(\)](#), [GiNaC::power::do_print_csrc\(\)](#), [GiNaC::power::do_print_csrc_cl_N\(\)](#), [GiNaC::power::do_print_dflt\(\)](#), [GiNaC::integral::do_print_latex\(\)](#), [GiNaC::mul::do_print_latex\(\)](#), [GiNaC::power::do_print_latex\(\)](#), [GiNaC::add::do_print_python_repr\(\)](#), [GiNaC::mul::do_print_python_repr\(\)](#), [GiNaC::power::do_print_python_repr\(\)](#), [GiNaC::pseries::do_print_python_repr\(\)](#), [GiNaC::relational::do_print_python_repr\(\)](#), [GiNaC::basic::do_print_tree\(\)](#), [GiNaC::clifford::do_print_tree\(\)](#), [GiNaC::expairseq::do_print_tree\(\)](#), [GiNaC::idx::do_print_tree\(\)](#), [GiNaC::varidx::do_print_tree\(\)](#), [GiNaC::spinidx::do_print_tree\(\)](#), [GiNaC::indexed::do_print_tree\(\)](#), [GiNaC::pseries::do_print_tree\(\)](#), [GiNaC::factorial_print_dflt_latex\(\)](#), [GiNaC::H_print_latex\(\)](#), [GiNaC::imag_part_print_latex\(\)](#), [GiNaC::operator<<\(\)](#), [GiNaC::expair::print\(\)](#), [GiNaC::add::print_add\(\)](#), [GiNaC::idx::print_index\(\)](#), [GiNaC::mul::print_overall_coeff\(\)](#), [GiNaC::power::print_power\(\)](#), [GiNaC::pseries::print_series\(\)](#), [GiNaC::print_sym_pow\(\)](#), [GiNaC::expairseq::printpair\(\)](#), [GiNaC::expairseq::printseq\(\)](#), [GiNaC::real_part_print_latex\(\)](#), [GiNaC::S_print_latex\(\)](#), and [GiNaC::zeta1_print_latex\(\)](#).

6.48.3.14 dbgprint()

```
void GiNaC::ex::dbgprint ( ) const
```

Little wrapper around print to be called within a debugger.

References [bp](#).

6.48.3.15 dbgprinttree()

```
void GiNaC::ex::dbgprinttree ( ) const
```

Little wrapper around printtree to be called within a debugger.

References [bp](#).

6.48.3.16 info()

```
bool GiNaC::ex::info (
    unsigned int ) const [inline]
```

References [bp](#).

Referenced by [GiNaC::abs_eval\(\)](#), [GiNaC::abs_info\(\)](#), [GiNaC::abs_power\(\)](#), [GiNaC::acos_eval\(\)](#), [GiNaC::acosh_eval\(\)](#), [GiNaC::asin_eval\(\)](#), [GiNaC::asin_info\(\)](#), [GiNaC::asinh_conjugate\(\)](#), [GiNaC::asinh_eval\(\)](#), [GiNaC::atan2_eval\(\)](#), [GiNaC::atan2_info\(\)](#), [GiNaC::atan_conjugate\(\)](#), [GiNaC::atan_eval\(\)](#), [GiNaC::atan_info\(\)](#), [GiNaC::atanh_eval\(\)](#), [GiNaC::beta_eval\(\)](#), [GiNaC::beta_series\(\)](#), [GiNaC::mul::can_be_further_expanded\(\)](#), [GiNaC::mul::can_make_flat\(\)](#), [GiNaC::power::conjugate\(\)](#), [GiNaC::pseries::conjugate\(\)](#), [GiNaC::expairseq::construct_from_2_ex\(\)](#), [content\(\)](#), [GiNaC::cos_eval\(\)](#), [GiNaC::cosh_eval\(\)](#), [GiNaC::csgn_series\(\)](#), [GiNaC::matrix::determinant\(\)](#), [GiNaC::divide\(\)](#), [GiNaC::divide_in_z\(\)](#), [GiNaC::add::do_print_csrc\(\)](#), [GiNaC::idx::do_print_csrc\(\)](#), [GiNaC::power::do_print_csrc\(\)](#), [GiNaC::eta_eval\(\)](#), [GiNaC::eta_evalf\(\)](#), [GiNaC::eta_series\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::exp_eval\(\)](#), [GiNaC::exp_info\(\)](#), [GiNaC::exp_power\(\)](#), [GiNaC::power::expand\(\)](#), [GiNaC::find_common_factor\(\)](#), [GiNaC::func_arg_info\(\)](#), [GiNaC::G2_eval\(\)](#), [GiNaC::G2_evalf\(\)](#), [GiNaC::G3_eval\(\)](#), [GiNaC::G3_evalf\(\)](#), [GiNaC::gcd\(\)](#), [GiNaC::H_eval\(\)](#), [GiNaC::H_evalf\(\)](#), [GiNaC::power::has\(\)](#), [GiNaC::heur_gcd\(\)](#), [GiNaC::idx::idx\(\)](#), [GiNaC::power::imag_part\(\)](#), [GiNaC::pseries::imag_part\(\)](#), [GiNaC::add::info\(\)](#), [GiNaC::mul::info\(\)](#), [GiNaC::power::info\(\)](#), [GiNaC::multiple_polylog_kernel::is_numeric\(\)](#), [GiNaC::ELi_kernel::is_numeric\(\)](#), [GiNaC::Ebar_kernel::is_numeric\(\)](#), [GiNaC::Kronecker_dtau_kernel::is_numeric\(\)](#), [GiNaC::Kronecker_dz_kernel::is_numeric\(\)](#), [GiNaC::Eisenstein_kernel::is_numeric\(\)](#), [GiNaC::Eisenstein_h_kernel::is_numeric\(\)](#), [GiNaC::modular_form_kernel::is_numeric\(\)](#), [GiNaC::user_defined_kernel::is_numeric\(\)](#), [GiNaC::power::is_polynomial\(\)](#), [GiNaC::iterated_integral2_eval\(\)](#), [GiNaC::iterated_integral3_eval\(\)](#), [GiNaC::iterated_integral_evalf_impl\(\)](#), [GiNaC::lcm\(\)](#), [GiNaC::lcmcoeff\(\)](#), [GiNaC::lgamma_conjugate\(\)](#), [GiNaC::lgamma_eval\(\)](#), [GiNaC::lgamma_series\(\)](#), [GiNaC::Li2_conjugate\(\)](#), [GiNaC::Li2_eval\(\)](#), [GiNaC::Li2_series\(\)](#), [GiNaC::Li_eval\(\)](#), [GiNaC::Li_evalf\(\)](#), [GiNaC::Li_series\(\)](#), [GiNaC::log_conjugate\(\)](#), [GiNaC::log_eval\(\)](#), [GiNaC::log_expand\(\)](#), [GiNaC::log_imag_part\(\)](#), [GiNaC::log_info\(\)](#), [GiNaC::log_real_part\(\)](#), [GiNaC::log_series\(\)](#), [GiNaC::lsolve\(\)](#), [GiNaC::basic::normal\(\)](#), [GiNaC::power::normal\(\)](#), [GiNaC::relational::operator safe_bool\(\)](#), [GiNaC::Order_imag_part\(\)](#), [GiNaC::Order_power\(\)](#), [GiNaC::matrix::pow\(\)](#), [GiNaC::prem\(\)](#), [GiNaC::psi1_eval\(\)](#), [GiNaC::psi1_series\(\)](#), [GiNaC::psi2_eval\(\)](#), [GiNaC::psi2_series\(\)](#), [GiNaC::quo\(\)](#), [GiNaC::power::real_part\(\)](#), [GiNaC::pseries::real_part\(\)](#), [GiNaC::rem\(\)](#), [GiNaC::resultant\(\)](#), [GiNaC::S_eval\(\)](#), [GiNaC::S_evalf\(\)](#), [GiNaC::S_series\(\)](#), [GiNaC::mul::series\(\)](#), [GiNaC::power::series\(\)](#), [GiNaC::sin_eval\(\)](#), [GiNaC::sinh_eval\(\)](#), [GiNaC::sprem\(\)](#), [GiNaC::step_series\(\)](#), [subs\(\)](#), [GiNaC::tan_eval\(\)](#), [GiNaC::tanh_eval\(\)](#), [GiNaC::tgamma_eval\(\)](#), [GiNaC::tgamma_series\(\)](#), [GiNaC::expairseq::to_polynomial\(\)](#), [GiNaC::power::to_polynomial\(\)](#), [GiNaC::expairseq::to_rational\(\)](#), [GiNaC::power::to_rational\(\)](#), [GiNaC::matrix::trace\(\)](#), [GiNaC::trig_info\(\)](#), [GiNaC::tryfactsubs\(\)](#), [unit\(\)](#), [unitcontprim\(\)](#), [GiNaC::zeta2_deriv\(\)](#), and [GiNaC::zeta2_eval\(\)](#).

6.48.3.17 nops()

```
size_t GiNaC::ex::nops ( ) const [inline]
```

References [bp](#).

Referenced by [GiNaC::abs_expand\(\)](#), [GiNaC::matrix::add_indexed\(\)](#), [GiNaC::algebraic_match_mul_with_mul\(\)](#), [GiNaC::ncmul::append_factors\(\)](#), [GiNaC::basic::collect\(\)](#), [GiNaC::collect_symbols\(\)](#), [GiNaC::color_trace\(\)](#), [GiNaC::ncmul::count_factors\(\)](#), [GiNaC::csgn_eval\(\)](#), [GiNaC::const_postorder_iterator::descend\(\)](#), [GiNaC::divide\(\)](#), [end\(\)](#), [GiNaC::indexed::eval\(\)](#), [GiNaC::exp_expand\(\)](#), [GiNaC::indexed::expand\(\)](#), [GiNaC::integral::expand\(\)](#), [GiNaC::mul::expand\(\)](#), [find\(\)](#), [GiNaC::find_common_factor\(\)](#), [GiNaC::G2_eval\(\)](#), [GiNaC::G2_evalf\(\)](#), [GiNaC::G3_eval\(\)](#), [GiNaC::G3_evalf\(\)](#), [GiNaC::gcd_pf_mul\(\)](#), [GiNaC::get_all_dummy_indices_safely\(\)](#), [GiNaC::get_first_symbol\(\)](#), [GiNaC::get_symbol_stats\(\)](#), [GiNaC::H_evalf\(\)](#), [GiNaC::hasindex\(\)](#), [GiNaC::haswild\(\)](#), [GiNaC::Gt_detail::TransformExpressionWithCache\(\)](#), [GiNaC::const_preorder_iterator::increment\(\)](#), [GiNaC::lcmcoeff\(\)](#), [GiNaC::Li2_series\(\)](#), [GiNaC::Li_deriv\(\)](#), [GiNaC::Li_eval\(\)](#), [GiNaC::Li_evalf\(\)](#), [GiNaC::log_expand\(\)](#), [GiNaC::lsolve\(\)](#), [GiNaC::expairseq::match\(\)](#), [GiNaC::basic::match\(\)](#), [GiNaC::multiply_lcm\(\)](#), [GiNaC::nops\(\)](#), [postorder_begin\(\)](#), [preorder_begin\(\)](#), [GiNaC::product_to_exvector\(\)](#), [GiNaC::rename_dummy_indices_uniquely\(\)](#), [GiNaC::rename_dummy_indices_uniquely\(\)](#), [GiNaC::rename_dummy_indices_uniquely\(\)](#), [GiNaC::matrix::scalar_mul_indexed\(\)](#), [GiNaC::integral::series\(\)](#), [GiNaC::Gt::setArgs\(\)](#), [GiNaC::step_eval\(\)](#), [GiNaC::symminfo::symminfo\(\)](#), [traverse_postorder\(\)](#), [traverse_preorder\(\)](#), [GiNaC::zeta1_evalf\(\)](#), and [GiNaC::zeta2_evalf\(\)](#).

6.48.3.18 op()

```
ex GiNaC::ex::op (
    size_t i ) const [inline]
```

References [bp](#), and [op\(\)](#).

Referenced by [GiNaC::abs_eval\(\)](#), [GiNaC::matrix::add_indexed\(\)](#), [GiNaC::algebraic_match_mul_with_mul\(\)](#), [GiNaC::ncmul::append_factors\(\)](#), [GiNaC::atan_series\(\)](#), [GiNaC::atanh_series\(\)](#), [GiNaC::mul::can_be_further_expanded\(\)](#), [GiNaC::basic::collect\(\)](#), [GiNaC::collect_symbols\(\)](#), [GiNaC::color_trace\(\)](#), [GiNaC::su3f::contract_with\(\)](#), [GiNaC::su3d::contract_with\(\)](#), [GiNaC::matrix::contract_with\(\)](#), [GiNaC::cos_eval\(\)](#), [GiNaC::cosh_eval\(\)](#), [GiNaC::ncmul::count_factors\(\)](#), [GiNaC::csgn_eval\(\)](#), [GiNaC::decomp_rational\(\)](#), [denom\(\)](#), [GiNaC::const_postorder_iterator::descend\(\)](#), [GiNaC::divide\(\)](#), [GiNaC::divide_in_z\(\)](#), [GiNaC::epsilon_tensor\(\)](#), [GiNaC::epsilon_tensor\(\)](#), [GiNaC::indexed::eval\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::integral::eval_integ\(\)](#), [GiNaC::exp_eval\(\)](#), [GiNaC::indexed::expand\(\)](#), [GiNaC::integral::expand\(\)](#), [GiNaC::mul::expand\(\)](#), [find\(\)](#), [GiNaC::find_common_factor\(\)](#), [GiNaC::matrix::fraction_free_elimination\(\)](#), [GiNaC::G2_eval\(\)](#), [GiNaC::G2_evalf\(\)](#), [GiNaC::G3_eval\(\)](#), [GiNaC::G3_evalf\(\)](#), [GiNaC::gcd_pf_mul\(\)](#), [GiNaC::gcd_pf_pow\(\)](#), [GiNaC::gcd_pf_pow_pow\(\)](#), [GiNaC::get_all_dummy_indices_safely\(\)](#), [GiNaC::get_first_symbol\(\)](#), [GiNaC::clifford::get_metric\(\)](#), [GiNaC::H_evalf\(\)](#), [GiNaC::power::has\(\)](#), [GiNaC::hasindex\(\)](#), [GiNaC::haswild\(\)](#), [GiNaC::Gt_detail::TransformExpressionWithCache< Type >::impl\(\)](#), [GiNaC::const_preorder_iterator::increment\(\)](#), [GiNaC::lcmcoeff\(\)](#), [GiNaC::Li2_series\(\)](#), [GiNaC::Li_deriv\(\)](#), [GiNaC::Li_evalf\(\)](#), [GiNaC::log_eval\(\)](#), [GiNaC::lorentz_eps\(\)](#), [GiNaC::lsolve\(\)](#), [GiNaC::expairseq::match\(\)](#), [GiNaC::basic::match\(\)](#), [GiNaC::multiply_lcm\(\)](#), [normal\(\)](#), [GiNaC::basic::normal\(\)](#), [GiNaC::power::normal\(\)](#), [numer\(\)](#), [GiNaC::op\(\)](#), [op\(\)](#), [GiNaC::op0_is_equal::operator\(\)\(\)](#), [GiNaC::ex_base_is_less::operator\(\)\(\)](#), [GiNaC::const_iterator::operator*\(\)](#), [GiNaC::const_iterator::operator\[\]\(\)](#), [GiNaC::Order_eval\(\)](#), [GiNaC::product_to_exvector\(\)](#), [GiNaC::replace_with_symbol\(\)](#), [GiNaC::clifford::same_metric\(\)](#), [GiNaC::matrix::scalar_mul_indexed\(\)](#), [GiNaC::Gt::setArgs\(\)](#), [GiNaC::sin_eval\(\)](#), [GiNaC::sinh_eval\(\)](#), [GiNaC::spmapkey::spmapkey\(\)](#), [GiNaC::sqrfree_parfrac\(\)](#), [GiNaC::step_eval\(\)](#), [subs\(\)](#), [GiNaC::symminfo::symminfo\(\)](#), [GiNaC::tan_eval\(\)](#), [GiNaC::tanh_eval\(\)](#), [traverse_postorder\(\)](#), [traverse_preorder\(\)](#), [GiNaC::tryfactsubs\(\)](#), and [GiNaC::zeta2_deriv\(\)](#).

6.48.3.19 operator[]() [1/4]

```
ex GiNaC::ex::operator[] (
    const ex & index ) const [inline]
```

References [bp](#).

6.48.3.20 operator[]() [2/4]

```
ex GiNaC::ex::operator[] (
    size_t i ) const [inline]
```

References [bp](#).

6.48.3.21 let_op()

```
ex & GiNaC::ex::let_op (
    size_t i )
```

Return modifiable operand/member at position i.

References [bp](#), and [makewritable\(\)](#).

Referenced by [GiNaC::H_evalf\(\)](#).

6.48.3.22 operator[]() [3/4]

```
ex & GiNaC::ex::operator[] (
    const ex & index )
```

References [bp](#), and [makewritable\(\)](#).

6.48.3.23 operator[]() [4/4]

```
ex & GiNaC::ex::operator[] (
    size_t i )
```

References [bp](#), and [makewritable\(\)](#).

6.48.3.24 lhs()

```
ex GiNaC::ex::lhs ( ) const
```

Left hand side of relational expression.

References [bp](#).

Referenced by [GiNaC::fsolve\(\)](#), [GiNaC::lhs\(\)](#), [GiNaC::pseries::pseries\(\)](#), [GiNaC::pseries::pseries\(\)](#), [GiNaC::Eisenstein_kernel::series\(\)](#), [GiNaC::Eisenstein_h_kernel::series\(\)](#), [GiNaC::modular_form_kernel::series\(\)](#), [GiNaC::mul::series\(\)](#), and [GiNaC::power::series\(\)](#).

6.48.3.25 rhs()

```
ex GiNaC::ex::rhs ( ) const
```

Right hand side of relational expression.

References [bp](#).

Referenced by [GiNaC::fsolve\(\)](#), [GiNaC::pseries::pseries\(\)](#), [GiNaC::pseries::pseries\(\)](#), [GiNaC::rhs\(\)](#), [GiNaC::Eisenstein_h_kernel::series\(\)](#), [GiNaC::pseries::series\(\)](#), [GiNaC::mul::series\(\)](#), [GiNaC::power::series\(\)](#), and [GiNaC::symbol::series\(\)](#).

6.48.3.26 conjugate()

```
ex GiNaC::ex::conjugate ( ) const [inline]
```

References [bp](#), and [conjugate\(\)](#).

Referenced by [GiNaC::abs_expl_derivative\(\)](#), [GiNaC::abs_power\(\)](#), [GiNaC::acos_conjugate\(\)](#), [GiNaC::acosh_conjugate\(\)](#), [GiNaC::asin_conjugate\(\)](#), [GiNaC::asinh_conjugate\(\)](#), [GiNaC::atan_conjugate\(\)](#), [GiNaC::atanh_conjugate\(\)](#), [conjugate\(\)](#), [GiNaC::expair::conjugate\(\)](#), [GiNaC::add::conjugate\(\)](#), [GiNaC::container< C >::conjugate\(\)](#), [GiNaC::expairseq::conjugate\(\)](#), [GiNaC::integral::conjugate\(\)](#), [GiNaC::matrix::conjugate\(\)](#), [GiNaC::mul::conjugate\(\)](#), [GiNaC::power::conjugate\(\)](#), [GiNaC::pseries::conjugate\(\)](#), [GiNaC::conjugate\(\)](#), [GiNaC::conjugate_eval\(\)](#), [GiNaC::conjugate_evalf\(\)](#), [GiNaC::conjugateepvector\(\)](#), [GiNaC::cos_conjugate\(\)](#), [GiNaC::cosh_conjugate\(\)](#), [GiNaC::exp_conjugate\(\)](#), [GiNaC::lgamma_conjugate\(\)](#), [GiNaC::Li2_conjugate\(\)](#), [GiNaC::log_conjugate\(\)](#), [GiNaC::sin_conjugate\(\)](#), [GiNaC::sinh_conjugate\(\)](#), [GiNaC::tan_conjugate\(\)](#), [GiNaC::tanh_conjugate\(\)](#), and [GiNaC::tgamma_conjugate\(\)](#).

6.48.3.27 real_part()

```
ex GiNaC::ex::real_part ( ) const [inline]
```

References [bp](#), and [real_part\(\)](#).

Referenced by [GiNaC::abs_eval\(\)](#), [GiNaC::asinh_conjugate\(\)](#), [GiNaC::atan_conjugate\(\)](#), [GiNaC::conjugate_real_part\(\)](#), [GiNaC::mul::find_real_imag\(\)](#), [GiNaC::power::imag_part\(\)](#), [GiNaC::pseries::imag_part\(\)](#), [real_part\(\)](#), [GiNaC::add::real_part\(\)](#), [GiNaC::power::real_part\(\)](#), [GiNaC::pseries::real_part\(\)](#), [GiNaC::real_part\(\)](#), and [GiNaC::real_part_eval\(\)](#).

6.48.3.28 imag_part()

```
ex GiNaC::ex::imag_part ( ) const [inline]
```

References [bp](#), and [imag_part\(\)](#).

Referenced by [GiNaC::acos_conjugate\(\)](#), [GiNaC::acosh_conjugate\(\)](#), [GiNaC::asin_conjugate\(\)](#), [GiNaC::asinh_conjugate\(\)](#), [GiNaC::atan_conjugate\(\)](#), [GiNaC::atanh_conjugate\(\)](#), [GiNaC::conjugate_imag_part\(\)](#), [GiNaC::mul::find_real_imag\(\)](#), [imag_part\(\)](#), [GiNaC::add::imag_part\(\)](#), [GiNaC::power::imag_part\(\)](#), [GiNaC::imag_part\(\)](#), [GiNaC::imag_part_eval\(\)](#), [GiNaC::lgamma_conjugate\(\)](#), [GiNaC::Li2_conjugate\(\)](#), [GiNaC::log_conjugate\(\)](#), and [GiNaC::power::real_part\(\)](#).

6.48.3.29 has()

```
bool GiNaC::ex::has (
    const ex & pattern,
    unsigned options = 0 ) const [inline]
```

References [bp](#), and [options](#).

Referenced by [GiNaC::power::degree\(\)](#), [GiNaC::integral::eval\(\)](#), [GiNaC::integral::eval_integ\(\)](#), [GiNaC::integral::expand\(\)](#), [GiNaC::basic::has\(\)](#), [GiNaC::Gt::has\(\)](#), [GiNaC::has\(\)](#), [GiNaC::power::is_polynomial\(\)](#), and [GiNaC::power::ldegree\(\)](#).

6.48.3.30 find()

```
bool GiNaC::ex::find (
    const ex & pattern,
    exset & found ) const
```

Find all occurrences of a pattern.

The found matches are appended to the "found" list. If the expression itself matches the pattern, the children are not further examined. This function returns true when any matches were found.

References [find\(\)](#), [match\(\)](#), [nops\(\)](#), and [op\(\)](#).

Referenced by [GiNaC::divide_in_z\(\)](#), [GiNaC::scalar_products::evaluate\(\)](#), [find\(\)](#), [GiNaC::find\(\)](#), [GiNaC::Gt_detail::TransformExpression\(\)](#), [GiNaC::pseries::mul_series\(\)](#), [GiNaC::replace_with_symbol\(\)](#), [GiNaC::idx::subs\(\)](#), and [GiNaC::basic::subs_one_level\(\)](#).

6.48.3.31 match() [1/2]

```
bool GiNaC::ex::match (
    const ex & pattern ) const
```

Check whether expression matches a specified pattern.

References [bp](#), and [match\(\)](#).

Referenced by [find\(\)](#), [GiNaC::power::has\(\)](#), [match\(\)](#), [GiNaC::basic::match\(\)](#), [GiNaC::match\(\)](#), and [GiNaC::tryfactsubs\(\)](#).

6.48.3.32 match() [2/2]

```
bool GiNaC::ex::match (
    const ex & pattern,
    exmap & repls ) const [inline]
```

References [bp](#).

6.48.3.33 subs() [1/3]

```
ex GiNaC::ex::subs (
    const exmap & m,
    unsigned options = 0 ) const [inline]
```

References [bp](#), [m](#), [options](#), and [subs\(\)](#).

Referenced by [GiNaC::adaptivesimpson\(\)](#), [GiNaC::mul::algebraic_subs_mul\(\)](#), [GiNaC::atan_series\(\)](#), [GiNaC::atanh_series\(\)](#), [GiNaC::beta_series\(\)](#), [GiNaC::collect_common_factors\(\)](#), [GiNaC::integral::conjugate\(\)](#), [GiNaC::csgn_series\(\)](#), [denom\(\)](#), [GiNaC::integral::derivative\(\)](#), [GiNaC::divide_in_z\(\)](#), [GiNaC::EllipticE_series\(\)](#), [GiNaC::EllipticK_series\(\)](#), [GiNaC::eta_series\(\)](#), [GiNaC::tensdelta::eval_indexed\(\)](#), [GiNaC::integral::eval_integ\(\)](#), [GiNaC::integral::evalf\(\)](#), [GiNaC::expand_dummy_sum\(\)](#), [GiNaC::fsolve\(\)](#), [GiNaC::gcd\(\)](#), [GiNaC::clifford::get_metric\(\)](#), [GiNaC::H_evalf\(\)](#), [GiNaC::heur_gcd_z\(\)](#), [GiNaC::modular_form_kernel::is_numeric\(\)](#), [GiNaC::user_defined_kernel::is_numeric\(\)](#), [GiNaC::user_defined_kernel::Laurent_series\(\)](#), [GiNaC::lgamma_series\(\)](#), [GiNaC::Li2_series\(\)](#), [GiNaC::Li_series\(\)](#), [GiNaC::log_series\(\)](#), [normal\(\)](#), [GiNaC::basic::normal\(\)](#), [GiNaC::power::normal\(\)](#), [numer\(\)](#), [numer_denom\(\)](#), [GiNaC::psi1_series\(\)](#), [GiNaC::psi2_series\(\)](#), [GiNaC::rename_dummy_indices\(\)](#), [GiNaC::rename_dummy_indices_uniquely\(\)](#), [GiNaC::rename_dummy_indices_uniquely\(\)](#), [GiNaC::rename_dummy_indices_uniquely\(\)](#), [GiNaC::tensor::replace_contr_index\(\)](#), [GiNaC::replace_with_symbol\(\)](#), [GiNaC::replace_with_symbol\(\)](#), [GiNaC::S_series\(\)](#), [GiNaC::basic::series\(\)](#), [GiNaC::integral::series\(\)](#), [GiNaC::power::series\(\)](#), [GiNaC::step_series\(\)](#), [subs\(\)](#), [GiNaC::subs\(\)](#), [GiNaC::subs\(\)](#), [GiNaC::subs\(\)](#), [GiNaC::basic::subs\(\)](#), [subs\(\)](#), [GiNaC::clifford::subs\(\)](#), [GiNaC::Gt::subs\(\)](#), [GiNaC::idx::subs\(\)](#), [GiNaC::power::subs\(\)](#), [GiNaC::pseries::subs\(\)](#), [GiNaC::relational::subs\(\)](#), [GiNaC::basic::subs_one_level\(\)](#), [GiNaC::container< C >::subschildren\(\)](#), [GiNaC::expairseq::subschildren\(\)](#), [GiNaC::subsvalue\(\)](#), [GiNaC::symm\(\)](#), [GiNaC::symmetrize_cyclic\(\)](#), [GiNaC::tan_series\(\)](#), [GiNaC::tanh_series\(\)](#), [GiNaC::tgamma_series\(\)](#), and [GiNaC::Gt::to_numeric\(\)](#).

6.48.3.34 subs() [2/3]

```
ex GiNaC::ex::subs (
    const lst & ls,
    const lst & lr,
    unsigned options = 0 ) const
```

Substitute objects in an expression (syntactic substitution) and return the result as a new expression.

References [GiNaC::container< C >::begin\(\)](#), [bp](#), [GiNaC::container< C >::end\(\)](#), [GINAC_ASSERT](#), [lr](#), [m](#), [GiNaC::container< C >::nops\(\)](#), [options](#), [GiNaC::subs_options::pattern_is_not_product](#), and [GiNaC::subs_options::pattern_is_product](#).

6.48.3.35 subs() [3/3]

```
ex GiNaC::ex::subs (
    const ex & e,
    unsigned options = 0 ) const
```

Substitute objects in an expression (syntactic substitution) and return the result as a new expression.

There are two valid types of replacement arguments: 1) a relational like `object==ex` and 2) a list of relationals `lst{object1==ex1,object2==ex2,...}`.

References [bp](#), [GINAC_ASSERT](#), [info\(\)](#), [GiNaC::info_flags::list](#), [m](#), [op\(\)](#), [options](#), [GiNaC::subs_options::pattern_is_not_product](#), [GiNaC::subs_options::pattern_is_product](#), [r](#), [GiNaC::info_flags::relation_equal](#), and [subs\(\)](#).

6.48.3.36 map() [1/2]

```
ex GiNaC::ex::map (
    map_function & f ) const [inline]
```

References [bp](#), and [map\(\)](#).

Referenced by [GiNaC::color_trace\(\)](#), [GiNaC::expand_dummy_sum\(\)](#), and [map\(\)](#).

6.48.3.37 map() [2/2]

```
ex GiNaC::ex::map (
    ex(*) (const ex &e) f ) const
```

6.48.3.38 accept()

```
void GiNaC::ex::accept (
    visitor & v ) const [inline]
```

References [bp](#).

Referenced by [traverse_postorder\(\)](#), and [traverse_preorder\(\)](#).

6.48.3.39 traverse_preorder()

```
void GiNaC::ex::traverse_preorder (
    visitor & v ) const
```

Traverse expression tree with given visitor, preorder traversal.

References [accept\(\)](#), [n](#), [nops\(\)](#), [op\(\)](#), and [traverse_preorder\(\)](#).

Referenced by [traverse\(\)](#), and [traverse_preorder\(\)](#).

6.48.3.40 traverse_postorder()

```
void GiNaC::ex::traverse_postorder (
    visitor & v ) const
```

Traverse expression tree with given visitor, postorder traversal.

References [accept\(\)](#), [n](#), [nops\(\)](#), [op\(\)](#), and [traverse_postorder\(\)](#).

Referenced by [traverse_postorder\(\)](#).

6.48.3.41 traverse()

```
void GiNaC::ex::traverse (
    visitor & v ) const [inline]
```

References [traverse_preorder\(\)](#).

6.48.3.42 is_polynomial()

```
bool GiNaC::ex::is_polynomial (
    const ex & vars ) const
```

Check whether expression is a polynomial.

References [bp](#).

Referenced by [GiNaC::is_polynomial\(\)](#), and [GiNaC::power::is_polynomial\(\)](#).

6.48.3.43 degree()

```
int GiNaC::ex::degree (
    const ex & s ) const [inline]
```

References [bp](#).

Referenced by [GiNaC::basic::collect\(\)](#), [content\(\)](#), [GiNaC::integral::degree\(\)](#), [GiNaC::mul::degree\(\)](#), [GiNaC::power::degree\(\)](#), [GiNaC::degree\(\)](#), [GiNaC::divide\(\)](#), [GiNaC::divide_in_z\(\)](#), [GiNaC::get_symbol_stats\(\)](#), [GiNaC::heur_gcd_z\(\)](#), [lcoeff\(\)](#), [GiNaC::prem\(\)](#), [GiNaC::quo\(\)](#), [GiNaC::rem\(\)](#), [GiNaC::resultant\(\)](#), [GiNaC::sprem\(\)](#), and [GiNaC::sr_gcd\(\)](#).

6.48.3.44 ldegree()

```
int GiNaC::ex::ldegree (
    const ex & s ) const [inline]
```

References [bp](#).

Referenced by [content\(\)](#), [GiNaC::gcd_pf_pow\(\)](#), [GiNaC::get_symbol_stats\(\)](#), [GiNaC::mul::ldegree\(\)](#), [GiNaC::power::ldegree\(\)](#), [GiNaC::ldegree\(\)](#), [GiNaC::Order_series\(\)](#), [GiNaC::resultant\(\)](#), [GiNaC::mul::series\(\)](#), [GiNaC::power::series\(\)](#), and [tcoeff\(\)](#).

6.48.3.45 coeff()

```
ex GiNaC::ex::coeff (
    const ex & s,
    int n = 1 ) const [inline]
```

References [bp](#), [coeff\(\)](#), and [n](#).

Referenced by [GiNaC::Bernoulli_polynomial\(\)](#), [coeff\(\)](#), [GiNaC::add::coeff\(\)](#), [GiNaC::mul::coeff\(\)](#), [GiNaC::ncmul::coeff\(\)](#), [GiNaC::pseries::coeff\(\)](#), [GiNaC::coeff\(\)](#), [GiNaC::basic::collect\(\)](#), [content\(\)](#), [GiNaC::pseries::convert_to_poly\(\)](#), [GiNaC::mul::derivative\(\)](#), [GiNaC::pseries::derivative\(\)](#), [GiNaC::divide\(\)](#), [GiNaC::divide_in_z\(\)](#), [GiNaC::mul::eval\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::pseries::eval_integ\(\)](#), [GiNaC::pseries::evalm\(\)](#), [GiNaC::pseries::expand\(\)](#), [GiNaC::power::expand_add](#), [GiNaC::expairseq::expandchildren\(\)](#), [GiNaC::pseries::exponop\(\)](#), [GiNaC::generalised_Bernoulli_number\(\)](#), [GiNaC::add::imag_part\(\)](#), [lcoeff\(\)](#), [GiNaC::lsolve\(\)](#), [GiNaC::expairseq::make_flat\(\)](#), [GiNaC::pseries::normal\(\)](#), [GiNaC::prem\(\)](#), [GiNaC::pseries::print_series\(\)](#), [GiNaC::quo\(\)](#), [GiNaC::add::real_part\(\)](#), [GiNaC::rem\(\)](#), [GiNaC::resultant\(\)](#), [GiNaC::mul::series\(\)](#), [GiNaC::integration_kernel::series_coeff\(\)](#), [GiNaC::sprem\(\)](#), [GiNaC::sqrfree_parfrac\(\)](#), [GiNaC::expairseq::subschildren\(\)](#), and [tcoeff\(\)](#).

6.48.3.46 lcoeff()

```
ex GiNaC::ex::lcoeff (
    const ex & s ) const [inline]
```

References [coeff\(\)](#), and [degree\(\)](#).

Referenced by [content\(\)](#), [GiNaC::get_symbol_stats\(\)](#), [GiNaC::sr_gcd\(\)](#), and [unit\(\)](#).

6.48.3.47 tcoeff()

```
ex GiNaC::ex::tcoeff (
    const ex & s ) const [inline]
```

References [coeff\(\)](#), and [ldegree\(\)](#).

6.48.3.48 expand()

```
ex GiNaC::ex::expand (
    unsigned options = 0 ) const
```

Expand an expression.

Parameters

<i>options</i>	see GiNaC::expand_options

References [bp](#), [expand\(\)](#), [GiNaC::status_flags::expanded](#), and [options](#).

Referenced by [GiNaC::abs_expand\(\)](#), [GiNaC::binomial_sym\(\)](#), [GiNaC::color_trace\(\)](#), [content\(\)](#), [GiNaC::matrix::determinant\(\)](#), [GiNaC::matrix::determinant_minor\(\)](#), [GiNaC::divide\(\)](#), [GiNaC::divide_in_z\(\)](#), [GiNaC::exp_expand\(\)](#), [GiNaC::expand\(\)](#), [GiNaC::expand\(\)](#), [expand\(\)](#), [GiNaC::indexed::expand\(\)](#), [GiNaC::integral::expand\(\)](#), [GiNaC::mul::expand\(\)](#), [GiNaC::power::expand\(\)](#), [GiNaC::pseries::expand\(\)](#), [GiNaC::expand_dummy_sum\(\)](#), [GiNaC::expairseq::expandchildren\(\)](#), [GiNaC::mul::expandchildren\(\)](#), [GiNaC::ncmul::expandchildren\(\)](#), [GiNaC::mul::find_real_imag\(\)](#), [GiNaC::frac_cancel\(\)](#), [GiNaC::matrix::fraction_free_elimination\(\)](#), [GiNaC::gcd\(\)](#), [GiNaC::heur_gcd_z\(\)](#), [GiNaC::log_expand\(\)](#), [GiNaC::lsolve\(\)](#), [GiNaC::expand_map_function::operator\(\)](#), [GiNaC::prem\(\)](#), [GiNaC::quo\(\)](#), [GiNaC::rem\(\)](#), [GiNaC::resultant\(\)](#), [GiNaC::basic::series\(\)](#), [GiNaC::mul::series\(\)](#), [GiNaC::power::series\(\)](#), [GiNaC::sprem\(\)](#), [GiNaC::sqrfree_parfrac\(\)](#), [GiNaC::sr_gcd\(\)](#), [GiNaC::matrix::trace\(\)](#), [unit\(\)](#), and [unitcontprim\(\)](#).

6.48.3.49 collect()

```
ex GiNaC::ex::collect (
    const ex & s,
    bool distributed = false ) const [inline]
```

References [bp](#), and [collect\(\)](#).

Referenced by [GiNaC::matrix::charpoly\(\)](#), [collect\(\)](#), and [GiNaC::collect\(\)](#).

6.48.3.50 diff()

```
ex GiNaC::ex::diff (
    const symbol & s,
    unsigned nth = 1 ) const
```

Compute partial derivative of an expression.

Parameters

<i>s</i>	symbol by which the expression is derived
<i>nth</i>	order of derivative (default 1)

Returns

partial derivative as a new expression

References [bp](#), and [diff\(\)](#).

Referenced by [GiNaC::abs_expl_derivative\(\)](#), [GiNaC::conjugate_expl_derivative\(\)](#), [GiNaC::fderivative::derivative\(\)](#), [GiNaC::function::derivative\(\)](#), [GiNaC::integral::derivative\(\)](#), [GiNaC::ncmul::derivative\(\)](#), [GiNaC::power::derivative\(\)](#), [GiNaC::pseries::derivative\(\)](#), [GiNaC::diff\(\)](#), [GiNaC::basic::diff\(\)](#), [diff\(\)](#), [GiNaC::fsolve\(\)](#), [GiNaC::imag_part_expl_derivative\(\)](#), [GiNaC::log_series\(\)](#), [GiNaC::Order_expl_derivative\(\)](#), [GiNaC::real_part_expl_derivative\(\)](#), [GiNaC::basic::series\(\)](#), and [GiNaC::sqrfree_yun\(\)](#).

6.48.3.51 series()

```
ex GiNaC::ex::series (
    const ex & r,
    int order,
    unsigned options = 0 ) const
```

Compute the truncated series expansion of an expression.

This function returns an expression containing an object of class pseries to represent the series. If the series does not terminate within the given truncation order, the last term of the series will be an order term.

Parameters

<i>r</i>	expansion relation, lhs holds variable and rhs holds point
<i>order</i>	truncation order of series calculations
<i>options</i>	of class series_options

Returns

an expression holding a pseries object

References [GiNaC::_ex0](#), [bp](#), [options](#), [order](#), [r](#), and [series\(\)](#).

Referenced by [GiNaC::Bernoulli_polynomial\(\)](#), [GiNaC::EllipticE_series\(\)](#), [GiNaC::EllipticK_series\(\)](#), [GiNaC::generalised_Bernoulli_nu](#), [GiNaC::modular_form_kernel::Laurent_series\(\)](#), [GiNaC::integration_kernel::Laurent_series\(\)](#), [GiNaC::Eisenstein_kernel::Laurent_ser](#), [GiNaC::Eisenstein_h_kernel::Laurent_series\(\)](#), [GiNaC::user_defined_kernel::Laurent_series\(\)](#), [GiNaC::Li2_series\(\)](#), [GiNaC::Li_series\(\)](#), [GiNaC::log_series\(\)](#), [GiNaC::Eisenstein_h_kernel::q_expansion_modular_form\(\)](#), [GiNaC::S_series\(\)](#), [series\(\)](#), [GiNaC::series\(\)](#), [GiNaC::add::series\(\)](#), [GiNaC::integral::series\(\)](#), [GiNaC::Eisenstein_kernel::series\(\)](#), [GiNaC::Eisenstein_h_kernel::series\(\)](#), [GiNaC::modular_form_kernel::series\(\)](#), [GiNaC::pseries::series\(\)](#), [GiNaC::mul::series\(\)](#), [GiNaC::power::series\(\)](#), and [GiNaC::tgamma_series\(\)](#).

6.48.3.52 normal()

```
ex GiNaC::ex::normal ( ) const
```

Normalization of rational functions.

This function converts an expression to its normal form "numerator/denominator", where numerator and denominator are (relatively prime) polynomials. Any subexpressions which are not rational functions (like non-rational numbers, non-integer powers or functions like [sin\(\)](#), [cos\(\)](#) etc.) are replaced by temporary symbols which are re-substituted by the (normalized) subexpressions before [normal\(\)](#) returns (this way, any expression can be treated as a rational function). [normal\(\)](#) is applied recursively to arguments of functions etc.

Returns

normalized expression

References [bp](#), [GINAC_ASSERT](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::container< C >::nops\(\)](#), [normal\(\)](#), [op\(\)](#), [GiNaC::container< C >::op\(\)](#), and [subs\(\)](#).

Referenced by [denom\(\)](#), [GiNaC::matrix::determinant\(\)](#), [GiNaC::matrix::fraction_free_elimination\(\)](#), [normal\(\)](#), [GiNaC::normal\(\)](#), [GiNaC::add::normal\(\)](#), [GiNaC::power::normal\(\)](#), [GiNaC::pseries::normal\(\)](#), [numer\(\)](#), [numer_denom\(\)](#), [GiNaC::matrix::trace\(\)](#), and [GiNaC::Gt::zisToFundamental\(\)](#).

6.48.3.53 to_rational()

```
ex GiNaC::ex::to_rational (
    exmap & repl ) const
```

Rationalization of non-rational functions.

This function converts a general expression to a rational function by replacing all non-rational subexpressions (like non-rational numbers, non-integer powers or functions like [sin\(\)](#), [cos\(\)](#) etc.) to temporary symbols. This makes it possible to use functions like [gcd\(\)](#) and [divide\(\)](#) on non-rational functions by applying [to_rational\(\)](#) on the arguments, calling the desired function and re-substituting the temporary symbols in the result. To make the last step possible, all temporary symbols and their associated expressions are collected in the map specified by the `repl` parameter, ready to be passed as an argument to [ex::subs\(\)](#).

Parameters

<code>repl</code>	collects all temporary symbols and their replacements

Returns

rationalized expression

References [bp](#), and [to_rational\(\)](#).

Referenced by [GiNaC::matrix::determinant\(\)](#), [GiNaC::matrix::fraction_free_elimination\(\)](#), [GiNaC::to_rational\(\)](#), [to_rational\(\)](#), [GiNaC::expairseq::to_rational\(\)](#), and [GiNaC::power::to_rational\(\)](#).

6.48.3.54 to_polynomial()

```
ex GiNaC::ex::to_polynomial (
    exmap & repl ) const
```

References [bp](#), and [to_polynomial\(\)](#).

Referenced by [GiNaC::find_common_factor\(\)](#), [GiNaC::to_polynomial\(\)](#), [to_polynomial\(\)](#), [GiNaC::expairseq::to_polynomial\(\)](#), and [GiNaC::power::to_polynomial\(\)](#).

6.48.3.55 numer()

```
ex GiNaC::ex::numer ( ) const
```

Get numerator of an expression.

If the expression is not of the normal form "numerator/denominator", it is first converted to this form and then the numerator is returned.

See also

[ex::normal](#)

Returns

numerator

References [bp](#), [GINAC_ASSERT](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::container< C >::nops\(\)](#), [normal\(\)](#), [op\(\)](#), [GiNaC::container< C >::op\(\)](#), and [subs\(\)](#).

Referenced by [GiNaC::numer\(\)](#), and [GiNaC::numer\(\)](#).

6.48.3.56 denom()

```
ex GiNaC::ex::denom ( ) const
```

Get denominator of an expression.

If the expression is not of the normal form "numerator/denominator", it is first converted to this form and then the denominator is returned.

See also

[ex::normal](#)

Returns

denominator

References [bp](#), [GINAC_ASSERT](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::container< C >::nops\(\)](#), [normal\(\)](#), [op\(\)](#), [GiNaC::container< C >::op\(\)](#), and [subs\(\)](#).

Referenced by [GiNaC::denom\(\)](#), and [GiNaC::denom\(\)](#).

6.48.3.57 numer_denom()

```
ex GiNaC::ex::numer_denom ( ) const
```

Get numerator and denominator of an expression.

If the expression is not of the normal form "numerator/denominator", it is first converted to this form and then a list [numerator, denominator] is returned.

See also

[ex::normal](#)

Returns

a list [numerator, denominator]

References [bp](#), [GINAC_ASSERT](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::container< C >::nops\(\)](#), [normal\(\)](#), [GiNaC::container< C >::op\(\)](#), and [subs\(\)](#).

Referenced by [GiNaC::matrix::fraction_free_elimination\(\)](#), and [GiNaC::numer_denom\(\)](#).

6.48.3.58 unit()

```
ex GiNaC::ex::unit (
    const ex & x ) const
```

Compute unit part (= sign of leading coefficient) of a multivariate polynomial in $\mathbb{Q}[x]$.

The product of unit part, content part, and primitive part is the polynomial itself.

Parameters

<code>x</code>	main variable

Returns

unit part

See also

[ex::content](#), [ex::primpart](#), [ex::unitcontprim](#)

References [GiNaC::_ex1](#), [GiNaC::_ex_1](#), [c](#), [expand\(\)](#), [GiNaC::get_first_symbol\(\)](#), [info\(\)](#), [lcoeff\(\)](#), [GiNaC::info_flags::negative](#), [unit\(\)](#), and [x](#).

Referenced by [content\(\)](#), [GiNaC::frac_cancel\(\)](#), and [unit\(\)](#).

6.48.3.59 content()

```
ex GiNaC::ex::content (
    const ex & x ) const
```

Compute content part (= unit normal GCD of all coefficients) of a multivariate polynomial in $\mathbb{Q}[x]$.

The product of unit part, content part, and primitive part is the polynomial itself.

Parameters

<code>x</code>	main variable

Returns

content part

See also

[ex::unit](#), [ex::primpart](#), [ex::unitcontprim](#)

References [GiNaC::_ex0](#), [c](#), [coeff\(\)](#), [cont](#), [degree\(\)](#), [expand\(\)](#), [GiNaC::gcd\(\)](#), [info\(\)](#), [GiNaC::info_flags::integer](#), [integer_content\(\)](#), [is_zero\(\)](#), [lcoeff\(\)](#), [ldegree\(\)](#), [GiNaC::info_flags::negative](#), [r](#), [unit\(\)](#), and [x](#).

Referenced by [GiNaC::sr_gcd\(\)](#), and [unitcontprim\(\)](#).

6.48.3.60 integer_content()

```
numeric GiNaC::ex::integer_content ( ) const
```

Compute the integer content (= GCD of all numeric coefficients) of an expanded polynomial.

For a polynomial with rational coefficients, this returns g/l where g is the GCD of the coefficients' numerators and l is the LCM of the coefficients' denominators.

Returns

integer content

References [bp](#), and [GiNaC::numeric::integer_content\(\)](#).

Referenced by [content\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::gcd\(\)](#), and [GiNaC::heur_gcd_z\(\)](#).

6.48.3.61 primpart() [1/2]

```
ex GiNaC::ex::primpart (
    const ex & x ) const
```

Compute primitive part of a multivariate polynomial in $\mathbb{Q}[x]$.

The result will be a unit-normal polynomial with a content part of 1. The product of unit part, content part, and primitive part is the polynomial itself.

Parameters

x	main variable

Returns

primitive part

See also

[ex::unit](#), [ex::content](#), [ex::unitcontprim](#)

References [c](#), [unitcontprim\(\)](#), and [x](#).

Referenced by [GiNaC::sr_gcd\(\)](#).

6.48.3.62 primpart() [2/2]

```
ex GiNaC::ex::primpart (
    const ex & x,
    const ex & c ) const
```

Compute primitive part of a multivariate polynomial in $\mathbb{Q}[x]$ when the content part is already known.

This function is faster in computing the primitive part than the previous function.

Parameters

x	main variable
c	previously computed content part

Returns

primitive part

References [GiNaC::_ex0](#), [GiNaC::_ex1](#), [c](#), [is_zero\(\)](#), [GiNaC::quo\(\)](#), [unit](#), and [x](#).

6.48.3.63 unitcontprim()

```
void GiNaC::ex::unitcontprim (
    const ex & x,
    ex & u,
    ex & c,
    ex & p ) const
```

Compute unit part, content part, and primitive part of a multivariate polynomial in $\mathbb{Q}[x]$.

The product of the three parts is the polynomial itself.

Parameters

x	main variable
u	unit part (returned)
c	content part (returned)
p	primitive part (returned)

See also

[ex::unit](#), [ex::content](#), [ex::primpart](#)

References [GiNaC::_ex0](#), [GiNaC::_ex1](#), [GiNaC::_ex_1](#), [GiNaC::abs\(\)](#), [c](#), [content\(\)](#), [expand\(\)](#), [info\(\)](#), [is_zero\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::quo\(\)](#), [unit](#), and [x](#).

Referenced by [GiNaC::gcd\(\)](#), and [primpart\(\)](#).

6.48.3.64 smod()

```
ex GiNaC::ex::smod (
    const numeric & xi ) const [inline]
```

References [bp](#), and [smod\(\)](#).

Referenced by [GiNaC::interpolate\(\)](#), and [smod\(\)](#).

6.48.3.65 max_coefficient()

```
numeric GiNaC::ex::max_coefficient ( ) const
```

Return maximum (absolute value) coefficient of a polynomial.

This function is used internally by [heur_gcd\(\)](#).

Returns

maximum coefficient

See also

[heur_gcd](#)

References [bp](#), and [GiNaC::numeric::max_coefficient\(\)](#).

Referenced by [GiNaC::heur_gcd_z\(\)](#).

6.48.3.66 get_free_indices()

```
exvector GiNaC::ex::get_free_indices ( ) const [inline]
```

References [bp](#).

Referenced by [antisymmetrize\(\)](#), [GiNaC::get_all_dummy_indices_safely\(\)](#), [GiNaC::add::get_free_indices\(\)](#), [GiNaC::integral::get_free_indices\(\)](#), [GiNaC::mul::get_free_indices\(\)](#), [GiNaC::ncmul::get_free_indices\(\)](#), [GiNaC::clifford::get_metric\(\)](#), [GiNaC::clifford::same_metric\(\)](#), [symmetrize\(\)](#), and [symmetrize_cyclic\(\)](#).

6.48.3.67 simplify_indexed() [1/2]

```
ex GiNaC::ex::simplify_indexed (
    unsigned options = 0 ) const
```

Simplify/canonicalize expression containing indexed objects.

This performs contraction of dummy indices where possible and checks whether the free indices in sums are consistent.

Parameters

<i>options</i>	Simplification options (currently unused)

Returns

simplified expression

References [GiNaC::simplify_indexed\(\)](#).

Referenced by [GiNaC::tensepsilon::contract_with\(\)](#), [GiNaC::simplify_indexed\(\)](#), and [GiNaC::simplify_indexed\(\)](#).

6.48.3.68 simplify_indexed() [2/2]

```
ex GiNaC::ex::simplify_indexed (
    const scalar_products & sp,
    unsigned options = 0 ) const
```

Simplify/canonicalize expression containing indexed objects.

This performs contraction of dummy indices where possible, checks whether the free indices in sums are consistent, and automatically replaces scalar products by known values if desired.

Parameters

<i>sp</i>	Scalar products to be replaced automatically
<i>options</i>	Simplification options (currently unused)

Returns

simplified expression

References [GiNaC::simplify_indexed\(\)](#).

6.48.3.69 compare()

```
int GiNaC::ex::compare (
    const ex & other ) const [inline]
```

References [bp](#), and [share\(\)](#).

Referenced by [GiNaC::expair::compare\(\)](#), [GiNaC::expairseq::construct_from_2_ex\(\)](#), [GiNaC::expair::is_less\(\)](#), [GiNaC::error_and_integral_is_less::operator\(\)](#), [GiNaC::ex_is_less::operator\(\)](#), [GiNaC::expair_rest_is_less::operator\(\)](#), [GiNaC::symminfo_is_less_by_symmterm::operator\(\)](#), [GiNaC::symminfo_is_less_by_orig::operator\(\)](#), [GiNaC::terminfo_is_less::operator\(\)](#), [GiNaC::spmapkey::operator<\(\)](#), and [GiNaC::spmapkey::spmapkey\(\)](#).

6.48.3.70 is_equal()

```
bool GiNaC::ex::is_equal (
    const ex & other ) const [inline]
```

References [bp](#), and [share\(\)](#).

Referenced by [GiNaC::abs_power\(\)](#), [GiNaC::acos_eval\(\)](#), [GiNaC::acosh_eval\(\)](#), [GiNaC::matrix::add_indexed\(\)](#), [GiNaC::asin_eval\(\)](#), [GiNaC::atan2_eval\(\)](#), [GiNaC::atan_eval\(\)](#), [GiNaC::atan_series\(\)](#), [GiNaC::atanh_eval\(\)](#), [GiNaC::atanh_series\(\)](#), [GiNaC::beta_eval\(\)](#), [GiNaC::power::coeff\(\)](#), [GiNaC::pseries::coeff\(\)](#), [GiNaC::add::combine_ex_with_coeff_to\(\)](#), [GiNaC::mul::combine_pair_with_coeff_to_pair\(\)](#), [GiNaC::conjugateepvector\(\)](#), [GiNaC::power::degree\(\)](#), [GiNaC::pseries::degree\(\)](#), [GiNaC::divide\(\)](#), [GiNaC::divide_in_z\(\)](#), [GiNaC::mul::do_print_csrc\(\)](#), [GiNaC::power::do_print_csrc\(\)](#), [GiNaC::power::do_print_csrc_cl\(\)](#), [GiNaC::power::do_print_dflt\(\)](#), [GiNaC::power::do_print_latex\(\)](#), [GiNaC::expairseq::do_print_tree\(\)](#), [GiNaC::epsilon_tensor\(\)](#), [GiNaC::epsilon_tensor\(\)](#), [GiNaC::mul::eval\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::matrix::eval_indexed\(\)](#), [GiNaC::tensdelta::eval_indexed\(\)](#), [GiNaC::tensmetric::eval_indexed\(\)](#), [GiNaC::mul::expand\(\)](#), [GiNaC::find_common_factor\(\)](#), [GiNaC::frac_cancel\(\)](#), [GiNaC::gcd\(\)](#), [GiNaC::gcd_pf_pow\(\)](#), [GiNaC::gcd_pf_pow_pow\(\)](#), [GiNaC::make_flat_inserter::handle_factor\(\)](#), [GiNaC::power::imag_part\(\)](#), [GiNaC::mul::info\(\)](#), [GiNaC::expair::is_canonical_numeric\(\)](#), [GiNaC::pseries::is_compatible_to\(\)](#), [GiNaC::idx::is_dummy_pair_same_type\(\)](#), [GiNaC::expair::is_equal\(\)](#), [GiNaC::expairseq::is_equal_same_type\(\)](#), [is_zero\(\)](#), [GiNaC::power::lddegree\(\)](#), [GiNaC::pseries::lddegree\(\)](#), [GiNaC::Li2_eval\(\)](#), [GiNaC::Li2_series\(\)](#), [GiNaC::Li_eval\(\)](#), [GiNaC::log_eval\(\)](#), [GiNaC::lorentz_eps\(\)](#), [GiNaC::expairseq::map\(\)](#), [GiNaC::expairseq::match\(\)](#), [GiNaC::idx::match_same_type\(\)](#), [GiNaC::minimal_dim\(\)](#), [GiNaC::pseries::mul_series\(\)](#), [GiNaC::multiply_lcm\(\)](#), [GiNaC::expairseq::nops\(\)](#), [GiNaC::power::normal\(\)](#), [GiNaC::expairseq::op\(\)](#), [GiNaC::ex_is_equal::operator\(\)](#), [GiNaC::op0_is_equal::operator\(\)](#), [GiNaC::idx_is_equal_ignore_dim::operator\(\)](#), [GiNaC::spmapkey::operator==\(\)](#), [GiNaC::add::print_add\(\)](#), [GiNaC::mul::print_overall_coeff\(\)](#), [GiNaC::expairseq::printseq\(\)](#), [GiNaC::product_to_exvector\(\)](#), [GiNaC::quo\(\)](#), [GiNaC::power::real_part\(\)](#), [GiNaC::mul::recombine_pair_to_ex\(\)](#), [GiNaC::rem\(\)](#), [GiNaC::replace_with_symbol\(\)](#), [GiNaC::clifford::same_metric\(\)](#), [GiNaC::pseries::series\(\)](#), [GiNaC::power::series\(\)](#), [GiNaC::add::split_ex_to_pair\(\)](#), and [GiNaC::sqrfree_yun\(\)](#).

6.48.3.71 is_zero()

```
bool GiNaC::ex::is_zero ( ) const [inline]
```

References [GiNaC::_ex0](#), and [is_equal\(\)](#).

Referenced by [GiNaC::acos_conjugate\(\)](#), [GiNaC::acos_eval\(\)](#), [GiNaC::acosh_conjugate\(\)](#), [GiNaC::acosh_eval\(\)](#), [GiNaC::pseries::add_series\(\)](#), [GiNaC::asin_conjugate\(\)](#), [GiNaC::asin_eval\(\)](#), [GiNaC::asinh_eval\(\)](#), [GiNaC::atan\(\)](#), [GiNaC::atan2_eval\(\)](#), [GiNaC::atan_eval\(\)](#), [GiNaC::atanh_conjugate\(\)](#), [GiNaC::atanh_eval\(\)](#), [GiNaC::add::coeff\(\)](#), [content\(\)](#), [GiNaC::cosh_eval\(\)](#), [GiNaC::add::degree\(\)](#), [GiNaC::fderivative::derivative\(\)](#), [GiNaC::function::derivative\(\)](#), [GiNaC::matrix::determinant\(\)](#), [GiNaC::matrix::determinant_minor\(\)](#), [GiNaC::basic::diff\(\)](#), [GiNaC::divide\(\)](#), [GiNaC::divide_in_z\(\)](#), [GiNaC::add::do_print_csrc\(\)](#), [GiNaC::matrix::echelon_form\(\)](#), [GiNaC::add::eval\(\)](#), [GiNaC::function::eval\(\)](#), [GiNaC::indexed::eval\(\)](#), [GiNaC::mul::eval\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::pseries::evalm\(\)](#), [GiNaC::Gt_detail::pathintegral_term::evaluate\(\)](#), [GiNaC::exp_eval\(\)](#), [GiNaC::mul::expand\(\)](#), [GiNaC::pseries::expand\(\)](#), [GiNaC::power::expand_add\(\)](#), [GiNaC::power::expand_add_2\(\)](#), [GiNaC::find_common_factor\(\)](#), [GiNaC::mul::find_real_imag\(\)](#), [GiNaC::frac_cancel\(\)](#), [GiNaC::matrix::fraction_free_elimination\(\)](#), [GiNaC::gcd\(\)](#), [GiNaC::heur_gcd_z\(\)](#), [GiNaC::add::imag_part\(\)](#), [GiNaC::add::info\(\)](#), [GiNaC::Gt_detail::pathintegral_term::integrate\(\)](#), [GiNaC::interpolate\(\)](#), [GiNaC::is_zero\(\)](#), [is_zero_matrix\(\)](#), [GiNaC::add::lddegree\(\)](#), [GiNaC::lgamma_conjugate\(\)](#), [GiNaC::Li2_conjugate\(\)](#), [GiNaC::Li2_eval\(\)](#), [GiNaC::Li2_series\(\)](#), [GiNaC::Li3_eval\(\)](#), [GiNaC::Li_series\(\)](#), [GiNaC::log\(\)](#), [GiNaC::log_conjugate\(\)](#), [GiNaC::log_eval\(\)](#), [GiNaC::log_series\(\)](#), [GiNaC::matrix::markowitz_elimination\(\)](#), [GiNaC::pseries::mul_series\(\)](#), [GiNaC::pseries::normal\(\)](#), [GiNaC::relational::operator safe_bool\(\)](#), [GiNaC::Order_eval\(\)](#), [GiNaC::Gt_detail::pathintegral_term::pathint](#), [GiNaC::prem\(\)](#), [primpart\(\)](#), [GiNaC::add::print_add\(\)](#), [GiNaC::pseries::print_series\(\)](#), [GiNaC::quo\(\)](#), [GiNaC::add::real_part\(\)](#), [GiNaC::rem\(\)](#), [GiNaC::S_series\(\)](#), [GiNaC::clifford::same_metric\(\)](#), [GiNaC::basic::series\(\)](#), [GiNaC::power::series\(\)](#), [GiNaC::symbol::series\(\)](#), [GiNaC::sinh_eval\(\)](#), [GiNaC::add::smod\(\)](#), [GiNaC::sprem\(\)](#), [GiNaC::sqrfree_yun\(\)](#), [GiNaC::tanh_eval\(\)](#), [GiNaC::Gt::tauToFundamental\(\)](#), [unitcontprim\(\)](#), and [GiNaC::indexed::validate\(\)](#).

6.48.3.72 is_zero_matrix()

```
bool GiNaC::ex::is_zero_matrix ( ) const
```

Check whether expression is zero or zero matrix.

References [evalm\(\)](#), and [is_zero\(\)](#).

6.48.3.73 symmetrize() [1/2]

```
ex GiNaC::ex::symmetrize ( ) const
```

Symmetrize expression over its free indices.

References [get_free_indices\(\)](#), and [GiNaC::symmetrize\(\)](#).

Referenced by [GiNaC::symmetrize\(\)](#), and [GiNaC::symmetrize\(\)](#).

6.48.3.74 symmetrize() [2/2]

```
ex GiNaC::ex::symmetrize (
    const lst & l ) const
```

Symmetrize expression over a list of objects (symbols, indices).

References [GiNaC::container< C >::begin\(\)](#), [GiNaC::container< C >::end\(\)](#), and [GiNaC::symm\(\)](#).

6.48.3.75 antisymmetrize() [1/2]

```
ex GiNaC::ex::antisymmetrize ( ) const
```

Antisymmetrize expression over its free indices.

References [GiNaC::antisymmetrize\(\)](#), and [get_free_indices\(\)](#).

Referenced by [GiNaC::antisymmetrize\(\)](#), and [GiNaC::antisymmetrize\(\)](#).

6.48.3.76 antisymmetrize() [2/2]

```
ex GiNaC::ex::antisymmetrize (
    const lst & l ) const
```

Antisymmetrize expression over a list of objects (symbols, indices).

References [GiNaC::container< C >::begin\(\)](#), [GiNaC::container< C >::end\(\)](#), and [GiNaC::symm\(\)](#).

6.48.3.77 symmetrize_cyclic() [1/2]

```
ex GiNaC::ex::symmetrize_cyclic ( ) const
```

Symmetrize expression by cyclic permutation over its free indices.

References [get_free_indices\(\)](#), and [GiNaC::symmetrize_cyclic\(\)](#).

Referenced by [GiNaC::symmetrize_cyclic\(\)](#), and [GiNaC::symmetrize_cyclic\(\)](#).

6.48.3.78 symmetrize_cyclic() [2/2]

```
ex GiNaC::ex::symmetrize_cyclic (
    const lst & l ) const
```

Symmetrize expression by cyclic permutation over a list of objects (symbols, indices).

References [GiNaC::container< C >::begin\(\)](#), [GiNaC::container< C >::end\(\)](#), and [GiNaC::symmetrize_cyclic\(\)](#).

6.48.3.79 return_type()

```
unsigned GiNaC::ex::return_type ( ) const [inline]
```

References [bp](#).

Referenced by [GiNaC::ncmul::append_factors\(\)](#), [GiNaC::ncmul::count_factors\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::exmul\(\)](#), [GiNaC::matrix::mul_scalar\(\)](#), [GiNaC::indexed::return_type\(\)](#), [GiNaC::integral::return_type\(\)](#), [GiNaC::power::return_type\(\)](#), and [GiNaC::relational::return_type\(\)](#).

6.48.3.80 return_type_tinfo()

```
return_type_t GiNaC::ex::return_type_tinfo ( ) const [inline]
```

References [bp](#).

Referenced by [GiNaC::color_trace\(\)](#), [GiNaC::indexed::return_type_tinfo\(\)](#), [GiNaC::integral::return_type_tinfo\(\)](#), [GiNaC::power::return_type_tinfo\(\)](#), and [GiNaC::relational::return_type_tinfo\(\)](#).

6.48.3.81 gethash()

```
unsigned GiNaC::ex::gethash ( ) const [inline]
```

References [bp](#).

Referenced by [GiNaC::basic::calchash\(\)](#), [GiNaC::expairseq::calchash\(\)](#), [GiNaC::function::calchash\(\)](#), [GiNaC::Gt::calchash\(\)](#), [GiNaC::idx::calchash\(\)](#), and [GiNaC::relational::calchash\(\)](#).

6.48.3.82 construct_from_basic()

```
ptr< basic > GiNaC::ex::construct_from_basic (
    const basic & other ) [static], [private]
```

Helper function for the ex-from-basic constructor.

This is where [GiNaC](#)'s automatic evaluator and memory management are implemented.

See also

[ex::ex\(const basic &\)](#)

References [bp](#), [GiNaC::basic::duplicate\(\)](#), [GiNaC::status_flags::dynallocated](#), [GiNaC::basic::eval\(\)](#), [GiNaC::status_flags::evaluated](#), [GiNaC::basic::flags](#), [GiNaC::refcounted::get_refcount\(\)](#), [GINAC_ASSERT](#), and [GiNaC::basic::setflag\(\)](#).

6.48.3.83 construct_from_int()

```
basic & GiNaC::ex::construct_from_int (
    int i ) [static], [private]
```

References [GiNaC::_num0_p](#), [GiNaC::_num10_p](#), [GiNaC::_num11_p](#), [GiNaC::_num12_p](#), [GiNaC::_num1_p](#), [GiNaC::_num2_p](#), [GiNaC::_num3_p](#), [GiNaC::_num4_p](#), [GiNaC::_num5_p](#), [GiNaC::_num6_p](#), [GiNaC::_num7_p](#), [GiNaC::_num8_p](#), [GiNaC::_num9_p](#), [GiNaC::_num_10_p](#), [GiNaC::_num_11_p](#), [GiNaC::_num_12_p](#), [GiNaC::_num_1_p](#), [GiNaC::_num_2_p](#), [GiNaC::_num_3_p](#), [GiNaC::_num_4_p](#), [GiNaC::_num_5_p](#), [GiNaC::_num_6_p](#), [GiNaC::_num_7_p](#), [GiNaC::_num_8_p](#), and [GiNaC::_num_9_p](#).

Referenced by [construct_from_longlong\(\)](#).

6.48.3.84 construct_from_uint()

```
basic & GiNaC::ex::construct_from_uint (
    unsigned int i ) [static], [private]
```

References [GiNaC::_num0_p](#), [GiNaC::_num10_p](#), [GiNaC::_num11_p](#), [GiNaC::_num12_p](#), [GiNaC::_num1_p](#), [GiNaC::_num2_p](#), [GiNaC::_num3_p](#), [GiNaC::_num4_p](#), [GiNaC::_num5_p](#), [GiNaC::_num6_p](#), [GiNaC::_num7_p](#), [GiNaC::_num8_p](#), and [GiNaC::_num9_p](#).

Referenced by [construct_from_ulonglong\(\)](#).

6.48.3.85 construct_from_long()

```
basic & GiNaC::ex::construct_from_long (
    long i ) [static], [private]
```

References [GiNaC::_num0_p](#), [GiNaC::_num10_p](#), [GiNaC::_num11_p](#), [GiNaC::_num12_p](#), [GiNaC::_num1_p](#), [GiNaC::_num2_p](#), [GiNaC::_num3_p](#), [GiNaC::_num4_p](#), [GiNaC::_num5_p](#), [GiNaC::_num6_p](#), [GiNaC::_num7_p](#), [GiNaC::_num8_p](#), [GiNaC::_num9_p](#), [GiNaC::_num_10_p](#), [GiNaC::_num_11_p](#), [GiNaC::_num_12_p](#), [GiNaC::_num_1_p](#), [GiNaC::_num_2_p](#), [GiNaC::_num_3_p](#), [GiNaC::_num_4_p](#), [GiNaC::_num_5_p](#), [GiNaC::_num_6_p](#), [GiNaC::_num_7_p](#), [GiNaC::_num_8_p](#), and [GiNaC::_num_9_p](#).

6.48.3.86 construct_from_ulong()

```
basic & GiNaC::ex::construct_from_ulong (
    unsigned long i ) [static], [private]
```

References [GiNaC::_num0_p](#), [GiNaC::_num10_p](#), [GiNaC::_num11_p](#), [GiNaC::_num12_p](#), [GiNaC::_num1_p](#), [GiNaC::_num2_p](#), [GiNaC::_num3_p](#), [GiNaC::_num4_p](#), [GiNaC::_num5_p](#), [GiNaC::_num6_p](#), [GiNaC::_num7_p](#), [GiNaC::_num8_p](#), and [GiNaC::_num9_p](#).

6.48.3.87 construct_from_longlong()

```
basic & GiNaC::ex::construct_from_longlong (
    long long i ) [static], [private]
```

References [construct_from_int\(\)](#).

6.48.3.88 construct_from_ulonglong()

```
basic & GiNaC::ex::construct_from_ulonglong (
    unsigned long long i ) [static], [private]
```

References [construct_from_uint\(\)](#).

6.48.3.89 construct_from_double()

```
basic & GiNaC::ex::construct_from_double (
    double d ) [static], [private]
```

6.48.3.90 construct_from_string_and_lst()

```
static ptr< basic > GiNaC::ex::construct_from_string_and_lst (
    const std::string & s,
    const ex & l ) [static], [private]
```

6.48.3.91 makewriteable()

```
void GiNaC::ex::makewriteable ( ) [private]
```

Make this ex writable (if more than one ex handle the same basic) by unlinking the object and creating an unshared copy of it.

References [bp](#), [GiNaC::status_flags::dynallocated](#), and [GINAC_ASSERT](#).

Referenced by [let_op\(\)](#), [operator\[\]\(\)](#), and [operator\[\]\(\)](#).

6.48.3.92 share()

```
void GiNaC::ex::share (
    const ex & other ) const [private]
```

Share equal objects between expressions.

See also

[ex::compare\(const ex &\)](#)

References [bp](#), and [GiNaC::status_flags::not_shareable](#).

Referenced by [compare\(\)](#), and [is_equal\(\)](#).

6.48.4 Friends And Related Symbol Documentation

6.48.4.1 archive_node

```
friend class archive_node [friend]
```

6.48.4.2 are_ex_trivially_equal

```
bool are_ex_trivially_equal (
    const ex & e1,
    const ex & e2 ) [friend]
```

Compare two objects of class quickly without doing a deep tree traversal.

Returns

"true" if they are equal "false" if equality cannot be established quickly (e1 and e2 may still be equal, in this case).

6.48.4.3 ex_to

```
template<class T >
const T & ex_to (
    const ex & e ) [friend]
```

Return a reference to the basic-derived class T object embedded in an expression.

This is fast but unsafe: the result is undefined if the expression does not contain a T object at its top level. Hence, you should generally check the type of e first. Also, you shouldn't cache the returned reference because [GiNaC's](#) garbage collector may destroy the referenced object any time it's used in another expression.

Parameters

<i>e</i>	expression

Returns

reference to object of class T

See also

[is_exactly_a<class T>\(\)](#)

6.48.4.4 is_a

```
template<class T >
bool is_a (
    const ex & obj ) [friend]
```

Check if *ex* is a handle to a T, including base classes.

6.48.4.5 is_exactly_a

```
template<class T >
bool is_exactly_a (
    const ex & obj ) [friend]
```

Check if *ex* is a handle to a T, not including base classes.

6.48.5 Member Data Documentation**6.48.5.1 bp**

```
ptr<basic> GiNaC::ex::bp [mutable], [private]
```

pointer to basic object managed by this

Referenced by [accept\(\)](#), [GiNaC::archive_node::archive_node\(\)](#), [coeff\(\)](#), [collect\(\)](#), [compare\(\)](#), [conjugate\(\)](#), [construct_from_basic\(\)](#), [dbgprint\(\)](#), [dbgprinttree\(\)](#), [degree\(\)](#), [denom\(\)](#), [diff\(\)](#), [eval\(\)](#), [eval_integ\(\)](#), [eval_ncmul\(\)](#), [evalf\(\)](#), [evalm\(\)](#), [ex\(\)](#), [ex\(\)](#), [ex\(\)](#), [ex\(\)](#), [ex\(\)](#), [ex\(\)](#), [ex\(\)](#), [ex\(\)](#), [ex\(\)](#), [ex\(\)](#), [ex\(\)](#), [expand\(\)](#), [get_free_indices\(\)](#), [gethash\(\)](#), [has\(\)](#), [GiNaC::archive_node::has_same_ex_as\(\)](#), [imag_part\(\)](#), [info\(\)](#), [integer_content\(\)](#), [is_equal\(\)](#), [is_polynomial\(\)](#), [ldegree\(\)](#), [let_op\(\)](#), [lhs\(\)](#), [makewriteable\(\)](#), [map\(\)](#), [match\(\)](#), [match\(\)](#), [max_coefficient\(\)](#), [nops\(\)](#), [normal\(\)](#), [numer\(\)](#), [numer_denom\(\)](#), [op\(\)](#), [operator\[\]\(\)](#), [operator\[\]\(\)](#), [operator\[\]\(\)](#), [operator\[\]\(\)](#), [print\(\)](#), [GiNaC::archive_node::printraw\(\)](#), [real_part\(\)](#), [return_type\(\)](#), [return_type_tinfo\(\)](#), [rhs\(\)](#), [series\(\)](#), [share\(\)](#), [smod\(\)](#), [subs\(\)](#), [subs\(\)](#), [subs\(\)](#), [swap\(\)](#), [to_polynomial\(\)](#), and [to_rational\(\)](#).

The documentation for this class was generated from the following files:

- [ex.h](#)
- [ex.cpp](#)
- [indexed.cpp](#)
- [normal.cpp](#)
- [pseries.cpp](#)
- [symmetry.cpp](#)

6.49 GiNaC::ex_base_is_less Struct Reference

Public Member Functions

- `bool operator()` (const `ex` &lh, const `ex` &rh) const

6.49.1 Member Function Documentation

6.49.1.1 operator()

```
bool GiNaC::ex_base_is_less::operator() (
    const ex & lh,
    const ex & rh ) const    [inline]
```

References [GiNaC::ex::op\(\)](#).

The documentation for this struct was generated from the following file:

- [indexed.cpp](#)

6.50 GiNaC::ex_is_equal Struct Reference

```
#include <ex.h>
```

Public Member Functions

- `bool operator()` (const `ex` &lh, const `ex` &rh) const

6.50.1 Member Function Documentation

6.50.1.1 operator()

```
bool GiNaC::ex_is_equal::operator() (
    const ex & lh,
    const ex & rh ) const    [inline]
```

References [GiNaC::ex::is_equal\(\)](#).

The documentation for this struct was generated from the following file:

- [ex.h](#)

6.51 GiNaC::ex_is_less Struct Reference

```
#include <ex.h>
```

Public Member Functions

- bool [operator\(\)](#) (const [ex](#) &lh, const [ex](#) &rh) const

6.51.1 Member Function Documentation

6.51.1.1 [operator\(\)](#)

```
bool GiNaC::ex_is_less::operator() (
    const ex & lh,
    const ex & rh ) const [inline]
```

References [GiNaC::ex::compare\(\)](#).

The documentation for this struct was generated from the following file:

- [ex.h](#)

6.52 GiNaC::ex_swap Struct Reference

```
#include <ex.h>
```

Public Member Functions

- void [operator\(\)](#) ([ex](#) &lh, [ex](#) &rh) const

6.52.1 Member Function Documentation

6.52.1.1 [operator\(\)](#)

```
void GiNaC::ex_swap::operator() (
    ex & lh,
    ex & rh ) const [inline]
```

References [GiNaC::ex::swap\(\)](#).

The documentation for this struct was generated from the following file:

- [ex.h](#)

6.53 GiNaC::expair Class Reference

A pair of expressions.

```
#include <expair.h>
```

Public Member Functions

- [expair](#) ()
- [expair](#) (const [ex](#) &r, const [ex](#) &c)
Construct an expair from two ex.
- bool [is_equal](#) (const [expair](#) &other) const
Member-wise check for canonical ordering equality.
- bool [is_less](#) (const [expair](#) &other) const
Member-wise check for canonical ordering lessness.
- int [compare](#) (const [expair](#) &other) const
Member-wise check for canonical ordering.
- void [print](#) (std::ostream &os) const
- bool [is_canonical_numeric](#) () const
True if this is of the form (numeric,ex(1)).
- void [swap](#) ([expair](#) &other)
Swap contents with other expair.
- const [expair](#) [conjugate](#) () const

Public Attributes

- [ex rest](#)
first member of pair, an arbitrary expression
- [ex coeff](#)
second member of pair, must be numeric

6.53.1 Detailed Description

A pair of expressions.

This is similar to STL's `pair<>`. It is slightly extended since we need to account for methods like `.compare()`. Also, since this is meant for use by class `expairseq` it must satisfy the invariance that the member `coeff` must be of type `numeric`.

6.53.2 Constructor & Destructor Documentation

6.53.2.1 `expair()` [1/2]

```
GiNaC::expair::expair ( ) [inline]
```

Referenced by `conjugate()`.

6.53.2.2 `expair()` [2/2]

```
GiNaC::expair::expair (
    const ex & r,
    const ex & c ) [inline]
```

Construct an expair from two ex.

References `coeff`, and `GINAC_ASSERT`.

6.53.3 Member Function Documentation

6.53.3.1 is_equal()

```
bool GiNaC::expair::is_equal (
    const expair & other ) const [inline]
```

Member-wise check for canonical ordering equality.

References [coeff](#), [GiNaC::ex::is_equal\(\)](#), and [rest](#).

Referenced by [GiNaC::expairseq::evalchildren\(\)](#), [GiNaC::mul::expair_needs_further_processing\(\)](#), and [GiNaC::expairseq::subschildre](#)

6.53.3.2 is_less()

```
bool GiNaC::expair::is_less (
    const expair & other ) const [inline]
```

Member-wise check for canonical ordering lessness.

References [coeff](#), [GiNaC::ex::compare\(\)](#), and [rest](#).

Referenced by [GiNaC::expair_is_less::operator\(\)\(\)](#).

6.53.3.3 compare()

```
int GiNaC::expair::compare (
    const expair & other ) const [inline]
```

Member-wise check for canonical ordering.

References [coeff](#), [GiNaC::ex::compare\(\)](#), and [rest](#).

6.53.3.4 print()

```
void GiNaC::expair::print (
    std::ostream & os ) const
```

References [c](#), [coeff](#), [GiNaC::ex::print\(\)](#), and [rest](#).

6.53.3.5 is_canonical_numeric()

```
bool GiNaC::expair::is_canonical_numeric ( ) const [inline]
```

True if this is of the form (numeric,ex(1)).

References [coeff](#), [GINAC_ASSERT](#), [GiNaC::ex::is_equal\(\)](#), and [rest](#).

6.53.3.6 swap()

```
void GiNaC::expair::swap (
    expair & other ) [inline]
```

Swap contents with other expair.

References [coeff](#), [rest](#), and [GiNaC::ex::swap\(\)](#).

Referenced by [GiNaC::mul::derivative\(\)](#), [GiNaC::expair_swap::operator\(\)](#), and [GiNaC::swap\(\)](#).

6.53.3.7 conjugate()

```
const expair GiNaC::expair::conjugate ( ) const
```

References [GiNaC::are_ex_trivially_equal\(\)](#), [coeff](#), [GiNaC::ex::conjugate\(\)](#), [expair\(\)](#), and [rest](#).

6.53.4 Member Data Documentation

6.53.4.1 rest

```
ex GiNaC::expair::rest
```

first member of pair, an arbitrary expression

Referenced by [GiNaC::expairseq::combine_pair_with_coeff_to_pair\(\)](#), [GiNaC::add::combine_pair_with_coeff_to_pair\(\)](#), [GiNaC::mul::combine_pair_with_coeff_to_pair\(\)](#), [compare\(\)](#), [conjugate\(\)](#), [GiNaC::expairseq::construct_from_2_ex\(\)](#), [GiNaC::expairseq::construct_from_expairseq_ex\(\)](#), [is_canonical_numeric\(\)](#), [is_equal\(\)](#), [is_less\(\)](#), [GiNaC::expair_rest_is_less::operator\(\)](#), [print\(\)](#), [GiNaC::expairseq::printpair\(\)](#), [GiNaC::expairseq::recombine_pair_to_ex\(\)](#), [GiNaC::add::recombine_pair_to_ex\(\)](#), [GiNaC::mul::recombine_pair_to_ex\(\)](#), and [swap\(\)](#).

6.53.4.2 coeff

```
ex GiNaC::expair::coeff
```

second member of pair, must be numeric

Referenced by [GiNaC::mul::can_make_flat\(\)](#), [GiNaC::expairseq::combine_pair_with_coeff_to_pair\(\)](#), [GiNaC::add::combine_pair_with_coeff_to_pair\(\)](#), [GiNaC::mul::combine_pair_with_coeff_to_pair\(\)](#), [compare\(\)](#), [conjugate\(\)](#), [GiNaC::expairseq::construct_from_2_ex\(\)](#), [GiNaC::expairseq::construct_from_expairseq_ex\(\)](#), [expair\(\)](#), [GiNaC::power::expand_mul\(\)](#), [is_canonical_numeric\(\)](#), [is_equal\(\)](#), [is_less\(\)](#), [print\(\)](#), [GiNaC::expairseq::printpair\(\)](#), [GiNaC::expairseq::recombine_pair_to_ex\(\)](#), [GiNaC::add::recombine_pair_to_ex\(\)](#), [GiNaC::mul::recombine_pair_to_ex\(\)](#), and [swap\(\)](#).

The documentation for this class was generated from the following files:

- [expair.h](#)
- [expair.cpp](#)

6.54 GiNaC::expair_is_less Struct Reference

Function object for insertion into third argument of STL's sort() etc.

```
#include <expair.h>
```

Public Member Functions

- bool [operator\(\)](#) (const [expair](#) &lh, const [expair](#) &rh) const

6.54.1 Detailed Description

Function object for insertion into third argument of STL's sort() etc.

6.54.2 Member Function Documentation

6.54.2.1 operator()

```
bool GiNaC::expair_is_less::operator() (
    const expair & lh,
    const expair & rh ) const    [inline]
```

References [GiNaC::expair::is_less\(\)](#).

The documentation for this struct was generated from the following file:

- [expair.h](#)

6.55 GiNaC::expair_rest_is_less Struct Reference

Function object not caring about the numerical coefficients for insertion into third argument of STL's sort().

```
#include <expair.h>
```

Public Member Functions

- bool [operator\(\)](#) (const [expair](#) &lh, const [expair](#) &rh) const

6.55.1 Detailed Description

Function object not caring about the numerical coefficients for insertion into third argument of STL's sort().

Note that this does not define a strict weak ordering since for any symbol x we have neither $3*x < 2*x$ or $2*x < 3*x$. Handle with care!

6.55.2 Member Function Documentation

6.55.2.1 operator()()

```
bool GiNaC::expair_rest_is_less::operator() (
    const expair & lh,
    const expair & rh ) const [inline]
```

References [GiNaC::ex::compare\(\)](#), and [GiNaC::expair::rest](#).

The documentation for this struct was generated from the following file:

- [expair.h](#)

6.56 GiNaC::expair_swap Struct Reference

```
#include <expair.h>
```

Public Member Functions

- void [operator\(\)](#) ([expair](#) &lh, [expair](#) &rh) const

6.56.1 Member Function Documentation

6.56.1.1 operator()()

```
void GiNaC::expair_swap::operator() (
    expair & lh,
    expair & rh ) const [inline]
```

References [GiNaC::expair::swap\(\)](#).

The documentation for this struct was generated from the following file:

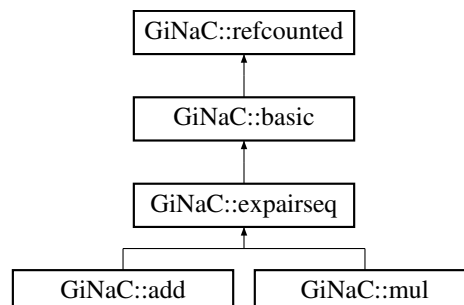
- [expair.h](#)

6.57 GiNaC::expairseq Class Reference

A sequence of class expair.

```
#include <expairseq.h>
```

Inheritance diagram for GiNaC::expairseq:



Public Member Functions

- `expairseq` (const `ex` &lh, const `ex` &rh)
- `expairseq` (const `exvector` &v)
- `expairseq` (const `epvector` &v, const `ex` &oc, bool do_index_renaming=false)
- `expairseq` (`epvector` &&vp, const `ex` &oc, bool do_index_renaming=false)
- unsigned `precedence` () const override
Return relative operator precedence (for parenthezing output).
- bool `info` (unsigned inf) const override
Information about the object.
- size_t `nops` () const override
Number of operands/members.
- `ex op` (size_t i) const override
Return operand/member at position i.
- `ex map` (`map_function` &f) const override
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- `ex eval` () const override
Perform coefficient-wise automatic term rewriting rules in this class.
- `ex to_rational` (`exmap` &repl) const override
Implementation of `ex::to_rational()` for expairseqs.
- `ex to_polynomial` (`exmap` &repl) const override
Implementation of `ex::to_polynomial()` for expairseqs.
- bool `match` (const `ex` &pattern, `exmap` &repl_lst) const override
Check whether the expression matches a given pattern.
- `ex subs` (const `exmap` &m, unsigned `options`=0) const override
Substitute a set of objects by arbitrary expressions.
- `ex conjugate` () const override
- void `archive` (`archive_node` &n) const override
Save (serialize) the object into archive node.
- void `read_archive` (const `archive_node` &n, `lst` &syms) override
Load (deserialize) the object from an archive node.

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic` ()
basic destructor, virtual because class `ex` will delete objects of derived classes via a `basic*`.
- `basic` (const `basic` &other)
- const `basic` & `operator=` (const `basic` &other)
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate` () const
Create a clone of this object on the heap.
- virtual `ex evalf` () const
Evaluate object numerically.
- virtual `ex evalm` () const
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ` () const
Evaluate integrals, if result is known.
- virtual `ex eval_indexed` (const `basic` &i) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void `print` (const `print_context` &c, unsigned level=0) const
Output to stream.

- virtual void [dbgprint](#) () const
Little wrapper around print to be called within a debugger.
- virtual void [dbgprinttree](#) () const
Little wrapper around printtree to be called within a debugger.
- virtual [ex operator\[\]](#) (const [ex](#) &index) const
- virtual [ex operator\[\]](#) (size_t i) const
- virtual [ex & let_op](#) (size_t i)
Return modifiable operand/member at position i.
- virtual [ex & operator\[\]](#) (const [ex](#) &index)
- virtual [ex & operator\[\]](#) (size_t i)
- virtual bool [has](#) (const [ex](#) &other, unsigned [options](#)=0) const
Test for occurrence of a pattern.
- virtual void [accept](#) (GiNaC::visitor &v) const
- virtual bool [is_polynomial](#) (const [ex](#) &var) const
Check whether this is a polynomial in the given variables.
- virtual int [degree](#) (const [ex](#) &s) const
Return degree of highest power in object s.
- virtual int [ldegree](#) (const [ex](#) &s) const
Return degree of lowest power in object s.
- virtual [ex coeff](#) (const [ex](#) &s, int n=1) const
Return coefficient of degree n in object s.
- virtual [ex collect](#) (const [ex](#) &s, bool distributed=false) const
Sort expanded expression in terms of powers of some object(s).
- virtual [ex series](#) (const [relational](#) &r, int [order](#), unsigned [options](#)=0) const
Default implementation of [ex::series\(\)](#).
- virtual [ex normal](#) (exmap &repl, exmap &rev_lookup, [lst](#) &modifier) const
Default implementation of [ex::normal\(\)](#).
- virtual [numeric integer_content](#) () const
- virtual [ex smod](#) (const [numeric](#) &xi) const
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual [numeric max_coefficient](#) () const
Implementation [ex::max_coefficient\(\)](#).
- virtual [exvector get_free_indices](#) () const
Return a vector containing the free indices of an expression.
- virtual [ex add_indexed](#) (const [ex](#) &self, const [ex](#) &other) const
Add two indexed expressions.
- virtual [ex scalar_mul_indexed](#) (const [ex](#) &self, const [numeric](#) &other) const
Multiply an indexed expression with a scalar.
- virtual bool [contract_with](#) (exvector::iterator self, exvector::iterator other, [exvector](#) &v) const
Try to contract two indexed expressions that appear in the same product.
- virtual [return_type_t return_type_tinfo](#) () const
- virtual [ex real_part](#) () const
- virtual [ex imag_part](#) () const
- template<class T >
void [print_dispatch](#) (const [print_context](#) &c, unsigned level) const
Like [print\(\)](#), but dispatch to the specified class.
- void [print_dispatch](#) (const [registered_class_info](#) &ri, const [print_context](#) &c, unsigned level) const
Like [print\(\)](#), but dispatch to the specified class.
- [ex subs_one_level](#) (const [exmap](#) &m, unsigned [options](#)) const
Helper function for [subs\(\)](#).
- [ex diff](#) (const [symbol](#) &s, unsigned nth=1) const

Default interface of nth derivative `ex::diff(s, n)`.

- int `compare` (const `basic` &other) const

Compare objects syntactically to establish canonical ordering.

- bool `is_equal` (const `basic` &other) const

Test for syntactic equality.

- const `basic` & `hold` () const

Stop further evaluation.

- unsigned `gethash` () const
- const `basic` & `setflag` (unsigned f) const

Set some `status_flags`.

- const `basic` & `clearflag` (unsigned f) const

Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Protected Member Functions

- bool `is_equal_same_type` (const `basic` &other) const override
Returns true if two objects of same type are equal.
- unsigned `return_type` () const override
- unsigned `calchash` () const override
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- `ex expand` (unsigned `options`=0) const override
Expand expression, i.e.
- virtual `ex thisexpairseq` (const `epvector` &v, const `ex` &oc, bool do_index_renaming=false) const
Create an object of this type.
- virtual `ex thisexpairseq` (`epvector` &&vp, const `ex` &oc, bool do_index_renaming=false) const
- virtual void `printseq` (const `print_context` &c, char delim, unsigned this_precedence, unsigned upper_← precedence) const
- virtual void `printpair` (const `print_context` &c, const `expair` &p, unsigned upper_precedence) const
- virtual `expair split_ex_to_pair` (const `ex` &e) const
Form an expair from an ex, using the corresponding semantics.
- virtual `expair combine_ex_with_coeff_to_pair` (const `ex` &e, const `ex` &c) const
- virtual `expair combine_pair_with_coeff_to_pair` (const `expair` &p, const `ex` &c) const
- virtual `ex recombine_pair_to_ex` (const `expair` &p) const
Form an ex out of an expair, using the corresponding semantics.
- virtual bool `expair_needs_further_processing` (`epp` it)
- virtual `ex default_overall_coeff` () const
- virtual void `combine_overall_coeff` (const `ex` &c)
- virtual void `combine_overall_coeff` (const `ex` &c1, const `ex` &c2)
- virtual bool `can_make_flat` (const `expair` &p) const
- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
- void `construct_from_2_ex` (const `ex` &lh, const `ex` &rh)
- void `construct_from_2_expairseq` (const `expairseq` &s1, const `expairseq` &s2)

- void `construct_from_expairseq_ex` (const `expairseq` &s, const `ex` &e)
- void `construct_from_exvector` (const `exvector` &v)
- void `construct_from_epvector` (const `epvector` &v, bool do_index_renaming=false)
- void `construct_from_epvector` (`epvector` &&v, bool do_index_renaming=false)
- void `make_flat` (const `exvector` &v)
Combine this expairseq with argument exvector.
- void `make_flat` (const `epvector` &v, bool do_index_renaming=false)
Combine this expairseq with argument epvector.
- void `canonicalize` ()
Brings this expairseq into a sorted (canonical) form.
- void `combine_same_terms_sorted_seq` ()
Compact a presorted expairseq by combining all matching expairs to one each.
- bool `is_canonical` () const
Check if this expairseq is in sorted (canonical) form.
- `epvector` `expandchildren` (unsigned `options`) const
Member-wise expand the expairs in this sequence.
- `epvector` `evalchildren` () const
Member-wise evaluate the expairs in this sequence.
- `epvector` `subchildren` (const `exmap` &m, unsigned `options`=0) const
Member-wise substitute in this sequence.

Protected Member Functions inherited from `GiNaC::basic`

- `basic` ()
- virtual `ex` `eval_ncmul` (const `exvector` &v) const
- virtual bool `match_same_type` (const `basic` &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual `ex` `derivative` (const `symbol` &s) const
Default implementation of `ex::diff()`.
- virtual int `compare_same_type` (const `basic` &other) const
Returns order relation between two objects of same type.
- void `ensure_if_modifiable` () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void `do_print` (const `print_context` &c, unsigned level) const
Default output to stream.
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
Tree output to stream.
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
Python parsable output to stream.

Protected Attributes

- `epvector` `seq`
- `ex` `overall_coeff`

Protected Attributes inherited from `GiNaC::basic`

- unsigned `flags`
of type `status_flags`
- unsigned `hashvalue`
hash value

6.57.1 Detailed Description

A sequence of class `expair`.

This is used for time-critical classes like sums and products of terms since handling a list of `coeff` and `rest` is much faster than handling a list of products or powers, respectively. (Not incidentally, Maple does it the same way, maybe others too.) The semantics is (at least) twofold: one for addition and one for multiplication and several methods have to be overridden by derived classes to reflect the change in semantics. However, most functionality turns out to be shared between addition and multiplication, which is the reason why there is this base class.

6.57.2 Constructor & Destructor Documentation

6.57.2.1 `expairseq()` [1/4]

```
GiNaC::expairseq::expairseq (
    const ex & lh,
    const ex & rh )
```

References [construct_from_2_ex\(\)](#), [GINAC_ASSERT](#), and [is_canonical\(\)](#).

6.57.2.2 `expairseq()` [2/4]

```
GiNaC::expairseq::expairseq (
    const exvector & v )
```

References [construct_from_exvector\(\)](#), [GINAC_ASSERT](#), and [is_canonical\(\)](#).

6.57.2.3 `expairseq()` [3/4]

```
GiNaC::expairseq::expairseq (
    const epvector & v,
    const ex & oc,
    bool do_index_renaming = false )
```

References [construct_from_epvector\(\)](#), [GINAC_ASSERT](#), and [is_canonical\(\)](#).

6.57.2.4 `expairseq()` [4/4]

```
GiNaC::expairseq::expairseq (
    epvector && vp,
    const ex & oc,
    bool do_index_renaming = false )
```

References [construct_from_epvector\(\)](#), [GINAC_ASSERT](#), and [is_canonical\(\)](#).

6.57.3 Member Function Documentation

6.57.3.1 precedence()

```
unsigned GiNaC::expairseq::precedence ( ) const [inline], [override], [virtual]
```

Return relative operator precedence (for parenthezing output).

Reimplemented from [GiNaC::basic](#).

Reimplemented in [GiNaC::mul](#).

Referenced by [do_print\(\)](#), and [printpair\(\)](#).

6.57.3.2 info()

```
bool GiNaC::expairseq::info (
    unsigned inf ) const [override], [virtual]
```

Information about the object.

See also

class [info_flags](#)

Reimplemented from [GiNaC::basic](#).

Reimplemented in [GiNaC::mul](#).

References [GiNaC::basic::clearflag\(\)](#), [GiNaC::status_flags::expanded](#), [GiNaC::info_flags::expanded](#), [GiNaC::basic::flags](#), [GiNaC::status_flags::has_indices](#), [GiNaC::info_flags::has_indices](#), [GiNaC::status_flags::has_no_indices](#), [seq](#), and [GiNaC::basic::setflag\(\)](#).

6.57.3.3 nops()

```
size_t GiNaC::expairseq::nops ( ) const [override], [virtual]
```

Number of operands/members.

Reimplemented from [GiNaC::basic](#).

References [default_overall_coeff\(\)](#), [GiNaC::ex::is_equal\(\)](#), [overall_coeff](#), and [seq](#).

Referenced by [GiNaC::algebraic_match_mul_with_mul\(\)](#), [GiNaC::mul::algebraic_subs_mul\(\)](#), [GiNaC::add::conjugate\(\)](#), [GiNaC::add::do_print_python_repr\(\)](#), [GiNaC::mul::do_print_python_repr\(\)](#), [do_print_tree\(\)](#), [GiNaC::power::expand_add\(\)](#), [GiNaC::power::expand_add_2\(\)](#), [GiNaC::add::get_free_indices\(\)](#), [GiNaC::mul::get_free_indices\(\)](#), [GiNaC::mul::has\(\)](#), and [match\(\)](#).

6.57.3.4 op()

```
ex GiNaC::expairseq::op (
    size_t i ) const [override], [virtual]
```

Return operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [default_overall_coeff\(\)](#), [GINAC_ASSERT](#), [GiNaC::ex::is_equal\(\)](#), [overall_coeff](#), [recombine_pair_to_ex\(\)](#), and [seq](#).

Referenced by [GiNaC::algebraic_match_mul_with_mul\(\)](#), [GiNaC::mul::algebraic_subs_mul\(\)](#), [GiNaC::add::conjugate\(\)](#), [GiNaC::add::do_print_python_repr\(\)](#), [GiNaC::mul::do_print_python_repr\(\)](#), [GiNaC::add::get_free_indices\(\)](#), [GiNaC::mul::get_free_indices\(\)](#), [match\(\)](#), [GiNaC::add::series\(\)](#), and [GiNaC::mul::series\(\)](#).

6.57.3.5 map()

```
ex GiNaC::expairseq::map (
    map_function & f ) const [override], [virtual]
```

Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).

Reimplemented from [GiNaC::basic](#).

References [default_overall_coeff\(\)](#), [GiNaC::ex::is_equal\(\)](#), [overall_coeff](#), [recombine_pair_to_ex\(\)](#), [seq](#), [split_ex_to_pair\(\)](#), and [thisexpairseq\(\)](#).

6.57.3.6 eval()

```
ex GiNaC::expairseq::eval ( ) const [override], [virtual]
```

Perform coefficient-wise automatic term rewriting rules in this class.

Reimplemented from [GiNaC::basic](#).

Reimplemented in [GiNaC::mul](#).

References [evalchildren\(\)](#), [GiNaC::status_flags::evaluated](#), [GiNaC::basic::flags](#), [GiNaC::basic::hold\(\)](#), and [overall_coeff](#).

6.57.3.7 to_rational()

```
ex GiNaC::expairseq::to_rational (
    exmap & repl ) const [override], [virtual]
```

Implementation of [ex::to_rational\(\)](#) for expairseqs.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex1](#), [default_overall_coeff\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::numeric](#), [overall_coeff](#), [recombine_pair_to_ex\(\)](#), [seq](#), [split_ex_to_pair\(\)](#), [thisexpairseq\(\)](#), [GiNaC::ex::to_rational\(\)](#), and [to_rational\(\)](#).

Referenced by [to_rational\(\)](#).

6.57.3.8 to_polynomial()

```
ex GiNaC::expairseq::to_polynomial (
    exmap & repl ) const [override], [virtual]
```

Implementation of [ex::to_polynomial\(\)](#) for expairseqs.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex1](#), [default_overall_coeff\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::numeric](#), [overall_coeff](#), [recombine_pair_to_ex\(\)](#), [seq](#), [split_ex_to_pair\(\)](#), [thisexpairseq\(\)](#), [GiNaC::ex::to_polynomial\(\)](#), and [to_polynomial\(\)](#).

Referenced by [to_polynomial\(\)](#).

6.57.3.9 match()

```
bool GiNaC::expairseq::match (
    const ex & pattern,
    exmap & repl_lst ) const [override], [virtual]
```

Check whether the expression matches a given pattern.

For every wildcard object in the pattern, a pair with the wildcard as a key and matching expression as a value is added to `repl_lst`.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::begin\(\)](#), [default_overall_coeff\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::nops\(\)](#), [nops\(\)](#), [GiNaC::ex::op\(\)](#), [op\(\)](#), [split_ex_to_pair\(\)](#), and [thisexpairseq\(\)](#).

6.57.3.10 subs()

```
ex GiNaC::expairseq::subs (
    const exmap & m,
    unsigned options = 0 ) const [override], [virtual]
```

Substitute a set of objects by arbitrary expressions.

The `ex` returned will already be evaluated.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::subs_options::algebraic](#), [m](#), [GiNaC::subs_options::no_index_renaming](#), [options](#), [overall_coeff](#), [GiNaC::basic::subs_one_level\(\)](#), [subschildren\(\)](#), and [thisexpairseq\(\)](#).

6.57.3.11 conjugate()

```
ex GiNaC::expairseq::conjugate ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

Reimplemented in [GiNaC::mul](#).

References [GiNaC::are_ex_trivially_equal\(\)](#), [GiNaC::ex::conjugate\(\)](#), [GiNaC::conjugateepvector\(\)](#), [overall_coeff](#), [seq](#), [thisexpairseq\(\)](#), and `x`.

6.57.3.12 archive()

```
void GiNaC::expairseq::archive (
    archive_node & n ) const [override], [virtual]
```

Save (serialize) the object into archive node.

Archive the object.

Losely speaking, this method turns an expression into a byte stream (which can be saved and restored later on, or sent via network, etc.)

Reimplemented from [GiNaC::basic](#).

References [n](#), [overall_coeff](#), and [seq](#).

6.57.3.13 read_archive()

```
void GiNaC::expairseq::read_archive (
    const archive_node & n,
    lst & syms ) [override], [virtual]
```

Load (deserialize) the object from an archive node.

Construct object from [archive_node](#).

Note

This method is essentially a constructor. However, constructors can't be virtual. So, if unarchiving routines are implemented as constructors one would need to define such a constructor in every class, even if all it does is simply calling constructor of a superclass.

Reimplemented from [GiNaC::basic](#).

References [canonicalize\(\)](#), [GiNaC::basic::coeff\(\)](#), [GINAC_ASSERT](#), [is_canonical\(\)](#), [n](#), [overall_coeff](#), and [seq](#).

6.57.3.14 is_equal_same_type()

```
bool GiNaC::expairseq::is_equal_same_type (
    const basic & other ) const [override], [protected], [virtual]
```

Returns true if two objects of same type are equal.

Normally needs not be reimplemented as long as it wasn't overwritten by some parent class, since it just calls [compare_same_type\(\)](#). The reason why this function exists is that sometimes it is easier to determine equality than an order relation and then it can be overridden.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::is_equal\(\)](#), [overall_coeff](#), and [seq](#).

6.57.3.15 return_type()

```
unsigned GiNaC::expairseq::return_type ( ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

Reimplemented in [GiNaC::mul](#).

References [GiNaC::return_types::noncommutative_composite](#).

6.57.3.16 calchash()

```
unsigned GiNaC::expairseq::calchash ( ) const [override], [protected], [virtual]
```

Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.

The method inherited from class basic computes a hash value based on the type and hash values of possible members. For this reason it is well suited for container classes but atomic classes should override this implementation because otherwise they would all end up with the same hashvalue.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::status_flags::evaluated](#), [GiNaC::basic::flags](#), [GiNaC::ex::gethash\(\)](#), [GiNaC::status_flags::hash_calculated](#), [GiNaC::basic::hashvalue](#), [GiNaC::make_hash_seed\(\)](#), [overall_coeff](#), [GiNaC::rotate_left\(\)](#), [seq](#), and [GiNaC::basic::setflag\(\)](#).

6.57.3.17 expand()

```
ex GiNaC::expairseq::expand (
    unsigned options = 0 ) const [override], [protected], [virtual]
```

Expand expression, i.e.

multiply it out and return the result as a new expression.

Reimplemented from [GiNaC::basic](#).

Reimplemented in [GiNaC::mul](#).

References [expandchildren\(\)](#), [GiNaC::status_flags::expanded](#), [options](#), [overall_coeff](#), [GiNaC::basic::setflag\(\)](#), and [thisexpairseq\(\)](#).

6.57.3.18 thisexpairseq() [1/2]

```
ex GiNaC::expairseq::thisexpairseq (
    const epvector & v,
    const ex & oc,
    bool do_index_renaming = false ) const [protected], [virtual]
```

Create an object of this type.

This method works similar to a constructor. It is useful because expairseq has (at least) two possible different semantics but we want to inherit methods thus avoiding code duplication. Sometimes a method in expairseq has to create a new one of the same semantics, which cannot be done by a ctor because the name (add, mul,...) is unknown on the expairseq level. In order for this trick to work a derived class must of course override this definition.

Reimplemented in [GiNaC::add](#), and [GiNaC::mul](#).

Referenced by [conjugate\(\)](#), [expand\(\)](#), [map\(\)](#), [match\(\)](#), [subs\(\)](#), [to_polynomial\(\)](#), and [to_rational\(\)](#).

6.57.3.19 thisexpairseq() [2/2]

```
ex GiNaC::expairseq::thisexpairseq (
    epvector && vp,
    const ex & oc,
    bool do_index_renaming = false ) const [protected], [virtual]
```

Reimplemented in [GiNaC::add](#), and [GiNaC::mul](#).

6.57.3.20 printseq()

```
void GiNaC::expairseq::printseq (
    const print_context & c,
    char delim,
    unsigned this_precedence,
    unsigned upper_precedence ) const [protected], [virtual]
```

References [c](#), [default_overall_coeff\(\)](#), [GiNaC::ex::is_equal\(\)](#), [overall_coeff](#), [GiNaC::ex::print\(\)](#), [printpair\(\)](#), and [seq](#).

Referenced by [do_print\(\)](#).

6.57.3.21 printpair()

```
void GiNaC::expairseq::printpair (
    const print_context & c,
    const expair & p,
    unsigned upper_precedence ) const [protected], [virtual]
```

References [c](#), [GiNaC::expair::coeff](#), [precedence\(\)](#), [GiNaC::ex::print\(\)](#), and [GiNaC::expair::rest](#).

Referenced by [is_canonical\(\)](#), and [printseq\(\)](#).

6.57.3.22 split_ex_to_pair()

```
expair GiNaC::expairseq::split_ex_to_pair (
    const ex & e ) const [protected], [virtual]
```

Form an expair from an ex, using the corresponding semantics.

See also

[expairseq::recombine_pair_to_ex\(\)](#)

Reimplemented in [GiNaC::add](#), and [GiNaC::mul](#).

References [GiNaC::_ex1](#).

Referenced by [construct_from_2_ex\(\)](#), [construct_from_expairseq_ex\(\)](#), [make_flat\(\)](#), [map\(\)](#), [match\(\)](#), [subschildren\(\)](#), [to_polynomial\(\)](#), and [to_rational\(\)](#).

6.57.3.23 combine_ex_with_coeff_to_pair()

```
expair GiNaC::expairseq::combine_ex_with_coeff_to_pair (
    const ex & e,
    const ex & c ) const [protected], [virtual]
```

Reimplemented in [GiNaC::add](#), and [GiNaC::mul](#).

References [c](#), and [GINAC_ASSERT](#).

Referenced by [evalchildren\(\)](#), and [subschildren\(\)](#).

6.57.3.24 combine_pair_with_coeff_to_pair()

```
expair GiNaC::expairseq::combine_pair_with_coeff_to_pair (
    const expair & p,
    const ex & c ) const [protected], [virtual]
```

Reimplemented in [GiNaC::add](#), and [GiNaC::mul](#).

References [c](#), [GiNaC::expair::coeff](#), [GINAC_ASSERT](#), and [GiNaC::expair::rest](#).

6.57.3.25 recombine_pair_to_ex()

```
ex GiNaC::expairseq::recombine_pair_to_ex (
    const expair & p ) const [protected], [virtual]
```

Form an ex out of an expair, using the corresponding semantics.

See also

[expairseq::split_ex_to_pair\(\)](#)

Reimplemented in [GiNaC::add](#), and [GiNaC::mul](#).

References [GiNaC::expair::coeff](#), and [GiNaC::expair::rest](#).

Referenced by [map\(\)](#), [op\(\)](#), [subschildren\(\)](#), [to_polynomial\(\)](#), and [to_rational\(\)](#).

6.57.3.26 expair_needs_further_processing()

```
bool GiNaC::expairseq::expair_needs_further_processing (
    exp it ) [protected], [virtual]
```

Reimplemented in [GiNaC::mul](#).

References [GiNaC::_ex1](#).

Referenced by [combine_same_terms_sorted_seq\(\)](#), [construct_from_2_expairseq\(\)](#), and [construct_from_expairseq_ex\(\)](#).

6.57.3.27 default_overall_coeff()

```
ex GiNaC::expairseq::default_overall_coeff ( ) const [protected], [virtual]
```

Reimplemented in [GiNaC::mul](#).

References [GiNaC::_ex0](#).

Referenced by [do_print_tree\(\)](#), [map\(\)](#), [match\(\)](#), [nops\(\)](#), [op\(\)](#), [printseq\(\)](#), [to_polynomial\(\)](#), and [to_rational\(\)](#).

6.57.3.28 combine_overall_coeff() [1/2]

```
void GiNaC::expairseq::combine_overall_coeff (
    const ex & c ) [protected], [virtual]
```

Reimplemented in [GiNaC::mul](#).

References [c](#), [GINAC_ASSERT](#), and [overall_coeff](#).

Referenced by [construct_from_2_ex\(\)](#), [construct_from_2_expairseq\(\)](#), [construct_from_expairseq_ex\(\)](#), [make_flat\(\)](#), and [make_flat\(\)](#).

6.57.3.29 combine_overall_coeff() [2/2]

```
void GiNaC::expairseq::combine_overall_coeff (
    const ex & c1,
    const ex & c2 ) [protected], [virtual]
```

Reimplemented in [GiNaC::mul](#).

References [GINAC_ASSERT](#), and [overall_coeff](#).

6.57.3.30 can_make_flat()

```
bool GiNaC::expairseq::can_make_flat (
    const expair & p ) const [protected], [virtual]
```

Reimplemented in [GiNaC::mul](#).

Referenced by [make_flat\(\)](#).

6.57.3.31 do_print()

```
void GiNaC::expairseq::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

References [c](#), [precedence\(\)](#), and [printseq\(\)](#).

6.57.3.32 do_print_tree()

```
void GiNaC::expairseq::do_print_tree (
    const print\_tree & c,
    unsigned level ) const [protected]
```

References [c](#), [default_overall_coeff\(\)](#), [GiNaC::basic::flags](#), [GiNaC::basic::hashvalue](#), [GiNaC::ex::is_equal\(\)](#), [nops\(\)](#), [overall_coeff](#), [GiNaC::ex::print\(\)](#), and [seq](#).

6.57.3.33 construct_from_2_ex()

```
void GiNaC::expairseq::construct_from_2_ex (
    const ex & lh,
    const ex & rh ) [protected]
```

References [GiNaC::expair::coeff](#), [combine_overall_coeff\(\)](#), [GiNaC::ex::compare\(\)](#), [construct_from_2_expairseq\(\)](#), [construct_from_expairseq_ex\(\)](#), [GiNaC::info_flags::has_indices](#), [GiNaC::ex::info\(\)](#), [GiNaC::rename_dummy_indices_uniquely\(\)](#), [GiNaC::expair::rest](#), [seq](#), and [split_ex_to_pair\(\)](#).

Referenced by [GiNaC::add::add\(\)](#), [expairseq\(\)](#), and [GiNaC::mul::mul\(\)](#).

6.57.3.34 construct_from_2_expairseq()

```
void GiNaC::expairseq::construct_from_2_expairseq (
    const expairseq & s1,
    const expairseq & s2 ) [protected]
```

References [combine_overall_coeff\(\)](#), [construct_from_epvector\(\)](#), [expair_needs_further_processing\(\)](#), [GiNaC::numeric::is_zero\(\)](#), [overall_coeff](#), and [seq](#).

Referenced by [construct_from_2_ex\(\)](#).

6.57.3.35 construct_from_expairseq_ex()

```
void GiNaC::expairseq::construct_from_expairseq_ex (
    const expairseq & s,
    const ex & e ) [protected]
```

References [GiNaC::expair::coeff](#), [combine_overall_coeff\(\)](#), [construct_from_epvector\(\)](#), [expair_needs_further_processing\(\)](#), [GiNaC::numeric::is_zero\(\)](#), [last](#), [overall_coeff](#), [GiNaC::expair::rest](#), [seq](#), and [split_ex_to_pair\(\)](#).

Referenced by [construct_from_2_ex\(\)](#).

6.57.3.36 construct_from_exvector()

```
void GiNaC::expairseq::construct_from_exvector (
    const exvector & v ) [protected]
```

References [canonicalize\(\)](#), [combine_same_terms_sorted_seq\(\)](#), and [make_flat\(\)](#).

Referenced by [GiNaC::add::add\(\)](#), [expairseq\(\)](#), [GiNaC::mul::mul\(\)](#), and [GiNaC::mul::mul\(\)](#).

6.57.3.37 construct_from_epvector() [1/2]

```
void GiNaC::expairseq::construct_from_epvector (
    const epvector & v,
    bool do_index_renaming = false ) [protected]
```

References [canonicalize\(\)](#), [combine_same_terms_sorted_seq\(\)](#), and [make_flat\(\)](#).

Referenced by [GiNaC::add::add\(\)](#), [GiNaC::add::add\(\)](#), [GiNaC::add::add\(\)](#), [GiNaC::add::add\(\)](#), [combine_same_terms_sorted_seq\(\)](#), [construct_from_2_expairseq\(\)](#), [construct_from_expairseq_ex\(\)](#), [expairseq\(\)](#), [expairseq\(\)](#), [GiNaC::mul::mul\(\)](#), [GiNaC::mul::mul\(\)](#), [GiNaC::mul::mul\(\)](#), and [GiNaC::mul::mul\(\)](#).

6.57.3.38 construct_from_epvector() [2/2]

```
void GiNaC::expairseq::construct_from_epvector (
    epvector && v,
    bool do_index_renaming = false ) [protected]
```

References [canonicalize\(\)](#), [combine_same_terms_sorted_seq\(\)](#), and [make_flat\(\)](#).

6.57.3.39 make_flat() [1/2]

```
void GiNaC::expairseq::make_flat (
    const exvector & v ) [protected]
```

Combine this expairseq with argument exvector.

It cares for associativity as well as for special handling of numerics.

References [GiNaC::_ex1](#), [combine_overall_coeff\(\)](#), [GiNaC::make_flat_inserter::handle_factor\(\)](#), [GiNaC::info_flags::has_indices](#), [overall_coeff](#), [seq](#), and [split_ex_to_pair\(\)](#).

Referenced by [construct_from_epvector\(\)](#), [construct_from_epvector\(\)](#), and [construct_from_exvector\(\)](#).

6.57.3.40 make_flat() [2/2]

```
void GiNaC::expairseq::make_flat (
    const epvector & v,
    bool do_index_renaming = false ) [protected]
```

Combine this expairseq with argument epvector.

It cares for associativity as well as for special handling of numerics.

References [GiNaC::_ex1](#), [GiNaC::are_ex_trivially_equal\(\)](#), [can_make_flat\(\)](#), [GiNaC::ex::coeff\(\)](#), [combine_overall_coeff\(\)](#), [GiNaC::make_flat_inserter::handle_factor\(\)](#), [GiNaC::info_flags::has_indices](#), [overall_coeff](#), and [seq](#).

6.57.3.41 canonicalize()

```
void GiNaC::expairseq::canonicalize ( ) [protected]
```

Brings this expairseq into a sorted (canonical) form.

References [seq](#).

Referenced by [construct_from_epvector\(\)](#), [construct_from_epvector\(\)](#), [construct_from_exvector\(\)](#), and [read_archive\(\)](#).

6.57.3.42 combine_same_terms_sorted_seq()

```
void GiNaC::expairseq::combine_same_terms_sorted_seq ( ) [protected]
```

Compact a presorted expairseq by combining all matching expairs to one each.

On an add object, this is responsible for $2*x+3*x+y \rightarrow 5*x+y$, for instance.

References [construct_from_epvector\(\)](#), [expair_needs_further_processing\(\)](#), [last](#), and [seq](#).

Referenced by [construct_from_epvector\(\)](#), [construct_from_epvector\(\)](#), and [construct_from_exvector\(\)](#).

6.57.3.43 is_canonical()

```
bool GiNaC::expairseq::is_canonical ( ) const [protected]
```

Check if this expairseq is in sorted (canonical) form.

Useful mainly for debugging or in assertions since being sorted is an invariance.

References [printpair\(\)](#), and [seq](#).

Referenced by [GiNaC::add::add\(\)](#), [GiNaC::add::add\(\)](#), [GiNaC::add::add\(\)](#), [GiNaC::add::add\(\)](#), [GiNaC::add::add\(\)](#), [GiNaC::add::add\(\)](#), [expairseq\(\)](#), [expairseq\(\)](#), [expairseq\(\)](#), [expairseq\(\)](#), [GiNaC::mul::mul\(\)](#), [GiNaC::mul::mul\(\)](#), [GiNaC::mul::mul\(\)](#), [GiNaC::mul::mul\(\)](#), [GiNaC::mul::mul\(\)](#), [GiNaC::mul::mul\(\)](#), [GiNaC::mul::mul\(\)](#), and [read_archive\(\)](#).

6.57.3.44 expandchildren()

```
epvector GiNaC::expairseq::expandchildren (
    unsigned options ) const [protected]
```

Member-wise expand the expairs in this sequence.

See also

[expairseq::expand\(\)](#)

Returns

epvector containing expanded pairs, empty if no members had to be changed.

References [GiNaC::are_ex_trivially_equal\(\)](#), [GiNaC::ex::coeff\(\)](#), [GiNaC::ex::expand\(\)](#), [last](#), [options](#), and [seq](#).

Referenced by [GiNaC::add::expand\(\)](#), and [expand\(\)](#).

6.57.3.45 evalchildren()

```
epvector GiNaC::expairseq::evalchildren ( ) const [protected]
```

Member-wise evaluate the expairs in this sequence.

See also

[expairseq::eval\(\)](#)

Returns

epvector containing evaluated pairs, empty if no members had to be changed.

References [combine_ex_with_coeff_to_pair\(\)](#), [GiNaC::expair::is_equal\(\)](#), [last](#), [seq](#), and [unlikely](#).

Referenced by [GiNaC::add::eval\(\)](#), [eval\(\)](#), and [GiNaC::mul::eval\(\)](#).

6.57.3.46 subschildren()

```
epvector GiNaC::expairseq::subschildren (
    const exmap & m,
    unsigned options = 0 ) const [protected]
```

Member-wise substitute in this sequence.

See also

[expairseq::subs\(\)](#)

Returns

epvector containing expanded pairs, empty if no members had to be changed.

References [GiNaC::are_ex_trivially_equal\(\)](#), [GiNaC::ex::begin\(\)](#), [GiNaC::ex::coeff\(\)](#), [combine_ex_with_coeff_to_pair\(\)](#), [GiNaC::expair::is_equal\(\)](#), [last](#), [m](#), [options](#), [GiNaC::subs_options::pattern_is_not_product](#), [GiNaC::subs_options::pattern_is_product](#), [recombine_pair_to_ex\(\)](#), [seq](#), [split_ex_to_pair\(\)](#), and [GiNaC::ex::subs\(\)](#).

Referenced by [subs\(\)](#).

6.57.4 Member Data Documentation

6.57.4.1 seq

```
epvector GiNaC::expairseq::seq [protected]
```

Referenced by [archive\(\)](#), [calchash\(\)](#), [canonicalize\(\)](#), [GiNaC::add::coeff\(\)](#), [GiNaC::mul::coeff\(\)](#), [combine_same_terms_sorted_seq\(\)](#), [conjugate\(\)](#), [GiNaC::mul::conjugate\(\)](#), [construct_from_2_ex\(\)](#), [construct_from_2_expairseq\(\)](#), [construct_from_expairseq_ex\(\)](#), [GiNaC::add::degree\(\)](#), [GiNaC::mul::degree\(\)](#), [GiNaC::add::derivative\(\)](#), [GiNaC::mul::derivative\(\)](#), [GiNaC::mul::do_print\(\)](#), [GiNaC::add::do_print_csrc\(\)](#), [GiNaC::mul::do_print_csrc\(\)](#), [GiNaC::mul::do_print_latex\(\)](#), [do_print_tree\(\)](#), [GiNaC::add::eval\(\)](#), [GiNaC::mul::eval\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::add::eval_ncmul\(\)](#), [GiNaC::mul::eval_ncmul\(\)](#), [evalchildren\(\)](#), [GiNaC::mul::evalf\(\)](#), [GiNaC::add::evalm\(\)](#), [GiNaC::mul::evalm\(\)](#), [GiNaC::mul::expand\(\)](#), [GiNaC::power::expand\(\)](#), [GiNaC::power::expand_add\(\)](#), [GiNaC::power::expand_add_2\(\)](#), [expandchildren\(\)](#), [GiNaC::mul::expandchildren\(\)](#), [GiNaC::mul::find_real_imag\(\)](#), [GiNaC::add::imag_part\(\)](#), [GiNaC::add::info\(\)](#), [info\(\)](#), [GiNaC::mul::info\(\)](#), [GiNaC::add::integer_content\(\)](#), [GiNaC::mul::integer_content\(\)](#), [is_canonical\(\)](#), [is_equal_same_type\(\)](#), [GiNaC::add::is_polynomial\(\)](#), [GiNaC::mul::is_polynomial\(\)](#), [GiNaC::add::ldegree\(\)](#), [GiNaC::mul::ldegree\(\)](#), [make_flat\(\)](#), [make_flat\(\)](#), [map\(\)](#), [GiNaC::add::max_coefficient\(\)](#), [GiNaC::mul::max_coefficient\(\)](#), [nops\(\)](#), [GiNaC::add::normal\(\)](#), [GiNaC::mul::normal\(\)](#), [op\(\)](#), [GiNaC::add::print_add\(\)](#), [printseq\(\)](#), [read_archive\(\)](#), [GiNaC::add::real_part\(\)](#), [GiNaC::add::return_type\(\)](#), [GiNaC::mul::return_type\(\)](#), [GiNaC::add::return_type_tinfo\(\)](#), [GiNaC::mul::return_type_tinfo\(\)](#), [GiNaC::add::series\(\)](#), [GiNaC::mul::series\(\)](#), [GiNaC::add::smod\(\)](#), [GiNaC::mul::smod\(\)](#), [subschildren\(\)](#), [to_polynomial\(\)](#), and [to_rational\(\)](#).

6.57.4.2 overall_coeff

ex GiNaC::expairseq::overall_coeff [protected]

Referenced by [GiNaC::add::add\(\)](#), [GiNaC::add::add\(\)](#), [GiNaC::add::add\(\)](#), [GiNaC::add::add\(\)](#), [GiNaC::add::add\(\)](#), [GiNaC::add::add\(\)](#), [archive\(\)](#), [calchash\(\)](#), [GiNaC::add::coeff\(\)](#), [GiNaC::mul::coeff\(\)](#), [GiNaC::add::combine_ex_with_coeff_to_pair\(\)](#), [combine_overall_coeff\(\)](#), [GiNaC::mul::combine_overall_coeff\(\)](#), [combine_overall_coeff\(\)](#), [GiNaC::mul::combine_overall_coeff\(\)](#), [conjugate\(\)](#), [GiNaC::mul::conjugate\(\)](#), [construct_from_2_expairseq\(\)](#), [construct_from_expairseq_ex\(\)](#), [GiNaC::add::degree\(\)](#), [GiNaC::mul::derivative\(\)](#), [GiNaC::add::do_print_csrc\(\)](#), [GiNaC::mul::do_print_csrc\(\)](#), [do_print_tree\(\)](#), [GiNaC::add::eval\(\)](#), [eval\(\)](#), [GiNaC::mul::eval\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::mul::evalf\(\)](#), [GiNaC::add::evalm\(\)](#), [GiNaC::mul::evalm\(\)](#), [GiNaC::add::expand\(\)](#), [expand\(\)](#), [GiNaC::mul::expand\(\)](#), [GiNaC::power::expand\(\)](#), [GiNaC::power::expand_add\(\)](#), [GiNaC::power::expand_add_2\(\)](#), [GiNaC::mul::find_real_imag\(\)](#), [GiNaC::add::imag_part\(\)](#), [GiNaC::add::info\(\)](#), [GiNaC::mul::info\(\)](#), [GiNaC::add::integer_content\(\)](#), [GiNaC::mul::integer_content\(\)](#), [is_equal_same_type\(\)](#), [GiNaC::add::lddegree\(\)](#), [make_flat\(\)](#), [make_flat\(\)](#), [map\(\)](#), [GiNaC::add::max_coefficient\(\)](#), [GiNaC::mul::max_coefficient\(\)](#), [GiNaC::mul::mul\(\)](#), [GiNaC::mul::mul\(\)](#), [GiNaC::mul::mul\(\)](#), [GiNaC::mul::mul\(\)](#), [GiNaC::mul::mul\(\)](#), [GiNaC::mul::mul\(\)](#), [GiNaC::mul::mul\(\)](#), [nops\(\)](#), [GiNaC::add::normal\(\)](#), [GiNaC::mul::normal\(\)](#), [op\(\)](#), [GiNaC::add::print_add\(\)](#), [GiNaC::mul::print_overall_coeff\(\)](#), [printseq\(\)](#), [read_archive\(\)](#), [GiNaC::add::real_part\(\)](#), [GiNaC::add::series\(\)](#), [GiNaC::mul::series\(\)](#), [GiNaC::add::smod\(\)](#), [GiNaC::mul::smod\(\)](#), [GiNaC::add::split_ex_to_pair\(\)](#), [subs\(\)](#), [to_polynomial\(\)](#), and [to_rational\(\)](#).

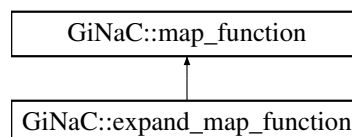
The documentation for this class was generated from the following files:

- [expairseq.h](#)
- [expairseq.cpp](#)
- [normal.cpp](#)

6.58 GiNaC::expand_map_function Struct Reference

Function object to be applied by [basic::expand\(\)](#).

Inheritance diagram for GiNaC::expand_map_function:



Public Member Functions

- [expand_map_function](#) (unsigned o)
- [ex operator\(\)](#) (const [ex](#) &e) override

Public Member Functions inherited from [GiNaC::map_function](#)

- virtual [~map_function](#) ()

Public Attributes

- unsigned [options](#)

Additional Inherited Members

Public Types inherited from [GiNaC::map_function](#)

- typedef const [ex](#) & [argument_type](#)
- typedef [ex](#) [result_type](#)

6.58.1 Detailed Description

Function object to be applied by [basic::expand\(\)](#).

6.58.2 Constructor & Destructor Documentation

6.58.2.1 [expand_map_function\(\)](#)

```
GiNaC::expand_map_function::expand_map_function (
    unsigned o ) [inline]
```

6.58.3 Member Function Documentation

6.58.3.1 [operator\(\)](#)

```
ex GiNaC::expand_map_function::operator() (
    const ex & e ) [inline], [override], [virtual]
```

Implements [GiNaC::map_function](#).

References [GiNaC::ex::expand\(\)](#), and [options](#).

6.58.4 Member Data Documentation

6.58.4.1 [options](#)

```
unsigned GiNaC::expand_map_function::options
```

The documentation for this struct was generated from the following file:

- [basic.cpp](#)

6.59 GiNaC::expand_options Class Reference

Flags to control the behavior of [expand\(\)](#).

```
#include <flags.h>
```

Public Types

- enum { [expand_indexed](#) = 0x0001 , [expand_function_args](#) = 0x0002 , [expand_rename_idx](#) = 0x0004 , [expand_transcendental](#) = 0x0008 }

6.59.1 Detailed Description

Flags to control the behavior of [expand\(\)](#).

6.59.2 Member Enumeration Documentation

6.59.2.1 anonymous enum

anonymous enum

Enumerator

expand_indexed	expands (a+b).i to a.i+b.i
expand_function_args	expands the arguments of functions
expand_rename_idx	used internally by mul::expand()
expand_transcendental	expands transcendental functions like log and exp

The documentation for this class was generated from the following file:

- [flags.h](#)

6.60 GiNaC::factor_options Class Reference

Flags to control the polynomial factorization.

```
#include <flags.h>
```

Public Types

- enum { [polynomial](#) = 0x0000 , [all](#) = 0x0001 }

6.60.1 Detailed Description

Flags to control the polynomial factorization.

6.60.2 Member Enumeration Documentation

6.60.2.1 anonymous enum

anonymous enum

Enumerator

polynomial	factor only expressions that are polynomials
all	factor all polynomial subexpressions

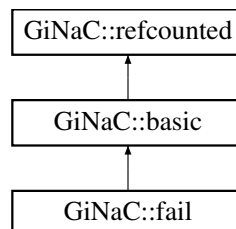
The documentation for this class was generated from the following file:

- [flags.h](#)

6.61 GiNaC::fail Class Reference

```
#include <fail.h>
```

Inheritance diagram for GiNaC::fail:



Protected Member Functions

- unsigned [return_type](#) () const override
- void [do_print](#) (const [print_context](#) &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual [ex derivative](#) (const [symbol](#) &s) const
Default implementation of [ex::diff\(\)](#).
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Additional Inherited Members

Public Member Functions inherited from [GiNaC::basic](#)

- virtual [~basic](#) ()
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- [basic](#) (const [basic](#) &other)
- const [basic](#) & [operator=](#) (const [basic](#) &other)
basic assignment operator: the other object might be of a derived class.
- virtual [basic](#) * [duplicate](#) () const
Create a clone of this object on the heap.
- virtual [ex eval](#) () const
Perform automatic non-interruptive term rewriting rules.
- virtual [ex evalf](#) () const
Evaluate object numerically.
- virtual [ex evalm](#) () const
Evaluate sums, products and integer powers of matrices.
- virtual [ex eval_integ](#) () const
Evaluate integrals, if result is known.
- virtual [ex eval_indexed](#) (const [basic](#) &i) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void [print](#) (const [print_context](#) &c, unsigned level=0) const
Output to stream.
- virtual void [dbgprint](#) () const
Little wrapper around print to be called within a debugger.
- virtual void [dbgprinttree](#) () const
Little wrapper around printtree to be called within a debugger.
- virtual unsigned [precedence](#) () const
Return relative operator precedence (for parenthezing output).
- virtual bool [info](#) (unsigned inf) const
Information about the object.
- virtual size_t [nops](#) () const
Number of operands/members.
- virtual [ex op](#) (size_t i) const
Return operand/member at position i.
- virtual [ex operator\[\]](#) (const [ex](#) &index) const
- virtual [ex operator\[\]](#) (size_t i) const
- virtual [ex](#) & [let_op](#) (size_t i)
Return modifiable operand/member at position i.
- virtual [ex](#) & [operator\[\]](#) (const [ex](#) &index)
- virtual [ex](#) & [operator\[\]](#) (size_t i)
- virtual bool [has](#) (const [ex](#) &other, unsigned [options](#)=0) const
Test for occurrence of a pattern.
- virtual bool [match](#) (const [ex](#) &pattern, [exmap](#) &repls) const
Check whether the expression matches a given pattern.
- virtual [ex subs](#) (const [exmap](#) &m, unsigned [options](#)=0) const
Substitute a set of objects by arbitrary expressions.
- virtual [ex map](#) ([map_function](#) &f) const
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void [accept](#) ([GiNaC::visitor](#) &v) const
- virtual bool [is_polynomial](#) (const [ex](#) &var) const

- Check whether this is a polynomial in the given variables.*

 - virtual int [degree](#) (const [ex](#) &s) const

Return degree of highest power in object s.
- virtual int [ldegree](#) (const [ex](#) &s) const

Return degree of lowest power in object s.
- virtual [ex](#) [coeff](#) (const [ex](#) &s, int [n](#)=1) const

Return coefficient of degree n in object s.
- virtual [ex](#) [expand](#) (unsigned [options](#)=0) const

Expand expression, i.e.
- virtual [ex](#) [collect](#) (const [ex](#) &s, bool [distributed](#)=false) const

Sort expanded expression in terms of powers of some object(s).
- virtual [ex](#) [series](#) (const [relational](#) &r, int [order](#), unsigned [options](#)=0) const

Default implementation of [ex::series\(\)](#).
- virtual [ex](#) [normal](#) ([exmap](#) &repl, [exmap](#) &rev_lookup, [lst](#) &modifier) const

Default implementation of [ex::normal\(\)](#).
- virtual [ex](#) [to_rational](#) ([exmap](#) &repl) const

Default implementation of [ex::to_rational\(\)](#).
- virtual [ex](#) [to_polynomial](#) ([exmap](#) &repl) const
- virtual [numeric](#) [integer_content](#) () const
- virtual [ex](#) [smod](#) (const [numeric](#) &xi) const

Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual [numeric](#) [max_coefficient](#) () const

Implementation [ex::max_coefficient\(\)](#).
- virtual [exvector](#) [get_free_indices](#) () const

Return a vector containing the free indices of an expression.
- virtual [ex](#) [add_indexed](#) (const [ex](#) &self, const [ex](#) &other) const

Add two indexed expressions.
- virtual [ex](#) [scalar_mul_indexed](#) (const [ex](#) &self, const [numeric](#) &other) const

Multiply an indexed expression with a scalar.
- virtual bool [contract_with](#) ([exvector::iterator](#) self, [exvector::iterator](#) other, [exvector](#) &v) const

Try to contract two indexed expressions that appear in the same product.
- virtual [return_type_t](#) [return_type_tinfo](#) () const
- virtual [ex](#) [conjugate](#) () const
- virtual [ex](#) [real_part](#) () const
- virtual [ex](#) [imag_part](#) () const
- template<class T >
- void [print_dispatch](#) (const [print_context](#) &c, unsigned level) const

Like [print\(\)](#), but dispatch to the specified class.
- void [print_dispatch](#) (const [registered_class_info](#) &ri, const [print_context](#) &c, unsigned level) const

Like [print\(\)](#), but dispatch to the specified class.
- virtual void [archive](#) ([archive_node](#) &n) const

Save (serialize) the object into archive node.
- virtual void [read_archive](#) (const [archive_node](#) &n, [lst](#) &syms)

Load (deserialize) the object from an archive node.
- [ex](#) [subs_one_level](#) (const [exmap](#) &m, unsigned [options](#)) const

Helper function for [subs\(\)](#).
- [ex](#) [diff](#) (const [symbol](#) &s, unsigned [nth](#)=1) const

Default interface of nth derivative [ex::diff\(s, n\)](#).
- int [compare](#) (const [basic](#) &other) const
- Compare objects syntactically to establish canonical ordering.*
- bool [is_equal](#) (const [basic](#) &other) const

Test for syntactic equality.

- const [basic](#) & [hold](#) () const

Stop further evaluation.

- unsigned [gethash](#) () const
- const [basic](#) & [setflag](#) (unsigned f) const

Set some [status_flags](#).

- const [basic](#) & [clearflag](#) (unsigned f) const

Clear some [status_flags](#).

Public Member Functions inherited from [GiNaC::refcounted](#)

- [refcounted](#) () noexcept
- unsigned int [add_reference](#) () noexcept
- unsigned int [remove_reference](#) () noexcept
- unsigned int [get_refcount](#) () const noexcept
- void [set_refcount](#) (unsigned int r) noexcept

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.61.1 Member Function Documentation

6.61.1.1 [return_type\(\)](#)

```
unsigned GiNaC::fail::return_type ( ) const [inline], [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::return_types::noncommutative_composite](#).

6.61.1.2 [do_print\(\)](#)

```
void GiNaC::fail::do_print (
    const print\_context & c,
    unsigned level ) const [protected]
```

The documentation for this class was generated from the following file:

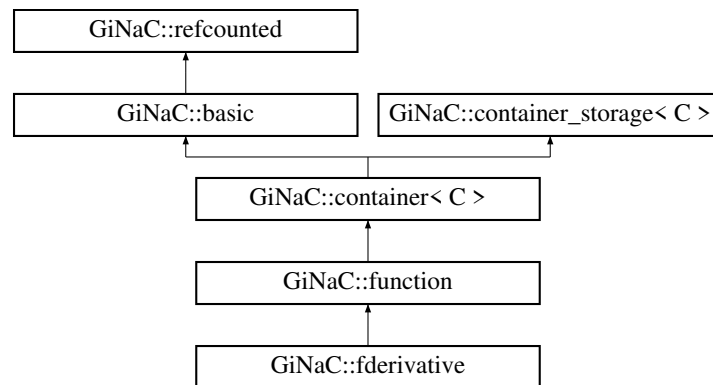
- [fail.h](#)

6.62 GiNaC::fderivative Class Reference

This class represents the (abstract) derivative of a symbolic function.

```
#include <fderivative.h>
```

Inheritance diagram for GiNaC::fderivative:



Public Member Functions

- **fderivative** (unsigned ser, unsigned param, const **exvector** &args)
Construct derivative with respect to one parameter.
- **fderivative** (unsigned ser, const **paramset** &params, const **exvector** &args)
Construct derivative with respect to multiple parameters.
- **fderivative** (unsigned ser, const **paramset** &params, **exvector** &&v)
- void **print** (const **print_context** &c, unsigned level=0) const override
Output to stream.
- **ex eval** () const override
Perform automatic non-interruptive term rewriting rules.
- **ex series** (const **relational** &r, int **order**, unsigned **options**=0) const override
The series expansion of derivatives falls back to Taylor expansion.
- **ex thiscontainer** (const **exvector** &v) const override
- **ex thiscontainer** (**exvector** &&v) const override
- void **archive** (**archive_node** &n) const override
Archive the object.
- void **read_archive** (const **archive_node** &n, **lst** &syms) override
Load (deserialize) the object from an archive node.
- const **paramset** & **derivatives** () const
Expose this object's derivative structure.

Public Member Functions inherited from GiNaC::function

- **function** (unsigned ser)
- **function** (unsigned ser, const **ex** &param1)
- **function** (unsigned ser, const **ex** &param1, const **ex** &param2)
- **function** (unsigned ser, const **ex** &param1, const **ex** &param2, const **ex** &param3)
- **function** (unsigned ser, const **ex** &param1, const **ex** &param2, const **ex** &param3, const **ex** &param4)

- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2, const [ex](#) &param3, const [ex](#) &param4, const [ex](#) &param5)
- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2, const [ex](#) &param3, const [ex](#) &param4, const [ex](#) &param5, const [ex](#) &param6)
- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2, const [ex](#) &param3, const [ex](#) &param4, const [ex](#) &param5, const [ex](#) &param6, const [ex](#) &param7)
- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2, const [ex](#) &param3, const [ex](#) &param4, const [ex](#) &param5, const [ex](#) &param6, const [ex](#) &param7, const [ex](#) &param8)
- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2, const [ex](#) &param3, const [ex](#) &param4, const [ex](#) &param5, const [ex](#) &param6, const [ex](#) &param7, const [ex](#) &param8, const [ex](#) &param9)
- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2, const [ex](#) &param3, const [ex](#) &param4, const [ex](#) &param5, const [ex](#) &param6, const [ex](#) &param7, const [ex](#) &param8, const [ex](#) &param9, const [ex](#) &param10)
- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2, const [ex](#) &param3, const [ex](#) &param4, const [ex](#) &param5, const [ex](#) &param6, const [ex](#) &param7, const [ex](#) &param8, const [ex](#) &param9, const [ex](#) &param10, const [ex](#) &param11)
- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2, const [ex](#) &param3, const [ex](#) &param4, const [ex](#) &param5, const [ex](#) &param6, const [ex](#) &param7, const [ex](#) &param8, const [ex](#) &param9, const [ex](#) &param10, const [ex](#) &param11, const [ex](#) &param12)
- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2, const [ex](#) &param3, const [ex](#) &param4, const [ex](#) &param5, const [ex](#) &param6, const [ex](#) &param7, const [ex](#) &param8, const [ex](#) &param9, const [ex](#) &param10, const [ex](#) &param11, const [ex](#) &param12, const [ex](#) &param13)
- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2, const [ex](#) &param3, const [ex](#) &param4, const [ex](#) &param5, const [ex](#) &param6, const [ex](#) &param7, const [ex](#) &param8, const [ex](#) &param9, const [ex](#) &param10, const [ex](#) &param11, const [ex](#) &param12, const [ex](#) &param13, const [ex](#) &param14)
- [function](#) (unsigned ser, const [exprseq](#) &es)
- [function](#) (unsigned ser, const [exvector](#) &v)
- [function](#) (unsigned ser, [exvector](#) &&v)
- void [print](#) (const [print_context](#) &c, unsigned level=0) const override
Output to stream.
- unsigned [precedence](#) () const override
Return relative operator precedence (for parenthezing output).
- [ex expand](#) (unsigned [options](#)=0) const override
Expand expression, i.e.
- [ex eval](#) () const override
Perform automatic non-interruptive term rewriting rules.
- [ex evalf](#) () const override
Evaluate object numerically.
- [ex eval_ncmul](#) (const [exvector](#) &v) const override
This method is defined to be in line with behavior of [function::return_type\(\)](#)
- unsigned [calchash](#) () const override
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- [ex series](#) (const [relational](#) &r, int [order](#), unsigned [options](#)=0) const override
Implementation of [ex::series](#) for functions.
- [ex thiscontainer](#) (const [exvector](#) &v) const override
- [ex thiscontainer](#) ([exvector](#) &&v) const override
- [ex conjugate](#) () const override
Implementation of [ex::conjugate](#) for functions.
- [ex real_part](#) () const override
Implementation of [ex::real_part](#) for functions.
- [ex imag_part](#) () const override
Implementation of [ex::imag_part](#) for functions.
- void [archive](#) ([archive_node](#) &n) const override
Archive the object.
- void [read_archive](#) (const [archive_node](#) &n, [lst](#) &symbols) override

Construct object from [archive_node](#).

- `bool info` (unsigned int) const override
Implementation of `ex::info` for functions.
- `ex power` (const `ex` &`exp`) const
- unsigned `get_serial` () const
- `std::string get_name` () const
Return the print name of the function.

Public Member Functions inherited from `GiNaC::container< C >`

- `container` (STLT const &s)
- `container` (STLT &&v)
- `container` (exvector::const_iterator b, exvector::const_iterator e)
- `container` (std::initializer_list< `ex` > il)
- `size_t nops` () const override
Number of operands/members.
- `ex op` (size_t i) const override
Return operand/member at position i.
- `ex & let_op` (size_t i) override
Return modifiable operand/member at position i.
- `ex subs` (const `exmap` &m, unsigned `options`=0) const override
Substitute a set of objects by arbitrary expressions.
- `container & prepend` (const `ex` &b)
Add element at front.
- `container & append` (const `ex` &b)
Add element at back.
- `container & remove_first` ()
Remove first element.
- `container & remove_last` ()
Remove last element.
- `container & remove_all` ()
Remove all elements.
- `container & sort` ()
Sort elements.
- `container & unique` ()
Remove adjacent duplicate elements.
- `const_iterator begin` () const
- `const_iterator end` () const
- `const_reverse_iterator rbegin` () const
- `const_reverse_iterator rend` () const

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic` ()
basic destructor, virtual because class `ex` will delete objects of derived classes via a `basic*`.
- `basic` (const `basic` &other)
- const `basic & operator=` (const `basic` &other)
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate` () const
Create a clone of this object on the heap.

- virtual `ex evalm () const`
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ () const`
Evaluate integrals, if result is known.
- virtual `ex eval_indexed (const basic &i) const`
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual `void dbgprint () const`
Little wrapper around print to be called within a debugger.
- virtual `void dbgprinttree () const`
Little wrapper around printtree to be called within a debugger.
- virtual `ex operator[] (const ex &index) const`
- virtual `ex operator[] (size_t i) const`
- virtual `ex & operator[] (const ex &index)`
- virtual `ex & operator[] (size_t i)`
- virtual `bool has (const ex &other, unsigned options=0) const`
Test for occurrence of a pattern.
- virtual `bool match (const ex &pattern, exmap &repls) const`
Check whether the expression matches a given pattern.
- virtual `ex map (map_function &f) const`
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual `void accept (GiNaC::visitor &v) const`
- virtual `bool is_polynomial (const ex &var) const`
Check whether this is a polynomial in the given variables.
- virtual `int degree (const ex &s) const`
Return degree of highest power in object s.
- virtual `int ldegree (const ex &s) const`
Return degree of lowest power in object s.
- virtual `ex coeff (const ex &s, int n=1) const`
Return coefficient of degree n in object s.
- virtual `ex collect (const ex &s, bool distributed=false) const`
Sort expanded expression in terms of powers of some object(s).
- virtual `ex normal (exmap &repl, exmap &rev_lookup, lst &modifier) const`
Default implementation of `ex::normal()`.
- virtual `ex to_rational (exmap &repl) const`
Default implementation of `ex::to_rational()`.
- virtual `ex to_polynomial (exmap &repl) const`
- virtual `numeric integer_content () const`
- virtual `ex smod (const numeric &xi) const`
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient () const`
Implementation `ex::max_coefficient()`.
- virtual `exvector get_free_indices () const`
Return a vector containing the free indices of an expression.
- virtual `ex add_indexed (const ex &self, const ex &other) const`
Add two indexed expressions.
- virtual `ex scalar_mul_indexed (const ex &self, const numeric &other) const`
Multiply an indexed expression with a scalar.
- virtual `bool contract_with (exvector::iterator self, exvector::iterator other, exvector &v) const`
Try to contract two indexed expressions that appear in the same product.
- `template<class T >`
`void print_dispatch (const print_context &c, unsigned level) const`

- Like `print()`, but dispatch to the specified class.
- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const
- Like `print()`, but dispatch to the specified class.
- `ex subs_one_level` (const `exmap` &m, unsigned `options`) const
- Helper function for `subs()`.
- `ex diff` (const `symbol` &s, unsigned nth=1) const
- Default interface of nth derivative `ex::diff(s, n)`.
- int `compare` (const `basic` &other) const
- Compare objects syntactically to establish canonical ordering.
- bool `is_equal` (const `basic` &other) const
- Test for syntactic equality.
- const `basic` & `hold` () const
- Stop further evaluation.
- unsigned `gethash` () const
- const `basic` & `setflag` (unsigned f) const
- Set some `status_flags`.
- const `basic` & `clearflag` (unsigned f) const
- Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Protected Member Functions

- `ex derivative` (const `symbol` &s) const override
- Implementation of `ex::diff()` for derivatives.
- bool `is_equal_same_type` (const `basic` &other) const override
- Returns true if two objects of same type are equal.
- bool `match_same_type` (const `basic` &other) const override
- Returns true if the attributes of two objects are similar enough for a match.
- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_latex` (const `print_context` &c, unsigned level) const
- void `do_print_csrc` (const `print_csrc` &c, unsigned level) const
- void `do_print_tree` (const `print_tree` &c, unsigned level) const

Protected Member Functions inherited from `GiNaC::function`

- `ex derivative` (const `symbol` &s) const override
- Implementation of `ex::diff()` for functions.
- bool `is_equal_same_type` (const `basic` &other) const override
- Returns true if two objects of same type are equal.
- bool `match_same_type` (const `basic` &other) const override
- Returns true if the attributes of two objects are similar enough for a match.
- unsigned `return_type` () const override
- `return_type_t return_type_tinfo` () const override
- `ex pderivative` (unsigned diff_param) const
- `ex expl_derivative` (const `symbol` &s) const
- bool `lookup_remember_table` (ex &result) const
- void `store_remember_table` (ex const &result) const

Protected Member Functions inherited from [GiNaC::container< C >](#)

- virtual [ex thiscontainer](#) (const [STLT](#) &v) const
Similar to [duplicate\(\)](#), but with a preset sequence.
- virtual [ex thiscontainer](#) ([STLT](#) &&v) const
Similar to [duplicate\(\)](#), but with a preset sequence (which gets pilfered).
- virtual void [printseq](#) (const [print_context](#) &c, char openbracket, char delim, char closebracket, unsigned this←_precedence, unsigned upper_precedence=0) const
Print sequence of contained elements.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
- void [do_print_python](#) (const [print_python](#) &c, unsigned level) const
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
- [STLT](#) [subchildren](#) (const [exmap](#) &m, unsigned [options](#)=0) const

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Protected Member Functions inherited from [GiNaC::container_storage< C >](#)

- [container_storage](#) ()
- [container_storage](#) (size_t n, const [ex](#) &e)
- [container_storage](#) (std::initializer_list< [ex](#) > il)
- template<class In >
 [container_storage](#) (In b, In e)
- void [reserve](#) (size_t)
- [~container_storage](#) ()
- void [reserve](#) (size_t n)
- void [reserve](#) (std::vector< [ex](#) > &v, size_t n)

Protected Attributes

- [paramset parameter_set](#)
Set of parameter numbers with respect to which to take the derivative.

Protected Attributes inherited from [GiNaC::function](#)

- unsigned [serial](#)

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

Protected Attributes inherited from [GiNaC::container_storage< C >](#)

- [STLT seq](#)

Additional Inherited Members

Public Types inherited from [GiNaC::container< C >](#)

- typedef [STLT::const_iterator](#) [const_iterator](#)
- typedef [STLT::const_reverse_iterator](#) [const_reverse_iterator](#)

Static Public Member Functions inherited from [GiNaC::function](#)

- static unsigned [register_new](#) ([function_options](#) const &opt)
- static unsigned [find_function](#) (const std::string &name, unsigned nparams)
Find serial number of function by name and number of parameters.
- static std::vector< [function_options](#) > [get_registered_functions](#) ()

Static Public Attributes inherited from [GiNaC::function](#)

- static unsigned [current_serial](#) = 0
This can be used as a hook for external applications.

Protected Types inherited from [GiNaC::container< C >](#)

- typedef [container_storage< C >::STLT](#) [STLT](#)

Protected Types inherited from [GiNaC::container_storage< C >](#)

- typedef C< [ex](#) > [STLT](#)

Static Protected Member Functions inherited from [GiNaC::function](#)

- static std::vector< [function_options](#) > & [registered_functions](#) ()

Static Protected Member Functions inherited from [GiNaC::container< C >](#)

- static unsigned [get_default_flags](#) ()
*Specialization of [container::get_default_flags\(\)](#) for *lst*.*
- static char [get_open_delim](#) ()
*Specialization of [container::get_open_delim\(\)](#) for *lst*.*
- static char [get_close_delim](#) ()
*Specialization of [container::get_close_delim\(\)](#) for *lst*.*

Static Protected Member Functions inherited from [GiNaC::container_storage< C >](#)

- static void [reserve](#) (STLT &, size_t)

6.62.1 Detailed Description

This class represents the (abstract) derivative of a symbolic function.

It is used to represent the derivatives of functions that do not have a derivative or series expansion procedure defined.

6.62.2 Constructor & Destructor Documentation

6.62.2.1 fderivative() [1/3]

```
GiNaC::fderivative::fderivative (
    unsigned ser,
    unsigned param,
    const exvector & args )
```

Construct derivative with respect to one parameter.

Parameters

<i>ser</i>	Serial number of function
<i>param</i>	Number of parameter with respect to which to take the derivative
<i>args</i>	Arguments of derivative function

References [parameter_set](#).

6.62.2.2 fderivative() [2/3]

```
GiNaC::fderivative::fderivative (
    unsigned ser,
    const paramset & params,
    const exvector & args )
```

Construct derivative with respect to multiple parameters.

Parameters

<i>ser</i>	Serial number of function
<i>params</i>	Set of numbers of parameters with respect to which to take the derivative
<i>args</i>	Arguments of derivative function

6.62.2.3 fderivative() [3/3]

```
GiNaC::fderivative::fderivative (
    unsigned ser,
    const paramset & params,
    exvector && v )
```

6.62.3 Member Function Documentation**6.62.3.1 print()**

```
void GiNaC::fderivative::print (
    const print\_context & c,
    unsigned level = 0 ) const [override], [virtual]
```

Output to stream.

This performs double dispatch on the dynamic type of *this and the dynamic type of the supplied print context.

Parameters

<i>c</i>	print context object that describes the output formatting
<i>level</i>	value that is used to identify the precedence or indentation level for placing parentheses and formatting

Reimplemented from [GiNaC::basic](#).

References [c](#), and [GiNaC::basic::print\(\)](#).

6.62.3.2 eval()

```
ex GiNaC::fderivative::eval ( ) const [override], [virtual]
```

Perform automatic non-interruptive term rewriting rules.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::basic::hold\(\)](#), [parameter_set](#), [GiNaC::function::pderivative\(\)](#), [GiNaC::function::registered_functions\(\)](#), [GiNaC::container_storage< C >::seq](#), and [GiNaC::function::serial](#).

6.62.3.3 series()

```
ex GiNaC::fderivative::series (
    const relational & r,
    int order,
    unsigned options = 0 ) const [override], [virtual]
```

The series expansion of derivatives falls back to Taylor expansion.

See also

[basic::series](#)

Reimplemented from [GiNaC::basic](#).

References [options](#), [order](#), [r](#), and [GiNaC::basic::series\(\)](#).

6.62.3.4 thiscontainer() [1/2]

```
ex GiNaC::fderivative::thiscontainer (
    const exvector & v ) const [override]
```

References [parameter_set](#), and [GiNaC::function::serial](#).

6.62.3.5 thiscontainer() [2/2]

```
ex GiNaC::fderivative::thiscontainer (
    exvector && v ) const [override]
```

References [parameter_set](#), and [GiNaC::function::serial](#).

6.62.3.6 archive()

```
void GiNaC::fderivative::archive (
    archive_node & n ) const [override], [virtual]
```

Archive the object.

Reimplemented from [GiNaC::container< C >](#).

References [GiNaC::container< C >::end\(\)](#), [n](#), and [parameter_set](#).

6.62.3.7 read_archive()

```
void GiNaC::fderivative::read_archive (
    const archive_node & n,
    lst & syms ) [override], [virtual]
```

Load (deserialize) the object from an archive node.

Construct object from [archive_node](#).

Note

This method is essentially a constructor. However, constructors can't be virtual. So, if unarchiving routines are implemented as constructors one would need to define such a constructor in every class, even if all it does is simply calling constructor of a superclass.

Reimplemented from [GiNaC::container< C >](#).

References [n](#), and [parameter_set](#).

6.62.3.8 derivative()

```
ex GiNaC::fderivative::derivative (
    const symbol & s ) const [override], [protected], [virtual]
```

Implementation of [ex::diff\(\)](#) for derivatives.

It applies the chain rule.

See also

[ex::diff](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::diff\(\)](#), [GiNaC::ex::is_zero\(\)](#), [parameter_set](#), [GiNaC::container_storage< C >::seq](#), and [GiNaC::function::serial](#).

6.62.3.9 is_equal_same_type()

```
bool GiNaC::fderivative::is_equal_same_type (
    const basic & other ) const [override], [protected], [virtual]
```

Returns true if two objects of same type are equal.

Normally needs not be reimplemented as long as it wasn't overwritten by some parent class, since it just calls [compare_same_type\(\)](#). The reason why this function exists is that sometimes it is easier to determine equality than an order relation and then it can be overridden.

Reimplemented from [GiNaC::container< C >](#).

References [GINAC_ASSERT](#), and [parameter_set](#).

6.62.3.10 match_same_type()

```
bool GiNaC::fderivative::match_same_type (
    const basic & other ) const [override], [protected], [virtual]
```

Returns true if the attributes of two objects are similar enough for a match.

This function must not match subexpressions (this is already done by [basic::match\(\)](#)). Only attributes not accessible by [op\(\)](#) should be compared. This is also the reason why this function doesn't take the wildcard replacement list from [match\(\)](#) as an argument: only subexpressions are subject to wildcard matches. Also, this function only needs to be implemented for container classes because [is_equal_same_type\(\)](#) is automatically used instead of [match_same_type\(\)](#) if [nops\(\)](#) == 0.

See also

[basic::match](#)

Reimplemented from [GiNaC::basic](#).

References [GINAC_ASSERT](#), and [parameter_set](#).

6.62.3.11 derivatives()

```
const paramset & GiNaC::fderivative::derivatives ( ) const
```

Expose this object's derivative structure.

Parameter numbers occurring more than once stand for repeated differentiation with respect to that parameter. If a symbolic function $f(x,y)$ is differentiated with respect to x , this method will return $\{0\}$. If $f(x,y)$ is differentiated twice with respect to y , it will return $\{1,1\}$. (This corresponds to the way this object is printed.)

Returns

multiset of function's parameter numbers that are abstractly differentiated.

References [parameter_set](#).

6.62.3.12 do_print()

```
void GiNaC::fderivative::do_print (
    const print\_context & c,
    unsigned level ) const [protected]
```

References [c](#), [GiNaC::container< C >::end\(\)](#), [parameter_set](#), [GiNaC::container< C >::precedence\(\)](#), [GiNaC::function::precedence\(\)](#), [GiNaC::container< C >::printseq\(\)](#), [GiNaC::function::registered_functions\(\)](#), and [GiNaC::function::serial](#).

6.62.3.13 do_print_latex()

```
void GiNaC::fderivative::do_print_latex (
    const print\_context & c,
    unsigned level ) const [protected]
```

References [c](#), [GiNaC::container< C >::end\(\)](#), [order](#), [parameter_set](#), [GiNaC::container< C >::precedence\(\)](#), [GiNaC::function::precedence\(\)](#), [GiNaC::container< C >::printseq\(\)](#), [GiNaC::function::registered_functions\(\)](#), and [GiNaC::function::serial](#).

6.62.3.14 do_print_csrc()

```
void GiNaC::fderivative::do_print_csrc (
    const print\_csrc & c,
    unsigned level ) const [protected]
```

References [c](#), [GiNaC::container< C >::end\(\)](#), [parameter_set](#), [GiNaC::container< C >::precedence\(\)](#), [GiNaC::function::precedence\(\)](#), [GiNaC::container< C >::printseq\(\)](#), [GiNaC::function::registered_functions\(\)](#), and [GiNaC::function::serial](#).

6.62.3.15 do_print_tree()

```
void GiNaC::fderivative::do_print_tree (
    const print\_tree & c,
    unsigned level ) const [protected]
```

References [c](#), [GiNaC::container< C >::end\(\)](#), [GiNaC::basic::flags](#), [GiNaC::basic::hashvalue](#), [GiNaC::container< C >::nops\(\)](#), [parameter_set](#), [GiNaC::function::registered_functions\(\)](#), [GiNaC::container_storage< C >::seq](#), and [GiNaC::function::serial](#).

6.62.4 Member Data Documentation

6.62.4.1 parameter_set

```
paramset GiNaC::fderivative::parameter_set [protected]
```

Set of parameter numbers with respect to which to take the derivative.

Referenced by [archive\(\)](#), [derivative\(\)](#), [derivatives\(\)](#), [do_print\(\)](#), [do_print_csrc\(\)](#), [do_print_latex\(\)](#), [do_print_tree\(\)](#), [eval\(\)](#), [fderivative\(\)](#), [is_equal_same_type\(\)](#), [match_same_type\(\)](#), [read_archive\(\)](#), [thiscontainer\(\)](#), and [thiscontainer\(\)](#).

The documentation for this class was generated from the following files:

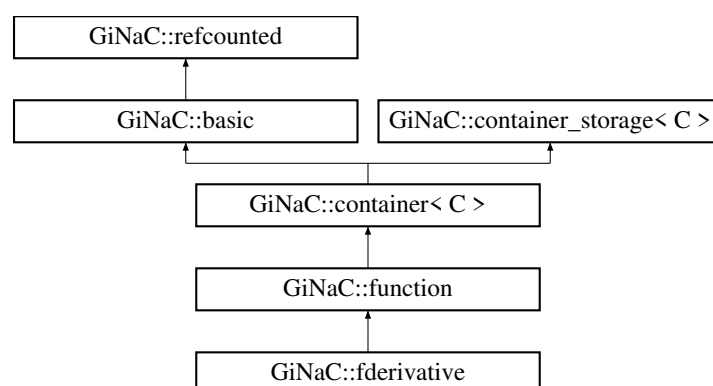
- [fderivative.h](#)
- [fderivative.cpp](#)

6.63 GiNaC::function Class Reference

The class function is used to implement builtin functions like sin, cos... and user defined functions.

```
#include <function.h>
```

Inheritance diagram for GiNaC::function:



Public Member Functions

- [function](#) (unsigned ser)
- [function](#) (unsigned ser, const [ex](#) &param1)
- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2)
- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2, const [ex](#) &param3)
- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2, const [ex](#) &param3, const [ex](#) &param4)
- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2, const [ex](#) &param3, const [ex](#) &param4, const [ex](#) &param5)
- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2, const [ex](#) &param3, const [ex](#) &param4, const [ex](#) &param5, const [ex](#) &param6)
- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2, const [ex](#) &param3, const [ex](#) &param4, const [ex](#) &param5, const [ex](#) &param6, const [ex](#) &param7)
- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2, const [ex](#) &param3, const [ex](#) &param4, const [ex](#) &param5, const [ex](#) &param6, const [ex](#) &param7, const [ex](#) &param8)
- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2, const [ex](#) &param3, const [ex](#) &param4, const [ex](#) &param5, const [ex](#) &param6, const [ex](#) &param7, const [ex](#) &param8, const [ex](#) &param9)
- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2, const [ex](#) &param3, const [ex](#) &param4, const [ex](#) &param5, const [ex](#) &param6, const [ex](#) &param7, const [ex](#) &param8, const [ex](#) &param9, const [ex](#) &param10)
- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2, const [ex](#) &param3, const [ex](#) &param4, const [ex](#) &param5, const [ex](#) &param6, const [ex](#) &param7, const [ex](#) &param8, const [ex](#) &param9, const [ex](#) &param10, const [ex](#) &param11)
- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2, const [ex](#) &param3, const [ex](#) &param4, const [ex](#) &param5, const [ex](#) &param6, const [ex](#) &param7, const [ex](#) &param8, const [ex](#) &param9, const [ex](#) &param10, const [ex](#) &param11, const [ex](#) &param12)
- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2, const [ex](#) &param3, const [ex](#) &param4, const [ex](#) &param5, const [ex](#) &param6, const [ex](#) &param7, const [ex](#) &param8, const [ex](#) &param9, const [ex](#) &param10, const [ex](#) &param11, const [ex](#) &param12, const [ex](#) &param13)
- [function](#) (unsigned ser, const [ex](#) &param1, const [ex](#) &param2, const [ex](#) &param3, const [ex](#) &param4, const [ex](#) &param5, const [ex](#) &param6, const [ex](#) &param7, const [ex](#) &param8, const [ex](#) &param9, const [ex](#) &param10, const [ex](#) &param11, const [ex](#) &param12, const [ex](#) &param13, const [ex](#) &param14)
- [function](#) (unsigned ser, const [exprseq](#) &es)
- [function](#) (unsigned ser, const [exvector](#) &v)
- [function](#) (unsigned ser, [exvector](#) &&v)
- void [print](#) (const [print_context](#) &c, unsigned level=0) const override
Output to stream.
- unsigned [precedence](#) () const override
Return relative operator precedence (for parenthezing output).
- [ex expand](#) (unsigned [options](#)=0) const override
Expand expression, i.e.
- [ex eval](#) () const override
Perform automatic non-interruptive term rewriting rules.
- [ex evalf](#) () const override
Evaluate object numerically.
- [ex eval_ncmul](#) (const [exvector](#) &v) const override
This method is defined to be in line with behavior of [function::return_type\(\)](#)
- unsigned [calchash](#) () const override
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- [ex series](#) (const [relational](#) &r, int [order](#), unsigned [options](#)=0) const override
Implementation of [ex::series](#) for functions.
- [ex thiscontainer](#) (const [exvector](#) &v) const override
- [ex thiscontainer](#) ([exvector](#) &&v) const override
- [ex conjugate](#) () const override
Implementation of [ex::conjugate](#) for functions.

- `ex real_part ()` const override
Implementation of `ex::real_part` for functions.
- `ex imag_part ()` const override
Implementation of `ex::imag_part` for functions.
- `void archive (archive_node &n)` const override
Archive the object.
- `void read_archive (const archive_node &n, lst &syms)` override
Construct object from `archive_node`.
- `bool info (unsigned inf)` const override
Implementation of `ex::info` for functions.
- `ex power (const ex &exp)` const
- `unsigned get_serial ()` const
- `std::string get_name ()` const
Return the print name of the function.

Public Member Functions inherited from `GiNaC::container< C >`

- `container (STLT const &s)`
- `container (STLT &&v)`
- `container (exvector::const_iterator b, exvector::const_iterator e)`
- `container (std::initializer_list< ex > il)`
- `size_t nops ()` const override
Number of operands/members.
- `ex op (size_t i)` const override
*Return operand/member at position *i*.*
- `ex & let_op (size_t i)` override
*Return modifiable operand/member at position *i*.*
- `ex subs (const exmap &m, unsigned options=0)` const override
Substitute a set of objects by arbitrary expressions.
- `container & prepend (const ex &b)`
Add element at front.
- `container & append (const ex &b)`
Add element at back.
- `container & remove_first ()`
Remove first element.
- `container & remove_last ()`
Remove last element.
- `container & remove_all ()`
Remove all elements.
- `container & sort ()`
Sort elements.
- `container & unique ()`
Remove adjacent duplicate elements.
- `const_iterator begin ()` const
- `const_iterator end ()` const
- `const_reverse_iterator rbegin ()` const
- `const_reverse_iterator rend ()` const

Public Member Functions inherited from **GiNaC::basic**

- virtual `~basic ()`
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic (const basic &other)`
- `const basic & operator= (const basic &other)`
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate () const`
Create a clone of this object on the heap.
- virtual `ex evalm () const`
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ () const`
Evaluate integrals, if result is known.
- virtual `ex eval_indexed (const basic &i) const`
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual `void dbgprint () const`
Little wrapper around print to be called within a debugger.
- virtual `void dbgprinttree () const`
Little wrapper around printtree to be called within a debugger.
- virtual `ex operator[] (const ex &index) const`
- virtual `ex operator[] (size_t i) const`
- virtual `ex & operator[] (const ex &index)`
- virtual `ex & operator[] (size_t i)`
- virtual `bool has (const ex &other, unsigned options=0) const`
Test for occurrence of a pattern.
- virtual `bool match (const ex &pattern, exmap &repls) const`
Check whether the expression matches a given pattern.
- virtual `ex map (map_function &f) const`
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual `void accept (GiNaC::visitor &v) const`
- virtual `bool is_polynomial (const ex &var) const`
Check whether this is a polynomial in the given variables.
- virtual `int degree (const ex &s) const`
Return degree of highest power in object s.
- virtual `int ldegree (const ex &s) const`
Return degree of lowest power in object s.
- virtual `ex coeff (const ex &s, int n=1) const`
Return coefficient of degree n in object s.
- virtual `ex collect (const ex &s, bool distributed=false) const`
Sort expanded expression in terms of powers of some object(s).
- virtual `ex normal (exmap &repl, exmap &rev_lookup, lst &modifier) const`
Default implementation of `ex::normal()`.
- virtual `ex to_rational (exmap &repl) const`
Default implementation of `ex::to_rational()`.
- virtual `ex to_polynomial (exmap &repl) const`
- virtual `numeric integer_content () const`
- virtual `ex smod (const numeric &xi) const`
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient () const`
Implementation `ex::max_coefficient()`.
- virtual `exvector get_free_indices () const`

- Return a vector containing the free indices of an expression.*

 - virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const

Add two indexed expressions.

 - virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const

Multiply an indexed expression with a scalar.

 - virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const

Try to contract two indexed expressions that appear in the same product.

 - template<class T >
void `print_dispatch` (const `print_context` &c, unsigned level) const

Like `print()`, but dispatch to the specified class.

 - void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const

Like `print()`, but dispatch to the specified class.

 - `ex subs_one_level` (const `exmap` &m, unsigned `options`) const

Helper function for `subs()`.

 - `ex diff` (const `symbol` &s, unsigned nth=1) const

Default interface of nth derivative `ex::diff(s, n)`.

 - int `compare` (const `basic` &other) const

Compare objects syntactically to establish canonical ordering.

 - bool `is_equal` (const `basic` &other) const

Test for syntactic equality.

 - const `basic` & `hold` () const

Stop further evaluation.

 - unsigned `gethash` () const
 - const `basic` & `setflag` (unsigned f) const

Set some `status_flags`.

 - const `basic` & `clearflag` (unsigned f) const

Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Static Public Member Functions

- static unsigned `register_new` (`function_options` const &opt)
 - static unsigned `find_function` (const std::string &name, unsigned nparams)
- Find serial number of function by name and number of parameters.*
- static std::vector< `function_options` > `get_registered_functions` ()

Static Public Attributes

- static unsigned `current_serial` = 0
- This can be used as a hook for external applications.*

Protected Member Functions

- `ex derivative` (const `symbol` &s) const override
Implementation of `ex::diff()` for functions.
- `bool is_equal_same_type` (const `basic` &other) const override
Returns true if two objects of same type are equal.
- `bool match_same_type` (const `basic` &other) const override
Returns true if the attributes of two objects are similar enough for a match.
- `unsigned return_type` () const override
- `return_type_t return_type_tinfo` () const override
- `ex pderivative` (unsigned diff_param) const
- `ex expl_derivative` (const `symbol` &s) const
- `bool lookup_remember_table` (ex &result) const
- `void store_remember_table` (ex const &result) const

Protected Member Functions inherited from `GiNaC::container< C >`

- `virtual ex thiscontainer` (const `STLT` &v) const
Similar to `duplicate()`, but with a preset sequence.
- `virtual ex thiscontainer` (`STLT` &&v) const
Similar to `duplicate()`, but with a preset sequence (which gets pilfered).
- `virtual void printseq` (const `print_context` &c, char openbracket, char delim, char closebracket, unsigned this←_precedence, unsigned upper_precedence=0) const
Print sequence of contained elements.
- `void do_print` (const `print_context` &c, unsigned level) const
- `void do_print_tree` (const `print_tree` &c, unsigned level) const
- `void do_print_python` (const `print_python` &c, unsigned level) const
- `void do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
- `STLT subschildren` (const `exmap` &m, unsigned options=0) const

Protected Member Functions inherited from `GiNaC::basic`

- `basic` ()
- `virtual int compare_same_type` (const `basic` &other) const
Returns order relation between two objects of same type.
- `void ensure_if_modifiable` () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- `void do_print` (const `print_context` &c, unsigned level) const
Default output to stream.
- `void do_print_tree` (const `print_tree` &c, unsigned level) const
Tree output to stream.
- `void do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
Python parsable output to stream.

Protected Member Functions inherited from `GiNaC::container_storage< C >`

- `container_storage` ()
- `container_storage` (size_t n, const `ex` &e)
- `container_storage` (std::initializer_list< `ex` > il)
- `template<class In >`
`container_storage` (In b, In e)
- `void reserve` (size_t)
- `~container_storage` ()
- `void reserve` (size_t n)
- `void reserve` (std::vector< `ex` > &v, size_t n)

Static Protected Member Functions

- static std::vector< [function_options](#) > & [registered_functions](#) ()

Static Protected Member Functions inherited from [GiNaC::container< C >](#)

- static unsigned [get_default_flags](#) ()
Specialization of [container::get_default_flags\(\)](#) for [lst](#).
- static char [get_open_delim](#) ()
Specialization of [container::get_open_delim\(\)](#) for [lst](#).
- static char [get_close_delim](#) ()
Specialization of [container::get_close_delim\(\)](#) for [lst](#).

Static Protected Member Functions inherited from [GiNaC::container_storage< C >](#)

- static void [reserve](#) (STLT &, size_t)

Protected Attributes

- unsigned [serial](#)

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

Protected Attributes inherited from [GiNaC::container_storage< C >](#)

- [STLT seq](#)

Friends

- class [remember_table_entry](#)

Additional Inherited Members

Public Types inherited from [GiNaC::container< C >](#)

- typedef STLT::const_iterator [const_iterator](#)
- typedef STLT::const_reverse_iterator [const_reverse_iterator](#)

Protected Types inherited from [GiNaC::container< C >](#)

- typedef [container_storage< C >::STLT](#) [STLT](#)

Protected Types inherited from [GiNaC::container_storage< C >](#)

- typedef C< [ex](#) > [STLT](#)

6.63.1 Detailed Description

The class function is used to implement builtin functions like sin, cos... and user defined functions.

6.63.2 Constructor & Destructor Documentation

6.63.2.1 function() [1/18]

```
GiNaC::function::function (
    unsigned ser )
```

6.63.2.2 function() [2/18]

```
GiNaC::function::function (
    unsigned ser,
    const ex & param1 )
```

6.63.2.3 function() [3/18]

```
GiNaC::function::function (
    unsigned ser,
    const ex & param1,
    const ex & param2 )
```

6.63.2.4 function() [4/18]

```
GiNaC::function::function (
    unsigned ser,
    const ex & param1,
    const ex & param2,
    const ex & param3 )
```

6.63.2.5 function() [5/18]

```
GiNaC::function::function (
    unsigned ser,
    const ex & param1,
    const ex & param2,
    const ex & param3,
    const ex & param4 )
```

6.63.2.6 function() [6/18]

```
GiNaC::function::function (
    unsigned ser,
    const ex & param1,
    const ex & param2,
    const ex & param3,
    const ex & param4,
    const ex & param5 )
```

6.63.2.7 function() [7/18]

```
GiNaC::function::function (
    unsigned ser,
    const ex & param1,
    const ex & param2,
    const ex & param3,
    const ex & param4,
    const ex & param5,
    const ex & param6 )
```

6.63.2.8 function() [8/18]

```
GiNaC::function::function (
    unsigned ser,
    const ex & param1,
    const ex & param2,
    const ex & param3,
    const ex & param4,
    const ex & param5,
    const ex & param6,
    const ex & param7 )
```

6.63.2.9 function() [9/18]

```
GiNaC::function::function (
    unsigned ser,
    const ex & param1,
    const ex & param2,
    const ex & param3,
    const ex & param4,
    const ex & param5,
    const ex & param6,
    const ex & param7,
    const ex & param8 )
```


6.63.2.10 function() [10/18]

```
GiNaC::function::function (
    unsigned ser,
    const ex & param1,
    const ex & param2,
    const ex & param3,
    const ex & param4,
    const ex & param5,
    const ex & param6,
    const ex & param7,
    const ex & param8,
    const ex & param9 )
```

6.63.2.11 function() [11/18]

```
GiNaC::function::function (
    unsigned ser,
    const ex & param1,
    const ex & param2,
    const ex & param3,
    const ex & param4,
    const ex & param5,
    const ex & param6,
    const ex & param7,
    const ex & param8,
    const ex & param9,
    const ex & param10 )
```

6.63.2.12 function() [12/18]

```
GiNaC::function::function (
    unsigned ser,
    const ex & param1,
    const ex & param2,
    const ex & param3,
    const ex & param4,
    const ex & param5,
    const ex & param6,
    const ex & param7,
    const ex & param8,
    const ex & param9,
    const ex & param10,
    const ex & param11 )
```

6.63.2.13 function() [13/18]

```
GiNaC::function::function (
    unsigned ser,
    const ex & param1,
    const ex & param2,
    const ex & param3,
```

```

    const ex & param4,
    const ex & param5,
    const ex & param6,
    const ex & param7,
    const ex & param8,
    const ex & param9,
    const ex & param10,
    const ex & param11,
    const ex & param12 )

```

6.63.2.14 [function\(\)](#) [14/18]

```

GiNaC::function::function (
    unsigned ser,
    const ex & param1,
    const ex & param2,
    const ex & param3,
    const ex & param4,
    const ex & param5,
    const ex & param6,
    const ex & param7,
    const ex & param8,
    const ex & param9,
    const ex & param10,
    const ex & param11,
    const ex & param12,
    const ex & param13 )

```

6.63.2.15 [function\(\)](#) [15/18]

```

GiNaC::function::function (
    unsigned ser,
    const ex & param1,
    const ex & param2,
    const ex & param3,
    const ex & param4,
    const ex & param5,
    const ex & param6,
    const ex & param7,
    const ex & param8,
    const ex & param9,
    const ex & param10,
    const ex & param11,
    const ex & param12,
    const ex & param13,
    const ex & param14 )

```

6.63.2.16 [function\(\)](#) [16/18]

```

GiNaC::function::function (
    unsigned ser,
    const exprseq & es )

```

References [GiNaC::basic::clearflag\(\)](#), and [GiNaC::status_flags::evaluated](#).

6.63.2.17 function() [17/18]

```
GiNaC::function::function (
    unsigned ser,
    const exvector & v )
```

6.63.2.18 function() [18/18]

```
GiNaC::function::function (
    unsigned ser,
    exvector && v )
```

6.63.3 Member Function Documentation**6.63.3.1 print()**

```
void GiNaC::function::print (
    const print_context & c,
    unsigned level = 0 ) const [override], [virtual]
```

Output to stream.

This performs double dispatch on the dynamic type of *this and the dynamic type of the supplied print context.

Parameters

<i>c</i>	print context object that describes the output formatting
<i>level</i>	value that is used to identify the precedence or indentation level for placing parentheses and formatting

Reimplemented from [GiNaC::basic](#).

References [c](#), [current_serial](#), [GiNaC::basic::flags](#), [GiNaC::class_info< OPT >::get_parent\(\)](#), [GINAC_ASSERT](#), [GiNaC::basic::hashvalue](#), [GiNaC::function_options::name](#), [GiNaC::container< C >::nops\(\)](#), [GiNaC::function_options::nparams](#), [GiNaC::class_info< OPT >::options](#), [GiNaC::container< C >::precedence\(\)](#), [precedence\(\)](#), [print\(\)](#), [GiNaC::function_options::print_di](#), [GiNaC::function_options::print_use_exvector_args](#), [GiNaC::container< C >::printseq\(\)](#), [registered_functions\(\)](#), [GiNaC::print_context::s](#), [GiNaC::container_storage< C >::seq](#), [serial](#), and [GiNaC::function_options::TeX_name](#).

Referenced by [print\(\)](#).

6.63.3.2 precedence()

```
unsigned GiNaC::function::precedence ( ) const [inline], [override], [virtual]
```

Return relative operator precedence (for parenthezing output).

Reimplemented from [GiNaC::container< C >](#).

Referenced by [GiNaC::fderivative::do_print\(\)](#), [GiNaC::fderivative::do_print_csrc\(\)](#), [GiNaC::fderivative::do_print_latex\(\)](#), and [print\(\)](#).

6.63.3.3 expand()

```
ex GiNaC::function::expand (
    unsigned options = 0 ) const [override], [virtual]
```

Expand expression, i.e.

multiply it out and return the result as a new expression.

Reimplemented from [GiNaC::basic](#).

References [current_serial](#), [GiNaC::function_options::expand_f](#), [GiNaC::expand_options::expand_function_args](#), [GiNaC::function_options::expand_use_exvector_args](#), [GiNaC::status_flags::expanded](#), [GINAC_ASSERT](#), [GiNaC::function_options::nparams](#), [registered_functions\(\)](#), [GiNaC::container_storage< C >::seq](#), [serial](#), and [GiNaC::basic::setflag\(\)](#).

6.63.3.4 eval()

```
ex GiNaC::function::eval ( ) const [override], [virtual]
```

Perform automatic non-interruptive term rewriting rules.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex0](#), [GiNaC::canonicalize\(\)](#), [current_serial](#), [GiNaC::function_options::eval_f](#), [GiNaC::function_options::eval_use_exvector_args](#), [GiNaC::status_flags::evaluated](#), [GiNaC::basic::flags](#), [GINAC_ASSERT](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::is_zero\(\)](#), [lookup_remember_table\(\)](#), [GiNaC::function_options::nparams](#), [registered_functions\(\)](#), [GiNaC::container_storage< C >::seq](#), [serial](#), [store_remember_table\(\)](#), [GiNaC::function_options::symtree](#), [thiscontainer\(\)](#), and [GiNaC::function_options::use_remember_table\(\)](#).

6.63.3.5 evalf()

```
ex GiNaC::function::evalf ( ) const [override], [virtual]
```

Evaluate object numerically.

Reimplemented from [GiNaC::basic](#).

References [current_serial](#), [GiNaC::function_options::evalf_f](#), [GiNaC::function_options::evalf_params_first](#), [GiNaC::function_options::evalf_use_exvector_args](#), [GINAC_ASSERT](#), [GiNaC::basic::hold\(\)](#), [GiNaC::function_options::nparams](#), [registered_functions\(\)](#), [GiNaC::container_storage< C >::seq](#), and [serial](#).

Referenced by [GiNaC::iterated_integral2_eval\(\)](#), and [GiNaC::iterated_integral3_eval\(\)](#).

6.63.3.6 eval_ncmul()

```
ex GiNaC::function::eval_ncmul (
    const exvector & v ) const [override], [virtual]
```

This method is defined to be in line with behavior of [function::return_type\(\)](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::begin\(\)](#), and [GiNaC::container_storage< C >::seq](#).

6.63.3.7 calchash()

```
unsigned GiNaC::function::calchash ( ) const [override], [virtual]
```

Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.

The method inherited from class `basic` computes a hash value based on the type and hash values of possible members. For this reason it is well suited for container classes but atomic classes should override this implementation because otherwise they would all end up with the same hashvalue.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::status_flags::evaluated](#), [GiNaC::basic::flags](#), [GiNaC::ex::gethash\(\)](#), [GiNaC::golden_ratio_hash\(\)](#), [GiNaC::status_flags::hash_calculated](#), [GiNaC::basic::hashvalue](#), [GiNaC::make_hash_seed\(\)](#), [GiNaC::container< C >::nops\(\)](#), [GiNaC::container< C >::op\(\)](#), [GiNaC::rotate_left\(\)](#), [serial](#), and [GiNaC::basic::setflag\(\)](#).

6.63.3.8 series()

```
ex GiNaC::function::series (
    const relational & r,
    int order,
    unsigned options = 0 ) const [override], [virtual]
```

Implementation of [ex::series](#) for functions.

@see [ex::series](#)

Reimplemented from [GiNaC::basic](#).

References [current_serial](#), [GINAC_ASSERT](#), [GiNaC::function_options::nparams](#), [options](#), [order](#), [r](#), [registered_functions\(\)](#), [GiNaC::container_storage< C >::seq](#), [serial](#), [GiNaC::basic::series\(\)](#), [GiNaC::function_options::series_f](#), and [GiNaC::function_options::series_use_exvector_args](#).

6.63.3.9 thiscontainer() [1/2]

```
ex GiNaC::function::thiscontainer (
    const exvector & v ) const [override]
```

References [serial](#).

Referenced by [eval\(\)](#).

6.63.3.10 thiscontainer() [2/2]

```
ex GiNaC::function::thiscontainer (
    exvector && v ) const [override]
```

References [serial](#).

6.63.3.11 conjugate()

```
ex GiNaC::function::conjugate ( ) const [override], [virtual]
```

Implementation of [ex::conjugate](#) for functions.

Reimplemented from [GiNaC::container< C >](#).

References [GiNaC::function_options::conjugate_f](#), [GiNaC::function_options::conjugate_use_exvector_args](#), [GINAC_ASSERT](#), [GiNaC::function_options::nparams](#), [registered_functions\(\)](#), [GiNaC::container_storage< C >::seq](#), and [serial](#).

6.63.3.12 real_part()

```
ex GiNaC::function::real_part ( ) const [override], [virtual]
```

Implementation of [ex::real_part](#) for functions.

Reimplemented from [GiNaC::container< C >](#).

References [GINAC_ASSERT](#), [GiNaC::function_options::nparams](#), [GiNaC::basic::real_part\(\)](#), [GiNaC::function_options::real_part_f](#), [GiNaC::function_options::real_part_use_exvector_args](#), [registered_functions\(\)](#), [GiNaC::container_storage< C >::seq](#), and [serial](#).

6.63.3.13 imag_part()

```
ex GiNaC::function::imag_part ( ) const [override], [virtual]
```

Implementation of [ex::imag_part](#) for functions.

Reimplemented from [GiNaC::container< C >](#).

References [GINAC_ASSERT](#), [GiNaC::basic::imag_part\(\)](#), [GiNaC::function_options::imag_part_f](#), [GiNaC::function_options::imag_part_f](#), [GiNaC::function_options::nparams](#), [registered_functions\(\)](#), [GiNaC::container_storage< C >::seq](#), and [serial](#).

6.63.3.14 archive()

```
void GiNaC::function::archive (
    archive_node & n ) const [override], [virtual]
```

Archive the object.

Reimplemented from [GiNaC::container< C >](#).

References [GINAC_ASSERT](#), [n](#), [registered_functions\(\)](#), and [serial](#).

6.63.3.15 read_archive()

```
void GiNaC::function::read_archive (
    const archive_node & n,
    lst & syms ) [override], [virtual]
```

Construct object from [archive_node](#).

Reimplemented from [GiNaC::container< C >](#).

References [n](#), [registered_functions\(\)](#), [GiNaC::container_storage< C >::seq](#), and [serial](#).

6.63.3.16 info()

```
bool GiNaC::function::info (
    unsigned inf ) const [override], [virtual]
```

Implementation of [ex::info](#) for functions.

Reimplemented from [GiNaC::container< C >](#).

References [GINAC_ASSERT](#), [GiNaC::basic::info\(\)](#), [GiNaC::function_options::info_f](#), [GiNaC::function_options::info_use_exvector_arg](#), [GiNaC::function_options::nparams](#), [registered_functions\(\)](#), [GiNaC::container_storage< C >::seq](#), and [serial](#).

6.63.3.17 derivative()

```
ex GiNaC::function::derivative (
    const symbol & s ) const [override], [protected], [virtual]
```

Implementation of [ex::diff\(\)](#) for functions.

It applies the chain rule, except for the Order term function. @see [ex::diff](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::diff\(\)](#), [expl_derivative\(\)](#), [GiNaC::ex::is_zero\(\)](#), [pderivative\(\)](#), and [GiNaC::container_storage< C >::seq](#).

6.63.3.18 is_equal_same_type()

```
bool GiNaC::function::is_equal_same_type (
    const basic & other ) const [override], [protected], [virtual]
```

Returns true if two objects of same type are equal.

Normally needs not be reimplemented as long as it wasn't overwritten by some parent class, since it just calls [compare_same_type\(\)](#). The reason why this function exists is that sometimes it is easier to determine equality than an order relation and then it can be overridden.

Reimplemented from [GiNaC::container< C >](#).

References [GINAC_ASSERT](#), [GiNaC::container< C >::is_equal_same_type\(\)](#), and [serial](#).

6.63.3.19 `match_same_type()`

```
bool GiNaC::function::match_same_type (
    const basic & other ) const [override], [protected], [virtual]
```

Returns true if the attributes of two objects are similar enough for a match.

This function must not match subexpressions (this is already done by [basic::match\(\)](#)). Only attributes not accessible by [op\(\)](#) should be compared. This is also the reason why this function doesn't take the wildcard replacement list from [match\(\)](#) as an argument: only subexpressions are subject to wildcard matches. Also, this function only needs to be implemented for container classes because [is_equal_same_type\(\)](#) is automatically used instead of [match_same_type\(\)](#) if [nops\(\)](#) == 0.

See also

[basic::match](#)

Reimplemented from [GiNaC::basic](#).

References [GINAC_ASSERT](#), and [serial](#).

6.63.3.20 `return_type()`

```
unsigned GiNaC::function::return_type ( ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::return_types::commutative](#), [GINAC_ASSERT](#), [registered_functions\(\)](#), [GiNaC::function_options::return_type](#), [GiNaC::container_storage< C >::seq](#), [serial](#), and [GiNaC::function_options::use_return_type](#).

6.63.3.21 `return_type_tinfo()`

```
return\_type\_t GiNaC::function::return_type_tinfo ( ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GINAC_ASSERT](#), [registered_functions\(\)](#), [GiNaC::function_options::return_type_tinfo](#), [GiNaC::container_storage< C >::seq](#), [serial](#), and [GiNaC::function_options::use_return_type](#).

6.63.3.22 `pderivative()`

```
ex GiNaC::function::pderivative (
    unsigned diff_param ) const [protected]
```

References [GiNaC::function_options::derivative_f](#), [GiNaC::function_options::derivative_use_exvector_args](#), [GINAC_ASSERT](#), and [GiNaC::function_options::nparams](#).

Referenced by [derivative\(\)](#), and [GiNaC::fderivative::eval\(\)](#).

6.63.3.23 expl_derivative()

```
ex GiNaC::function::expl_derivative (
    const symbol & s ) const [protected]
```

References [GiNaC::function_options::expl_derivative_f](#), [GiNaC::function_options::expl_derivative_use_exvector_args](#), [GINAC_ASSERT](#), and [GiNaC::function_options::nparams](#).

Referenced by [derivative\(\)](#).

6.63.3.24 registered_functions()

```
std::vector< function_options > & GiNaC::function::registered_functions ( ) [static], [protected]
```

Referenced by [archive\(\)](#), [conjugate\(\)](#), [GiNaC::fderivative::do_print\(\)](#), [GiNaC::fderivative::do_print_csrc\(\)](#), [GiNaC::fderivative::do_print_latex\(\)](#), [GiNaC::fderivative::do_print_tree\(\)](#), [GiNaC::fderivative::eval\(\)](#), [eval\(\)](#), [evalf\(\)](#), [expand\(\)](#), [find_function\(\)](#), [get_name\(\)](#), [get_registered_functions\(\)](#), [imag_part\(\)](#), [info\(\)](#), [print\(\)](#), [read_archive\(\)](#), [real_part\(\)](#), [register_new\(\)](#), [return_type\(\)](#), [return_type_tinfo\(\)](#), and [series\(\)](#).

6.63.3.25 lookup_remember_table()

```
bool GiNaC::function::lookup_remember_table (
    ex & result ) const [protected]
```

References [GiNaC::remember_table::remember_tables\(\)](#), and [serial](#).

Referenced by [eval\(\)](#).

6.63.3.26 store_remember_table()

```
void GiNaC::function::store_remember_table (
    ex const & result ) const [protected]
```

References [GiNaC::remember_table::remember_tables\(\)](#), and [serial](#).

Referenced by [eval\(\)](#).

6.63.3.27 power()

```
ex GiNaC::function::power (
    const ex & exp ) const
```

References [GiNaC::status_flags::evaluated](#), [GINAC_ASSERT](#), [GiNaC::function_options::nparams](#), [GiNaC::function_options::power_f](#), and [GiNaC::function_options::power_use_exvector_args](#).

6.63.3.28 register_new()

```
unsigned GiNaC::function::register_new (
    function_options const & opt ) [static]
```

References [GiNaC::function_options::functions_with_same_name](#), [GiNaC::function_options::name](#), [registered_functions\(\)](#), [GiNaC::function_options::remember_assoc_size](#), [GiNaC::function_options::remember_size](#), [GiNaC::function_options::remember_str](#), [GiNaC::remember_table::remember_tables\(\)](#), and [GiNaC::function_options::use_remember](#).

6.63.3.29 find_function()

```
unsigned GiNaC::function::find_function (
    const std::string & name,
    unsigned nparams ) [static]
```

Find serial number of function by name and number of parameters.

Throws exception if function was not found.

References [registered_functions\(\)](#), and [serial](#).

6.63.3.30 get_registered_functions()

```
static std::vector< function_options > GiNaC::function::get_registered_functions ( ) [inline],
[static]
```

References [registered_functions\(\)](#).

6.63.3.31 get_serial()

```
unsigned GiNaC::function::get_serial ( ) const [inline]
```

References [serial](#).

6.63.3.32 get_name()

```
std::string GiNaC::function::get_name ( ) const
```

Return the print name of the function.

References [GINAC_ASSERT](#), [registered_functions\(\)](#), and [serial](#).

6.63.4 Friends And Related Symbol Documentation

6.63.4.1 remember_table_entry

```
friend class remember_table_entry [friend]
```

6.63.5 Member Data Documentation

6.63.5.1 current_serial

```
unsigned GiNaC::function::current_serial = 0 [static]
```

This can be used as a hook for external applications.

Referenced by [eval\(\)](#), [evalf\(\)](#), [expand\(\)](#), [print\(\)](#), and [series\(\)](#).

6.63.5.2 serial

```
unsigned GiNaC::function::serial [protected]
```

Referenced by [archive\(\)](#), [calchash\(\)](#), [conjugate\(\)](#), [GiNaC::fderivative::derivative\(\)](#), [GiNaC::fderivative::do_print\(\)](#), [GiNaC::fderivative::do_print_csrf\(\)](#), [GiNaC::fderivative::do_print_latex\(\)](#), [GiNaC::fderivative::do_print_tree\(\)](#), [GiNaC::fderivative::eval\(\)](#), [eval\(\)](#), [evalf\(\)](#), [expand\(\)](#), [find_function\(\)](#), [get_name\(\)](#), [get_serial\(\)](#), [imag_part\(\)](#), [info\(\)](#), [is_equal_same_type\(\)](#), [lookup_remember_table\(\)](#), [match_same_type\(\)](#), [print\(\)](#), [read_archive\(\)](#), [real_part\(\)](#), [return_type\(\)](#), [return_type_tinfo\(\)](#), [series\(\)](#), [store_remember_table\(\)](#), [GiNaC::fderivative::thiscontainer\(\)](#), [thiscontainer\(\)](#), [GiNaC::fderivative::thiscontainer\(\)](#), and [thiscontainer\(\)](#).

The documentation for this class was generated from the following files:

- [function.h](#)
- [function.cpp](#)

6.64 GiNaC::function_options Class Reference

```
#include <function.h>
```

Public Member Functions

- [function_options](#) ()
- [function_options](#) (std::string const &n, std::string const &tn=std::string())
- [function_options](#) (std::string const &n, unsigned np)
- [~function_options](#) ()
- [void initialize](#) ()
- [function_options & dummy](#) ()
- [function_options & set_name](#) (std::string const &n, std::string const &tn=std::string())
- [function_options & latex_name](#) (std::string const &tn)
- [function_options & eval_func](#) (eval_funcp_1 e)
- [function_options & eval_func](#) (eval_funcp_2 e)
- [function_options & eval_func](#) (eval_funcp_3 e)
- [function_options & eval_func](#) (eval_funcp_4 e)
- [function_options & eval_func](#) (eval_funcp_5 e)
- [function_options & eval_func](#) (eval_funcp_6 e)
- [function_options & eval_func](#) (eval_funcp_7 e)
- [function_options & eval_func](#) (eval_funcp_8 e)
- [function_options & eval_func](#) (eval_funcp_9 e)
- [function_options & eval_func](#) (eval_funcp_10 e)
- [function_options & eval_func](#) (eval_funcp_11 e)
- [function_options & eval_func](#) (eval_funcp_12 e)
- [function_options & eval_func](#) (eval_funcp_13 e)
- [function_options & eval_func](#) (eval_funcp_14 e)
- [function_options & evalf_func](#) (evalf_funcp_1 e)
- [function_options & evalf_func](#) (evalf_funcp_2 e)
- [function_options & evalf_func](#) (evalf_funcp_3 e)
- [function_options & evalf_func](#) (evalf_funcp_4 e)
- [function_options & evalf_func](#) (evalf_funcp_5 e)
- [function_options & evalf_func](#) (evalf_funcp_6 e)
- [function_options & evalf_func](#) (evalf_funcp_7 e)

- Generated by Doxygen

- [function_options](#) & [series_func](#) ([series_funcp_6](#) e)
- [function_options](#) & [series_func](#) ([series_funcp_7](#) e)
- [function_options](#) & [series_func](#) ([series_funcp_8](#) e)
- [function_options](#) & [series_func](#) ([series_funcp_9](#) e)
- [function_options](#) & [series_func](#) ([series_funcp_10](#) e)
- [function_options](#) & [series_func](#) ([series_funcp_11](#) e)
- [function_options](#) & [series_func](#) ([series_funcp_12](#) e)
- [function_options](#) & [series_func](#) ([series_funcp_13](#) e)
- [function_options](#) & [series_func](#) ([series_funcp_14](#) e)
- [function_options](#) & [info_func](#) ([info_funcp_1](#) e)
- [function_options](#) & [info_func](#) ([info_funcp_2](#) e)
- [function_options](#) & [info_func](#) ([info_funcp_3](#) e)
- [function_options](#) & [info_func](#) ([info_funcp_4](#) e)
- [function_options](#) & [info_func](#) ([info_funcp_5](#) e)
- [function_options](#) & [info_func](#) ([info_funcp_6](#) e)
- [function_options](#) & [info_func](#) ([info_funcp_7](#) e)
- [function_options](#) & [info_func](#) ([info_funcp_8](#) e)
- [function_options](#) & [info_func](#) ([info_funcp_9](#) e)
- [function_options](#) & [info_func](#) ([info_funcp_10](#) e)
- [function_options](#) & [info_func](#) ([info_funcp_11](#) e)
- [function_options](#) & [info_func](#) ([info_funcp_12](#) e)
- [function_options](#) & [info_func](#) ([info_funcp_13](#) e)
- [function_options](#) & [info_func](#) ([info_funcp_14](#) e)
- [function_options](#) & [eval_func](#) ([eval_funcp_exvector](#) e)
- [function_options](#) & [evalf_func](#) ([evalf_funcp_exvector](#) e)
- [function_options](#) & [conjugate_func](#) ([conjugate_funcp_exvector](#) e)
- [function_options](#) & [real_part_func](#) ([real_part_funcp_exvector](#) e)
- [function_options](#) & [imag_part_func](#) ([imag_part_funcp_exvector](#) e)
- [function_options](#) & [expand_func](#) ([expand_funcp_exvector](#) e)
- [function_options](#) & [derivative_func](#) ([derivative_funcp_exvector](#) e)
- [function_options](#) & [expl_derivative_func](#) ([expl_derivative_funcp_exvector](#) e)
- [function_options](#) & [power_func](#) ([power_funcp_exvector](#) e)
- [function_options](#) & [series_func](#) ([series_funcp_exvector](#) e)
- [function_options](#) & [info_func](#) ([info_funcp_exvector](#) e)
- [template<class Ctx >](#)
[function_options](#) & [print_func](#) ([print_funcp_1](#) p)
- [template<class Ctx >](#)
[function_options](#) & [print_func](#) ([print_funcp_2](#) p)
- [template<class Ctx >](#)
[function_options](#) & [print_func](#) ([print_funcp_3](#) p)
- [template<class Ctx >](#)
[function_options](#) & [print_func](#) ([print_funcp_4](#) p)
- [template<class Ctx >](#)
[function_options](#) & [print_func](#) ([print_funcp_5](#) p)
- [template<class Ctx >](#)
[function_options](#) & [print_func](#) ([print_funcp_6](#) p)
- [template<class Ctx >](#)
[function_options](#) & [print_func](#) ([print_funcp_7](#) p)
- [template<class Ctx >](#)
[function_options](#) & [print_func](#) ([print_funcp_8](#) p)
- [template<class Ctx >](#)
[function_options](#) & [print_func](#) ([print_funcp_9](#) p)
- [template<class Ctx >](#)
[function_options](#) & [print_func](#) ([print_funcp_10](#) p)

- `template<class Ctx >`
`function_options & print_func (print_funcp_11 p)`
- `template<class Ctx >`
`function_options & print_func (print_funcp_12 p)`
- `template<class Ctx >`
`function_options & print_func (print_funcp_13 p)`
- `template<class Ctx >`
`function_options & print_func (print_funcp_14 p)`
- `template<class Ctx >`
`function_options & print_func (print_funcp_exvector p)`
- `function_options & set_return_type (unsigned rt, const return_type_t *rtt=nullptr)`
- `function_options & do_not_evalf_params ()`
- `function_options & remember (unsigned size, unsigned assoc_size=0, unsigned strategy=remember_strategies::delete_never)`
- `function_options & overloaded (unsigned o)`
- `function_options & set_symmetry (const symmetry &s)`
- `std::string get_name () const`
- `unsigned get_nparams () const`

Protected Member Functions

- `bool has_derivative () const`
- `bool has_power () const`
- `void test_and_set_nparams (unsigned n)`
- `void set_print_func (unsigned id, print_funcp f)`

Protected Attributes

- `std::string name`
- `std::string TeX_name`
- `unsigned nparams`
- `eval_funcp eval_f`
- `evalf_funcp evalf_f`
- `conjugate_funcp conjugate_f`
- `real_part_funcp real_part_f`
- `imag_part_funcp imag_part_f`
- `expand_funcp expand_f`
- `derivative_funcp derivative_f`
- `expl_derivative_funcp expl_derivative_f`
- `power_funcp power_f`
- `series_funcp series_f`
- `std::vector< print_funcp > print_dispatch_table`
- `info_funcp info_f`
- `bool evalf_params_first`
- `bool use_return_type`
- `unsigned return_type`
- `return_type_t return_type_tinfo`
- `bool use_remember`
- `unsigned remember_size`
- `unsigned remember_assoc_size`
- `unsigned remember_strategy`
- `bool eval_use_exvector_args`
- `bool evalf_use_exvector_args`
- `bool conjugate_use_exvector_args`

- [bool real_part_use_exvector_args](#)
- [bool imag_part_use_exvector_args](#)
- [bool expand_use_exvector_args](#)
- [bool derivative_use_exvector_args](#)
- [bool expl_derivative_use_exvector_args](#)
- [bool power_use_exvector_args](#)
- [bool series_use_exvector_args](#)
- [bool print_use_exvector_args](#)
- [bool info_use_exvector_args](#)
- [unsigned functions_with_same_name](#)
- [ex symtree](#)

Friends

- [class function](#)
- [class fderivative](#)

6.64.1 Constructor & Destructor Documentation

6.64.1.1 `function_options()` [1/3]

`GiNaC::function_options::function_options ( )`

References [initialize\(\)](#).

6.64.1.2 `function_options()` [2/3]

```
GiNaC::function_options::function_options (
    std::string const & n,
    std::string const & tn = std::string() )
```

References [initialize\(\)](#), [n](#), [print_func\(\)](#), and [set_name\(\)](#).

6.64.1.3 `function_options()` [3/3]

```
GiNaC::function_options::function_options (
    std::string const & n,
    unsigned np )
```

References [initialize\(\)](#), [n](#), [nparams](#), [print_func\(\)](#), and [set_name\(\)](#).

6.64.1.4 `~function_options()`

`GiNaC::function_options::~~function_options ( )`

6.64.2 Member Function Documentation

6.64.2.1 initialize()

```
void GiNaC::function_options::initialize ( )
```

References [conjugate_f](#), [conjugate_use_exvector_args](#), [derivative_f](#), [derivative_use_exvector_args](#), [eval_f](#), [eval_use_exvector_args](#), [evalf_f](#), [evalf_params_first](#), [evalf_use_exvector_args](#), [expand_f](#), [expand_use_exvector_args](#), [expl_derivative_f](#), [expl_derivative_use_exvector_args](#), [functions_with_same_name](#), [imag_part_f](#), [imag_part_use_exvector_args](#), [info_f](#), [info_use_exvector_args](#), [nparams](#), [power_f](#), [power_use_exvector_args](#), [print_use_exvector_args](#), [real_part_f](#), [real_part_use_exvector_args](#), [series_f](#), [series_use_exvector_args](#), [set_name\(\)](#), [symtree](#), [use_remember](#), and [use_return_type](#).

Referenced by [function_options\(\)](#), [function_options\(\)](#), and [function_options\(\)](#).

6.64.2.2 dummy()

```
function_options & GiNaC::function_options::dummy ( ) [inline]
```

6.64.2.3 set_name()

```
function_options & GiNaC::function_options::set_name (
    std::string const & n,
    std::string const & tn = std::string() )
```

References [n](#), [name](#), [print_func\(\)](#), and [TeX_name](#).

Referenced by [function_options\(\)](#), [function_options\(\)](#), and [initialize\(\)](#).

6.64.2.4 latex_name()

```
function_options & GiNaC::function_options::latex_name (
    std::string const & tn )
```

References [print_func\(\)](#), and [TeX_name](#).

6.64.2.5 eval_func() [1/15]

```
function_options & GiNaC::function_options::eval_func (
    eval_funcp_1 e )
```

References [eval_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.6 eval_func() [2/15]

```
function_options & GiNaC::function_options::eval_func (
    eval_funcp_2 e )
```

References [eval_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.7 eval_func() [3/15]

```
function_options & GiNaC::function_options::eval_func (
    eval_funcp_3 e )
```

References [eval_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.8 eval_func() [4/15]

```
function_options & GiNaC::function_options::eval_func (
    eval_funcp_4 e )
```

References [eval_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.9 eval_func() [5/15]

```
function_options & GiNaC::function_options::eval_func (
    eval_funcp_5 e )
```

References [eval_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.10 eval_func() [6/15]

```
function_options & GiNaC::function_options::eval_func (
    eval_funcp_6 e )
```

References [eval_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.11 eval_func() [7/15]

```
function_options & GiNaC::function_options::eval_func (
    eval_funcp_7 e )
```

References [eval_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.12 eval_func() [8/15]

```
function_options & GiNaC::function_options::eval_func (
    eval_funcp_8 e )
```

References [eval_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.13 eval_func() [9/15]

```
function_options & GiNaC::function_options::eval_func (
    eval_funcp_9 e )
```

References [eval_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.14 eval_func() [10/15]

```
function_options & GiNaC::function_options::eval_func (
    eval_funcp_10 e )
```

References [eval_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.15 eval_func() [11/15]

```
function_options & GiNaC::function_options::eval_func (
    eval_funcp_11 e )
```

References [eval_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.16 eval_func() [12/15]

```
function_options & GiNaC::function_options::eval_func (
    eval_funcp_12 e )
```

References [eval_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.17 eval_func() [13/15]

```
function_options & GiNaC::function_options::eval_func (
    eval_funcp_13 e )
```

References [eval_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.18 eval_func() [14/15]

```
function_options & GiNaC::function_options::eval_func (
    eval_funcp_14 e )
```

References [eval_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.19 evalf_func() [1/15]

```
function_options & GiNaC::function_options::evalf_func (
    evalf_funcp_1 e )
```

References [evalf_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.20 evalf_func() [2/15]

```
function_options & GiNaC::function_options::evalf_func (
    evalf_funcp_2 e )
```

References [evalf_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.21 evalf_func() [3/15]

```
function_options & GiNaC::function_options::evalf_func (
    evalf_funcp_3 e )
```

References [evalf_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.22 evalf_func() [4/15]

```
function_options & GiNaC::function_options::evalf_func (
    evalf_funcp_4 e )
```

References [evalf_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.23 evalf_func() [5/15]

```
function_options & GiNaC::function_options::evalf_func (
    evalf_funcp_5 e )
```

References [evalf_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.24 evalf_func() [6/15]

```
function_options & GiNaC::function_options::evalf_func (
    evalf_funcp_6 e )
```

References [evalf_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.25 evalf_func() [7/15]

```
function_options & GiNaC::function_options::evalf_func (
    evalf_funcp_7 e )
```

References [evalf_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.26 evalf_func() [8/15]

```
function_options & GiNaC::function_options::evalf_func (
    evalf_funcp_8 e )
```

References [evalf_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.27 evalf_func() [9/15]

```
function_options & GiNaC::function_options::evalf_func (
    evalf_funcp_9 e )
```

References [evalf_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.28 evalf_func() [10/15]

```
function_options & GiNaC::function_options::evalf_func (
    evalf_funcp_10 e )
```

References [evalf_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.29 evalf_func() [11/15]

```
function_options & GiNaC::function_options::evalf_func (
    evalf_funcp_11 e )
```

References [evalf_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.30 evalf_func() [12/15]

```
function_options & GiNaC::function_options::evalf_func (
    evalf_funcp_12 e )
```

References [evalf_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.31 evalf_func() [13/15]

```
function_options & GiNaC::function_options::evalf_func (
    evalf_funcp_13 e )
```

References [evalf_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.32 evalf_func() [14/15]

```
function_options & GiNaC::function_options::evalf_func (
    evalf_funcp_14 e )
```

References [evalf_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.33 conjugate_func() [1/15]

```
function_options & GiNaC::function_options::conjugate_func (
    conjugate_funcp_1 e )
```

References [conjugate_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.34 conjugate_func() [2/15]

```
function_options & GiNaC::function_options::conjugate_func (
    conjugate_funcp_2 e )
```

References [conjugate_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.35 conjugate_func() [3/15]

```
function_options & GiNaC::function_options::conjugate_func (
    conjugate_funcp_3 e )
```

References [conjugate_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.36 conjugate_func() [4/15]

```
function_options & GiNaC::function_options::conjugate_func (
    conjugate_funcp_4 e )
```

References [conjugate_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.37 conjugate_func() [5/15]

```
function_options & GiNaC::function_options::conjugate_func (
    conjugate_funcp_5 e )
```

References [conjugate_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.38 conjugate_func() [6/15]

```
function_options & GiNaC::function_options::conjugate_func (
    conjugate_funcp_6 e )
```

References [conjugate_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.39 conjugate_func() [7/15]

```
function_options & GiNaC::function_options::conjugate_func (
    conjugate_funcp_7 e )
```

References [conjugate_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.40 conjugate_func() [8/15]

```
function_options & GiNaC::function_options::conjugate_func (
    conjugate_funcp_8 e )
```

References [conjugate_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.41 conjugate_func() [9/15]

```
function_options & GiNaC::function_options::conjugate_func (
    conjugate_funcp_9 e )
```

References [conjugate_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.42 conjugate_func() [10/15]

```
function_options & GiNaC::function_options::conjugate_func (  
    conjugate_funcp_10 e )
```

References [conjugate_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.43 conjugate_func() [11/15]

```
function_options & GiNaC::function_options::conjugate_func (  
    conjugate_funcp_11 e )
```

References [conjugate_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.44 conjugate_func() [12/15]

```
function_options & GiNaC::function_options::conjugate_func (  
    conjugate_funcp_12 e )
```

References [conjugate_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.45 conjugate_func() [13/15]

```
function_options & GiNaC::function_options::conjugate_func (  
    conjugate_funcp_13 e )
```

References [conjugate_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.46 conjugate_func() [14/15]

```
function_options & GiNaC::function_options::conjugate_func (  
    conjugate_funcp_14 e )
```

References [conjugate_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.47 real_part_func() [1/15]

```
function_options & GiNaC::function_options::real_part_func (  
    real_part_funcp_1 e )
```

References [real_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.48 real_part_func() [2/15]

```
function_options & GiNaC::function_options::real_part_func (  
    real_part_funcp_2 e )
```

References [real_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.49 real_part_func() [3/15]

```
function_options & GiNaC::function_options::real_part_func (
    real_part_funcp_3 e )
```

References [real_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.50 real_part_func() [4/15]

```
function_options & GiNaC::function_options::real_part_func (
    real_part_funcp_4 e )
```

References [real_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.51 real_part_func() [5/15]

```
function_options & GiNaC::function_options::real_part_func (
    real_part_funcp_5 e )
```

References [real_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.52 real_part_func() [6/15]

```
function_options & GiNaC::function_options::real_part_func (
    real_part_funcp_6 e )
```

References [real_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.53 real_part_func() [7/15]

```
function_options & GiNaC::function_options::real_part_func (
    real_part_funcp_7 e )
```

References [real_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.54 real_part_func() [8/15]

```
function_options & GiNaC::function_options::real_part_func (
    real_part_funcp_8 e )
```

References [real_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.55 real_part_func() [9/15]

```
function_options & GiNaC::function_options::real_part_func (
    real_part_funcp_9 e )
```

References [real_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.56 real_part_func() [10/15]

```
function_options & GiNaC::function_options::real_part_func (
    real_part_funcp_10 e )
```

References [real_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.57 real_part_func() [11/15]

```
function_options & GiNaC::function_options::real_part_func (
    real_part_funcp_11 e )
```

References [real_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.58 real_part_func() [12/15]

```
function_options & GiNaC::function_options::real_part_func (
    real_part_funcp_12 e )
```

References [real_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.59 real_part_func() [13/15]

```
function_options & GiNaC::function_options::real_part_func (
    real_part_funcp_13 e )
```

References [real_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.60 real_part_func() [14/15]

```
function_options & GiNaC::function_options::real_part_func (
    real_part_funcp_14 e )
```

References [real_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.61 imag_part_func() [1/15]

```
function_options & GiNaC::function_options::imag_part_func (
    imag_part_funcp_1 e )
```

References [imag_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.62 imag_part_func() [2/15]

```
function_options & GiNaC::function_options::imag_part_func (
    imag_part_funcp_2 e )
```

References [imag_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.63 imag_part_func() [3/15]

```
function_options & GiNaC::function_options::imag_part_func (
    imag_part_funcp_3 e )
```

References [imag_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.64 imag_part_func() [4/15]

```
function_options & GiNaC::function_options::imag_part_func (
    imag_part_funcp_4 e )
```

References [imag_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.65 imag_part_func() [5/15]

```
function_options & GiNaC::function_options::imag_part_func (
    imag_part_funcp_5 e )
```

References [imag_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.66 imag_part_func() [6/15]

```
function_options & GiNaC::function_options::imag_part_func (
    imag_part_funcp_6 e )
```

References [imag_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.67 imag_part_func() [7/15]

```
function_options & GiNaC::function_options::imag_part_func (
    imag_part_funcp_7 e )
```

References [imag_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.68 imag_part_func() [8/15]

```
function_options & GiNaC::function_options::imag_part_func (
    imag_part_funcp_8 e )
```

References [imag_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.69 imag_part_func() [9/15]

```
function_options & GiNaC::function_options::imag_part_func (
    imag_part_funcp_9 e )
```

References [imag_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.70 imag_part_func() [10/15]

```
function_options & GiNaC::function_options::imag_part_func (
    imag_part_funcp_10 e )
```

References [imag_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.71 imag_part_func() [11/15]

```
function_options & GiNaC::function_options::imag_part_func (
    imag_part_funcp_11 e )
```

References [imag_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.72 imag_part_func() [12/15]

```
function_options & GiNaC::function_options::imag_part_func (
    imag_part_funcp_12 e )
```

References [imag_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.73 imag_part_func() [13/15]

```
function_options & GiNaC::function_options::imag_part_func (
    imag_part_funcp_13 e )
```

References [imag_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.74 imag_part_func() [14/15]

```
function_options & GiNaC::function_options::imag_part_func (
    imag_part_funcp_14 e )
```

References [imag_part_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.75 expand_func() [1/15]

```
function_options & GiNaC::function_options::expand_func (
    expand_funcp_1 e )
```

References [expand_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.76 expand_func() [2/15]

```
function_options & GiNaC::function_options::expand_func (
    expand_funcp_2 e )
```

References [expand_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.77 expand_func() [3/15]

```
function_options & GiNaC::function_options::expand_func (
    expand_funcp_3 e )
```

References [expand_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.78 expand_func() [4/15]

```
function_options & GiNaC::function_options::expand_func (
    expand_funcp_4 e )
```

References [expand_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.79 expand_func() [5/15]

```
function_options & GiNaC::function_options::expand_func (
    expand_funcp_5 e )
```

References [expand_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.80 expand_func() [6/15]

```
function_options & GiNaC::function_options::expand_func (
    expand_funcp_6 e )
```

References [expand_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.81 expand_func() [7/15]

```
function_options & GiNaC::function_options::expand_func (
    expand_funcp_7 e )
```

References [expand_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.82 expand_func() [8/15]

```
function_options & GiNaC::function_options::expand_func (
    expand_funcp_8 e )
```

References [expand_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.83 expand_func() [9/15]

```
function_options & GiNaC::function_options::expand_func (
    expand_funcp_9 e )
```

References [expand_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.84 expand_func() [10/15]

```
function_options & GiNaC::function_options::expand_func (
    expand_funcp_10 e )
```

References [expand_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.85 expand_func() [11/15]

```
function_options & GiNaC::function_options::expand_func (
    expand_funcp_11 e )
```

References [expand_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.86 expand_func() [12/15]

```
function_options & GiNaC::function_options::expand_func (
    expand_funcp_12 e )
```

References [expand_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.87 expand_func() [13/15]

```
function_options & GiNaC::function_options::expand_func (
    expand_funcp_13 e )
```

References [expand_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.88 expand_func() [14/15]

```
function_options & GiNaC::function_options::expand_func (
    expand_funcp_14 e )
```

References [expand_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.89 derivative_func() [1/15]

```
function_options & GiNaC::function_options::derivative_func (
    derivative_funcp_1 e )
```

References [derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.90 derivative_func() [2/15]

```
function_options & GiNaC::function_options::derivative_func (
    derivative_funcp_2 e )
```

References [derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.91 derivative_func() [3/15]

```
function_options & GiNaC::function_options::derivative_func (  
    derivative_funcp_3 e )
```

References [derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.92 derivative_func() [4/15]

```
function_options & GiNaC::function_options::derivative_func (  
    derivative_funcp_4 e )
```

References [derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.93 derivative_func() [5/15]

```
function_options & GiNaC::function_options::derivative_func (  
    derivative_funcp_5 e )
```

References [derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.94 derivative_func() [6/15]

```
function_options & GiNaC::function_options::derivative_func (  
    derivative_funcp_6 e )
```

References [derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.95 derivative_func() [7/15]

```
function_options & GiNaC::function_options::derivative_func (  
    derivative_funcp_7 e )
```

References [derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.96 derivative_func() [8/15]

```
function_options & GiNaC::function_options::derivative_func (  
    derivative_funcp_8 e )
```

References [derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.97 derivative_func() [9/15]

```
function_options & GiNaC::function_options::derivative_func (  
    derivative_funcp_9 e )
```

References [derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.98 derivative_func() [10/15]

```
function_options & GiNaC::function_options::derivative_func (  
    derivative_funcp_10 e )
```

References [derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.99 derivative_func() [11/15]

```
function_options & GiNaC::function_options::derivative_func (  
    derivative_funcp_11 e )
```

References [derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.100 derivative_func() [12/15]

```
function_options & GiNaC::function_options::derivative_func (  
    derivative_funcp_12 e )
```

References [derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.101 derivative_func() [13/15]

```
function_options & GiNaC::function_options::derivative_func (  
    derivative_funcp_13 e )
```

References [derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.102 derivative_func() [14/15]

```
function_options & GiNaC::function_options::derivative_func (  
    derivative_funcp_14 e )
```

References [derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.103 expl_derivative_func() [1/15]

```
function_options & GiNaC::function_options::expl_derivative_func (  
    expl_derivative_funcp_1 e )
```

References [expl_derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.104 expl_derivative_func() [2/15]

```
function_options & GiNaC::function_options::expl_derivative_func (  
    expl_derivative_funcp_2 e )
```

References [expl_derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.105 expl_derivative_func() [3/15]

```
function_options & GiNaC::function_options::expl_derivative_func (
    expl_derivative_funcp_3 e )
```

References [expl_derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.106 expl_derivative_func() [4/15]

```
function_options & GiNaC::function_options::expl_derivative_func (
    expl_derivative_funcp_4 e )
```

References [expl_derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.107 expl_derivative_func() [5/15]

```
function_options & GiNaC::function_options::expl_derivative_func (
    expl_derivative_funcp_5 e )
```

References [expl_derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.108 expl_derivative_func() [6/15]

```
function_options & GiNaC::function_options::expl_derivative_func (
    expl_derivative_funcp_6 e )
```

References [expl_derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.109 expl_derivative_func() [7/15]

```
function_options & GiNaC::function_options::expl_derivative_func (
    expl_derivative_funcp_7 e )
```

References [expl_derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.110 expl_derivative_func() [8/15]

```
function_options & GiNaC::function_options::expl_derivative_func (
    expl_derivative_funcp_8 e )
```

References [expl_derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.111 expl_derivative_func() [9/15]

```
function_options & GiNaC::function_options::expl_derivative_func (
    expl_derivative_funcp_9 e )
```

References [expl_derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.112 expl_derivative_func() [10/15]

```
function_options & GiNaC::function_options::expl_derivative_func (
    expl_derivative_funcp_10 e )
```

References [expl_derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.113 expl_derivative_func() [11/15]

```
function_options & GiNaC::function_options::expl_derivative_func (
    expl_derivative_funcp_11 e )
```

References [expl_derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.114 expl_derivative_func() [12/15]

```
function_options & GiNaC::function_options::expl_derivative_func (
    expl_derivative_funcp_12 e )
```

References [expl_derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.115 expl_derivative_func() [13/15]

```
function_options & GiNaC::function_options::expl_derivative_func (
    expl_derivative_funcp_13 e )
```

References [expl_derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.116 expl_derivative_func() [14/15]

```
function_options & GiNaC::function_options::expl_derivative_func (
    expl_derivative_funcp_14 e )
```

References [expl_derivative_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.117 power_func() [1/15]

```
function_options & GiNaC::function_options::power_func (
    power_funcp_1 e )
```

References [power_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.118 power_func() [2/15]

```
function_options & GiNaC::function_options::power_func (
    power_funcp_2 e )
```

References [power_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.119 power_func() [3/15]

```
function_options & GiNaC::function_options::power_func (
    power_funcp_3 e )
```

References [power_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.120 power_func() [4/15]

```
function_options & GiNaC::function_options::power_func (
    power_funcp_4 e )
```

References [power_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.121 power_func() [5/15]

```
function_options & GiNaC::function_options::power_func (
    power_funcp_5 e )
```

References [power_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.122 power_func() [6/15]

```
function_options & GiNaC::function_options::power_func (
    power_funcp_6 e )
```

References [power_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.123 power_func() [7/15]

```
function_options & GiNaC::function_options::power_func (
    power_funcp_7 e )
```

References [power_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.124 power_func() [8/15]

```
function_options & GiNaC::function_options::power_func (
    power_funcp_8 e )
```

References [power_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.125 power_func() [9/15]

```
function_options & GiNaC::function_options::power_func (
    power_funcp_9 e )
```

References [power_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.126 power_func() [10/15]

```
function_options & GiNaC::function_options::power_func (
    power_funcp_10 e )
```

References [power_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.127 power_func() [11/15]

```
function_options & GiNaC::function_options::power_func (
    power_funcp_11 e )
```

References [power_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.128 power_func() [12/15]

```
function_options & GiNaC::function_options::power_func (
    power_funcp_12 e )
```

References [power_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.129 power_func() [13/15]

```
function_options & GiNaC::function_options::power_func (
    power_funcp_13 e )
```

References [power_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.130 power_func() [14/15]

```
function_options & GiNaC::function_options::power_func (
    power_funcp_14 e )
```

References [power_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.131 series_func() [1/15]

```
function_options & GiNaC::function_options::series_func (
    series_funcp_1 e )
```

References [series_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.132 series_func() [2/15]

```
function_options & GiNaC::function_options::series_func (
    series_funcp_2 e )
```

References [series_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.133 series_func() [3/15]

```
function_options & GiNaC::function_options::series_func (
    series_funcp_3 e )
```

References [series_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.134 series_func() [4/15]

```
function_options & GiNaC::function_options::series_func (
    series_funcp_4 e )
```

References [series_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.135 series_func() [5/15]

```
function_options & GiNaC::function_options::series_func (
    series_funcp_5 e )
```

References [series_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.136 series_func() [6/15]

```
function_options & GiNaC::function_options::series_func (
    series_funcp_6 e )
```

References [series_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.137 series_func() [7/15]

```
function_options & GiNaC::function_options::series_func (
    series_funcp_7 e )
```

References [series_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.138 series_func() [8/15]

```
function_options & GiNaC::function_options::series_func (
    series_funcp_8 e )
```

References [series_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.139 series_func() [9/15]

```
function_options & GiNaC::function_options::series_func (
    series_funcp_9 e )
```

References [series_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.140 series_func() [10/15]

```
function_options & GiNaC::function_options::series_func (
    series_funcp_10 e )
```

References [series_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.141 series_func() [11/15]

```
function_options & GiNaC::function_options::series_func (
    series_funcp_11 e )
```

References [series_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.142 series_func() [12/15]

```
function_options & GiNaC::function_options::series_func (
    series_funcp_12 e )
```

References [series_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.143 series_func() [13/15]

```
function_options & GiNaC::function_options::series_func (
    series_funcp_13 e )
```

References [series_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.144 series_func() [14/15]

```
function_options & GiNaC::function_options::series_func (
    series_funcp_14 e )
```

References [series_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.145 info_func() [1/15]

```
function_options & GiNaC::function_options::info_func (
    info_funcp_1 e )
```

References [info_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.146 info_func() [2/15]

```
function_options & GiNaC::function_options::info_func (
    info_funcp_2 e )
```

References [info_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.147 info_func() [3/15]

```
function_options & GiNaC::function_options::info_func (  
    info_funcp_3 e )
```

References [info_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.148 info_func() [4/15]

```
function_options & GiNaC::function_options::info_func (  
    info_funcp_4 e )
```

References [info_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.149 info_func() [5/15]

```
function_options & GiNaC::function_options::info_func (  
    info_funcp_5 e )
```

References [info_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.150 info_func() [6/15]

```
function_options & GiNaC::function_options::info_func (  
    info_funcp_6 e )
```

References [info_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.151 info_func() [7/15]

```
function_options & GiNaC::function_options::info_func (  
    info_funcp_7 e )
```

References [info_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.152 info_func() [8/15]

```
function_options & GiNaC::function_options::info_func (  
    info_funcp_8 e )
```

References [info_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.153 info_func() [9/15]

```
function_options & GiNaC::function_options::info_func (  
    info_funcp_9 e )
```

References [info_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.154 info_func() [10/15]

```
function_options & GiNaC::function_options::info_func (  
    info_funcp_10 e )
```

References [info_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.155 info_func() [11/15]

```
function_options & GiNaC::function_options::info_func (  
    info_funcp_11 e )
```

References [info_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.156 info_func() [12/15]

```
function_options & GiNaC::function_options::info_func (  
    info_funcp_12 e )
```

References [info_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.157 info_func() [13/15]

```
function_options & GiNaC::function_options::info_func (  
    info_funcp_13 e )
```

References [info_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.158 info_func() [14/15]

```
function_options & GiNaC::function_options::info_func (  
    info_funcp_14 e )
```

References [info_f](#), and [test_and_set_nparams\(\)](#).

6.64.2.159 eval_func() [15/15]

```
function_options & GiNaC::function_options::eval_func (  
    eval_funcp_exvector e )
```

References [eval_f](#), and [eval_use_exvector_args](#).

6.64.2.160 evalf_func() [15/15]

```
function_options & GiNaC::function_options::evalf_func (  
    evalf_funcp_exvector e )
```

References [evalf_f](#), and [evalf_use_exvector_args](#).

6.64.2.161 conjugate_func() [15/15]

```
function_options & GiNaC::function_options::conjugate_func (
    conjugate_funcp_exvector e )
```

References [conjugate_f](#), and [conjugate_use_exvector_args](#).

6.64.2.162 real_part_func() [15/15]

```
function_options & GiNaC::function_options::real_part_func (
    real_part_funcp_exvector e )
```

References [real_part_f](#), and [real_part_use_exvector_args](#).

6.64.2.163 imag_part_func() [15/15]

```
function_options & GiNaC::function_options::imag_part_func (
    imag_part_funcp_exvector e )
```

References [imag_part_f](#), and [imag_part_use_exvector_args](#).

6.64.2.164 expand_func() [15/15]

```
function_options & GiNaC::function_options::expand_func (
    expand_funcp_exvector e )
```

References [expand_f](#), and [expand_use_exvector_args](#).

6.64.2.165 derivative_func() [15/15]

```
function_options & GiNaC::function_options::derivative_func (
    derivative_funcp_exvector e )
```

References [derivative_f](#), and [derivative_use_exvector_args](#).

6.64.2.166 expl_derivative_func() [15/15]

```
function_options & GiNaC::function_options::expl_derivative_func (
    expl_derivative_funcp_exvector e )
```

References [expl_derivative_f](#), and [expl_derivative_use_exvector_args](#).

6.64.2.167 power_func() [15/15]

```
function_options & GiNaC::function_options::power_func (
    power_funcp_exvector e )
```

References [power_f](#), and [power_use_exvector_args](#).

6.64.2.168 series_func() [15/15]

```
function_options & GiNaC::function_options::series_func (
    series_funcp_exvector e )
```

References [series_f](#), and [series_use_exvector_args](#).

6.64.2.169 info_func() [15/15]

```
function_options & GiNaC::function_options::info_func (
    info_funcp_exvector e )
```

References [info_f](#), and [info_use_exvector_args](#).

6.64.2.170 print_func() [1/15]

```
template<class Ctx >
function_options & GiNaC::function_options::print_func (
    print_funcp_1 p ) [inline]
```

References [options](#), [set_print_func\(\)](#), and [test_and_set_nparams\(\)](#).

Referenced by [function_options\(\)](#), [function_options\(\)](#), [latex_name\(\)](#), [remember\(\)](#), [set_name\(\)](#), and [set_return_type\(\)](#).

6.64.2.171 print_func() [2/15]

```
template<class Ctx >
function_options & GiNaC::function_options::print_func (
    print_funcp_2 p ) [inline]
```

References [options](#), [set_print_func\(\)](#), and [test_and_set_nparams\(\)](#).

6.64.2.172 print_func() [3/15]

```
template<class Ctx >
function_options & GiNaC::function_options::print_func (
    print_funcp_3 p ) [inline]
```

References [options](#), [set_print_func\(\)](#), and [test_and_set_nparams\(\)](#).

6.64.2.173 print_func() [4/15]

```
template<class Ctx >
function_options & GiNaC::function_options::print_func (
    print_funcp_4 p ) [inline]
```

References [options](#), [set_print_func\(\)](#), and [test_and_set_nparams\(\)](#).

6.64.2.174 print_func() [5/15]

```
template<class Ctx >
function_options & GiNaC::function_options::print_func (
    print_funcp_5 p ) [inline]
```

References [options](#), [set_print_func\(\)](#), and [test_and_set_nparams\(\)](#).

6.64.2.175 print_func() [6/15]

```
template<class Ctx >
function_options & GiNaC::function_options::print_func (
    print_funcp_6 p ) [inline]
```

References [options](#), [set_print_func\(\)](#), and [test_and_set_nparams\(\)](#).

6.64.2.176 print_func() [7/15]

```
template<class Ctx >
function_options & GiNaC::function_options::print_func (
    print_funcp_7 p ) [inline]
```

References [options](#), [set_print_func\(\)](#), and [test_and_set_nparams\(\)](#).

6.64.2.177 print_func() [8/15]

```
template<class Ctx >
function_options & GiNaC::function_options::print_func (
    print_funcp_8 p ) [inline]
```

References [options](#), [set_print_func\(\)](#), and [test_and_set_nparams\(\)](#).

6.64.2.178 print_func() [9/15]

```
template<class Ctx >
function_options & GiNaC::function_options::print_func (
    print_funcp_9 p ) [inline]
```

References [options](#), [set_print_func\(\)](#), and [test_and_set_nparams\(\)](#).

6.64.2.179 print_func() [10/15]

```
template<class Ctx >
function_options & GiNaC::function_options::print_func (
    print_funcp_10 p ) [inline]
```

References [options](#), [set_print_func\(\)](#), and [test_and_set_nparams\(\)](#).

6.64.2.180 print_func() [11/15]

```
template<class Ctx >
function_options & GiNaC::function_options::print_func (
    print_funcp_11 p ) [inline]
```

References [options](#), [set_print_func\(\)](#), and [test_and_set_nparams\(\)](#).

6.64.2.181 print_func() [12/15]

```
template<class Ctx >
function_options & GiNaC::function_options::print_func (
    print_funcp_12 p ) [inline]
```

References [options](#), [set_print_func\(\)](#), and [test_and_set_nparams\(\)](#).

6.64.2.182 print_func() [13/15]

```
template<class Ctx >
function_options & GiNaC::function_options::print_func (
    print_funcp_13 p ) [inline]
```

References [options](#), [set_print_func\(\)](#), and [test_and_set_nparams\(\)](#).

6.64.2.183 print_func() [14/15]

```
template<class Ctx >
function_options & GiNaC::function_options::print_func (
    print_funcp_14 p ) [inline]
```

References [options](#), [set_print_func\(\)](#), and [test_and_set_nparams\(\)](#).

6.64.2.184 print_func() [15/15]

```
template<class Ctx >
function_options & GiNaC::function_options::print_func (
    print_funcp_exvector p ) [inline]
```

References [options](#), [print_use_exvector_args](#), and [set_print_func\(\)](#).

6.64.2.185 set_return_type()

```
function_options & GiNaC::function_options::set_return_type (
    unsigned rt,
    const return_type_t * rtt = nullptr )
```

References [print_func\(\)](#), [return_type](#), [return_type_tinfo](#), and [use_return_type](#).

6.64.2.186 do_not_evalf_params()

```
function_options & GiNaC::function_options::do_not_evalf_params ( )
```

References [evalf_params_first](#).

6.64.2.187 remember()

```
function_options & GiNaC::function_options::remember (
    unsigned size,
    unsigned assoc_size = 0,
    unsigned strategy = remember_strategies::delete_never )
```

References [print_func\(\)](#), [remember_assoc_size](#), [remember_size](#), [remember_strategy](#), and [use_remember](#).

6.64.2.188 overloaded()

```
function_options & GiNaC::function_options::overloaded (
    unsigned o )
```

References [functions_with_same_name](#).

6.64.2.189 set_symmetry()

```
function_options & GiNaC::function_options::set_symmetry (
    const symmetry & s )
```

References [symtree](#).

6.64.2.190 get_name()

```
std::string GiNaC::function_options::get_name ( ) const [inline]
```

References [name](#).

6.64.2.191 get_nparams()

```
unsigned GiNaC::function_options::get_nparams ( ) const [inline]
```

References [nparams](#).

6.64.2.192 has_derivative()

```
bool GiNaC::function_options::has_derivative ( ) const [inline], [protected]
```

References [derivative_f](#).

```
bool GiNaC::function_options::has_power ( ) const [inline], [protected]
```

6.64.2.194 test_and_set_nparams()

References `n`, `name`, and `nparams`.

6.64.2.195 set_print_func()

References [print_dispatch_table.](#)

6.64.3 Friends And Related Symbol Documentation

```
friend class function [friend]
```

6.64.3.2 fderivative

```
friend class fderivative [friend]
```

6.64.4 Member Data Documentation

6.64.4.1 name

```
std::string GiNaC::function_options::name [protected]
```

Referenced by [get_name\(\)](#), [GiNaC::function::print\(\)](#), [GiNaC::function::register_new\(\)](#), [set_name\(\)](#), and [test_and_set_nparams\(\)](#).

6.64.4.2 TeX_name

```
std::string GiNaC::function_options::TeX_name [protected]
```

Referenced by [latex_name\(\)](#), [GiNaC::function::print\(\)](#), and [set_name\(\)](#).

6.64.4.3 nparams

```
unsigned GiNaC::function_options::nparams [protected]
```

Referenced by [GiNaC::function::conjugate\(\)](#), [GiNaC::function::eval\(\)](#), [GiNaC::function::evalf\(\)](#), [GiNaC::function::expand\(\)](#), [GiNaC::function::expl_derivative\(\)](#), [function_options\(\)](#), [get_nparams\(\)](#), [GiNaC::function::imag_part\(\)](#), [GiNaC::function::info\(\)](#), [initialize\(\)](#), [GiNaC::function::pderivative\(\)](#), [GiNaC::function::power\(\)](#), [GiNaC::function::print\(\)](#), [GiNaC::function::real_part\(\)](#), [GiNaC::function::series\(\)](#), and [test_and_set_nparams\(\)](#).

6.64.4.4 eval_f

```
eval_funcp GiNaC::function_options::eval_f [protected]
```

Referenced by [GiNaC::function::eval\(\)](#), [eval_func\(\)](#), [eval_func\(\)](#), [eval_func\(\)](#), [eval_func\(\)](#), [eval_func\(\)](#), [eval_func\(\)](#), [eval_func\(\)](#), [eval_func\(\)](#), [eval_func\(\)](#), [eval_func\(\)](#), [eval_func\(\)](#), [eval_func\(\)](#), and [initialize\(\)](#).

6.64.4.5 evalf_f

```
evalf_funcp GiNaC::function_options::evalf_f [protected]
```

Referenced by [GiNaC::function::evalf\(\)](#), [evalf_func\(\)](#), [evalf_func\(\)](#), [evalf_func\(\)](#), [evalf_func\(\)](#), [evalf_func\(\)](#), [evalf_func\(\)](#), [evalf_func\(\)](#), [evalf_func\(\)](#), [evalf_func\(\)](#), [evalf_func\(\)](#), [evalf_func\(\)](#), [evalf_func\(\)](#), and [initialize\(\)](#).

6.64.4.6 conjugate_f

`conjugate_funcp` GiNaC::function_options::conjugate_f [protected]

Referenced by `GiNaC::function::conjugate()`, `conjugate_func()`, `conjugate_func()`, `conjugate_func()`, `conjugate_func()`, `conjugate_func()`, `conjugate_func()`, `conjugate_func()`, `conjugate_func()`, `conjugate_func()`, `conjugate_func()`, `conjugate_func()`, `conjugate_func()`, and `initialize()`.

6.64.4.7 real_part_f

`real_part_funcp` GiNaC::function_options::real_part_f [protected]

Referenced by `initialize()`, `GiNaC::function::real_part()`, `real_part_func()`, `real_part_func()`, `real_part_func()`, `real_part_func()`, `real_part_func()`, `real_part_func()`, `real_part_func()`, `real_part_func()`, `real_part_func()`, `real_part_func()`, `real_part_func()`, and `real_part_func()`.

6.64.4.8 imag_part_f

`imag_part_funcp` GiNaC::function_options::imag_part_f [protected]

Referenced by `GiNaC::function::imag_part()`, `imag_part_func()`, `imag_part_func()`, `imag_part_func()`, `imag_part_func()`, `imag_part_func()`, `imag_part_func()`, `imag_part_func()`, `imag_part_func()`, `imag_part_func()`, `imag_part_func()`, `imag_part_func()`, `imag_part_func()`, and `initialize()`.

6.64.4.9 expand_f

`expand_funcp` GiNaC::function_options::expand_f [protected]

Referenced by `GiNaC::function::expand()`, `expand_func()`, `expand_func()`, `expand_func()`, `expand_func()`, `expand_func()`, `expand_func()`, `expand_func()`, `expand_func()`, `expand_func()`, `expand_func()`, `expand_func()`, `expand_func()`, and `initialize()`.

6.64.4.10 derivative_f

`derivative_funcp` GiNaC::function_options::derivative_f [protected]

Referenced by `derivative_func()`, `derivative_func()`, `derivative_func()`, `derivative_func()`, `derivative_func()`, `derivative_func()`, `derivative_func()`, `derivative_func()`, `derivative_func()`, `derivative_func()`, `derivative_func()`, `derivative_func()`, `derivative_func()`, `has_derivative()`, `initialize()`, and `GiNaC::function::pderivative()`.

6.64.4.11 expl_derivative_f

`expl_derivative_funcp` GiNaC::function_options::expl_derivative_f [protected]

Referenced by `GiNaC::function::expl_derivative()`, `expl_derivative_func()`, `expl_derivative_func()`, `expl_derivative_func()`, `expl_derivative_func()`, `expl_derivative_func()`, `expl_derivative_func()`, `expl_derivative_func()`, `expl_derivative_func()`, `expl_derivative_func()`, `expl_derivative_func()`, `expl_derivative_func()`, `expl_derivative_func()`, and `initialize()`.

6.64.4.12 power_f

`power_funcp` `GiNaC::function_options::power_f` [protected]

Referenced by `has_power()`, `initialize()`, `GiNaC::function::power()`, `power_func()`, `power_func()`, `power_func()`, `power_func()`, `power_func()`, `power_func()`, `power_func()`, `power_func()`, `power_func()`, `power_func()`, `power_func()`, `power_func()`, and `power_func()`.

6.64.4.13 series_f

`series_funcp` `GiNaC::function_options::series_f` [protected]

Referenced by `initialize()`, `GiNaC::function::series()`, `series_func()`, `series_func()`, `series_func()`, `series_func()`, `series_func()`, `series_func()`, `series_func()`, `series_func()`, `series_func()`, `series_func()`, `series_func()`, `series_func()`, `series_func()`, and `series_func()`.

6.64.4.14 print_dispatch_table

`std::vector<print_funcp>` `GiNaC::function_options::print_dispatch_table` [protected]

Referenced by `GiNaC::function::print()`, and `set_print_func()`.

6.64.4.15 info_f

`info_funcp` `GiNaC::function_options::info_f` [protected]

Referenced by `GiNaC::function::info()`, `info_func()`, `info_func()`, `info_func()`, `info_func()`, `info_func()`, `info_func()`, `info_func()`, `info_func()`, `info_func()`, `info_func()`, `info_func()`, `info_func()`, `info_func()`, and `initialize()`.

6.64.4.16 evalf_params_first

`bool` `GiNaC::function_options::evalf_params_first` [protected]

Referenced by `do_not_evalf_params()`, `GiNaC::function::evalf()`, and `initialize()`.

6.64.4.17 use_return_type

`bool` `GiNaC::function_options::use_return_type` [protected]

Referenced by `initialize()`, `GiNaC::function::return_type()`, `GiNaC::function::return_type_tinfo()`, and `set_return_type()`.

6.64.4.18 return_type

`unsigned` `GiNaC::function_options::return_type` [protected]

Referenced by `GiNaC::function::return_type()`, and `set_return_type()`.

6.64.4.19 return_type_tinfo

`return_type_t` GiNaC::function_options::return_type_tinfo [protected]

Referenced by [GiNaC::function::return_type_tinfo\(\)](#), and [set_return_type\(\)](#).

6.64.4.20 use_remember

`bool` GiNaC::function_options::use_remember [protected]

Referenced by [GiNaC::function::eval\(\)](#), [initialize\(\)](#), [GiNaC::function::register_new\(\)](#), and [remember\(\)](#).

6.64.4.21 remember_size

`unsigned` GiNaC::function_options::remember_size [protected]

Referenced by [GiNaC::function::register_new\(\)](#), and [remember\(\)](#).

6.64.4.22 remember_assoc_size

`unsigned` GiNaC::function_options::remember_assoc_size [protected]

Referenced by [GiNaC::function::register_new\(\)](#), and [remember\(\)](#).

6.64.4.23 remember_strategy

`unsigned` GiNaC::function_options::remember_strategy [protected]

Referenced by [GiNaC::function::register_new\(\)](#), and [remember\(\)](#).

6.64.4.24 eval_use_exvector_args

`bool` GiNaC::function_options::eval_use_exvector_args [protected]

Referenced by [GiNaC::function::eval\(\)](#), [eval_func\(\)](#), and [initialize\(\)](#).

6.64.4.25 evalf_use_exvector_args

`bool` GiNaC::function_options::evalf_use_exvector_args [protected]

Referenced by [GiNaC::function::evalf\(\)](#), [evalf_func\(\)](#), and [initialize\(\)](#).

6.64.4.26 conjugate_use_exvector_args

`bool` GiNaC::function_options::conjugate_use_exvector_args [protected]

Referenced by [GiNaC::function::conjugate\(\)](#), [conjugate_func\(\)](#), and [initialize\(\)](#).

6.64.4.27 `real_part_use_exvector_args`

`bool` `GiNaC::function_options::real_part_use_exvector_args` [protected]

Referenced by [initialize\(\)](#), [GiNaC::function::real_part\(\)](#), and [real_part_func\(\)](#).

6.64.4.28 `imag_part_use_exvector_args`

`bool` `GiNaC::function_options::imag_part_use_exvector_args` [protected]

Referenced by [GiNaC::function::imag_part\(\)](#), [imag_part_func\(\)](#), and [initialize\(\)](#).

6.64.4.29 `expand_use_exvector_args`

`bool` `GiNaC::function_options::expand_use_exvector_args` [protected]

Referenced by [GiNaC::function::expand\(\)](#), [expand_func\(\)](#), and [initialize\(\)](#).

6.64.4.30 `derivative_use_exvector_args`

`bool` `GiNaC::function_options::derivative_use_exvector_args` [protected]

Referenced by [derivative_func\(\)](#), [initialize\(\)](#), and [GiNaC::function::pderivative\(\)](#).

6.64.4.31 `expl_derivative_use_exvector_args`

`bool` `GiNaC::function_options::expl_derivative_use_exvector_args` [protected]

Referenced by [GiNaC::function::expl_derivative\(\)](#), [expl_derivative_func\(\)](#), and [initialize\(\)](#).

6.64.4.32 `power_use_exvector_args`

`bool` `GiNaC::function_options::power_use_exvector_args` [protected]

Referenced by [initialize\(\)](#), [GiNaC::function::power\(\)](#), and [power_func\(\)](#).

6.64.4.33 `series_use_exvector_args`

`bool` `GiNaC::function_options::series_use_exvector_args` [protected]

Referenced by [initialize\(\)](#), [GiNaC::function::series\(\)](#), and [series_func\(\)](#).

6.64.4.34 `print_use_exvector_args`

`bool` `GiNaC::function_options::print_use_exvector_args` [protected]

Referenced by [initialize\(\)](#), [GiNaC::function::print\(\)](#), and [print_func\(\)](#).

6.64.4.35 info_use_exvector_args

`bool` GiNaC::function_options::info_use_exvector_args [protected]

Referenced by [GiNaC::function::info\(\)](#), [info_func\(\)](#), and [initialize\(\)](#).

6.64.4.36 functions_with_same_name

`unsigned` GiNaC::function_options::functions_with_same_name [protected]

Referenced by [initialize\(\)](#), [overloaded\(\)](#), and [GiNaC::function::register_new\(\)](#).

6.64.4.37 symtree

`ex` GiNaC::function_options::symtree [protected]

Referenced by [GiNaC::function::eval\(\)](#), [initialize\(\)](#), and [set_symmetry\(\)](#).

The documentation for this class was generated from the following files:

- [function.h](#)
- [function.cpp](#)

6.65 GiNaC::G2_SERIAL Class Reference

Generalized multiple polylogarithm.

```
#include <inifcns.h>
```

Static Public Attributes

- static unsigned [serial](#)

6.65.1 Detailed Description

Generalized multiple polylogarithm.

6.65.2 Member Data Documentation

6.65.2.1 serial

`unsigned` GiNaC::G2_SERIAL::serial [static]

Initial value:

```
= function::register_new(function_options("G", 2).
    evalf_func(G2_evalf).
    eval_func(G2_eval).
    overloaded(2))
```

Referenced by [GiNaC::G\(\)](#).

The documentation for this class was generated from the following files:

- [inifcns.h](#)
- [inifcns_nstdsums.cpp](#)

6.66 GiNaC::G3_SERIAL Class Reference

Generalized multiple polylogarithm with explicit imaginary parts.

```
#include <inifcns.h>
```

Static Public Attributes

- static unsigned [serial](#)

6.66.1 Detailed Description

Generalized multiple polylogarithm with explicit imaginary parts.

6.66.2 Member Data Documentation

6.66.2.1 serial

```
unsigned GiNaC::G3_SERIAL::serial [static]
```

Initial value:

```
= function::register_new(function_options("G", 3).
    evalf_func(G3_evalf).
    eval_func(G3_eval).
    overloaded(2))
```

Referenced by [GiNaC::G\(\)](#).

The documentation for this class was generated from the following files:

- [inifcns.h](#)
- [inifcns_nstdsums.cpp](#)

6.67 GiNaC::gcd_options Struct Reference

Flags to control the behavior of [gcd\(\)](#) and friends.

```
#include <normal.h>
```

Public Types

- enum { [no_heur_gcd](#) = 2 , [no_part_factored](#) = 4 , [use_sr_gcd](#) = 8 }

6.67.1 Detailed Description

Flags to control the behavior of [gcd\(\)](#) and friends.

6.67.2 Member Enumeration Documentation

6.67.2.1 anonymous enum

```
anonymous enum
```

Enumerator

no_heur_gcd	Usually GiNaC tries heuristic GCD first, because typically it's much faster than anything else. Even if heuristic algorithm fails, the overhead is negligible w.r.t. cost of computing the GCD by some other method. However, some people dislike it, so here's a flag which tells GiNaC to NOT use the heuristic algorithm.
no_part_factored	GiNaC tries to avoid expanding expressions when computing GCDs. This is a good idea, but some people dislike it. Hence the flag to disable special handling of partially factored polynomials. DON'T SET THIS unless you <i>really</i> know what are you doing!
use_sr_gcd	By default GiNaC uses modular GCD algorithm. Typically it's much faster than PRS (pseudo remainder sequence) algorithm. This flag forces GiNaC to use PRS algorithm

The documentation for this struct was generated from the following file:

- [normal.h](#)

6.68 GiNaC::gcdheu_failed Class Reference

Exception thrown by [heur_gcd\(\)](#) to signal failure.

6.68.1 Detailed Description

Exception thrown by [heur_gcd\(\)](#) to signal failure.

The documentation for this class was generated from the following file:

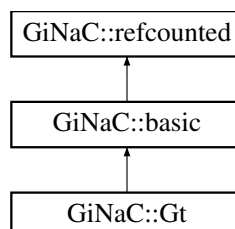
- [normal.cpp](#)

6.69 GiNaC::Gt Class Reference

Elliptic Multiple Polylogarithm.

```
#include <Gt.h>
```

Inheritance diagram for GiNaC::Gt:



Classes

- struct [kernel](#)

Public Member Functions

- `Gt` (const `ex` &`args`, const `ex` &`z`, const `ex` &`tau`)
- const `ex` & `getTau` () const
- const `ex` & `getZ` () const
- const `std::vector< kernel >` & `getArgs` () const
- unsigned `precedence` () const override
Return relative operator precedence (for parenthezing output).
- `ex eval` () const override
Perform automatic non-interruptive term rewriting rules.
- `ex evalf` () const override
Evaluate object numerically.
- `size_t nops` () const override
Number of operands/members.
- `ex subs` (const `exmap` &`m`, unsigned `options`=0) const override
Substitute a set of objects by arbitrary expressions.
- unsigned `calchash` () const override
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- bool `has` (const `ex` &`other`, unsigned `options`=0) const override
Test for occurrence of a pattern.
- bool `match` (const `ex` &`pattern`, `exmap` &`repl_lst`) const override
Check whether the expression matches a given pattern.
- void `archive` (`archive_node` &`n`) const override
Save (a.k.a.
- void `read_archive` (const `archive_node` &`n`, `lst` &`syms`) override
Read (a.k.a.
- `ex zisToFundamental` (const `ex` *`points`=nullptr) const
- `ex regularise` (const `ex` *`points`=nullptr) const
- `matrix findMoebiusTransform` (const `ex` *`points`=nullptr) const
- `ex tauToFundamental` (const `ex` *`points`=nullptr) const
- `std::vector< ex > decomposeIntegrationPath` (const `ex` *`points`=nullptr) const
- `ex applyIntegrationPath` (const `std::vector< ex >` &`endpoints`) const
- `numeric qExpand` (const `ex` *`points`=nullptr) const
- `numeric evaluate` (const `ex` *`points`=nullptr) const

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic` ()
basic destructor, virtual because class `ex` will delete objects of derived classes via a `basic*`.
- `basic` (const `basic` &`other`)
- const `basic` & `operator=` (const `basic` &`other`)
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate` () const
Create a clone of this object on the heap.
- virtual `ex evalm` () const
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ` () const
Evaluate integrals, if result is known.
- virtual `ex eval_indexed` (const `basic` &`i`) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void `print` (const `print_context` &`c`, unsigned `level`=0) const

- virtual void `dbgprint` () const
Output to stream.
- virtual void `dbgprinttree` () const
Little wrapper around print to be called within a debugger.
- virtual bool `info` (unsigned inf) const
Information about the object.
- virtual `ex op` (size_t i) const
Return operand/member at position i.
- virtual `ex operator[]` (const `ex` &index) const
- virtual `ex operator[]` (size_t i) const
- virtual `ex & let_op` (size_t i)
Return modifiable operand/member at position i.
- virtual `ex & operator[]` (const `ex` &index)
- virtual `ex & operator[]` (size_t i)
- virtual `ex map` (map_function &f) const
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void `accept` (GiNaC::visitor &v) const
- virtual bool `is_polynomial` (const `ex` &var) const
Check whether this is a polynomial in the given variables.
- virtual int `degree` (const `ex` &s) const
Return degree of highest power in object s.
- virtual int `ldegree` (const `ex` &s) const
Return degree of lowest power in object s.
- virtual `ex coeff` (const `ex` &s, int n=1) const
Return coefficient of degree n in object s.
- virtual `ex expand` (unsigned options=0) const
Expand expression, i.e.
- virtual `ex collect` (const `ex` &s, bool distributed=false) const
Sort expanded expression in terms of powers of some object(s).
- virtual `ex series` (const relational &r, int order, unsigned options=0) const
Default implementation of ex::series().
- virtual `ex normal` (exmap &repl, exmap &rev_lookup, lst &modifier) const
Default implementation of ex::normal().
- virtual `ex to_rational` (exmap &repl) const
Default implementation of ex::to_rational().
- virtual `ex to_polynomial` (exmap &repl) const
- virtual `numeric integer_content` () const
- virtual `ex smod` (const numeric &xi) const
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient` () const
Implementation ex::max_coefficient().
- virtual `exvector get_free_indices` () const
Return a vector containing the free indices of an expression.
- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
Add two indexed expressions.
- virtual `ex scalar_mul_indexed` (const `ex` &self, const numeric &other) const
Multiply an indexed expression with a scalar.
- virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const
Try to contract two indexed expressions that appear in the same product.
- virtual unsigned `return_type` () const

- virtual [return_type_t return_type_tinfo](#) () const
- virtual [ex conjugate](#) () const
- virtual [ex real_part](#) () const
- virtual [ex imag_part](#) () const
- template<class T >
void [print_dispatch](#) (const [print_context](#) &c, unsigned level) const
Like [print\(\)](#), but dispatch to the specified class.
- void [print_dispatch](#) (const [registered_class_info](#) &ri, const [print_context](#) &c, unsigned level) const
Like [print\(\)](#), but dispatch to the specified class.
- [ex subs_one_level](#) (const [exmap](#) &m, unsigned [options](#)) const
Helper function for [subs\(\)](#).
- [ex diff](#) (const [symbol](#) &s, unsigned nth=1) const
Default interface of nth derivative [ex::diff\(s, n\)](#).
- int [compare](#) (const [basic](#) &other) const
Compare objects syntactically to establish canonical ordering.
- bool [is_equal](#) (const [basic](#) &other) const
Test for syntactic equality.
- const [basic](#) & [hold](#) () const
Stop further evaluation.
- unsigned [gethash](#) () const
- const [basic](#) & [setflag](#) (unsigned f) const
Set some [status_flags](#).
- const [basic](#) & [clearflag](#) (unsigned f) const
Clear some [status_flags](#).

Public Member Functions inherited from [GiNaC::refcounted](#)

- [refcounted](#) () noexcept
- unsigned int [add_reference](#) () noexcept
- unsigned int [remove_reference](#) () noexcept
- unsigned int [get_refcount](#) () const noexcept
- void [set_refcount](#) (unsigned int r) noexcept

Static Public Member Functions

- static [ex ex_zisToFundamental](#) (const [ex](#) &expr, const [ex](#) *points=nullptr)
- static [ex ex_regularise](#) (const [ex](#) &expr, const [ex](#) *points=nullptr)
- static [ex ex_tauToFundamental](#) (const [ex](#) &expr, const [ex](#) *points=nullptr)
- static [ex ex_cutIntegrationPath](#) (const [ex](#) &expr, const [ex](#) *points=nullptr)
- static [numeric ex_qExpand](#) (const [ex](#) &expr, const [ex](#) *points=nullptr)
- static [ex ex_prepare](#) (const [ex](#) &expr, const [ex](#) *points=nullptr)
- static [numeric ex_evaluate](#) (const [ex](#) &expr, const [ex](#) *points=nullptr)
- static [lst lst_evaluate](#) (const [lst](#) &list, const [ex](#) *points=nullptr)
- static [ex ex_zisToFundamental](#) (const [ex](#) &expr, const [ex](#) &points)
- static [ex ex_regularise](#) (const [ex](#) &expr, const [ex](#) &points)
- static [ex ex_tauToFundamental](#) (const [ex](#) &expr, const [ex](#) &points)
- static [ex ex_cutIntegrationPath](#) (const [ex](#) &expr, const [ex](#) &points)
- static [numeric ex_qExpand](#) (const [ex](#) &expr, const [ex](#) &points)
- static [ex ex_prepare](#) (const [ex](#) &expr, const [ex](#) &points)
- static [numeric ex_evaluate](#) (const [ex](#) &expr, const [ex](#) &points)
- static [lst lst_evaluate](#) (const [lst](#) &list, const [ex](#) &points)
- static [ex apply_function_recursive](#) (const [ex](#) &input, const std::function< [ex](#)(const [Gt](#) &)> &func)
- static void [clear_polylog_cache](#) ()

Static Public Attributes

- static double `cutIntegrationPath_threshold_real` = 0.4
- static double `cutIntegrationPath_threshold_imag` = 0.4
- static size_t `qExpand_minOrder` = 4
- static size_t `qExpand_stepSize` = 4
- static bool `auto_clear_polylog_cache` = false
- static bool `enable_tauToFundamental` = true

Protected Member Functions

- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_latex` (const `print_latex` &c, unsigned level) const

Protected Member Functions inherited from `GiNaC::basic`

- `basic` ()
- virtual `ex eval_ncmul` (const `exvector` &v) const
- virtual bool `match_same_type` (const `basic` &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual `ex derivative` (const `symbol` &s) const
Default implementation of `ex::diff()`.
- virtual int `compare_same_type` (const `basic` &other) const
Returns order relation between two objects of same type.
- virtual bool `is_equal_same_type` (const `basic` &other) const
Returns true if two objects of same type are equal.
- void `ensure_if_modifiable` () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void `do_print` (const `print_context` &c, unsigned level) const
Default output to stream.
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
Tree output to stream.
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
Python parsable output to stream.

Private Member Functions

- `Gt` (std::vector< `kernel` > args, `ex` z, `ex` tau)
- void `setArgs` (const `ex` &args)

Static Private Member Functions

- static `numeric to_numeric` (const `ex` &expr, const `ex` *points=nullptr)
- static long `to_integer` (const `ex` &expr, const bool allow_negative)

Private Attributes

- size_t `nargs`
- std::vector< `kernel` > `args`
- `ex` z
- `ex` tau

Static Private Attributes

- static [Gt_detail::TransformExpressionWithCache](#)< [G2_SERIAL](#) > [ngexand_transformer](#)

Additional Inherited Members

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.69.1 Detailed Description

Elliptic Multiple Polylogarithm.

6.69.2 Constructor & Destructor Documentation

6.69.2.1 [Gt\(\)](#) [1/2]

```
GiNaC::Gt::Gt (
    const ex & args,
    const ex & z,
    const ex & tau )
```

References [GiNaC::ex::evalf\(\)](#), [nargs](#), [setArgs\(\)](#), and [tau](#).

6.69.2.2 [Gt\(\)](#) [2/2]

```
GiNaC::Gt::Gt (
    std::vector< kernel > args,
    ex z,
    ex tau ) [private]
```

References [args](#), [GiNaC::ex::evalf\(\)](#), [k](#), [nargs](#), and [tau](#).

6.69.3 Member Function Documentation

6.69.3.1 [setArgs\(\)](#)

```
void GiNaC::Gt::setArgs (
    const ex & args ) [private]
```

References [args](#), [nargs](#), [GiNaC::ex::nops\(\)](#), and [GiNaC::ex::op\(\)](#).

Referenced by [Gt\(\)](#), and [read_archive\(\)](#).

6.69.3.2 getTau()

```
const ex & GiNaC::Gt::getTau ( ) const [inline]
```

References [tau](#).

6.69.3.3 getZ()

```
const ex & GiNaC::Gt::getZ ( ) const [inline]
```

References [z](#).

6.69.3.4 getArgs()

```
const std::vector< kernel > & GiNaC::Gt::getArgs ( ) const [inline]
```

References [args](#).

6.69.3.5 precedence()

```
unsigned GiNaC::Gt::precedence ( ) const [inline], [override], [virtual]
```

Return relative operator precedence (for parenthezing output).

Reimplemented from [GiNaC::basic](#).

6.69.3.6 eval()

```
ex GiNaC::Gt::eval ( ) const [override], [virtual]
```

Perform automatic non-interruptive term rewriting rules.

Reimplemented from [GiNaC::basic](#).

References [args](#), [GiNaC::status_flags::evaluated](#), [GiNaC::factorial\(\)](#), [GiNaC::basic::flags](#), [GiNaC::basic::hold\(\)](#), [nargs](#), [GiNaC::pow\(\)](#), [tau](#), and [z](#).

6.69.3.7 evalf()

```
ex GiNaC::Gt::evalf ( ) const [override], [virtual]
```

Evaluate object numerically.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::are_ex_trivially_equal\(\)](#), [args](#), [GiNaC::ex::evalf\(\)](#), [evaluate\(\)](#), [k](#), [nargs](#), [tau](#), and [z](#).

Referenced by [qExpand\(\)](#), and [to_numeric\(\)](#).

6.69.3.8 nops()

```
size_t GiNaC::Gt::nops ( ) const [inline], [override], [virtual]
```

Number of operands/members.

Reimplemented from [GiNaC::basic](#).

6.69.3.9 subs()

```
ex GiNaC::Gt::subs (
    const exmap & m,
    unsigned options = 0 ) const [override], [virtual]
```

Substitute a set of objects by arbitrary expressions.

The ex returned will already be evaluated.

Reimplemented from [GiNaC::basic](#).

References [args](#), [k](#), [m](#), [nargs](#), [options](#), [GiNaC::ex::subs\(\)](#), [tau](#), and [z](#).

6.69.3.10 calchash()

```
unsigned GiNaC::Gt::calchash ( ) const [override], [virtual]
```

Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.

The method inherited from class basic computes a hash value based on the type and hash values of possible members. For this reason it is well suited for container classes but atomic classes should override this implementation because otherwise they would all end up with the same hashvalue.

Reimplemented from [GiNaC::basic](#).

References [args](#), [GiNaC::status_flags::evaluated](#), [GiNaC::basic::flags](#), [GiNaC::basic::gethash\(\)](#), [GiNaC::ex::gethash\(\)](#), [GiNaC::status_flags::hash_calculated](#), [GiNaC::basic::hashvalue](#), [k](#), [GiNaC::make_hash_seed\(\)](#), [GiNaC::rotate_left\(\)](#), [GiNaC::basic::setflag\(\)](#), [tau](#), and [z](#).

6.69.3.11 has()

```
bool GiNaC::Gt::has (
    const ex & pattern,
    unsigned options = 0 ) const [override], [virtual]
```

Test for occurrence of a pattern.

An object 'has' a pattern if it matches the pattern itself or one of the children 'has' it. As a consequence (according to the definition of children) given $e=x+y+z$, $e.has(x)$ is true but $e.has(x+y)$ is false.

Reimplemented from [GiNaC::basic](#).

References [args](#), [GiNaC::ex::has\(\)](#), [k](#), [options](#), [tau](#), and [z](#).

6.69.3.12 match()

```
bool GiNaC::Gt::match (
    const ex & pattern,
    exmap & repl_lst ) const [override], [virtual]
```

Check whether the expression matches a given pattern.

For every wildcard object in the pattern, a pair with the wildcard as a key and matching expression as a value is added to *repl_lst*.

Reimplemented from [GiNaC::basic](#).

6.69.3.13 archive()

```
void GiNaC::Gt::archive (
    archive_node & n ) const [override], [virtual]
```

Save (a.k.a.

serialize) object into archive.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::container< C >::append\(\)](#), [n](#), [nargs](#), [tau](#), and [z](#).

6.69.3.14 read_archive()

```
void GiNaC::Gt::read_archive (
    const archive_node & n,
    lst & syms ) [override], [virtual]
```

Read (a.k.a.

deserialize) object from archive.

Reimplemented from [GiNaC::basic](#).

References [args](#), [n](#), [setArgs\(\)](#), [tau](#), and [z](#).

6.69.3.15 zisToFundamental()

```
ex GiNaC::Gt::zisToFundamental (
    const ex * points = nullptr ) const
```

References [args](#), [DEBUG_GT](#), [GiNaC::factor\(\)](#), [GiNaC::factorial\(\)](#), [GiNaC::I](#), [GiNaC::numeric::imag\(\)](#), [GiNaC::is_zero\(\)](#), [k](#), [nargs](#), [GiNaC::ex::normal\(\)](#), [GiNaC::Pi](#), [GiNaC::pow\(\)](#), [GiNaC::numeric::real\(\)](#), [tau](#), [to_numeric\(\)](#), and [z](#).

Referenced by [ex_zisToFundamental\(\)](#).

6.69.3.16 regularise()

```
ex GiNaC::Gt::regularise (
    const ex * points = nullptr ) const
```

References [args](#), [DEBUG_GT](#), [GiNaC::factorial\(\)](#), [GiNaC::is_zero\(\)](#), [k](#), [n](#), [nargs](#), [GiNaC::pow\(\)](#), [tau](#), [to_numeric\(\)](#), and [z](#).

Referenced by [ex_regularise\(\)](#).

6.69.3.17 findMoebiusTransform()

```
matrix GiNaC::Gt::findMoebiusTransform (
    const ex * points = nullptr ) const
```

References [GiNaC::abs\(\)](#), [GiNaC::numeric::imag\(\)](#), [GiNaC::numeric::is_positive\(\)](#), [GiNaC::matrix::mul\(\)](#), [GiNaC::numeric::mul\(\)](#), [tau](#), [GiNaC::numeric::to_cl_N\(\)](#), and [to_numeric\(\)](#).

Referenced by [tauToFundamental\(\)](#).

6.69.3.18 tauToFundamental()

```
ex GiNaC::Gt::tauToFundamental (
    const ex * points = nullptr ) const
```

References [args](#), [GiNaC::binomial\(\)](#), [c](#), [DEBUG_GT](#), [GiNaC::factorial\(\)](#), [findMoebiusTransform\(\)](#), [GiNaC::I](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::numeric::is_zero\(\)](#), [k](#), [GiNaC::log\(\)](#), [nargs](#), [GiNaC::Gt::kernel::ni](#), [GiNaC::basic::normal\(\)](#), [GiNaC::Pi](#), [GiNaC::pow\(\)](#), [tau](#), [to_numeric\(\)](#), and [z](#).

Referenced by [ex_tauToFundamental\(\)](#).

6.69.3.19 decomposeIntegrationPath()

```
std::vector< ex > GiNaC::Gt::decomposeIntegrationPath (
    const ex * points = nullptr ) const
```

References [args](#), [cutIntegrationPath_threshold_imag](#), [cutIntegrationPath_threshold_real](#), [DEBUG_GT](#), [GiNaC::basic::diff\(\)](#), [GiNaC::numeric::imag\(\)](#), [GiNaC::numeric::is_integer\(\)](#), [k](#), [GiNaC::numeric::real\(\)](#), [tau](#), [GiNaC::numeric::to_double\(\)](#), [to_numeric\(\)](#), and [z](#).

Referenced by [ex_cutIntegrationPath\(\)](#).

6.69.3.20 applyIntegrationPath()

```
ex GiNaC::Gt::applyIntegrationPath (
    const std::vector< ex > & endpoints ) const
```

References [args](#), [k](#), and [tau](#).

Referenced by [ex_cutIntegrationPath\(\)](#).

6.69.3.21 qExpand()

```
numeric GiNaC::Gt::qExpand (
    const ex * points = nullptr ) const
```

References [GiNaC::abs\(\)](#), [GiNaC::add::add\(\)](#), [args](#), [GiNaC::binomial\(\)](#), [DEBUG_GT](#), [GiNaC::Digits](#), [GiNaC::ex::evalf\(\)](#), [evalf\(\)](#), [GiNaC::exp\(\)](#), [GiNaC::factor\(\)](#), [GiNaC::factorial\(\)](#), [GiNaC::I](#), [GiNaC::numeric::imag\(\)](#), [GiNaC::numeric::is_negative\(\)](#), [GiNaC::numeric::is_positive\(\)](#), [k](#), [GiNaC::log\(\)](#), [nargs](#), [ngexand_transformer](#), [GiNaC::Pi](#), [GiNaC::pow\(\)](#), [qExpand_minOrder](#), [qExpand_stepSize](#), [GiNaC::numeric::real\(\)](#), [GiNaC::real\(\)](#), [tau](#), [GiNaC::numeric::to_double\(\)](#), [GiNaC::to_double\(\)](#), [to_numeric\(\)](#), [z](#), and [GiNaC::zeta\(\)](#).

Referenced by [ex_qExpand\(\)](#).

6.69.3.22 evaluate()

```
numeric GiNaC::Gt::evaluate (
    const ex * points = nullptr ) const
```

References [ex_evaluate\(\)](#).

Referenced by [evalf\(\)](#).

6.69.3.23 ex_zisToFundamental() [1/2]

```
ex GiNaC::Gt::ex_zisToFundamental (
    const ex & expr,
    const ex * points = nullptr ) [static]
```

References [apply_function_recursive\(\)](#), and [zisToFundamental\(\)](#).

Referenced by [ex_prepare\(\)](#), and [ex_zisToFundamental\(\)](#).

6.69.3.24 ex_regularise() [1/2]

```
ex GiNaC::Gt::ex_regularise (
    const ex & expr,
    const ex * points = nullptr ) [static]
```

References [apply_function_recursive\(\)](#), and [regularise\(\)](#).

Referenced by [ex_prepare\(\)](#), and [ex_regularise\(\)](#).

6.69.3.25 ex_tauToFundamental() [1/2]

```
ex GiNaC::Gt::ex_tauToFundamental (
    const ex & expr,
    const ex * points = nullptr ) [static]
```

References [apply_function_recursive\(\)](#), and [tauToFundamental\(\)](#).

Referenced by [ex_prepare\(\)](#), and [ex_tauToFundamental\(\)](#).

6.69.3.26 ex_cutIntegrationPath() [1/2]

```
ex GiNaC::Gt::ex_cutIntegrationPath (
    const ex & expr,
    const ex * points = nullptr ) [static]
```

References [apply_function_recursive\(\)](#), [applyIntegrationPath\(\)](#), and [decomposeIntegrationPath\(\)](#).

Referenced by [ex_cutIntegrationPath\(\)](#), and [ex_prepare\(\)](#).

6.69.3.27 ex_qExpand() [1/2]

```
numeric GiNaC::Gt::ex_qExpand (
    const ex & expr,
    const ex * points = nullptr ) [static]
```

References [apply_function_recursive\(\)](#), [qExpand\(\)](#), and [to_numeric\(\)](#).

Referenced by [ex_evaluate\(\)](#), [ex_qExpand\(\)](#), and [lst_evaluate\(\)](#).

6.69.3.28 ex_prepare() [1/2]

```
ex GiNaC::Gt::ex_prepare (
    const ex & expr,
    const ex * points = nullptr ) [static]
```

References [DEBUG_GT](#), [enable_tauToFundamental](#), [ex_cutIntegrationPath\(\)](#), [ex_regularise\(\)](#), [ex_tauToFundamental\(\)](#), and [ex_zisToFundamental\(\)](#).

Referenced by [ex_evaluate\(\)](#), [ex_prepare\(\)](#), and [lst_evaluate\(\)](#).

6.69.3.29 ex_evaluate() [1/2]

```
numeric GiNaC::Gt::ex_evaluate (
    const ex & expr,
    const ex * points = nullptr ) [static]
```

References [auto_clear_polylog_cache](#), [clear_polylog_cache\(\)](#), [DEBUG_GT](#), [ex_prepare\(\)](#), [ex_qExpand\(\)](#), and [to_numeric\(\)](#).

Referenced by [evaluate\(\)](#), and [ex_evaluate\(\)](#).

6.69.3.30 lst_evaluate() [1/2]

```
lst GiNaC::Gt::lst_evaluate (
    const lst & list,
    const ex * points = nullptr ) [static]
```

References [GiNaC::container< C >::append\(\)](#), [auto_clear_polylog_cache](#), [clear_polylog_cache\(\)](#), [DEBUG_GT](#), [ex_prepare\(\)](#), [ex_qExpand\(\)](#), [GiNaC::container< C >::nops\(\)](#), [GiNaC::container< C >::op\(\)](#), and [to_numeric\(\)](#).

Referenced by [lst_evaluate\(\)](#).

6.69.3.31 `ex_zisToFundamental()` [2/2]

```
ex GiNaC::Gt::ex_zisToFundamental (
    const ex & expr,
    const ex & points ) [static]
```

References [ex_zisToFundamental\(\)](#).

6.69.3.32 `ex_regularise()` [2/2]

```
ex GiNaC::Gt::ex_regularise (
    const ex & expr,
    const ex & points ) [static]
```

References [ex_regularise\(\)](#).

6.69.3.33 `ex_tauToFundamental()` [2/2]

```
ex GiNaC::Gt::ex_tauToFundamental (
    const ex & expr,
    const ex & points ) [static]
```

References [ex_tauToFundamental\(\)](#).

6.69.3.34 `ex_cutIntegrationPath()` [2/2]

```
ex GiNaC::Gt::ex_cutIntegrationPath (
    const ex & expr,
    const ex & points ) [static]
```

References [ex_cutIntegrationPath\(\)](#).

6.69.3.35 `ex_qExpand()` [2/2]

```
numeric GiNaC::Gt::ex_qExpand (
    const ex & expr,
    const ex & points ) [static]
```

References [ex_qExpand\(\)](#).

6.69.3.36 `ex_prepare()` [2/2]

```
ex GiNaC::Gt::ex_prepare (
    const ex & expr,
    const ex & points ) [static]
```

References [ex_prepare\(\)](#).

6.69.3.37 ex_evaluate() [2/2]

```
numeric GiNaC::Gt::ex_evaluate (
    const ex & expr,
    const ex & points ) [static]
```

References [ex_evaluate\(\)](#).

6.69.3.38 lst_evaluate() [2/2]

```
lst GiNaC::Gt::lst_evaluate (
    const lst & list,
    const ex & points ) [static]
```

References [lst_evaluate\(\)](#).

6.69.3.39 apply_function_recursive()

```
ex GiNaC::Gt::apply_function_recursive (
    const ex & input,
    const std::function< ex(const Gt &)> & func ) [static]
```

Referenced by [ex_cutIntegrationPath\(\)](#), [ex_qExpand\(\)](#), [ex_regularise\(\)](#), [ex_tauToFundamental\(\)](#), and [ex_zisToFundamental\(\)](#).

6.69.3.40 clear_polylog_cache()

```
void GiNaC::Gt::clear_polylog_cache ( ) [static]
```

References [nqexand_transformer](#).

Referenced by [ex_evaluate\(\)](#), and [lst_evaluate\(\)](#).

6.69.3.41 do_print()

```
void GiNaC::Gt::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

References [args](#), [c](#), [nargs](#), [tau](#), and [z](#).

6.69.3.42 do_print_latex()

```
void GiNaC::Gt::do_print_latex (
    const print_latex & c,
    unsigned level ) const [protected]
```

References [args](#), [c](#), [nargs](#), [tau](#), and [z](#).

6.69.3.43 to_numeric()

```
numeric GiNaC::Gt::to_numeric (
    const ex & expr,
    const ex * points = nullptr ) [static], [private]
```

References [evalf\(\)](#), and [GiNaC::ex::subs\(\)](#).

Referenced by [decomposeIntegrationPath\(\)](#), [ex_evaluate\(\)](#), [ex_qExpand\(\)](#), [findMoebiusTransform\(\)](#), [lst_evaluate\(\)](#), [qExpand\(\)](#), [regularise\(\)](#), [tauToFundamental\(\)](#), and [zisToFundamental\(\)](#).

6.69.3.44 to_integer()

```
long GiNaC::Gt::to_integer (
    const ex & expr,
    const bool allow_negative ) [static], [private]
```

References [n](#).

Referenced by [GiNaC::Gt::kernel::kernel\(\)](#).

6.69.4 Member Data Documentation

6.69.4.1 cutIntegrationPath_threshold_real

```
double GiNaC::Gt::cutIntegrationPath_threshold_real = 0.4 [static]
```

Referenced by [decomposeIntegrationPath\(\)](#).

6.69.4.2 cutIntegrationPath_threshold_imag

```
double GiNaC::Gt::cutIntegrationPath_threshold_imag = 0.4 [static]
```

Referenced by [decomposeIntegrationPath\(\)](#).

6.69.4.3 qExpand_minOrder

```
size_t GiNaC::Gt::qExpand_minOrder = 4 [static]
```

Referenced by [qExpand\(\)](#).

6.69.4.4 qExpand_stepSize

```
size_t GiNaC::Gt::qExpand_stepSize = 4 [static]
```

Referenced by [qExpand\(\)](#).

6.69.4.5 auto_clear_polylog_cache

```
bool GiNaC::Gt::auto_clear_polylog_cache = false [static]
```

Referenced by [ex_evaluate\(\)](#), and [lst_evaluate\(\)](#).

6.69.4.6 enable_tauToFundamental

```
bool GiNaC::Gt::enable_tauToFundamental = true [static]
```

Referenced by [ex_prepare\(\)](#).

6.69.4.7 nargs

```
size_t GiNaC::Gt::nargs [private]
```

Referenced by [archive\(\)](#), [do_print\(\)](#), [do_print_latex\(\)](#), [eval\(\)](#), [evalf\(\)](#), [Gt\(\)](#), [Gt\(\)](#), [qExpand\(\)](#), [regularise\(\)](#), [setArgs\(\)](#), [subs\(\)](#), [tauToFundamental\(\)](#), and [zisToFundamental\(\)](#).

6.69.4.8 args

```
std::vector<kernel> GiNaC::Gt::args [private]
```

Referenced by [applyIntegrationPath\(\)](#), [calchash\(\)](#), [decomposeIntegrationPath\(\)](#), [do_print\(\)](#), [do_print_latex\(\)](#), [eval\(\)](#), [evalf\(\)](#), [getArgs\(\)](#), [Gt\(\)](#), [has\(\)](#), [qExpand\(\)](#), [read_archive\(\)](#), [regularise\(\)](#), [setArgs\(\)](#), [subs\(\)](#), [tauToFundamental\(\)](#), and [zisToFundamental\(\)](#).

6.69.4.9 z

```
ex GiNaC::Gt::z [private]
```

Referenced by [archive\(\)](#), [calchash\(\)](#), [decomposeIntegrationPath\(\)](#), [do_print\(\)](#), [do_print_latex\(\)](#), [eval\(\)](#), [evalf\(\)](#), [getZ\(\)](#), [has\(\)](#), [qExpand\(\)](#), [read_archive\(\)](#), [regularise\(\)](#), [subs\(\)](#), [tauToFundamental\(\)](#), and [zisToFundamental\(\)](#).

6.69.4.10 tau

```
ex GiNaC::Gt::tau [private]
```

Referenced by [applyIntegrationPath\(\)](#), [archive\(\)](#), [calchash\(\)](#), [decomposeIntegrationPath\(\)](#), [do_print\(\)](#), [do_print_latex\(\)](#), [eval\(\)](#), [evalf\(\)](#), [findMoebiusTransform\(\)](#), [getTau\(\)](#), [Gt\(\)](#), [Gt\(\)](#), [has\(\)](#), [qExpand\(\)](#), [read_archive\(\)](#), [regularise\(\)](#), [subs\(\)](#), [tauToFundamental\(\)](#), and [zisToFundamental\(\)](#).

6.69.4.11 nqexand_transformer

```
Gt_detail::TransformExpressionWithCache< G2_SERIAL > GiNaC::Gt::nqexand_transformer [static],
[private]
```

Initial value:

```
= [] ()
{
    Digits.add_callback([](long change) -> void {
        if (change>0) Gt::clear_polylog_cache();
    });
    return Gt_detail::TransformExpressionWithCache<G2_SERIAL>{
        [](const ex& obj) {return obj.evalf();},
        [](const ex& obj) {return obj.evalf();}
    };
}()
```

Referenced by [clear_polylog_cache\(\)](#), and [qExpand\(\)](#).

The documentation for this class was generated from the following files:

- [Gt.h](#)
- [Gt.cpp](#)

6.70 GiNaC::has_distance< T > Class Template Reference

SFINAE test for distance.

```
#include <utils_multi_iterator.h>
```

Public Types

- enum { [value](#) = sizeof(test<T>(0)) == sizeof(yes_type) }

Private Types

- typedef char [yes_type](#)[1]
- typedef char [no_type](#)[2]

Static Private Member Functions

- template<typename C >
static [yes_type](#) & [test](#) (decltype(std::distance< C >))
- template<typename C >
static [no_type](#) & [test](#) (...)

6.70.1 Detailed Description

```
template<typename T>
class GiNaC::has_distance< T >
```

SFINAE test for distance.

6.70.2 Member Typedef Documentation

6.70.2.1 yes_type

```
template<typename T >
typedef char GiNaC::has_distance< T >::yes_type[1] [private]
```

6.70.2.2 no_type

```
template<typename T >
typedef char GiNaC::has_distance< T >::no_type[2] [private]
```

6.70.3 Member Enumeration Documentation

6.70.3.1 anonymous enum

```
template<typename T >
anonymous enum
```

Enumerator

value	

6.70.4 Member Function Documentation

6.70.4.1 test() [1/2]

```
template<typename T >
template<typename C >
static yes_type & GiNaC::has_distance< T >::test (
    decltype(std::distance< C >) ) [static], [private]
```

6.70.4.2 test() [2/2]

```
template<typename T >
template<typename C >
static no_type & GiNaC::has_distance< T >::test (
    ... ) [static], [private]
```

The documentation for this class was generated from the following file:

- [utils_multi_iterator.h](#)

6.71 GiNaC::has_options Class Reference

Flags to control the behavior of [has\(\)](#).

```
#include <flags.h>
```

Public Types

- enum { [algebraic](#) = 0x0001 }

6.71.1 Detailed Description

Flags to control the behavior of [has\(\)](#).

6.71.2 Member Enumeration Documentation

6.71.2.1 anonymous enum

```
anonymous enum
```

Enumerator

algebraic	enable algebraic matching

The documentation for this class was generated from the following file:

- [flags.h](#)

6.72 std::hash< GiNaC::ex > Struct Reference

Specialization of std::hash() for ex objects.

```
#include <ex.h>
```

Public Member Functions

- std::size_t [operator\(\)](#) (const [GiNaC::ex](#) &e) const noexcept

6.72.1 Detailed Description

Specialization of std::hash() for ex objects.

6.72.2 Member Function Documentation

6.72.2.1 operator()

```
std::size_t std::hash< GiNaC::ex >::operator() (
    const GiNaC::ex & e ) const [inline], [noexcept]
```

The documentation for this struct was generated from the following file:

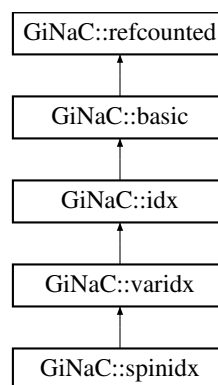
- [ex.h](#)

6.73 GiNaC::idx Class Reference

This class holds one index of an indexed object.

```
#include <idx.h>
```

Inheritance diagram for GiNaC::idx:



Public Member Functions

- [idx](#) (const [ex](#) &v, const [ex](#) &dim)
Construct index with given value and dimension.
- bool [info](#) (unsigned inf) const override
Information about the object.
- size_t [nops](#) () const override
Number of operands/members.
- [ex op](#) (size_t i) const override
Return operand/member at position i.
- [ex map](#) (map_function &f) const override
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- [ex evalf](#) () const override
By default, [basic::evalf](#) would evaluate the index value but we don't want a.1 to become a.
- [ex subs](#) (const [exmap](#) &m, unsigned [options](#)=0) const override
Substitute a set of objects by arbitrary expressions.
- void [archive](#) ([archive_node](#) &n) const override
Save (serialize) the object into archive node.

- void `read_archive` (const `archive_node` &n, `lst` &syms) override
Load (deserialize) the object from an archive node.
- virtual bool `is_dummy_pair_same_type` (const `basic` &other) const
Check whether the index forms a dummy index pair with another index of the same type.
- `ex get_value` () const
Get value of index.
- bool `is_numeric` () const
Check whether the index is numeric.
- bool `is_symbolic` () const
Check whether the index is symbolic.
- `ex get_dim` () const
Get dimension of index space.
- bool `is_dim_numeric` () const
Check whether the dimension is numeric.
- bool `is_dim_symbolic` () const
Check whether the dimension is symbolic.
- `ex replace_dim` (const `ex` &new_dim) const
Make a new index with the same value but a different dimension.
- `ex minimal_dim` (const `idx` &other) const
Return the minimum of the dimensions of this and another index.

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic` ()
basic destructor, virtual because class `ex` will delete objects of derived classes via a `basic*`.
- `basic` (const `basic` &other)
- const `basic` & `operator=` (const `basic` &other)
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate` () const
Create a clone of this object on the heap.
- virtual `ex eval` () const
Perform automatic non-interruptive term rewriting rules.
- virtual `ex evalm` () const
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ` () const
Evaluate integrals, if result is known.
- virtual `ex eval_indexed` (const `basic` &i) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void `print` (const `print_context` &c, unsigned level=0) const
Output to stream.
- virtual void `dbgprint` () const
Little wrapper around `print` to be called within a debugger.
- virtual void `dbgprinttree` () const
Little wrapper around `printtree` to be called within a debugger.
- virtual unsigned `precedence` () const
Return relative operator precedence (for parenthesizing output).
- virtual `ex operator[]` (const `ex` &index) const
- virtual `ex operator[]` (size_t i) const
- virtual `ex & let_op` (size_t i)
Return modifiable operand/member at position i.

- virtual `ex & operator[]` (const `ex` &index)
- virtual `ex & operator[]` (size_t i)
- virtual bool `has` (const `ex` &other, unsigned `options`=0) const
Test for occurrence of a pattern.
- virtual bool `match` (const `ex` &pattern, `exmap` &repls) const
Check whether the expression matches a given pattern.
- virtual void `accept` (`GiNaC::visitor` &v) const
- virtual bool `is_polynomial` (const `ex` &var) const
Check whether this is a polynomial in the given variables.
- virtual int `degree` (const `ex` &s) const
Return degree of highest power in object s.
- virtual int `ldegree` (const `ex` &s) const
Return degree of lowest power in object s.
- virtual `ex coeff` (const `ex` &s, int `n`=1) const
Return coefficient of degree n in object s.
- virtual `ex expand` (unsigned `options`=0) const
Expand expression, i.e.
- virtual `ex collect` (const `ex` &s, bool `distributed`=false) const
Sort expanded expression in terms of powers of some object(s).
- virtual `ex series` (const `relational` &r, int `order`, unsigned `options`=0) const
Default implementation of `ex::series()`.
- virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const
Default implementation of `ex::normal()`.
- virtual `ex to_rational` (`exmap` &repl) const
Default implementation of `ex::to_rational()`.
- virtual `ex to_polynomial` (`exmap` &repl) const
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &xi) const
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient` () const
Implementation `ex::max_coefficient()`.
- virtual `exvector get_free_indices` () const
Return a vector containing the free indices of an expression.
- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
Add two indexed expressions.
- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
Multiply an indexed expression with a scalar.
- virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const
Try to contract two indexed expressions that appear in the same product.
- virtual unsigned `return_type` () const
- virtual `return_type_t return_type_tinfo` () const
- virtual `ex conjugate` () const
- virtual `ex real_part` () const
- virtual `ex imag_part` () const
- template<class T >
void `print_dispatch` (const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- `ex subs_one_level` (const `exmap` &m, unsigned `options`) const
Helper function for `subs()`.

- `ex diff` (const `symbol` &s, unsigned nth=1) const
Default interface of nth derivative `ex::diff(s, n)`.
- int `compare` (const `basic` &other) const
Compare objects syntactically to establish canonical ordering.
- bool `is_equal` (const `basic` &other) const
Test for syntactic equality.
- const `basic` & `hold` () const
Stop further evaluation.
- unsigned `gethash` () const
- const `basic` & `setflag` (unsigned f) const
Set some `status_flags`.
- const `basic` & `clearflag` (unsigned f) const
Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Protected Member Functions

- `ex derivative` (const `symbol` &s) const override
Implementation of `ex::diff()` for an index always returns 0.
- bool `match_same_type` (const `basic` &other) const override
Returns true if the attributes of two objects are similar enough for a match.
- unsigned `calchash` () const override
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- void `print_index` (const `print_context` &c, unsigned level) const
- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_csrc` (const `print_csrc` &c, unsigned level) const
- void `do_print_latex` (const `print_latex` &c, unsigned level) const
- void `do_print_tree` (const `print_tree` &c, unsigned level) const

Protected Member Functions inherited from `GiNaC::basic`

- `basic` ()
- virtual `ex eval_ncmul` (const `exvector` &v) const
- virtual int `compare_same_type` (const `basic` &other) const
Returns order relation between two objects of same type.
- virtual bool `is_equal_same_type` (const `basic` &other) const
Returns true if two objects of same type are equal.
- void `ensure_if_modifiable` () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void `do_print` (const `print_context` &c, unsigned level) const
Default output to stream.
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
Tree output to stream.
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
Python parsable output to stream.

Protected Attributes

- [ex value](#)
Expression that constitutes the index (numeric or symbolic name)
- [ex dim](#)
Dimension of space (can be symbolic or numeric)

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.73.1 Detailed Description

This class holds one index of an indexed object.

Indices can theoretically consist of any symbolic expression but they are usually only just a symbol (e.g. "mu", "i") or numeric (integer). Indices belong to a space with a certain numeric or symbolic dimension.

6.73.2 Constructor & Destructor Documentation

6.73.2.1 [idx\(\)](#)

```
GiNaC::idx::idx (
    const ex & v,
    const ex & dim ) [explicit]
```

Construct index with given value and dimension.

Parameters

v	Value of index (numeric or symbolic)
dim	Dimension of index space (numeric or symbolic)

References [dim](#), [GiNaC::ex::info\(\)](#), [is_dim_numeric\(\)](#), and [GiNaC::info_flags::posint](#).

6.73.3 Member Function Documentation

6.73.3.1 [info\(\)](#)

```
bool GiNaC::idx::info (
    unsigned inf ) const [override], [virtual]
```

Information about the object.

See also

class [info_flags](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::info_flags::has_indices](#), and [GiNaC::info_flags::idx](#).

6.73.3.2 nops()

```
size_t GiNaC::idx::nops ( ) const [override], [virtual]
```

Number of operands/members.

Reimplemented from [GiNaC::basic](#).

6.73.3.3 op()

```
ex GiNaC::idx::op (
    size_t i ) const [override], [virtual]
```

Return operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [GINAC_ASSERT](#), and [value](#).

6.73.3.4 map()

```
ex GiNaC::idx::map (
    map_function & f ) const [override], [virtual]
```

Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).

Reimplemented from [GiNaC::basic](#).

References [GiNaC::are_ex_trivially_equal\(\)](#), [GiNaC::basic::clearflag\(\)](#), [GiNaC::basic::duplicate\(\)](#), [GiNaC::status_flags::hash_calculate](#) and [value](#).

6.73.3.5 evalf()

```
ex GiNaC::idx::evalf ( ) const [override], [virtual]
```

By default, [basic::evalf](#) would evaluate the index value but we don't want a.1 to become a.

(1.0).

Reimplemented from [GiNaC::basic](#).

6.73.3.6 subs()

```
ex GiNaC::idx::subs (
    const exmap & m,
    unsigned options = 0 ) const [override], [virtual]
```

Substitute a set of objects by arbitrary expressions.

The ex returned will already be evaluated.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::are_ex_trivially_equal\(\)](#), [GiNaC::basic::clearflag\(\)](#), [GiNaC::basic::duplicate\(\)](#), [GiNaC::ex::find\(\)](#), [GiNaC::status_flags::hash_calculated](#), [m](#), [options](#), [GiNaC::subs_options::really_subs_idx](#), [GiNaC::ex::subs\(\)](#), and [value](#).

6.73.3.7 archive()

```
void GiNaC::idx::archive (
    archive_node & n ) const [override], [virtual]
```

Save (serialize) the object into archive node.

Archive the object.

Losely speaking, this method turns an expression into a byte stream (which can be saved and restored later on, or sent via network, etc.)

Reimplemented from [GiNaC::basic](#).

Reimplemented in [GiNaC::varidx](#), and [GiNaC::spinidx](#).

References [dim](#), [n](#), and [value](#).

6.73.3.8 read_archive()

```
void GiNaC::idx::read_archive (
    const archive_node & n,
    lst & syms ) [override], [virtual]
```

Load (deserialize) the object from an archive node.

Construct object from [archive_node](#).

Note

This method is essentially a constructor. However, constructors can't be virtual. So, if unarchiving routines are implemented as constructors one would need to define such a constructor in every class, even if all it does is simply calling constructor of a superclass.

Reimplemented from [GiNaC::basic](#).

Reimplemented in [GiNaC::varidx](#), and [GiNaC::spinidx](#).

References [dim](#), [n](#), and [value](#).

6.73.3.9 derivative()

```
ex GiNaC::idx::derivative (
    const symbol & s ) const [override], [protected], [virtual]
```

Implementation of [ex::diff\(\)](#) for an index always returns 0.

See also

[ex::diff](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex0](#).

6.73.3.10 match_same_type()

```
bool GiNaC::idx::match_same_type (
    const basic & other ) const [override], [protected], [virtual]
```

Returns true if the attributes of two objects are similar enough for a match.

This function must not match subexpressions (this is already done by [basic::match\(\)](#)). Only attributes not accessible by [op\(\)](#) should be compared. This is also the reason why this function doesn't take the wildcard replacement list from [match\(\)](#) as an argument: only subexpressions are subject to wildcard matches. Also, this function only needs to be implemented for container classes because [is_equal_same_type\(\)](#) is automatically used instead of [match_same_type\(\)](#) if [nops\(\)](#) == 0.

See also

[basic::match](#)

Reimplemented from [GiNaC::basic](#).

Reimplemented in [GiNaC::varidx](#), and [GiNaC::spinidx](#).

References [dim](#), [GINAC_ASSERT](#), and [GiNaC::ex::is_equal\(\)](#).

6.73.3.11 calchash()

```
unsigned GiNaC::idx::calchash ( ) const [override], [protected], [virtual]
```

Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.

The method inherited from class basic computes a hash value based on the type and hash values of possible members. For this reason it is well suited for container classes but atomic classes should override this implementation because otherwise they would all end up with the same hashvalue.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::status_flags::evaluated](#), [GiNaC::basic::flags](#), [GiNaC::ex::gethash\(\)](#), [GiNaC::status_flags::hash_calculated](#), [GiNaC::basic::hashvalue](#), [GiNaC::make_hash_seed\(\)](#), [GiNaC::rotate_left\(\)](#), [GiNaC::basic::setflag\(\)](#), and [value](#).

6.73.3.12 is_dummy_pair_same_type()

```
bool GiNaC::idx::is_dummy_pair_same_type (
    const basic & other ) const [virtual]
```

Check whether the index forms a dummy index pair with another index of the same type.

Reimplemented in [GiNaC::varidx](#), and [GiNaC::spinidx](#).

References [dim](#), [GiNaC::ex::is_equal\(\)](#), and [value](#).

Referenced by [GiNaC::is_dummy_pair\(\)](#).

6.73.3.13 get_value()

```
ex GiNaC::idx::get_value ( ) const [inline]
```

Get value of index.

References [value](#).

Referenced by [GiNaC::matrix::eval_indexed\(\)](#), [GiNaC::tensdelta::eval_indexed\(\)](#), [GiNaC::minkmetric::eval_indexed\(\)](#), and [GiNaC::spinmetric::eval_indexed\(\)](#).

6.73.3.14 is_numeric()

```
bool GiNaC::idx::is_numeric ( ) const [inline]
```

Check whether the index is numeric.

References [value](#).

6.73.3.15 is_symbolic()

```
bool GiNaC::idx::is_symbolic ( ) const [inline]
```

Check whether the index is symbolic.

References [value](#).

Referenced by [GiNaC::spinmetric::contract_with\(\)](#), and [GiNaC::tensor::replace_contr_index\(\)](#).

6.73.3.16 get_dim()

```
ex GiNaC::idx::get_dim ( ) const [inline]
```

Get dimension of index space.

References [dim](#).

Referenced by [GiNaC::matrix::eval_indexed\(\)](#), [GiNaC::tensdelta::eval_indexed\(\)](#), and [GiNaC::tensmetric::eval_indexed\(\)](#).

6.73.3.17 is_dim_numeric()

```
bool GiNaC::idx::is_dim_numeric ( ) const [inline]
```

Check whether the dimension is numeric.

References [dim](#).

Referenced by [idx\(\)](#).

6.73.3.18 is_dim_symbolic()

```
bool GiNaC::idx::is_dim_symbolic ( ) const [inline]
```

Check whether the dimension is symbolic.

References [dim](#).

6.73.3.19 replace_dim()

```
ex GiNaC::idx::replace_dim (
    const ex & new_dim ) const
```

Make a new index with the same value but a different dimension.

References [GiNaC::basic::clearflag\(\)](#), [dim](#), [GiNaC::basic::duplicate\(\)](#), and [GiNaC::status_flags::hash_calculated](#).

Referenced by [GiNaC::tensdelta::eval_indexed\(\)](#), [GiNaC::tensmetric::eval_indexed\(\)](#), and [GiNaC::tensor::replace_contr_index\(\)](#).

6.73.3.20 minimal_dim()

```
ex GiNaC::idx::minimal_dim (
    const idx & other ) const
```

Return the minimum of the dimensions of this and another index.

If this is undecidable, throw an exception.

References [dim](#), and [GiNaC::minimal_dim\(\)](#).

Referenced by [GiNaC::tensdelta::eval_indexed\(\)](#), [GiNaC::tensmetric::eval_indexed\(\)](#), and [GiNaC::tensor::replace_contr_index\(\)](#).

6.73.3.21 print_index()

```
void GiNaC::idx::print_index (
    const print_context & c,
    unsigned level ) const [protected]
```

References [c](#), [dim](#), [GiNaC::ex::print\(\)](#), [GiNaC::print_options::print_index_dimensions](#), and [value](#).

Referenced by [do_print\(\)](#), [GiNaC::varidx::do_print\(\)](#), [GiNaC::spinidx::do_print\(\)](#), [do_print_latex\(\)](#), and [GiNaC::spinidx::do_print_latex\(\)](#).

6.73.3.22 do_print()

```
void GiNaC::idx::do_print (
    const print\_context & c,
    unsigned level ) const [protected]
```

References [c](#), and [print_index\(\)](#).

6.73.3.23 do_print_csrc()

```
void GiNaC::idx::do_print_csrc (
    const print\_csrc & c,
    unsigned level ) const [protected]
```

References [c](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::ex::print\(\)](#), and [value](#).

6.73.3.24 do_print_latex()

```
void GiNaC::idx::do_print_latex (
    const print\_latex & c,
    unsigned level ) const [protected]
```

References [c](#), and [print_index\(\)](#).

6.73.3.25 do_print_tree()

```
void GiNaC::idx::do_print_tree (
    const print\_tree & c,
    unsigned level ) const [protected]
```

References [c](#), [dim](#), [GiNaC::basic::flags](#), [GiNaC::basic::hashvalue](#), [GiNaC::ex::print\(\)](#), and [value](#).

6.73.4 Member Data Documentation

6.73.4.1 value

```
ex GiNaC::idx::value [protected]
```

Expression that constitutes the index (numeric or symbolic name)

Referenced by [archive\(\)](#), [calchash\(\)](#), [do_print_csrc\(\)](#), [do_print_tree\(\)](#), [GiNaC::varidx::do_print_tree\(\)](#), [GiNaC::spinidx::do_print_tree\(\)](#), [get_value\(\)](#), [is_dummy_pair_same_type\(\)](#), [is_numeric\(\)](#), [is_symbolic\(\)](#), [map\(\)](#), [op\(\)](#), [print_index\(\)](#), [read_archive\(\)](#), and [subs\(\)](#).

6.73.4.2 dim

`ex` GiNaC::idx::dim [protected]

Dimension of space (can be symbolic or numeric)

Referenced by [archive\(\)](#), [do_print_tree\(\)](#), [GiNaC::varidx::do_print_tree\(\)](#), [GiNaC::spinidx::do_print_tree\(\)](#), [get_dim\(\)](#), [idx\(\)](#), [is_dim_numeric\(\)](#), [is_dim_symbolic\(\)](#), [is_dummy_pair_same_type\(\)](#), [match_same_type\(\)](#), [minimal_dim\(\)](#), [print_index\(\)](#), [read_archive\(\)](#), and [replace_dim\(\)](#).

The documentation for this class was generated from the following files:

- [idx.h](#)
- [idx.cpp](#)

6.74 GiNaC::idx_is_equal_ignore_dim Struct Reference

Public Member Functions

- `bool operator()` (const `ex` &lh, const `ex` &rh) const

6.74.1 Member Function Documentation

6.74.1.1 operator()

```
bool GiNaC::idx_is_equal_ignore_dim::operator() (
    const ex & lh,
    const ex & rh ) const [inline]
```

References [GiNaC::ex::is_equal\(\)](#).

The documentation for this struct was generated from the following file:

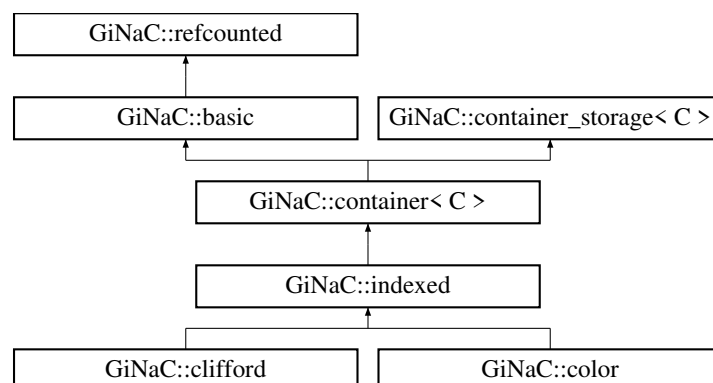
- [indexed.cpp](#)

6.75 GiNaC::indexed Class Reference

This class holds an indexed expression.

```
#include <indexed.h>
```

Inheritance diagram for GiNaC::indexed:



Public Member Functions

- [indexed](#) (const [ex](#) &b)
Construct indexed object with no index.
- [indexed](#) (const [ex](#) &b, const [ex](#) &i1)
Construct indexed object with one index.
- [indexed](#) (const [ex](#) &b, const [ex](#) &i1, const [ex](#) &i2)
Construct indexed object with two indices.
- [indexed](#) (const [ex](#) &b, const [ex](#) &i1, const [ex](#) &i2, const [ex](#) &i3)
Construct indexed object with three indices.
- [indexed](#) (const [ex](#) &b, const [ex](#) &i1, const [ex](#) &i2, const [ex](#) &i3, const [ex](#) &i4)
Construct indexed object with four indices.
- [indexed](#) (const [ex](#) &b, const [symmetry](#) &symm, const [ex](#) &i1, const [ex](#) &i2)
Construct indexed object with two indices and a specified symmetry.
- [indexed](#) (const [ex](#) &b, const [symmetry](#) &symm, const [ex](#) &i1, const [ex](#) &i2, const [ex](#) &i3)
Construct indexed object with three indices and a specified symmetry.
- [indexed](#) (const [ex](#) &b, const [symmetry](#) &symm, const [ex](#) &i1, const [ex](#) &i2, const [ex](#) &i3, const [ex](#) &i4)
Construct indexed object with four indices and a specified symmetry.
- [indexed](#) (const [ex](#) &b, const [exvector](#) &iv)
Construct indexed object with a specified vector of indices.
- [indexed](#) (const [ex](#) &b, const [symmetry](#) &symm, const [exvector](#) &iv)
Construct indexed object with a specified vector of indices and symmetry.
- [indexed](#) (const [symmetry](#) &symm, const [exprseq](#) &es)
- [indexed](#) (const [symmetry](#) &symm, const [exvector](#) &v)
- [indexed](#) (const [symmetry](#) &symm, [exvector](#) &&v)
- unsigned [precedence](#) () const override
Return relative operator precedence (for parenthezing output).
- bool [info](#) (unsigned inf) const override
Information about the object.
- [ex eval](#) () const override
Perform automatic non-interruptive term rewriting rules.
- [ex real_part](#) () const override
- [ex imag_part](#) () const override
- [exvector get_free_indices](#) () const override
Return a vector containing the free indices of an expression.
- void [archive](#) ([archive_node](#) &n) const override
Save (a.k.a.
- void [read_archive](#) (const [archive_node](#) &n, [lst](#) &syms) override
Read (a.k.a.
- bool [all_index_values_are](#) (unsigned inf) const
Check whether all index values have a certain property.
- [exvector get_indices](#) () const
Return a vector containing the object's indices.
- [exvector get_dummy_indices](#) () const
Return a vector containing the dummy indices of the object, if any.
- [exvector get_dummy_indices](#) (const [indexed](#) &other) const
Return a vector containing the dummy indices in the contraction with another indexed object.
- bool [has_dummy_index_for](#) (const [ex](#) &i) const
Check whether the object has an index that forms a dummy index pair with a given index.
- [ex get_symmetry](#) () const
Return symmetry properties.

Public Member Functions inherited from GiNaC::container< C >

- [container](#) (STLT const &s)
- [container](#) (STLT &&v)
- [container](#) (exvector::const_iterator b, exvector::const_iterator e)
- [container](#) (std::initializer_list< [ex](#) > il)
- [size_t nops](#) () const override
Number of operands/members.
- [ex op](#) (size_t i) const override
Return operand/member at position i.
- [ex & let_op](#) (size_t i) override
Return modifiable operand/member at position i.
- [ex subs](#) (const [exmap](#) &m, unsigned [options](#)=0) const override
Substitute a set of objects by arbitrary expressions.
- [container & prepend](#) (const [ex](#) &b)
Add element at front.
- [container & append](#) (const [ex](#) &b)
Add element at back.
- [container & remove_first](#) ()
Remove first element.
- [container & remove_last](#) ()
Remove last element.
- [container & remove_all](#) ()
Remove all elements.
- [container & sort](#) ()
Sort elements.
- [container & unique](#) ()
Remove adjacent duplicate elements.
- [const_iterator begin](#) () const
- [const_iterator end](#) () const
- [const_reverse_iterator rbegin](#) () const
- [const_reverse_iterator rend](#) () const

Public Member Functions inherited from GiNaC::basic

- virtual [~basic](#) ()
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- [basic](#) (const [basic](#) &other)
- const [basic & operator=](#) (const [basic](#) &other)
basic assignment operator: the other object might be of a derived class.
- virtual [basic * duplicate](#) () const
Create a clone of this object on the heap.
- virtual [ex evalf](#) () const
Evaluate object numerically.
- virtual [ex evalm](#) () const
Evaluate sums, products and integer powers of matrices.
- virtual [ex eval_integ](#) () const
Evaluate integrals, if result is known.
- virtual [ex eval_indexed](#) (const [basic](#) &i) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void [print](#) (const [print_context](#) &c, unsigned level=0) const

- Output to stream.*

 - virtual void `dbgprint` () const

Little wrapper around print to be called within a debugger.
- virtual void `dbgprinttree` () const

Little wrapper around printtree to be called within a debugger.
- virtual `ex operator[]` (const `ex` &index) const
- virtual `ex operator[]` (size_t i) const
- virtual `ex & operator[]` (const `ex` &index)
- virtual `ex & operator[]` (size_t i)
- virtual bool `has` (const `ex` &other, unsigned `options`=0) const
- Test for occurrence of a pattern.*

 - virtual bool `match` (const `ex` &pattern, `exmap` &repls) const

Check whether the expression matches a given pattern.
- virtual `ex map` (`map_function` &f) const
- Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).*

 - virtual void `accept` (`GiNaC::visitor` &v) const
 - virtual bool `is_polynomial` (const `ex` &var) const

Check whether this is a polynomial in the given variables.
- virtual int `degree` (const `ex` &s) const
- Return degree of highest power in object s.*

 - virtual int `ldegree` (const `ex` &s) const

Return degree of lowest power in object s.
- virtual `ex coeff` (const `ex` &s, int `n`=1) const
- Return coefficient of degree n in object s.*

 - virtual `ex collect` (const `ex` &s, bool `distributed`=false) const

Sort expanded expression in terms of powers of some object(s).
- virtual `ex series` (const `relational` &r, int `order`, unsigned `options`=0) const
- Default implementation of `ex::series()`.*

 - virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const

Default implementation of `ex::normal()`.
- virtual `ex to_rational` (`exmap` &repl) const
- Default implementation of `ex::to_rational()`.*

 - virtual `ex to_polynomial` (`exmap` &repl) const
 - virtual `numeric integer_content` () const
 - virtual `ex smod` (const `numeric` &xi) const

Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient` () const
- Implementation `ex::max_coefficient()`.*

 - virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const

Add two indexed expressions.
- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
- Multiply an indexed expression with a scalar.*

 - virtual bool `contract_with` (`exvector::iterator` self, `exvector::iterator` other, `exvector` &v) const

Try to contract two indexed expressions that appear in the same product.
- template<class T >
void `print_dispatch` (const `print_context` &c, unsigned level) const
- Like `print()`, but dispatch to the specified class.*

 - void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const

Like `print()`, but dispatch to the specified class.
- `ex subs_one_level` (const `exmap` &m, unsigned `options`) const
- Helper function for `subs()`.*

- `ex diff` (const `symbol` &s, unsigned nth=1) const
Default interface of nth derivative `ex::diff(s, n)`.
- int `compare` (const `basic` &other) const
Compare objects syntactically to establish canonical ordering.
- bool `is_equal` (const `basic` &other) const
Test for syntactic equality.
- const `basic` & `hold` () const
Stop further evaluation.
- unsigned `gethash` () const
- const `basic` & `setflag` (unsigned f) const
Set some `status_flags`.
- const `basic` & `clearflag` (unsigned f) const
Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Protected Member Functions

- `ex derivative` (const `symbol` &s) const override
Implementation of `ex::diff()` for an indexed object always returns 0.
- `ex thiscontainer` (const `exvector` &v) const override
- `ex thiscontainer` (`exvector` &&v) const override
- unsigned `return_type` () const override
- `return_type_t` `return_type_tinfo` () const override
- `ex expand` (unsigned `options`=0) const override
Expand expression, i.e.
- void `printindices` (const `print_context` &c, unsigned level) const
- void `print_indexed` (const `print_context` &c, const char *openbrace, const char *closebrace, unsigned level) const
- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_latex` (const `print_latex` &c, unsigned level) const
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
- void `validate` () const
Check whether all indices are of class `idx` and validate the symmetry tree.

Protected Member Functions inherited from [GiNaC::container< C >](#)

- [ex conjugate](#) () const override
- bool [is_equal_same_type](#) (const [basic](#) &other) const override
Returns true if two objects of same type are equal.
- virtual [ex thiscontainer](#) (const [STLT](#) &v) const
Similar to [duplicate\(\)](#), but with a preset sequence.
- virtual [ex thiscontainer](#) ([STLT](#) &&v) const
Similar to [duplicate\(\)](#), but with a preset sequence (which gets pilfered).
- virtual void [printseq](#) (const [print_context](#) &c, char openbracket, char delim, char closebracket, unsigned this←_precedence, unsigned upper_precedence=0) const
Print sequence of contained elements.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
- void [do_print_python](#) (const [print_python](#) &c, unsigned level) const
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
- [STLT](#) [subchildren](#) (const [exmap](#) &m, unsigned [options](#)=0) const

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Protected Member Functions inherited from [GiNaC::container_storage< C >](#)

- [container_storage](#) ()
- [container_storage](#) (size_t n, const [ex](#) &e)
- [container_storage](#) (std::initializer_list< [ex](#) > il)
- template<class In >
 [container_storage](#) (In b, In e)
- void [reserve](#) (size_t)
- [~container_storage](#) ()
- void [reserve](#) (size_t n)
- void [reserve](#) (std::vector< [ex](#) > &v, size_t n)

Protected Attributes

- [ex symtree](#)

Index symmetry (tree of symmetry objects)

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

Protected Attributes inherited from [GiNaC::container_storage< C >](#)

- [STLT seq](#)

Friends

- [ex simplify_indexed](#) (const [ex](#) &e, [exvector](#) &free_indices, [exvector](#) &dummy_indices, const [scalar_products](#) &sp)
Simplify indexed expression, return list of free indices.
- [ex simplify_indexed_product](#) (const [ex](#) &e, [exvector](#) &free_indices, [exvector](#) &dummy_indices, const [scalar_products](#) &sp)
Simplify product of indexed expressions (commutative, noncommutative and simple squares), return list of free indices.
- bool [reposition_dummy_indices](#) ([ex](#) &e, [exvector](#) &variant_dummy_indices, [exvector](#) &moved_indices)
Raise/lower dummy indices in a single indexed objects to canonicalize their variance.

Additional Inherited Members**Public Types inherited from [GiNaC::container< C >](#)**

- typedef [STLT::const_iterator](#) [const_iterator](#)
- typedef [STLT::const_reverse_iterator](#) [const_reverse_iterator](#)

Protected Types inherited from [GiNaC::container< C >](#)

- typedef [container_storage< C >::STLT](#) [STLT](#)

Protected Types inherited from [GiNaC::container_storage< C >](#)

- typedef [C< ex > STLT](#)

Static Protected Member Functions inherited from [GiNaC::container< C >](#)

- static unsigned [get_default_flags](#) ()
Specialization of [container::get_default_flags\(\)](#) for [lst](#).
- static char [get_open_delim](#) ()
Specialization of [container::get_open_delim\(\)](#) for [lst](#).
- static char [get_close_delim](#) ()
Specialization of [container::get_close_delim\(\)](#) for [lst](#).

Static Protected Member Functions inherited from [GiNaC::container_storage< C >](#)

- static void [reserve](#) (STLT &, size_t)

6.75.1 Detailed Description

This class holds an indexed expression.

It consists of a 'base' expression (the expression being indexed) which can be accessed as `op(0)`, and `n` ($n \geq 0$) indices (all of class `idx`), accessible as `op(1)..op(n)`.

6.75.2 Constructor & Destructor Documentation

6.75.2.1 `indexed()` [1/13]

```
GiNaC::indexed::indexed (
    const ex & b )
```

Construct indexed object with no index.

Parameters

<i>b</i>	Base expression

References [GiNaC::not_symmetric\(\)](#), and [validate\(\)](#).

6.75.2.2 `indexed()` [2/13]

```
GiNaC::indexed::indexed (
    const ex & b,
    const ex & i1 )
```

Construct indexed object with one index.

The index must be of class `idx`.

Parameters

<i>b</i>	Base expression
<i>i1</i>	The index

References [validate\(\)](#).

6.75.2.3 indexed() [3/13]

```
GiNaC::indexed::indexed (
    const ex & b,
    const ex & i1,
    const ex & i2 )
```

Construct indexed object with two indices.

The indices must be of class idx.

Parameters

<i>b</i>	Base expression
<i>i1</i>	First index
<i>i2</i>	Second index

References [validate\(\)](#).

6.75.2.4 indexed() [4/13]

```
GiNaC::indexed::indexed (
    const ex & b,
    const ex & i1,
    const ex & i2,
    const ex & i3 )
```

Construct indexed object with three indices.

The indices must be of class idx.

Parameters

<i>b</i>	Base expression
<i>i1</i>	First index
<i>i2</i>	Second index
<i>i3</i>	Third index

References [validate\(\)](#).

6.75.2.5 indexed() [5/13]

```
GiNaC::indexed::indexed (
    const ex & b,
    const ex & i1,
    const ex & i2,
    const ex & i3,
    const ex & i4 )
```

Construct indexed object with four indices.

The indices must be of class `idx`.

Parameters

<i>b</i>	Base expression
<i>i1</i>	First index
<i>i2</i>	Second index
<i>i3</i>	Third index
<i>i4</i>	Fourth index

References [validate\(\)](#).

6.75.2.6 indexed() [6/13]

```
GiNaC::indexed::indexed (
    const ex & b,
    const symmetry & symm,
    const ex & i1,
    const ex & i2 )
```

Construct indexed object with two indices and a specified symmetry.

The indices must be of class `idx`.

Parameters

<i>b</i>	Base expression
<i>symm</i>	Symmetry of indices
<i>i1</i>	First index
<i>i2</i>	Second index

References [validate\(\)](#).

6.75.2.7 indexed() [7/13]

```
GiNaC::indexed::indexed (
    const ex & b,
    const symmetry & symm,
    const ex & i1,
    const ex & i2,
    const ex & i3 )
```

Construct indexed object with three indices and a specified symmetry.

The indices must be of class `idx`.

Parameters

<i>b</i>	Base expression

Parameters

<i>symm</i>	Symmetry of indices
<i>i1</i>	First index
<i>i2</i>	Second index
<i>i3</i>	Third index

References [validate\(\)](#).

6.75.2.8 indexed() [8/13]

```
GiNaC::indexed::indexed (
    const ex & b,
    const symmetry & symm,
    const ex & i1,
    const ex & i2,
    const ex & i3,
    const ex & i4 )
```

Construct indexed object with four indices and a specified symmetry.

The indices must be of class `idx`.

Parameters

<i>b</i>	Base expression
<i>symm</i>	Symmetry of indices
<i>i1</i>	First index
<i>i2</i>	Second index
<i>i3</i>	Third index
<i>i4</i>	Fourth index

References [validate\(\)](#).

6.75.2.9 indexed() [9/13]

```
GiNaC::indexed::indexed (
    const ex & b,
    const exvector & iv )
```

Construct indexed object with a specified vector of indices.

The indices must be of class `idx`.

Parameters

<i>b</i>	Base expression
<i>iv</i>	Vector of indices

References [GiNaC::container_storage< C >::seq](#), and [validate\(\)](#).

6.75.2.10 indexed() [10/13]

```
GiNaC::indexed::indexed (
    const ex & b,
    const symmetry & symm,
    const exvector & iv )
```

Construct indexed object with a specified vector of indices and symmetry.

The indices must be of class `idx`.

Parameters

<i>b</i>	Base expression
<i>symm</i>	Symmetry of indices
<i>iv</i>	Vector of indices

References [GiNaC::container_storage< C >::seq](#), and [validate\(\)](#).

6.75.2.11 indexed() [11/13]

```
GiNaC::indexed::indexed (
    const symmetry & symm,
    const exprseq & es )
```

6.75.2.12 indexed() [12/13]

```
GiNaC::indexed::indexed (
    const symmetry & symm,
    const exvector & v )
```

6.75.2.13 indexed() [13/13]

```
GiNaC::indexed::indexed (
    const symmetry & symm,
    exvector && v )
```

6.75.3 Member Function Documentation**6.75.3.1 precedence()**

```
unsigned GiNaC::indexed::precedence ( ) const [inline], [override], [virtual]
```

Return relative operator precedence (for parenthezing output).

Reimplemented from [GiNaC::container< C >](#).

Referenced by [print_indexed\(\)](#).

6.75.3.2 info()

```
bool GiNaC::indexed::info (
    unsigned int ) const [override], [virtual]
```

Information about the object.

See also

class [info_flags](#)

Reimplemented from [GiNaC::container< C >](#).

References [GiNaC::info_flags::has_indices](#), [GiNaC::info_flags::indexed](#), and [GiNaC::container_storage< C >::seq](#).

Referenced by [imag_part\(\)](#), and [real_part\(\)](#).

6.75.3.3 eval()

```
ex GiNaC::indexed::eval ( ) const [override], [virtual]
```

Perform automatic non-interruptive term rewriting rules.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex0](#), [GiNaC::canonicalize\(\)](#), [GINAC_ASSERT](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), [GiNaC::container_storage< C >::seq](#), [symtree](#), and [thiscontainer\(\)](#).

6.75.3.4 real_part()

```
ex GiNaC::indexed::real_part ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::container< C >](#).

References [info\(\)](#), [GiNaC::container< C >::op\(\)](#), and [GiNaC::info_flags::real](#).

6.75.3.5 imag_part()

```
ex GiNaC::indexed::imag_part ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::container< C >](#).

References [info\(\)](#), [GiNaC::container< C >::op\(\)](#), and [GiNaC::info_flags::real](#).

6.75.3.6 get_free_indices()

```
exvector GiNaC::indexed::get_free_indices ( ) const [override], [virtual]
```

Return a vector containing the free indices of an expression.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::find_free_and_dummy\(\)](#), and [GiNaC::container_storage< C >::seq](#).

Referenced by [get_dummy_indices\(\)](#).

6.75.3.7 archive()

```
void GiNaC::indexed::archive (
    archive_node & n ) const [override], [virtual]
```

Save (a.k.a.

serialize) indexed object into archive.

Reimplemented from [GiNaC::container< C >](#).

References [n](#), and [symtree](#).

6.75.3.8 read_archive()

```
void GiNaC::indexed::read_archive (
    const archive_node & n,
    lst & syms ) [override], [virtual]
```

Read (a.k.a.

deserialize) indexed object from archive.

Reimplemented from [GiNaC::container< C >](#).

References [n](#), [GiNaC::not_symmetric\(\)](#), [GiNaC::container_storage< C >::seq](#), [GiNaC::sy_anti\(\)](#), [GiNaC::sy_symm\(\)](#), [GiNaC::syymm\(\)](#), [symtree](#), and [validate\(\)](#).

6.75.3.9 derivative()

```
ex GiNaC::indexed::derivative (
    const symbol & s ) const [override], [protected], [virtual]
```

Implementation of [ex::diff\(\)](#) for an indexed object always returns 0.

See also

[ex::diff](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex0](#).

6.75.3.10 thiscontainer() [1/2]

```
ex GiNaC::indexed::thiscontainer (
    const exvector & v ) const [override], [protected]
```

References [symtree](#).

Referenced by [eval\(\)](#), and [expand\(\)](#).

6.75.3.11 thiscontainer() [2/2]

```
ex GiNaC::indexed::thiscontainer (
    exvector && v ) const [override], [protected]
```

References [symtree](#).

6.75.3.12 return_type()

```
unsigned GiNaC::indexed::return_type ( ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::return_types::commutative](#), [GiNaC::container< C >::op\(\)](#), and [GiNaC::ex::return_type\(\)](#).

6.75.3.13 return_type_tinfo()

```
return_type_t GiNaC::indexed::return_type_tinfo ( ) const [inline], [override], [protected],
[virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::container< C >::op\(\)](#), and [GiNaC::ex::return_type_tinfo\(\)](#).

6.75.3.14 expand()

```
ex GiNaC::indexed::expand (
    unsigned options = 0 ) const [override], [protected], [virtual]
```

Expand expression, i.e.

multiply it out and return the result as a new expression.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex0](#), [GiNaC::are_ex_trivially_equal\(\)](#), [GiNaC::ex::expand\(\)](#), [GiNaC::expand_options::expand_indexed](#), [GINAC_ASSERT](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), [options](#), [GiNaC::container_storage< C >::seq](#), and [thiscontainer\(\)](#).

6.75.3.15 all_index_values_are()

```
bool GiNaC::indexed::all_index_values_are (
    unsigned inf ) const
```

Check whether all index values have a certain property.

See also

class [info_flags](#)

References [GiNaC::container_storage< C >::seq](#).

6.75.3.16 get_indices()

```
exvector GiNaC::indexed::get_indices ( ) const
```

Return a vector containing the object's indices.

References [GINAC_ASSERT](#), and [GiNaC::container_storage< C >::seq](#).

6.75.3.17 get_dummy_indices() [1/2]

```
exvector GiNaC::indexed::get_dummy_indices ( ) const
```

Return a vector containing the dummy indices of the object, if any.

References [GiNaC::find_free_and_dummy\(\)](#), and [GiNaC::container_storage< C >::seq](#).

6.75.3.18 get_dummy_indices() [2/2]

```
exvector GiNaC::indexed::get_dummy_indices (
    const indexed & other ) const
```

Return a vector containing the dummy indices in the contraction with another indexed object.

This is symmetric: `a.get_dummy_indices(b) == b.get_dummy_indices(a)`

References [GiNaC::find_dummy_indices\(\)](#), and [get_free_indices\(\)](#).

6.75.3.19 has_dummy_index_for()

```
bool GiNaC::indexed::has_dummy_index_for (
    const ex & i ) const
```

Check whether the object has an index that forms a dummy index pair with a given index.

References [GiNaC::is_dummy_pair\(\)](#), and [GiNaC::container_storage< C >::seq](#).

6.75.3.20 get_symmetry()

```
ex GiNaC::indexed::get_symmetry ( ) const [inline]
```

Return symmetry properties.

References [symtree](#).

Referenced by [GiNaC::clifford::get_metric\(\)](#).

6.75.3.21 printindices()

```
void GiNaC::indexed::printindices (
    const print\_context & c,
    unsigned level ) const [protected]
```

References [c](#), and [GiNaC::container_storage< C >::seq](#).

Referenced by [GiNaC::clifford::do_print_dflt\(\)](#), [GiNaC::clifford::do_print_tree\(\)](#), [do_print_tree\(\)](#), and [print_indexed\(\)](#).

6.75.3.22 print_indexed()

```
void GiNaC::indexed::print_indexed (
    const print\_context & c,
    const char * openbrace,
    const char * closebrace,
    unsigned level ) const [protected]
```

References [c](#), [precedence\(\)](#), [printindices\(\)](#), and [GiNaC::container_storage< C >::seq](#).

Referenced by [do_print\(\)](#), and [do_print_latex\(\)](#).

6.75.3.23 do_print()

```
void GiNaC::indexed::do_print (
    const print\_context & c,
    unsigned level ) const [protected]
```

References [c](#), and [print_indexed\(\)](#).

6.75.3.24 do_print_latex()

```
void GiNaC::indexed::do_print_latex (
    const print\_latex & c,
    unsigned level ) const [protected]
```

References [c](#), and [print_indexed\(\)](#).

6.75.3.25 do_print_tree()

```
void GiNaC::indexed::do_print_tree (
    const print\_tree & c,
    unsigned level ) const [protected]
```

References [c](#), [GiNaC::basic::flags](#), [GiNaC::basic::hashvalue](#), [GiNaC::ex::print\(\)](#), [printindices\(\)](#), [GiNaC::container_storage< C >::seq](#), and [symtree](#).

6.75.3.26 validate()

```
void GiNaC::indexed::validate ( ) const [protected]
```

Check whether all indices are of class `idx` and validate the symmetry tree.

This function is used internally to make sure that all constructed indexed objects really carry indices and not some other classes.

References [GINAC_ASSERT](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::container_storage< C >::seq](#), [symtree](#), and [validate\(\)](#).

Referenced by [indexed\(\)](#), [indexed\(\)](#), [indexed\(\)](#), [indexed\(\)](#), [indexed\(\)](#), [indexed\(\)](#), [indexed\(\)](#), [indexed\(\)](#), [indexed\(\)](#), [indexed\(\)](#), [read_archive\(\)](#), and [validate\(\)](#).

6.75.4 Friends And Related Symbol Documentation

6.75.4.1 simplify_indexed

```
ex simplify_indexed (
    const ex & e,
    exvector & free_indices,
    exvector & dummy_indices,
    const scalar_products & sp ) [friend]
```

Simplify indexed expression, return list of free indices.

Referenced by [GiNaC::clifford::get_metric\(\)](#), and [GiNaC::clifford::same_metric\(\)](#).

6.75.4.2 simplify_indexed_product

```
ex simplify_indexed_product (
    const ex & e,
    exvector & free_indices,
    exvector & dummy_indices,
    const scalar_products & sp ) [friend]
```

Simplify product of indexed expressions (commutative, noncommutative and simple squares), return list of free indices.

6.75.4.3 reposition_dummy_indices

```
bool reposition_dummy_indices (
    ex & e,
    exvector & variant_dummy_indices,
    exvector & moved_indices ) [friend]
```

Raise/lower dummy indices in a single indexed objects to canonicalize their variance.

Parameters

<code>e</code>	Object to work on
<code>variant_dummy_indices</code>	The set of indices that might need repositioning (will be changed by this function)
<code>moved_indices</code>	The set of indices that have been repositioned (will be changed by this function)

Returns

true if 'e' was changed

6.75.5 Member Data Documentation**6.75.5.1 symtree**

```
ex GiNaC::indexed::symtree [protected]
```

Index symmetry (tree of symmetry objects)

Referenced by [archive\(\)](#), [GiNaC::clifford::do_print_tree\(\)](#), [do_print_tree\(\)](#), [eval\(\)](#), [get_symmetry\(\)](#), [read_archive\(\)](#), [thiscontainer\(\)](#), [thiscontainer\(\)](#), and [validate\(\)](#).

The documentation for this class was generated from the following files:

- [indexed.h](#)
- [indexed.cpp](#)

6.76 GiNaC::info_flags Class Reference

Possible attributes an object can have.

```
#include <flags.h>
```

Public Types

- enum {
[numeric](#) , [real](#) , [rational](#) , [integer](#) ,
[crational](#) , [cinteger](#) , [positive](#) , [negative](#) ,
[nonnegative](#) , [posint](#) , [negint](#) , [nonnegint](#) ,
[even](#) , [odd](#) , [prime](#) , [relation](#) ,
[relation_equal](#) , [relation_not_equal](#) , [relation_less](#) , [relation_less_or_equal](#) ,
[relation_greater](#) , [relation_greater_or_equal](#) , [symbol](#) , [list](#) ,
[exprseq](#) , [polynomial](#) , [integer_polynomial](#) , [cinteger_polynomial](#) ,
[rational_polynomial](#) , [crational_polynomial](#) , [rational_function](#) , [indexed](#) ,
[has_indices](#) , [idx](#) , [expanded](#) , [indefinite](#) }

6.76.1 Detailed Description

Possible attributes an object can have.

6.76.2 Member Enumeration Documentation**6.76.2.1 anonymous enum**

```
anonymous enum
```

Enumerator

numeric	
real	
rational	
integer	
crational	
cinteger	
positive	
negative	
nonnegative	
posint	
negint	
nonnegint	
even	
odd	
prime	
relation	
relation_equal	
relation_not_equal	
relation_less	
relation_less_or_equal	
relation_greater	
relation_greater_or_equal	
symbol	
list	
exprseq	
polynomial	
integer_polynomial	
cinteger_polynomial	
rational_polynomial	
crational_polynomial	
rational_function	
indexed	
has_indices	
idx	
expanded	
indefinite	

The documentation for this class was generated from the following file:

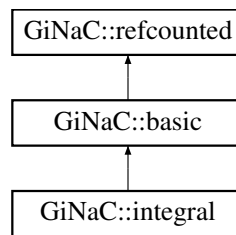
- [flags.h](#)

6.77 GiNaC::integral Class Reference

Symbolic integral.

```
#include <integral.h>
```

Inheritance diagram for GiNaC::integral:



Public Member Functions

- `integral` (const `ex` &x_, const `ex` &a_, const `ex` &b_, const `ex` &f_)
- unsigned `precedence` () const override
Return relative operator precedence (for parenthezing output).
- `ex eval` () const override
Perform automatic non-interruptive term rewriting rules.
- `ex evalf` () const override
Evaluate object numerically.
- int `degree` (const `ex` &s) const override
Return degree of highest power in object s.
- int `ldegree` (const `ex` &s) const override
Return degree of lowest power in object s.
- `ex eval_ncmul` (const `exvector` &v) const override
- size_t `nops` () const override
Number of operands/members.
- `ex op` (size_t i) const override
Return operand/member at position i.
- `ex & let_op` (size_t i) override
Return modifiable operand/member at position i.
- `ex expand` (unsigned `options`=0) const override
Expand expression, i.e.
- `exvector get_free_indices` () const override
Return a vector containing the free indices of an expression.
- unsigned `return_type` () const override
- `return_type_t return_type_tinfo` () const override
- `ex conjugate` () const override
- `ex eval_integ` () const override
Evaluate integrals, if result is known.
- void `archive` (`archive_node` &n) const override
Save (a.k.a.
- void `read_archive` (const `archive_node` &n, `lst` &syms) override
Read (a.k.a.

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic ()`
basic destructor, virtual because class `ex` will delete objects of derived classes via a `basic*`.
- `basic (const basic &other)`
- `const basic & operator= (const basic &other)`
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate () const`
Create a clone of this object on the heap.
- virtual `ex evalm () const`
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_indexed (const basic &i) const`
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual `void print (const print_context &c, unsigned level=0) const`
Output to stream.
- virtual `void dbgprint () const`
Little wrapper around `print` to be called within a debugger.
- virtual `void dbgprinttree () const`
Little wrapper around `printtree` to be called within a debugger.
- virtual `bool info (unsigned inf) const`
Information about the object.
- virtual `ex operator[] (const ex &index) const`
- virtual `ex operator[] (size_t i) const`
- virtual `ex & operator[] (const ex &index)`
- virtual `ex & operator[] (size_t i)`
- virtual `bool has (const ex &other, unsigned options=0) const`
Test for occurrence of a pattern.
- virtual `bool match (const ex &pattern, exmap &repls) const`
Check whether the expression matches a given pattern.
- virtual `ex subs (const exmap &m, unsigned options=0) const`
Substitute a set of objects by arbitrary expressions.
- virtual `ex map (map_function &f) const`
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual `void accept (GiNaC::visitor &v) const`
- virtual `bool is_polynomial (const ex &var) const`
Check whether this is a polynomial in the given variables.
- virtual `ex coeff (const ex &s, int n=1) const`
Return coefficient of degree `n` in object `s`.
- virtual `ex collect (const ex &s, bool distributed=false) const`
Sort expanded expression in terms of powers of some object(s).
- virtual `ex normal (exmap &repl, exmap &rev_lookup, lst &modifier) const`
Default implementation of `ex::normal()`.
- virtual `ex to_rational (exmap &repl) const`
Default implementation of `ex::to_rational()`.
- virtual `ex to_polynomial (exmap &repl) const`
- virtual `numeric integer_content () const`
- virtual `ex smod (const numeric &xi) const`
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient () const`
Implementation `ex::max_coefficient()`.
- virtual `ex add_indexed (const ex &self, const ex &other) const`

- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
Add two indexed expressions.
- virtual `bool contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const
Multiply an indexed expression with a scalar.
- virtual `ex real_part` () const
Try to contract two indexed expressions that appear in the same product.
- virtual `ex imag_part` () const
- template<class T >
void `print_dispatch` (const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- `ex subs_one_level` (const `exmap` &m, unsigned `options`) const
Helper function for `subs()`.
- `ex diff` (const `symbol` &s, unsigned nth=1) const
Default interface of nth derivative `ex::diff(s, n)`.
- int `compare` (const `basic` &other) const
Compare objects syntactically to establish canonical ordering.
- bool `is_equal` (const `basic` &other) const
Test for syntactic equality.
- const `basic` & `hold` () const
Stop further evaluation.
- unsigned `gethash` () const
- const `basic` & `setflag` (unsigned f) const
Set some `status_flags`.
- const `basic` & `clearflag` (unsigned f) const
Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Static Public Attributes

- static int `max_integration_level` = 15
- static `ex relative_integration_error` = 1e-8

Protected Member Functions

- `ex derivative` (const `symbol` &s) const override
Default implementation of `ex::diff()`.
- `ex series` (const `relational` &r, int `order`, unsigned `options`=0) const override
Default implementation of `ex::series()`.
- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_latex` (const `print_latex` &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Private Attributes

- [ex](#) x
- [ex](#) a
- [ex](#) b
- [ex](#) f

Additional Inherited Members

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.77.1 Detailed Description

Symbolic integral.

6.77.2 Constructor & Destructor Documentation

6.77.2.1 [integral\(\)](#)

```
GiNaC::integral::integral (
    const ex & x_,
    const ex & a_,
    const ex & b_,
    const ex & f_ )
```

References [x](#).

6.77.3 Member Function Documentation

6.77.3.1 precedence()

```
unsigned GiNaC::integral::precedence ( ) const [inline], [override], [virtual]
```

Return relative operator precedence (for parenthezing output).

Reimplemented from [GiNaC::basic](#).

Referenced by [do_print_latex\(\)](#).

6.77.3.2 eval()

```
ex GiNaC::integral::eval ( ) const [override], [virtual]
```

Perform automatic non-interruptive term rewriting rules.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex0](#), [a](#), [b](#), [GiNaC::status_flags::evaluated](#), [f](#), [GiNaC::basic::flags](#), [GiNaC::ex::has\(\)](#), [GiNaC::haswild\(\)](#), [GiNaC::basic::hold\(\)](#), and [x](#).

6.77.3.3 evalf()

```
ex GiNaC::integral::evalf ( ) const [override], [virtual]
```

Evaluate object numerically.

Reimplemented from [GiNaC::basic](#).

References [a](#), [GiNaC::adaptivesimpson\(\)](#), [GiNaC::are_ex_trivially_equal\(\)](#), [b](#), [GiNaC::ex::evalf\(\)](#), [f](#), [GiNaC::ex::subs\(\)](#), and [x](#).

6.77.3.4 degree()

```
int GiNaC::integral::degree (
    const ex & s ) const [override], [virtual]
```

Return degree of highest power in object s.

Reimplemented from [GiNaC::basic](#).

References [a](#), [b](#), [GiNaC::ex::degree\(\)](#), and [f](#).

6.77.3.5 ldegree()

```
int GiNaC::integral::ldegree (
    const ex & s ) const [override], [virtual]
```

Return degree of lowest power in object s.

Reimplemented from [GiNaC::basic](#).

References [a](#), [b](#), and [f](#).

6.77.3.6 eval_ncmul()

```
ex GiNaC::integral::eval_ncmul (
    const exvector & v ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::eval_ncmul\(\)](#), and [f](#).

6.77.3.7 nops()

```
size_t GiNaC::integral::nops ( ) const [override], [virtual]
```

Number of operands/members.

Reimplemented from [GiNaC::basic](#).

6.77.3.8 op()

```
ex GiNaC::integral::op (
    size_t i ) const [override], [virtual]
```

Return operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [a](#), [b](#), [f](#), [GINAC_ASSERT](#), and [x](#).

6.77.3.9 let_op()

```
ex & GiNaC::integral::let_op (
    size_t i ) [override], [virtual]
```

Return modifiable operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [a](#), [b](#), [GiNaC::basic::ensure_if_modifiable\(\)](#), [f](#), and [x](#).

6.77.3.10 expand()

```
ex GiNaC::integral::expand (
    unsigned options = 0 ) const [override], [virtual]
```

Expand expression, i.e.

multiply it out and return the result as a new expression.

Reimplemented from [GiNaC::basic](#).

References [a](#), [GiNaC::are_ex_trivially_equal\(\)](#), [b](#), [GiNaC::basic::ex](#), [GiNaC::ex::expand\(\)](#), [expand\(\)](#), [GiNaC::status_flags::expanded](#), [f](#), [GiNaC::basic::flags](#), [GiNaC::ex::has\(\)](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), [options](#), [GiNaC::basic::setflag\(\)](#), and [x](#).

Referenced by [eval_integ\(\)](#), and [expand\(\)](#).

6.77.3.11 `get_free_indices()`

```
exvector GiNaC::integral::get_free_indices ( ) const [override], [virtual]
```

Return a vector containing the free indices of an expression.

Reimplemented from [GiNaC::basic](#).

References [a](#), [b](#), [f](#), and [GiNaC::ex::get_free_indices\(\)](#).

6.77.3.12 `return_type()`

```
unsigned GiNaC::integral::return_type ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [f](#), and [GiNaC::ex::return_type\(\)](#).

6.77.3.13 `return_type_tinfo()`

```
return_type_t GiNaC::integral::return_type_tinfo ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [f](#), and [GiNaC::ex::return_type_tinfo\(\)](#).

6.77.3.14 `conjugate()`

```
ex GiNaC::integral::conjugate ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [a](#), [GiNaC::are_ex_trivially_equal\(\)](#), [b](#), [GiNaC::ex::conjugate\(\)](#), [f](#), [GiNaC::ex::subs\(\)](#), and [x](#).

6.77.3.15 `eval_integ()`

```
ex GiNaC::integral::eval_integ ( ) const [override], [virtual]
```

Evaluate integrals, if result is known.

Reimplemented from [GiNaC::basic](#).

References [a](#), [b](#), [GiNaC::ex::eval_integ\(\)](#), [expand\(\)](#), [GiNaC::status_flags::expanded](#), [f](#), [GiNaC::basic::flags](#), [GiNaC::ex::has\(\)](#), [GiNaC::log\(\)](#), [GiNaC::ex::op\(\)](#), [GiNaC::ex::subs\(\)](#), and [x](#).

6.77.3.16 archive()

```
void GiNaC::integral::archive (
    archive_node & n ) const [override], [virtual]
```

Save (a.k.a.

serialize) object into archive.

Reimplemented from [GiNaC::basic](#).

References [a](#), [b](#), [f](#), [n](#), and [x](#).

6.77.3.17 read_archive()

```
void GiNaC::integral::read_archive (
    const archive_node & n,
    lst & syms ) [override], [virtual]
```

Read (a.k.a.

deserialize) object from archive.

Reimplemented from [GiNaC::basic](#).

References [a](#), [b](#), [f](#), [n](#), and [x](#).

6.77.3.18 derivative()

```
ex GiNaC::integral::derivative (
    const symbol & s ) const [override], [protected], [virtual]
```

Default implementation of [ex::diff\(\)](#).

It maps the operation on the operands (or returns 0 when the object has no operands).

See also

[ex::diff](#)

Reimplemented from [GiNaC::basic](#).

References [a](#), [b](#), [GiNaC::ex::diff\(\)](#), [f](#), [GiNaC::ex::subs\(\)](#), and [x](#).

6.77.3.19 series()

```
ex GiNaC::integral::series (
    const relational & r,
    int order,
    unsigned options = 0 ) const [override], [protected], [virtual]
```

Default implementation of [ex::series\(\)](#).

This performs Taylor expansion.

See also

[ex::series](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex1](#), [a](#), [b](#), [f](#), [GiNaC::is_order_function\(\)](#), [GiNaC::ex::nops\(\)](#), [options](#), [order](#), [r](#), [GiNaC::ex::series\(\)](#), [series\(\)](#), [GiNaC::ex::subs\(\)](#), and [x](#).

Referenced by [series\(\)](#).

6.77.3.20 do_print()

```
void GiNaC::integral::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

References [a](#), [b](#), [c](#), [f](#), [GiNaC::ex::print\(\)](#), and [x](#).

6.77.3.21 do_print_latex()

```
void GiNaC::integral::do_print_latex (
    const print_latex & c,
    unsigned level ) const [protected]
```

References [a](#), [b](#), [c](#), [f](#), [precedence\(\)](#), [GiNaC::ex::print\(\)](#), and [x](#).

6.77.4 Member Data Documentation

6.77.4.1 max_integration_level

```
int GiNaC::integral::max_integration_level = 15 [static]
```

Referenced by [GiNaC::adaptivesimpson\(\)](#).

6.77.4.2 relative_integration_error

```
ex GiNaC::integral::relative_integration_error = 1e-8 [static]
```

6.77.4.3 x

`ex GiNaC::integral::x [private]`

Referenced by [archive\(\)](#), [conjugate\(\)](#), [derivative\(\)](#), [do_print\(\)](#), [do_print_latex\(\)](#), [eval\(\)](#), [eval_integ\(\)](#), [evalf\(\)](#), [expand\(\)](#), [integral\(\)](#), [let_op\(\)](#), [op\(\)](#), [read_archive\(\)](#), and [series\(\)](#).

6.77.4.4 a

`ex GiNaC::integral::a [private]`

Referenced by [archive\(\)](#), [conjugate\(\)](#), [degree\(\)](#), [derivative\(\)](#), [do_print\(\)](#), [do_print_latex\(\)](#), [eval\(\)](#), [eval_integ\(\)](#), [evalf\(\)](#), [expand\(\)](#), [get_free_indices\(\)](#), [ldegree\(\)](#), [let_op\(\)](#), [op\(\)](#), [read_archive\(\)](#), and [series\(\)](#).

6.77.4.5 b

`ex GiNaC::integral::b [private]`

Referenced by [archive\(\)](#), [conjugate\(\)](#), [degree\(\)](#), [derivative\(\)](#), [do_print\(\)](#), [do_print_latex\(\)](#), [eval\(\)](#), [eval_integ\(\)](#), [evalf\(\)](#), [expand\(\)](#), [get_free_indices\(\)](#), [ldegree\(\)](#), [let_op\(\)](#), [op\(\)](#), [read_archive\(\)](#), and [series\(\)](#).

6.77.4.6 f

`ex GiNaC::integral::f [private]`

Referenced by [archive\(\)](#), [conjugate\(\)](#), [degree\(\)](#), [derivative\(\)](#), [do_print\(\)](#), [do_print_latex\(\)](#), [eval\(\)](#), [eval_integ\(\)](#), [eval_ncmul\(\)](#), [evalf\(\)](#), [expand\(\)](#), [get_free_indices\(\)](#), [ldegree\(\)](#), [let_op\(\)](#), [op\(\)](#), [read_archive\(\)](#), [return_type\(\)](#), [return_type_tinfo\(\)](#), and [series\(\)](#).

The documentation for this class was generated from the following files:

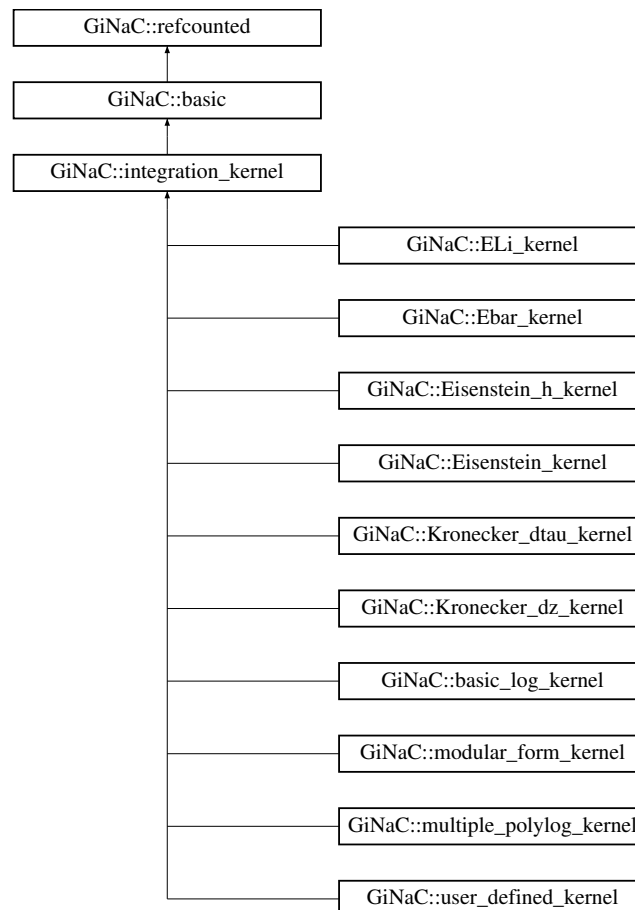
- [integral.h](#)
- [indexed.cpp](#)
- [integral.cpp](#)
- [pseries.cpp](#)

6.78 GiNaC::integration_kernel Class Reference

The base class for integration kernels for iterated integrals.

```
#include <integration_kernel.h>
```

Inheritance diagram for GiNaC::integration_kernel:



Public Member Functions

- `ex series` (const relational &r, int order, unsigned options=0) const override
Default implementation of `ex::series()`.
- virtual bool `has_trailing_zero` (void) const
This routine returns true, if the integration kernel has a trailing zero.
- virtual bool `is_numeric` (void) const
This routine returns true, if the integration kernel can be evaluated numerically.
- virtual `ex Laurent_series` (const ex &x, int order) const
Returns the Laurent series, starting possibly with the pole term.
- virtual `ex get_numerical_value` (const ex &lambda, int N_trunc=0) const
Evaluates the integrand at lambda.
- size_t `get_cache_size` (void) const
Returns the current size of the cache.
- void `set_cache_step` (int cache_steps) const
Sets the step size by which the cache is increased.

- `ex get_series_coeff (int i) const`
Wrapper around `series_coeff(i)`, converts `cl_N` to numeric.
- `cln::cl_N series_coeff (int i) const`
Subclasses have either to implement `series_coeff_impl` or the two methods `Laurent_series` and `uses_Laurent_series`.

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic ()`
basic destructor, virtual because class `ex` will delete objects of derived classes via a `basic*`.
- `basic (const basic &other)`
- const `basic & operator= (const basic &other)`
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate () const`
Create a clone of this object on the heap.
- virtual `ex eval () const`
Perform automatic non-interruptive term rewriting rules.
- virtual `ex evalf () const`
Evaluate object numerically.
- virtual `ex evalm () const`
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ () const`
Evaluate integrals, if result is known.
- virtual `ex eval_indexed (const basic &i) const`
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void `print (const print_context &c, unsigned level=0) const`
Output to stream.
- virtual void `dbgprint () const`
Little wrapper around `print` to be called within a debugger.
- virtual void `dbgprinttree () const`
Little wrapper around `printtree` to be called within a debugger.
- virtual unsigned `precedence () const`
Return relative operator precedence (for parenthezing output).
- virtual bool `info (unsigned inf) const`
Information about the object.
- virtual `size_t nops () const`
Number of operands/members.
- virtual `ex op (size_t i) const`
Return operand/member at position `i`.
- virtual `ex operator[] (const ex &index) const`
- virtual `ex operator[] (size_t i) const`
- virtual `ex & let_op (size_t i)`
Return modifiable operand/member at position `i`.
- virtual `ex & operator[] (const ex &index)`
- virtual `ex & operator[] (size_t i)`
- virtual bool `has (const ex &other, unsigned options=0) const`
Test for occurrence of a pattern.
- virtual bool `match (const ex &pattern, exmap &repls) const`
Check whether the expression matches a given pattern.
- virtual `ex subs (const exmap &m, unsigned options=0) const`
Substitute a set of objects by arbitrary expressions.

- virtual `ex map` (`map_function` &`f`) const
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void `accept` (`GiNaC::visitor` &`v`) const
- virtual bool `is_polynomial` (const `ex` &`var`) const
Check whether this is a polynomial in the given variables.
- virtual int `degree` (const `ex` &`s`) const
Return degree of highest power in object s.
- virtual int `ldegree` (const `ex` &`s`) const
Return degree of lowest power in object s.
- virtual `ex coeff` (const `ex` &`s`, int `n`=1) const
Return coefficient of degree n in object s.
- virtual `ex expand` (unsigned `options`=0) const
Expand expression, i.e.
- virtual `ex collect` (const `ex` &`s`, bool `distributed`=false) const
Sort expanded expression in terms of powers of some object(s).
- virtual `ex normal` (`exmap` &`repl`, `exmap` &`rev_lookup`, `lst` &`modifier`) const
Default implementation of `ex::normal()`.
- virtual `ex to_rational` (`exmap` &`repl`) const
Default implementation of `ex::to_rational()`.
- virtual `ex to_polynomial` (`exmap` &`repl`) const
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &`xi`) const
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient` () const
Implementation `ex::max_coefficient()`.
- virtual `exvector get_free_indices` () const
Return a vector containing the free indices of an expression.
- virtual `ex add_indexed` (const `ex` &`self`, const `ex` &`other`) const
Add two indexed expressions.
- virtual `ex scalar_mul_indexed` (const `ex` &`self`, const `numeric` &`other`) const
Multiply an indexed expression with a scalar.
- virtual bool `contract_with` (`exvector::iterator` `self`, `exvector::iterator` `other`, `exvector` &`v`) const
Try to contract two indexed expressions that appear in the same product.
- virtual unsigned `return_type` () const
- virtual `return_type_t return_type_tinfo` () const
- virtual `ex conjugate` () const
- virtual `ex real_part` () const
- virtual `ex imag_part` () const
- template<class T >
void `print_dispatch` (const `print_context` &`c`, unsigned `level`) const
Like `print()`, but dispatch to the specified class.
- void `print_dispatch` (const `registered_class_info` &`ri`, const `print_context` &`c`, unsigned `level`) const
Like `print()`, but dispatch to the specified class.
- virtual void `archive` (`archive_node` &`n`) const
Save (serialize) the object into archive node.
- virtual void `read_archive` (const `archive_node` &`n`, `lst` &`syms`)
Load (deserialize) the object from an archive node.
- `ex subs_one_level` (const `exmap` &`m`, unsigned `options`) const
Helper function for `subs()`.
- `ex diff` (const `symbol` &`s`, unsigned `nth`=1) const
Default interface of nth derivative `ex::diff(s, n)`.

- int `compare` (const `basic` &other) const
Compare objects syntactically to establish canonical ordering.
- bool `is_equal` (const `basic` &other) const
Test for syntactic equality.
- const `basic` & `hold` () const
Stop further evaluation.
- unsigned `gethash` () const
- const `basic` & `setflag` (unsigned f) const
Set some `status_flags`.
- const `basic` & `clearflag` (unsigned f) const
Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Protected Member Functions

- virtual bool `uses_Laurent_series` () const
Returns true, if the coefficients are computed from the Laurent series (in which case the method `Laurent_series` needs to be implemented).
- virtual `cln::cl_N` `series_coeff_impl` (int i) const
For $\omega = d\lambda$ only the coefficient of λ^0 is non-zero.
- `ex` `get_numerical_value_impl` (const `ex` &lambda, const `ex` &pre, int shift, int N_trunc) const
The actual implementation for computing a numerical value for the integrand.
- void `do_print` (const `print_context` &c, unsigned level) const

Protected Member Functions inherited from `GiNaC::basic`

- `basic` ()
- virtual `ex` `eval_ncmul` (const `exvector` &v) const
- virtual bool `match_same_type` (const `basic` &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual `ex` `derivative` (const `symbol` &s) const
Default implementation of `ex::diff()`.
- virtual int `compare_same_type` (const `basic` &other) const
Returns order relation between two objects of same type.
- virtual bool `is_equal_same_type` (const `basic` &other) const
Returns true if two objects of same type are equal.
- virtual unsigned `calchash` () const
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- void `ensure_if_modifiable` () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void `do_print` (const `print_context` &c, unsigned level) const
Default output to stream.
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
Tree output to stream.
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
Python parsable output to stream.

Protected Attributes

- int [cache_step_size](#)
- std::vector< cln::cl_N > [series_vec](#)

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.78.1 Detailed Description

The base class for integration kernels for iterated integrals.

This class represents the differential one-form

$$\omega = d\lambda$$

The integration variable is a dummy variable and does not need to be specified.

6.78.2 Member Function Documentation

6.78.2.1 [series\(\)](#)

```
ex GiNaC::integration_kernel::series (
    const relational & r,
    int order,
    unsigned options = 0 ) const [override], [virtual]
```

Default implementation of [ex::series\(\)](#).

This performs Taylor expansion.

See also

[ex::series](#)

Reimplemented from [GiNaC::basic](#).

Reimplemented in [GiNaC::modular_form_kernel](#).

References [order](#), and [r](#).

6.78.2.2 [has_trailing_zero\(\)](#)

```
bool GiNaC::integration_kernel::has_trailing_zero (
    void ) const [virtual]
```

This routine returns true, if the integration kernel has a trailing zero.

6.78.2.3 is_numeric()

```
bool GiNaC::integration_kernel::is_numeric (
    void ) const [virtual]
```

This routine returns true, if the integration kernel can be evaluated numerically.

Reimplemented in [GiNaC::multiple_polylog_kernel](#), [GiNaC::ELi_kernel](#), [GiNaC::Ebar_kernel](#), [GiNaC::Kronecker_dtau_kernel](#), [GiNaC::Kronecker_dz_kernel](#), [GiNaC::Eisenstein_kernel](#), [GiNaC::Eisenstein_h_kernel](#), [GiNaC::modular_form_kernel](#), and [GiNaC::user_defined_kernel](#).

6.78.2.4 Laurent_series()

```
ex GiNaC::integration_kernel::Laurent_series (
    const ex & x,
    int order ) const [virtual]
```

Returns the Laurent series, starting possibly with the pole term.

Neglected terms are of order $O(x^{order})$.

Reimplemented in [GiNaC::modular_form_kernel](#), [GiNaC::Eisenstein_kernel](#), [GiNaC::Eisenstein_h_kernel](#), and [GiNaC::user_defined_kernel](#).

References [n](#), [order](#), [GiNaC::pow\(\)](#), [GiNaC::ex::series\(\)](#), and [x](#).

6.78.2.5 get_numerical_value()

```
ex GiNaC::integration_kernel::get_numerical_value (
    const ex & lambda,
    int N_trunc = 0 ) const [virtual]
```

Evaluates the integrand at lambda.

Reimplemented in [GiNaC::ELi_kernel](#), [GiNaC::Ebar_kernel](#), [GiNaC::Kronecker_dtau_kernel](#), [GiNaC::Eisenstein_kernel](#), [GiNaC::Eisenstein_h_kernel](#), [GiNaC::modular_form_kernel](#), and [GiNaC::Kronecker_dz_kernel](#).

6.78.2.6 uses_Laurent_series()

```
bool GiNaC::integration_kernel::uses_Laurent_series ( ) const [protected], [virtual]
```

Returns true, if the coefficients are computed from the Laurent series (in which case the method `Laurent_series` needs to be implemented).

Returns false if this is not the case (and the class has an implementation of `series_coeff_impl`).

Reimplemented in [GiNaC::Eisenstein_kernel](#), [GiNaC::Eisenstein_h_kernel](#), [GiNaC::modular_form_kernel](#), and [GiNaC::user_defined_kernel](#).

6.78.2.7 series_coeff_impl()

```
cln::cl_N GiNaC::integration_kernel::series_coeff_impl (
    int i ) const [protected], [virtual]
```

For $\omega = d\lambda$ only the coefficient of λ^0 is non-zero.

The i-th coefficient corresponds to the power λ^{i-1} .

Reimplemented in [GiNaC::basic_log_kernel](#), [GiNaC::multiple_polylog_kernel](#), [GiNaC::ELi_kernel](#), [GiNaC::Ebar_kernel](#), [GiNaC::Kronecker_dtau_kernel](#), and [GiNaC::Kronecker_dz_kernel](#).

6.78.2.8 get_cache_size()

```
size_t GiNaC::integration_kernel::get_cache_size (
    void ) const
```

Returns the current size of the cache.

6.78.2.9 set_cache_step()

```
void GiNaC::integration_kernel::set_cache_step (
    int cache_steps ) const
```

Sets the step size by which the cache is increased.

6.78.2.10 get_series_coeff()

```
ex GiNaC::integration_kernel::get_series_coeff (
    int i ) const
```

Wrapper around `series_coeff(i)`, converts `cl_N` to numeric.

6.78.2.11 series_coeff()

```
cln::cl_N GiNaC::integration_kernel::series_coeff (
    int i ) const
```

Subclasses have either to implement `series_coeff_impl` or the two methods `Laurent_series` and `uses_Laurent_series`.

The method `series_coeff_impl` can be used, if a single coefficient can be computed independently of the others.

The method `Laurent_series` can be used, if it is more efficient to compute a Laurent series in one shot and to determine a range of coefficients from this Laurent series.

References [GiNaC::ex::coeff\(\)](#), [GiNaC::ex::evalf\(\)](#), and `x`.

6.78.2.12 `get_numerical_value_impl()`

```
ex GiNaC::integration_kernel::get_numerical_value_impl (
    const ex & lambda,
    const ex & pre,
    int shift,
    int N_trunc ) const [protected]
```

The actual implementation for computing a numerical value for the integrand.

References [GiNaC::Digits](#), [GiNaC::ex::evalf\(\)](#), and [one](#).

Referenced by [GiNaC::Eli_kernel::get_numerical_value\(\)](#), [GiNaC::Ebar_kernel::get_numerical_value\(\)](#), [GiNaC::Kronecker_dtau_kernel::get_numerical_value\(\)](#), [GiNaC::Eisenstein_kernel::get_numerical_value\(\)](#), [GiNaC::Eisenstein_h_kernel::get_numerical_value\(\)](#), [GiNaC::modular_form_kernel::get_numerical_value\(\)](#), and [GiNaC::Kronecker_dz_kernel::get_numerical_value\(\)](#).

6.78.2.13 `do_print()`

```
void GiNaC::integration_kernel::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

References [c](#).

6.78.3 Member Data Documentation

6.78.3.1 `cache_step_size`

```
int GiNaC::integration_kernel::cache_step_size [mutable], [protected]
```

6.78.3.2 `series_vec`

```
std::vector<cln::cl_N> GiNaC::integration_kernel::series_vec [mutable], [protected]
```

The documentation for this class was generated from the following files:

- [integration_kernel.h](#)
- [integration_kernel.cpp](#)

6.79 `GiNaC::is_not_a_clifford` Struct Reference

Predicate for finding non-clifford objects.

Public Member Functions

- `bool operator() (const ex &e)`

6.79.1 Detailed Description

Predicate for finding non-clifford objects.

6.79.2 Member Function Documentation

6.79.2.1 operator()

```
bool GiNaC::is_not_a_clifford::operator() (
    const ex & e ) [inline]
```

The documentation for this struct was generated from the following file:

- [clifford.cpp](#)

6.80 GiNaC::is_summation_idx Struct Reference

Public Member Functions

- bool [operator\(\)](#) (const [ex](#) &e)

6.80.1 Member Function Documentation

6.80.1.1 operator()

```
bool GiNaC::is_summation_idx::operator() (
    const ex & e ) [inline]
```

References [GiNaC::is_dummy_pair\(\)](#).

The documentation for this struct was generated from the following file:

- [indexed.cpp](#)

6.81 GiNaC::iterated_integral2_SERIAL Class Reference

Complete elliptic integral of the first kind.

```
#include <inifcns.h>
```

Static Public Attributes

- static unsigned [serial](#)

6.81.1 Detailed Description

Complete elliptic integral of the first kind.

Complete elliptic integral of the second kind. Iterated integral.

6.81.2 Member Data Documentation

6.81.2.1 serial

```
unsigned GiNaC::iterated_integral2_SERIAL::serial [static]
```

Initial value:

```
=
    function::register_new(function_options("iterated_integral", 2).
        eval_func(iterated_integral2_eval).
        evalf_func(iterated_integral2_evalf).
        do_not_evalf_params().
        overloaded(2))
```

Referenced by [GiNaC::iterated_integral\(\)](#).

The documentation for this class was generated from the following files:

- [inifcns.h](#)
- [inifcns_elliptic.cpp](#)

6.82 GiNaC::iterated_integral3_SERIAL Class Reference

Iterated integral with explicit truncation.

```
#include <inifcns.h>
```

Static Public Attributes

- static unsigned [serial](#)

6.82.1 Detailed Description

Iterated integral with explicit truncation.

6.82.2 Member Data Documentation

6.82.2.1 serial

```
unsigned GiNaC::iterated_integral3_SERIAL::serial [static]
```

Initial value:

```
=
    function::register_new(function_options("iterated_integral", 3).
        eval_func(iterated_integral3_eval).
        evalf_func(iterated_integral3_evalf).
        do_not_evalf_params().
        overloaded(2))
```

Referenced by [GiNaC::iterated_integral\(\)](#).

The documentation for this class was generated from the following files:

- [inifcns.h](#)
- [inifcns_elliptic.cpp](#)

6.83 GiNaC::Gt::kernel Struct Reference

```
#include <Gt.h>
```

Public Member Functions

- [kernel](#) ()
- [kernel](#) (size_t [ni](#), [ex zi](#), int [deform](#)=0)
- [kernel](#) ([lst](#) arg)

Public Attributes

- size_t [ni](#)
- [ex zi](#)
- int [deform](#)

Private Member Functions

- void [check](#) ()

Friends

- class [Gt](#)

6.83.1 Constructor & Destructor Documentation

6.83.1.1 kernel() [1/3]

```
GiNaC::Gt::kernel::kernel ( )
```

6.83.1.2 kernel() [2/3]

```
GiNaC::Gt::kernel::kernel (
    size_t ni,
    ex zi,
    int deform = 0 )
```

References [check\(\)](#).

6.83.1.3 kernel() [3/3]

```
GiNaC::Gt::kernel::kernel (
    lst arg )
```

References [GiNaC::container< C >::nops\(\)](#), [GiNaC::container< C >::op\(\)](#), and [GiNaC::Gt::to_integer\(\)](#).

6.83.2 Member Function Documentation**6.83.2.1 check()**

```
void GiNaC::Gt::kernel::check ( ) [private]
```

Referenced by [kernel\(\)](#).

6.83.3 Friends And Related Symbol Documentation**6.83.3.1 Gt**

```
friend class Gt [friend]
```

6.83.4 Member Data Documentation**6.83.4.1 ni**

```
size_t GiNaC::Gt::kernel::ni
```

Referenced by [GiNaC::Gt::tauToFundamental\(\)](#).

6.83.4.2 zi

```
ex GiNaC::Gt::kernel::zi
```

6.83.4.3 deform

```
int GiNaC::Gt::kernel::deform
```

The documentation for this struct was generated from the following files:

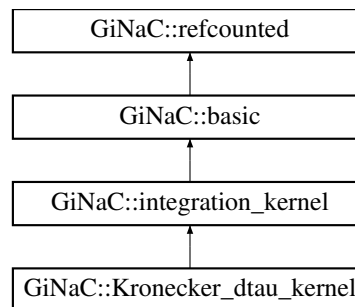
- [Gt.h](#)
- [Gt.cpp](#)

6.84 GiNaC::Kronecker_dtau_kernel Class Reference

The kernel corresponding to integrating the Kronecker coefficient function $g^{(n)}(z_j, K\tau)$ in τ (or equivalently in $\bar{q}$).

```
#include <integration_kernel.h>
```

Inheritance diagram for GiNaC::Kronecker_dtau_kernel:



Public Member Functions

- [Kronecker_dtau_kernel](#) (const [ex](#) &n, const [ex](#) &z, const [ex](#) &K=numeric(1), const [ex](#) &C_norm=numeric(1))
- [size_t nops](#) () const override
Number of operands/members.
- [ex op](#) (size_t i) const override
Return operand/member at position i.
- [ex & let_op](#) (size_t i) override
Return modifiable operand/member at position i.
- [bool is_numeric](#) (void) const override
This routine returns true, if the integration kernel can be evaluated numerically.
- [ex get_numerical_value](#) (const [ex](#) &qbar, int N_trunc=0) const override
Returns the value of the $g^{(n)}(z, K\tau)$, where tau is given by qbar.

Public Member Functions inherited from `GiNaC::integration_kernel`

- `ex series` (const relational &r, int order, unsigned options=0) const override
Default implementation of `ex::series()`.
- virtual bool `has_trailing_zero` (void) const
This routine returns true, if the integration kernel has a trailing zero.
- virtual `ex Laurent_series` (const ex &x, int order) const
Returns the Laurent series, starting possibly with the pole term.
- size_t `get_cache_size` (void) const
Returns the current size of the cache.
- void `set_cache_step` (int cache_steps) const
Sets the step size by which the cache is increased.
- `ex get_series_coeff` (int i) const
Wrapper around `series_coeff(i)`, converts `cl_N` to numeric.
- cln::cl_N `series_coeff` (int i) const
Subclasses have either to implement `series_coeff_impl` or the two methods `Laurent_series` and `uses_Laurent_series`.

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic` ()
basic destructor, virtual because class `ex` will delete objects of derived classes via a `basic*`.
- `basic` (const basic &other)
- const `basic & operator=` (const basic &other)
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate` () const
Create a clone of this object on the heap.
- virtual `ex eval` () const
Perform automatic non-interruptive term rewriting rules.
- virtual `ex evalf` () const
Evaluate object numerically.
- virtual `ex evalm` () const
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ` () const
Evaluate integrals, if result is known.
- virtual `ex eval_indexed` (const basic &i) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void `print` (const print_context &c, unsigned level=0) const
Output to stream.
- virtual void `dbgprint` () const
Little wrapper around `print` to be called within a debugger.
- virtual void `dbgprinttree` () const
Little wrapper around `printtree` to be called within a debugger.
- virtual unsigned `precedence` () const
Return relative operator precedence (for parenthezing output).
- virtual bool `info` (unsigned inf) const
Information about the object.
- virtual `ex operator[]` (const ex &index) const
- virtual `ex operator[]` (size_t i) const
- virtual `ex & operator[]` (const ex &index)
- virtual `ex & operator[]` (size_t i)
- virtual bool `has` (const ex &other, unsigned options=0) const

- Test for occurrence of a pattern.*

 - virtual bool `match` (const `ex` &pattern, `exmap` &repls) const
- Check whether the expression matches a given pattern.*

 - virtual `ex subs` (const `exmap` &m, unsigned `options`=0) const
- Substitute a set of objects by arbitrary expressions.*

 - virtual `ex map` (`map_function` &f) const
- Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).*

 - virtual void `accept` (`GiNaC::visitor` &v) const
 - virtual bool `is_polynomial` (const `ex` &var) const
- Check whether this is a polynomial in the given variables.*

 - virtual int `degree` (const `ex` &s) const
- Return degree of highest power in object s.*

 - virtual int `ldegree` (const `ex` &s) const
- Return degree of lowest power in object s.*

 - virtual `ex coeff` (const `ex` &s, int `n`=1) const
- Return coefficient of degree n in object s.*

 - virtual `ex expand` (unsigned `options`=0) const
- Expand expression, i.e.*

 - virtual `ex collect` (const `ex` &s, bool `distributed`=false) const
- Sort expanded expression in terms of powers of some object(s).*

 - virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const
- Default implementation of `ex::normal()`.*

 - virtual `ex to_rational` (`exmap` &repl) const
- Default implementation of `ex::to_rational()`.*

 - virtual `ex to_polynomial` (`exmap` &repl) const
 - virtual `numeric integer_content` () const
 - virtual `ex smod` (const `numeric` &xi) const
- Apply symmetric modular homomorphism to an expanded multivariate polynomial.*

 - virtual `numeric max_coefficient` () const
- Implementation `ex::max_coefficient()`.*

 - virtual `exvector get_free_indices` () const
- Return a vector containing the free indices of an expression.*

 - virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
- Add two indexed expressions.*

 - virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
- Multiply an indexed expression with a scalar.*

 - virtual bool `contract_with` (`exvector::iterator` self, `exvector::iterator` other, `exvector` &v) const
- Try to contract two indexed expressions that appear in the same product.*

 - virtual unsigned `return_type` () const
 - virtual `return_type_t return_type_tinfo` () const
 - virtual `ex conjugate` () const
 - virtual `ex real_part` () const
 - virtual `ex imag_part` () const
- template<class T >

 - void `print_dispatch` (const `print_context` &c, unsigned level) const
- Like `print()`, but dispatch to the specified class.*

 - void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const
- Like `print()`, but dispatch to the specified class.*

 - virtual void `archive` (`archive_node` &n) const
- Save (serialize) the object into archive node.*

 - virtual void `read_archive` (const `archive_node` &n, `lst` &syms)

- Load (deserialize) the object from an archive node.*

 - `ex subs_one_level` (const `exmap` &m, unsigned `options`) const

Helper function for `subs()`.
- `ex diff` (const `symbol` &s, unsigned nth=1) const

Default interface of nth derivative `ex::diff(s, n)`.
- `int compare` (const `basic` &other) const

Compare objects syntactically to establish canonical ordering.
- `bool is_equal` (const `basic` &other) const

Test for syntactic equality.
- `const basic & hold` () const

Stop further evaluation.
- `unsigned gethash` () const
- `const basic & setflag` (unsigned f) const

Set some `status_flags`.
- `const basic & clearflag` (unsigned f) const

Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- `unsigned int add_reference` () noexcept
- `unsigned int remove_reference` () noexcept
- `unsigned int get_refcount` () const noexcept
- `void set_refcount` (unsigned int r) noexcept

Protected Member Functions

- `cl::cl_N series_coeff_impl` (int i) const override
- For $\omega = d\lambda$ only the coefficient of λ^0 is non-zero.*
- `void do_print` (const `print_context` &c, unsigned level) const

Protected Member Functions inherited from `GiNaC::integration_kernel`

- `virtual bool uses_Laurent_series` () const
- Returns true, if the coefficients are computed from the Laurent series (in which case the method `Laurent_series` needs to be implemented).*
- `ex get_numerical_value_impl` (const `ex` &lambda, const `ex` &pre, int shift, int N_trunc) const
- The actual implementation for computing a numerical value for the integrand.*
- `void do_print` (const `print_context` &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual [ex derivative](#) (const [symbol](#) &s) const
Default implementation of [ex::diff\(\)](#).
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Protected Attributes

- [ex n](#)
- [ex z](#)
- [ex K](#)
- [ex C_norm](#)

Protected Attributes inherited from [GiNaC::integration_kernel](#)

- int [cache_step_size](#)
- std::vector< [cln::cl_N](#) > [series_vec](#)

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.84.1 Detailed Description

The kernel corresponding to integrating the Kronecker coefficient function $g^{(n)}(z_j, K\tau)$ in τ (or equivalently in $\bar{q}$).

This class represents the differential one-form

$$\omega_{n,K}^{\text{Kronecker},\tau}(z_j) = \frac{C_n K(n-1)}{(2\pi i)^n} g^{(n)}(z_j, K\tau) \frac{d\bar{q}}{\bar{q}}$$

6.84.2 Constructor & Destructor Documentation

6.84.2.1 Kronecker_dtau_kernel()

```
GiNaC::Kronecker_dtau_kernel::Kronecker_dtau_kernel (
    const ex & n,
    const ex & z,
    const ex & K = numeric(1),
    const ex & C_norm = numeric(1) )
```

6.84.3 Member Function Documentation

6.84.3.1 nops()

```
size_t GiNaC::Kronecker_dtau_kernel::nops ( ) const [override], [virtual]
```

Number of operands/members.

Reimplemented from [GiNaC::basic](#).

6.84.3.2 op()

```
ex GiNaC::Kronecker_dtau_kernel::op (
    size_t i ) const [override], [virtual]
```

Return operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [C_norm](#), [K](#), [n](#), and [z](#).

6.84.3.3 let_op()

```
ex & GiNaC::Kronecker_dtau_kernel::let_op (
    size_t i ) [override], [virtual]
```

Return modifiable operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [C_norm](#), [GiNaC::basic::ensure_if_modifiable\(\)](#), [K](#), [n](#), and [z](#).

6.84.3.4 is_numeric()

```
bool GiNaC::Kronecker_dtau_kernel::is_numeric (
    void ) const [override], [virtual]
```

This routine returns true, if the integration kernel can be evaluated numerically.

Reimplemented from [GiNaC::integration_kernel](#).

References [C_norm](#), [GiNaC::ex::evalf\(\)](#), [GiNaC::ex::info\(\)](#), [K](#), [n](#), [GiNaC::info_flags::nonnegint](#), [GiNaC::info_flags::numeric](#), [GiNaC::info_flags::posint](#), and [z](#).

6.84.3.5 get_numerical_value()

```
ex GiNaC::Kronecker_dtau_kernel::get_numerical_value (
    const ex & qbar,
    int N_trunc = 0 ) const [override], [virtual]
```

Returns the value of the $g^{(n)}(z, K \cdot \tau)$, where τ is given by $qbar$.

Reimplemented from [GiNaC::integration_kernel](#).

References [C_norm](#), [GiNaC::basic::evalf\(\)](#), [GiNaC::ex::evalf\(\)](#), [GiNaC::exp\(\)](#), [GiNaC::Ebar_kernel::get_numerical_value\(\)](#), [GiNaC::integration_kernel::get_numerical_value_impl\(\)](#), [GiNaC::l](#), K , n , [GiNaC::Pi](#), [GiNaC::pow\(\)](#), $qbar$, and z .

6.84.3.6 series_coeff_impl()

```
cln::cl_N GiNaC::Kronecker_dtau_kernel::series_coeff_impl (
    int i ) const [override], [protected], [virtual]
```

For $\omega = d\lambda$ only the coefficient of λ^0 is non-zero.

The i -th coefficient corresponds to the power λ^{i-1} .

Reimplemented from [GiNaC::integration_kernel](#).

References [GiNaC::bernoulli\(\)](#), [C_norm](#), [GiNaC::basic::evalf\(\)](#), [GiNaC::ex::evalf\(\)](#), [GiNaC::exp\(\)](#), [GiNaC::factorial\(\)](#), [GiNaC::l](#), K , n , [GiNaC::Pi](#), [GiNaC::numeric::to_cl_N\(\)](#), [GiNaC::numeric::to_int\(\)](#), and z .

6.84.3.7 do_print()

```
void GiNaC::Kronecker_dtau_kernel::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

References [c](#), [C_norm](#), K , n , [GiNaC::ex::print\(\)](#), and z .

6.84.4 Member Data Documentation

6.84.4.1 n

```
ex GiNaC::Kronecker_dtau_kernel::n [protected]
```

Referenced by [do_print\(\)](#), [get_numerical_value\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [series_coeff_impl\(\)](#).

6.84.4.2 z

```
ex GiNaC::Kronecker_dtau_kernel::z [protected]
```

Referenced by [do_print\(\)](#), [get_numerical_value\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [series_coeff_impl\(\)](#).

6.84.4.3 K

`ex GiNaC::Kronecker_dtau_kernel::K` [protected]

Referenced by [do_print\(\)](#), [get_numerical_value\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [series_coeff_impl\(\)](#).

6.84.4.4 C_norm

`ex GiNaC::Kronecker_dtau_kernel::C_norm` [protected]

Referenced by [do_print\(\)](#), [get_numerical_value\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [series_coeff_impl\(\)](#).

The documentation for this class was generated from the following files:

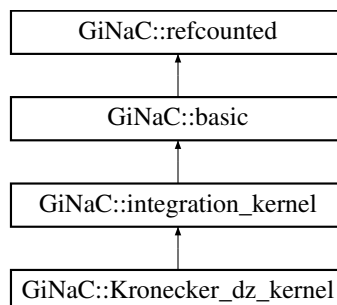
- [integration_kernel.h](#)
- [integration_kernel.cpp](#)

6.85 GiNaC::Kronecker_dz_kernel Class Reference

The kernel corresponding to integrating the Kronecker coefficient function $g^{(n-1)}(z - z_j, K\tau)$ in z .

```
#include <integration_kernel.h>
```

Inheritance diagram for GiNaC::Kronecker_dz_kernel:



Public Member Functions

- [Kronecker_dz_kernel](#) (const [ex](#) &n, const [ex](#) &z_j, const [ex](#) &tau, const [ex](#) &K=numeric(1), const [ex](#) &C_norm=numeric(1))
- [size_t nops](#) () const override
Number of operands/members.
- [ex op](#) (size_t i) const override
Return operand/member at position i.
- [ex](#) & [let_op](#) (size_t i) override
Return modifiable operand/member at position i.
- bool [is_numeric](#) (void) const override
This routine returns true, if the integration kernel can be evaluated numerically.
- [ex get_numerical_value](#) (const [ex](#) &z, int N_trunc=0) const override
Returns the value of the $g^{(n-1)}(z-z_j, K*\tau)$.

Public Member Functions inherited from GiNaC::integration_kernel

- **ex series** (const relational &r, int order, unsigned options=0) const override
Default implementation of ex::series().
- virtual bool **has_trailing_zero** (void) const
This routine returns true, if the integration kernel has a trailing zero.
- virtual **ex Laurent_series** (const ex &x, int order) const
Returns the Laurent series, starting possibly with the pole term.
- size_t **get_cache_size** (void) const
Returns the current size of the cache.
- void **set_cache_step** (int cache_steps) const
Sets the step size by which the cache is increased.
- **ex get_series_coeff** (int i) const
Wrapper around series_coeff(i), converts cl_N to numeric.
- cln::cl_N **series_coeff** (int i) const
Subclasses have either to implement series_coeff_impl or the two methods Laurent_series and uses_Laurent_series.

Public Member Functions inherited from GiNaC::basic

- virtual **~basic** ()
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- **basic** (const basic &other)
- const **basic & operator=** (const basic &other)
basic assignment operator: the other object might be of a derived class.
- virtual **basic * duplicate** () const
Create a clone of this object on the heap.
- virtual **ex eval** () const
Perform automatic non-interruptive term rewriting rules.
- virtual **ex evalf** () const
Evaluate object numerically.
- virtual **ex evalm** () const
Evaluate sums, products and integer powers of matrices.
- virtual **ex eval_integ** () const
Evaluate integrals, if result is known.
- virtual **ex eval_indexed** (const basic &i) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void **print** (const print_context &c, unsigned level=0) const
Output to stream.
- virtual void **dbgprint** () const
Little wrapper around print to be called within a debugger.
- virtual void **dbgprinttree** () const
Little wrapper around printtree to be called within a debugger.
- virtual unsigned **precedence** () const
Return relative operator precedence (for parenthezing output).
- virtual bool **info** (unsigned inf) const
Information about the object.
- virtual **ex operator[]** (const ex &index) const
- virtual **ex operator[]** (size_t i) const
- virtual **ex & operator[]** (const ex &index)
- virtual **ex & operator[]** (size_t i)
- virtual bool **has** (const ex &other, unsigned options=0) const

- Test for occurrence of a pattern.*

 - virtual bool `match` (const `ex` &pattern, `exmap` &repls) const
- Check whether the expression matches a given pattern.*

 - virtual `ex subs` (const `exmap` &m, unsigned `options`=0) const
- Substitute a set of objects by arbitrary expressions.*

 - virtual `ex map` (`map_function` &f) const
- Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).*

 - virtual void `accept` (`GiNaC::visitor` &v) const
 - virtual bool `is_polynomial` (const `ex` &var) const
- Check whether this is a polynomial in the given variables.*

 - virtual int `degree` (const `ex` &s) const
- Return degree of highest power in object s.*

 - virtual int `ldegree` (const `ex` &s) const
- Return degree of lowest power in object s.*

 - virtual `ex coeff` (const `ex` &s, int `n`=1) const
- Return coefficient of degree n in object s.*

 - virtual `ex expand` (unsigned `options`=0) const
- Expand expression, i.e.*

 - virtual `ex collect` (const `ex` &s, bool `distributed`=false) const
- Sort expanded expression in terms of powers of some object(s).*

 - virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const
- Default implementation of `ex::normal()`.*

 - virtual `ex to_rational` (`exmap` &repl) const
- Default implementation of `ex::to_rational()`.*

 - virtual `ex to_polynomial` (`exmap` &repl) const
 - virtual `numeric integer_content` () const
 - virtual `ex smod` (const `numeric` &xi) const
- Apply symmetric modular homomorphism to an expanded multivariate polynomial.*

 - virtual `numeric max_coefficient` () const
- Implementation `ex::max_coefficient()`.*

 - virtual `exvector get_free_indices` () const
- Return a vector containing the free indices of an expression.*

 - virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
- Add two indexed expressions.*

 - virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
- Multiply an indexed expression with a scalar.*

 - virtual bool `contract_with` (`exvector::iterator` self, `exvector::iterator` other, `exvector` &v) const
- Try to contract two indexed expressions that appear in the same product.*

 - virtual unsigned `return_type` () const
 - virtual `return_type_t return_type_tinfo` () const
 - virtual `ex conjugate` () const
 - virtual `ex real_part` () const
 - virtual `ex imag_part` () const
- template<class T >

 - void `print_dispatch` (const `print_context` &c, unsigned level) const
- Like `print()`, but dispatch to the specified class.*

 - void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const
- Like `print()`, but dispatch to the specified class.*

 - virtual void `archive` (`archive_node` &n) const
- Save (serialize) the object into archive node.*

 - virtual void `read_archive` (const `archive_node` &n, `lst` &syms)

- *Load (deserialize) the object from an archive node.*
- `ex subs_one_level` (const `exmap` &m, unsigned `options`) const
Helper function for `subs()`.
- `ex diff` (const `symbol` &s, unsigned `nth`=1) const
Default interface of `nth` derivative `ex::diff(s, n)`.
- `int compare` (const `basic` &other) const
Compare objects syntactically to establish canonical ordering.
- `bool is_equal` (const `basic` &other) const
Test for syntactic equality.
- `const basic & hold` () const
Stop further evaluation.
- `unsigned gethash` () const
- `const basic & setflag` (unsigned `f`) const
Set some `status_flags`.
- `const basic & clearflag` (unsigned `f`) const
Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- `unsigned int add_reference` () noexcept
- `unsigned int remove_reference` () noexcept
- `unsigned int get_refcount` () const noexcept
- `void set_refcount` (unsigned int `r`) noexcept

Protected Member Functions

- `cln::cl_N series_coeff_impl` (int `i`) const override
For $\omega = d\lambda$ only the coefficient of λ^0 is non-zero.
- `void do_print` (const `print_context` &c, unsigned `level`) const

Protected Member Functions inherited from `GiNaC::integration_kernel`

- `virtual bool uses_Laurent_series` () const
Returns true, if the coefficients are computed from the Laurent series (in which case the method `Laurent_series` needs to be implemented).
- `ex get_numerical_value_impl` (const `ex` &lambda, const `ex` &pre, int `shift`, int `N_trunc`) const
The actual implementation for computing a numerical value for the integrand.
- `void do_print` (const `print_context` &c, unsigned `level`) const

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual [ex derivative](#) (const [symbol](#) &s) const
Default implementation of [ex::diff\(\)](#).
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Protected Attributes

- [ex n](#)
- [ex z_j](#)
- [ex tau](#)
- [ex K](#)
- [ex C_norm](#)

Protected Attributes inherited from [GiNaC::integration_kernel](#)

- int [cache_step_size](#)
- `std::vector< cln::cl\_N >` [series_vec](#)

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.85.1 Detailed Description

The kernel corresponding to integrating the Kronecker coefficient function $g^{(n-1)}(z - z_j, K\tau)$ in z .

This class represents the differential one-form

$$\omega_{n,K}^{\text{Kronecker},z}(z_j, \tau) = C_n (2\pi i)^{2-n} g^{(n-1)}(z - z_j, K\tau) dz$$

6.85.2 Constructor & Destructor Documentation

6.85.2.1 Kronecker_dz_kernel()

```
GiNaC::Kronecker_dz_kernel::Kronecker_dz_kernel (
    const ex & n,
    const ex & z_j,
    const ex & tau,
    const ex & K = numeric(1),
    const ex & C_norm = numeric(1) )
```

6.85.3 Member Function Documentation

6.85.3.1 nops()

```
size_t GiNaC::Kronecker_dz_kernel::nops ( ) const [override], [virtual]
```

Number of operands/members.

Reimplemented from [GiNaC::basic](#).

6.85.3.2 op()

```
ex GiNaC::Kronecker_dz_kernel::op (
    size_t i ) const [override], [virtual]
```

Return operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [C_norm](#), [K](#), [n](#), [tau](#), and [z_j](#).

6.85.3.3 let_op()

```
ex & GiNaC::Kronecker_dz_kernel::let_op (
    size_t i ) [override], [virtual]
```

Return modifiable operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [C_norm](#), [GiNaC::basic::ensure_if_modifiable\(\)](#), [K](#), [n](#), [tau](#), and [z_j](#).

6.85.3.4 is_numeric()

```
bool GiNaC::Kronecker_dz_kernel::is_numeric (
    void ) const [override], [virtual]
```

This routine returns true, if the integration kernel can be evaluated numerically.

Reimplemented from [GiNaC::integration_kernel](#).

References [C_norm](#), [GiNaC::ex::evalf\(\)](#), [GiNaC::ex::info\(\)](#), [K](#), [n](#), [GiNaC::info_flags::nonnegint](#), [GiNaC::info_flags::numeric](#), [GiNaC::info_flags::posint](#), [tau](#), and [z_j](#).

6.85.3.5 `get_numerical_value()`

```
ex GiNaC::Kronecker_dz_kernel::get_numerical_value (
    const ex & z,
    int N_trunc = 0 ) const [override], [virtual]
```

Returns the value of the $g^{(n-1)}(z-z_j, K\tau)$.

Reimplemented from [GiNaC::integration_kernel](#).

References [C_norm](#), [GiNaC::integration_kernel::get_numerical_value_impl\(\)](#), [GiNaC::l](#), [n](#), [GiNaC::Pi](#), and [GiNaC::pow\(\)](#).

6.85.3.6 `series_coeff_impl()`

```
cln::cl_N GiNaC::Kronecker_dz_kernel::series_coeff_impl (
    int i ) const [override], [protected], [virtual]
```

For $\omega = d\lambda$ only the coefficient of λ^0 is non-zero.

The i -th coefficient corresponds to the power λ^{i-1} .

Reimplemented from [GiNaC::integration_kernel](#).

References [GiNaC::bernoulli\(\)](#), [C_norm](#), [GiNaC::basic::evalf\(\)](#), [GiNaC::ex::evalf\(\)](#), [GiNaC::exp\(\)](#), [GiNaC::factorial\(\)](#), [GiNaC::Ebar_kernel::get_numerical_value\(\)](#), [GiNaC::l](#), [GiNaC::is_even\(\)](#), [K](#), [n](#), [GiNaC::Pi](#), [GiNaC::pow\(\)](#), [qbar](#), [tau](#), and [z_j](#).

6.85.3.7 `do_print()`

```
void GiNaC::Kronecker_dz_kernel::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

References [c](#), [C_norm](#), [K](#), [n](#), [GiNaC::ex::print\(\)](#), [tau](#), and [z_j](#).

6.85.4 Member Data Documentation

6.85.4.1 `n`

```
ex GiNaC::Kronecker_dz_kernel::n [protected]
```

Referenced by [do_print\(\)](#), [get_numerical_value\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [series_coeff_impl\(\)](#).

6.85.4.2 `z_j`

```
ex GiNaC::Kronecker_dz_kernel::z_j [protected]
```

Referenced by [do_print\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [series_coeff_impl\(\)](#).

6.85.4.3 tau

`ex GiNaC::Kronecker_dz_kernel::tau` [protected]

Referenced by [do_print\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [series_coeff_impl\(\)](#).

6.85.4.4 K

`ex GiNaC::Kronecker_dz_kernel::K` [protected]

Referenced by [do_print\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [series_coeff_impl\(\)](#).

6.85.4.5 C_norm

`ex GiNaC::Kronecker_dz_kernel::C_norm` [protected]

Referenced by [do_print\(\)](#), [get_numerical_value\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [series_coeff_impl\(\)](#).

The documentation for this class was generated from the following files:

- [integration_kernel.h](#)
- [integration_kernel.cpp](#)

6.86 GiNaC::lanczos_coeffs Class Reference

Public Member Functions

- [lanczos_coeffs](#) ()
- bool [sufficiently_accurate](#) (int digits)
- int [get_order](#) () const
- cln::cl_N [calc_lanczos_A](#) (const cln::cl_N &) const

Private Attributes

- std::vector< cln::cl_N > * [current_vector](#)

Static Private Attributes

- static std::vector< cln::cl_N > * [coeffs](#) = nullptr

6.86.1 Constructor & Destructor Documentation

6.86.1.1 lanczos_coeffs()

`GiNaC::lanczos_coeffs::lanczos_coeffs ( )`

References [coeffs](#).

6.86.2 Member Function Documentation

6.86.2.1 sufficiently_accurate()

```
bool GiNaC::lanczos_coeffs::sufficiently_accurate (
    int digits )
```

References [coeffs](#), and [current_vector](#).

Referenced by [GiNaC::lgamma\(\)](#), and [GiNaC::tgamma\(\)](#).

6.86.2.2 get_order()

```
int GiNaC::lanczos_coeffs::get_order ( ) const [inline]
```

References [current_vector](#).

Referenced by [GiNaC::lgamma\(\)](#), and [GiNaC::tgamma\(\)](#).

6.86.2.3 calc_lanczos_A()

```
cln::cl_N GiNaC::lanczos_coeffs::calc_lanczos_A (
    const cln::cl_N & x ) const
```

References [current_vector](#), and [x](#).

Referenced by [GiNaC::lgamma\(\)](#), and [GiNaC::tgamma\(\)](#).

6.86.3 Member Data Documentation

6.86.3.1 coeffs

```
std::vector< cln::cl_N > * GiNaC::lanczos_coeffs::coeffs = nullptr [static], [private]
```

Referenced by [lanczos_coeffs\(\)](#), and [sufficiently_accurate\(\)](#).

6.86.3.2 current_vector

```
std::vector<cln::cl_N>* GiNaC::lanczos_coeffs::current_vector [private]
```

Referenced by [calc_lanczos_A\(\)](#), [get_order\(\)](#), and [sufficiently_accurate\(\)](#).

The documentation for this class was generated from the following file:

- [numeric.cpp](#)

6.87 std::less< GiNaC::ptr< T > > Struct Template Reference

Specialization of std::less for ptr<T> to enable ordering of ptr<T> objects (e.g.

Public Member Functions

- bool [operator\(\)](#) (const [GiNaC::ptr< T >](#) &lhs, const [GiNaC::ptr< T >](#) &rhs) const

6.87.1 Detailed Description

```
template<class T>
struct std::less< GiNaC::ptr< T > >
```

Specialization of std::less for ptr<T> to enable ordering of ptr<T> objects (e.g.
for the use as std::map keys).

6.87.2 Member Function Documentation

6.87.2.1 operator()

```
template<class T >
bool std::less< GiNaC::ptr< T > >::operator() (
    const GiNaC::ptr< T > & lhs,
    const GiNaC::ptr< T > & rhs ) const [inline]
```

The documentation for this struct was generated from the following file:

- [ptr.h](#)

6.88 GiNaC::library_init Class Reference

Helper class to initialize the library.

```
#include <ex.h>
```

Public Member Functions

- [library_init](#) ()
Ctor of static initialization helpers.
- [~library_init](#) ()
Dtor of static initialization helpers.

Static Private Member Functions

- static void [init_unarchivers](#) ()

Static Private Attributes

- static int `count` = 0

How many static objects were created? Only the first one must create the static flyweights on the heap.

6.88.1 Detailed Description

Helper class to initialize the library.

There must be one static object of this class in every object file that makes use of our flyweights in order to guarantee proper initialization. Hence we put it into this file which is included by every relevant file anyways. This is modeled after section [ios::Init] of the C++ standard, where cout and friends are set up.

See also

[utils.cpp](#)

6.88.2 Constructor & Destructor Documentation

6.88.2.1 `library_init()`

```
GiNaC::library_init::library_init ( )
```

Ctor of static initialization helpers.

The first call to this is going to initialize the library, the others do nothing.

References [GiNaC::_ex0](#), [GiNaC::_ex1](#), [GiNaC::_ex10](#), [GiNaC::_ex11](#), [GiNaC::_ex12](#), [GiNaC::_ex120](#), [GiNaC::_ex15](#), [GiNaC::_ex18](#), [GiNaC::_ex1_2](#), [GiNaC::_ex1_3](#), [GiNaC::_ex1_4](#), [GiNaC::_ex2](#), [GiNaC::_ex20](#), [GiNaC::_ex24](#), [GiNaC::_ex25](#), [GiNaC::_ex3](#), [GiNaC::_ex30](#), [GiNaC::_ex4](#), [GiNaC::_ex48](#), [GiNaC::_ex5](#), [GiNaC::_ex6](#), [GiNaC::_ex60](#), [GiNaC::_ex7](#), [GiNaC::_ex8](#), [GiNaC::_ex9](#), [GiNaC::_ex_1](#), [GiNaC::_ex_10](#), [GiNaC::_ex_11](#), [GiNaC::_ex_12](#), [GiNaC::_ex_120](#), [GiNaC::_ex_15](#), [GiNaC::_ex_18](#), [GiNaC::_ex_1_2](#), [GiNaC::_ex_1_3](#), [GiNaC::_ex_1_4](#), [GiNaC::_ex_2](#), [GiNaC::_ex_20](#), [GiNaC::_ex_24](#), [GiNaC::_ex_25](#), [GiNaC::_ex_3](#), [GiNaC::_ex_30](#), [GiNaC::_ex_4](#), [GiNaC::_ex_48](#), [GiNaC::_ex_5](#), [GiNaC::_ex_6](#), [GiNaC::_ex_60](#), [GiNaC::_ex_7](#), [GiNaC::_ex_8](#), [GiNaC::_ex_9](#), [GiNaC::_num0_bp](#), [GiNaC::_num0_p](#), [GiNaC::_num10_p](#), [GiNaC::_num11_p](#), [GiNaC::_num120_p](#), [GiNaC::_num12_p](#), [GiNaC::_num15_p](#), [GiNaC::_num18_p](#), [GiNaC::_num1_2_p](#), [GiNaC::_num1_3_p](#), [GiNaC::_num1_4_p](#), [GiNaC::_num1_p](#), [GiNaC::_num20_p](#), [GiNaC::_num24_p](#), [GiNaC::_num25_p](#), [GiNaC::_num2_p](#), [GiNaC::_num30_p](#), [GiNaC::_num3_p](#), [GiNaC::_num48_p](#), [GiNaC::_num4_p](#), [GiNaC::_num5_p](#), [GiNaC::_num60_p](#), [GiNaC::_num6_p](#), [GiNaC::_num7_p](#), [GiNaC::_num8_p](#), [GiNaC::_num9_p](#), [GiNaC::_num_10_p](#), [GiNaC::_num_11_p](#), [GiNaC::_num_120_p](#), [GiNaC::_num_12_p](#), [GiNaC::_num_15_p](#), [GiNaC::_num_18_p](#), [GiNaC::_num_1_2_p](#), [GiNaC::_num_1_3_p](#), [GiNaC::_num_1_4_p](#), [GiNaC::_num_1_p](#), [GiNaC::_num_20_p](#), [GiNaC::_num_24_p](#), [GiNaC::_num_25_p](#), [GiNaC::_num_2_p](#), [GiNaC::_num_30_p](#), [GiNaC::_num_3_p](#), [GiNaC::_num_48_p](#), [GiNaC::_num_4_p](#), [GiNaC::_num_5_p](#), [GiNaC::_num_60_p](#), [GiNaC::_num_6_p](#), [GiNaC::_num_7_p](#), [GiNaC::_num_8_p](#), [GiNaC::_num_9_p](#), and `count`.

6.88.2.2 `~library_init()`

```
GiNaC::library_init::~~library_init ( )
```

Dtor of static initialization helpers.

The last call to this is going to shut down the library, the others do nothing.

References [GiNaC::_ex0](#), [GiNaC::_ex1](#), [GiNaC::_ex10](#), [GiNaC::_ex11](#), [GiNaC::_ex12](#), [GiNaC::_ex120](#), [GiNaC::_ex15](#), [GiNaC::_ex18](#), [GiNaC::_ex1_2](#), [GiNaC::_ex1_3](#), [GiNaC::_ex1_4](#), [GiNaC::_ex2](#), [GiNaC::_ex20](#), [GiNaC::_ex24](#), [GiNaC::_ex25](#), [GiNaC::_ex3](#), [GiNaC::_ex30](#), [GiNaC::_ex4](#), [GiNaC::_ex48](#), [GiNaC::_ex5](#), [GiNaC::_ex6](#), [GiNaC::_ex60](#), [GiNaC::_ex7](#), [GiNaC::_ex8](#), [GiNaC::_ex9](#), [GiNaC::_ex_1](#), [GiNaC::_ex_10](#), [GiNaC::_ex_11](#), [GiNaC::_ex_12](#), [GiNaC::_ex_120](#), [GiNaC::_ex_15](#), [GiNaC::_ex_18](#), [GiNaC::_ex_1_2](#), [GiNaC::_ex_1_3](#), [GiNaC::_ex_1_4](#), [GiNaC::_ex_2](#), [GiNaC::_ex_20](#), [GiNaC::_ex_24](#), [GiNaC::_ex_25](#), [GiNaC::_ex_3](#), [GiNaC::_ex_30](#), [GiNaC::_ex_4](#), [GiNaC::_ex_48](#), [GiNaC::_ex_5](#), [GiNaC::_ex_6](#), [GiNaC::_ex_60](#), [GiNaC::_ex_7](#), [GiNaC::_ex_8](#), [GiNaC::_ex_9](#), and `count`.

6.88.3 Member Function Documentation

6.88.3.1 init_unarchivers()

```
void GiNaC::library_init::init_unarchivers ( ) [static], [private]
```

6.88.4 Member Data Documentation

6.88.4.1 count

```
int GiNaC::library_init::count = 0 [static], [private]
```

How many static objects were created? Only the first one must create the static flyweights on the heap.

Referenced by [library_init\(\)](#), and [~library_init\(\)](#).

The documentation for this class was generated from the following files:

- [ex.h](#)
- [utils.cpp](#)

6.89 GiNaC::make_flat_inserter Class Reference

Class to handle the renaming of dummy indices.

```
#include <expairseq.h>
```

Public Member Functions

- [make_flat_inserter](#) (const [epvector](#) &epv, bool b)
- [make_flat_inserter](#) (const [exvector](#) &v, bool b)
- [ex_handle_factor](#) (const [ex](#) &x, const [ex](#) &coeff)

Private Member Functions

- void [combine_indices](#) (const [exvector](#) &dummies_of_factor)

Private Attributes

- bool [do_renaming](#)
- [exvector](#) [used_indices](#)

6.89.1 Detailed Description

Class to handle the renaming of dummy indices.

It holds a vector of indices that are being used in the expression so far. If the same index occurs again as a dummy index in a factor, it is to be renamed. Unless dummy index renaming was switched off, of course ;-)

6.89.2 Constructor & Destructor Documentation

6.89.2.1 `make_flat_inserter()` [1/2]

```
GiNaC::make_flat_inserter::make_flat_inserter (
    const epvector & epv,
    bool b ) [inline]
```

References [GiNaC::are_ex_trivially_equal\(\)](#), [combine_indices\(\)](#), and [do_renaming](#).

6.89.2.2 `make_flat_inserter()` [2/2]

```
GiNaC::make_flat_inserter::make_flat_inserter (
    const exvector & v,
    bool b ) [inline]
```

References [combine_indices\(\)](#), and [do_renaming](#).

6.89.3 Member Function Documentation

6.89.3.1 `handle_factor()`

```
ex GiNaC::make_flat_inserter::handle_factor (
    const ex & x,
    const ex & coeff ) [inline]
```

References [GiNaC::coeff\(\)](#), [combine_indices\(\)](#), [do_renaming](#), [GiNaC::get_all_dummy_indices_safely\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::rename_dummy_indices_uniquely\(\)](#), [used_indices](#), and [x](#).

Referenced by [GiNaC::ncmul::eval\(\)](#), [GiNaC::expairseq::make_flat\(\)](#), and [GiNaC::expairseq::make_flat\(\)](#).

6.89.3.2 `combine_indices()`

```
void GiNaC::make_flat_inserter::combine_indices (
    const exvector & dummies_of_factor ) [inline], [private]
```

References [used_indices](#).

Referenced by [handle_factor\(\)](#), [make_flat_inserter\(\)](#), and [make_flat_inserter\(\)](#).

6.89.4 Member Data Documentation

6.89.4.1 `do_renaming`

```
bool GiNaC::make_flat_inserter::do_renaming [private]
```

Referenced by [handle_factor\(\)](#), [make_flat_inserter\(\)](#), and [make_flat_inserter\(\)](#).

6.89.4.2 used_indices

`exvector` GiNaC::make_flat_inserter::used_indices [private]

Referenced by [combine_indices\(\)](#), and [handle_factor\(\)](#).

The documentation for this class was generated from the following file:

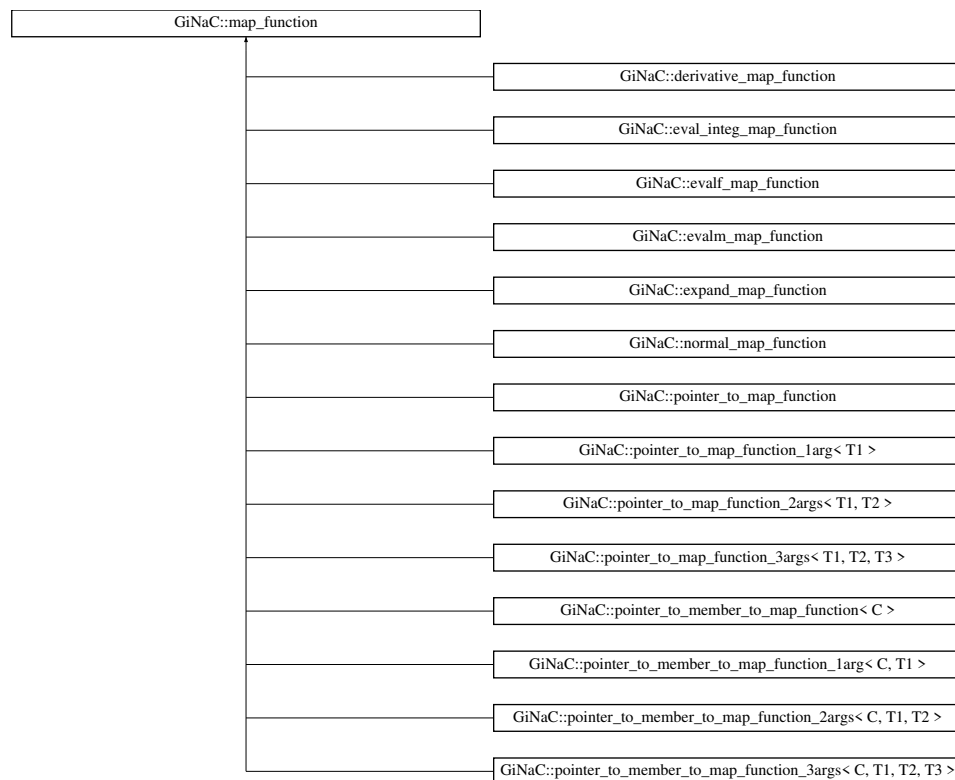
- [expairseq.h](#)

6.90 GiNaC::map_function Struct Reference

Function object for `map()`.

```
#include <basic.h>
```

Inheritance diagram for GiNaC::map_function:



Public Types

- typedef const [ex](#) & [argument_type](#)
- typedef [ex](#) [result_type](#)

Public Member Functions

- virtual [~map_function](#) ()
- virtual [ex operator\(\)](#) (const [ex](#) &e)=0

6.90.1 Detailed Description

Function object for map().

6.90.2 Member Typedef Documentation

6.90.2.1 argument_type

```
typedef const ex& GiNaC::map\_function::argument\_type
```

6.90.2.2 result_type

```
typedef ex GiNaC::map\_function::result\_type
```

6.90.3 Constructor & Destructor Documentation

6.90.3.1 ~map_function()

```
virtual GiNaC::map\_function::~~map\_function ( ) \[inline\], \[virtual\]
```

6.90.4 Member Function Documentation

6.90.4.1 operator>()

```
virtual ex GiNaC::map\_function::operator() (
    const ex & e ) \[pure virtual\]
```

Implemented in [GiNaC::evalf_map_function](#), [GiNaC::evalm_map_function](#), [GiNaC::eval_integ_map_function](#), [GiNaC::derivative_map_function](#), [GiNaC::expand_map_function](#), [GiNaC::pointer_to_map_function](#), [GiNaC::pointer_to_map_function_2args< T1, T2 >](#), [GiNaC::pointer_to_map_function_3args< T1, T2, T3 >](#), [GiNaC::pointer_to_member_to_map_function< C >](#), [GiNaC::pointer_to_member_to_map_function_1arg< C, T1 >](#), [GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 >](#), [GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2 >](#) and [GiNaC::normal_map_function](#).

The documentation for this struct was generated from the following file:

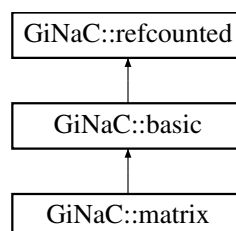
- [basic.h](#)

6.91 GiNaC::matrix Class Reference

Symbolic matrices.

```
#include <matrix.h>
```

Inheritance diagram for [GiNaC::matrix](#):



Public Member Functions

- [matrix](#) (unsigned [r](#), unsigned [c](#))
Very common ctor.
- [matrix](#) (unsigned [r](#), unsigned [c](#), const [lst](#) &l)
Construct matrix from (flat) list of elements.
- [matrix](#) (std::initializer_list< std::initializer_list< [ex](#) > > l)
Construct a matrix from an 2 dimensional initializer list.
- [size_t nops](#) () const override
nops is defined to be rows x columns.
- [ex op](#) (size_t i) const override
returns matrix entry at position (i/col, icol).
- [ex & let_op](#) (size_t i) override
returns writable matrix entry at position (i/col, icol).
- [ex evalm](#) () const override
Evaluate sums, products and integer powers of matrices.
- [ex subs](#) (const [exmap](#) &m, unsigned [options](#)=0) const override
Substitute a set of objects by arbitrary expressions.
- [ex eval_indexed](#) (const [basic](#) &i) const override
Automatic symbolic evaluation of an indexed matrix.
- [ex add_indexed](#) (const [ex](#) &self, const [ex](#) &other) const override
Sum of two indexed matrices.
- [ex scalar_mul_indexed](#) (const [ex](#) &self, const [numeric](#) &other) const override
Product of an indexed matrix with a number.
- bool [contract_with](#) (exvector::iterator self, exvector::iterator other, [exvector](#) &v) const override
Contraction of an indexed matrix with something else.
- [ex conjugate](#) () const override
Complex conjugate every matrix entry.
- [ex real_part](#) () const override
- [ex imag_part](#) () const override
- void [archive](#) ([archive_node](#) &n) const override
Save (a.k.a.
- void [read_archive](#) (const [archive_node](#) &n, [lst](#) &syms) override
Read (a.k.a.
- unsigned [rows](#) () const
Get number of rows.
- unsigned [cols](#) () const
Get number of columns.
- [matrix add](#) (const [matrix](#) &other) const
Sum of matrices.
- [matrix sub](#) (const [matrix](#) &other) const
Difference of matrices.
- [matrix mul](#) (const [matrix](#) &other) const
Product of matrices.
- [matrix mul](#) (const [numeric](#) &other) const
Product of matrix and scalar.
- [matrix mul_scalar](#) (const [ex](#) &other) const
Product of matrix and scalar expression.
- [matrix pow](#) (const [ex](#) &expn) const
Power of a matrix.
- const [ex & operator\(\)](#) (unsigned ro, unsigned co) const

- operator()* to access elements for reading.
- **ex & operator()** (unsigned ro, unsigned co)
 - operator()* to access elements for writing.
- **matrix & set** (unsigned ro, unsigned co, const **ex &value**)
- **matrix transpose** () const
 - Transposed of an $m \times n$ matrix, producing a new $n \times m$ matrix object that represents the transposed.*
- **ex determinant** (unsigned algo=**determinant_algo::automatic**) const
 - Determinant of square matrix.*
- **ex trace** () const
 - Trace of a matrix.*
- **ex charpoly** (const **ex &lambda**) const
 - Characteristic Polynomial.*
- **matrix inverse** () const
 - Inverse of this matrix, with automatic algorithm selection.*
- **matrix inverse** (unsigned algo) const
 - Inverse of this matrix.*
- **matrix solve** (const **matrix &vars**, const **matrix &rhs**, unsigned algo=**solve_algo::automatic**) const
 - Solve a linear system consisting of a $m \times n$ matrix and a $m \times p$ right hand side by applying an elimination scheme to the augmented matrix.*
- unsigned **rank** () const
 - Compute the rank of this matrix.*
- unsigned **rank** (unsigned **solve_algo**) const
 - Compute the rank of this matrix using the given algorithm, which should be a member of enum **solve_algo**.*
- bool **is_zero_matrix** () const
 - Function to check that all elements of the matrix are zero.*

Public Member Functions inherited from **GiNaC::basic**

- virtual **~basic** ()
 - basic destructor, virtual because class **ex** will delete objects of derived classes via a **basic***.*
- **basic** (const **basic &other**)
- const **basic & operator=** (const **basic &other**)
 - basic assignment operator: the other object might be of a derived class.*
- virtual **basic * duplicate** () const
 - Create a clone of this object on the heap.*
- virtual **ex eval** () const
 - Perform automatic non-interruptive term rewriting rules.*
- virtual **ex evalf** () const
 - Evaluate object numerically.*
- virtual **ex eval_integ** () const
 - Evaluate integrals, if result is known.*
- virtual void **print** (const **print_context &c**, unsigned level=0) const
 - Output to stream.*
- virtual void **dbgprint** () const
 - Little wrapper around **print** to be called within a debugger.*
- virtual void **dbgprinttree** () const
 - Little wrapper around **printtree** to be called within a debugger.*
- virtual unsigned **precedence** () const
 - Return relative operator precedence (for parenthezing output).*
- virtual bool **info** (unsigned inf) const

Information about the object.

- virtual `ex operator[]` (const `ex` &index) const
- virtual `ex operator[]` (size_t i) const
- virtual `ex & operator[]` (const `ex` &index)
- virtual `ex & operator[]` (size_t i)
- virtual bool `has` (const `ex` &other, unsigned `options`=0) const

Test for occurrence of a pattern.

- virtual bool `match` (const `ex` &pattern, `exmap` &repls) const

Check whether the expression matches a given pattern.

- virtual `ex map` (`map_function` &f) const

Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).

- virtual void `accept` (GiNaC::visitor &v) const
- virtual bool `is_polynomial` (const `ex` &var) const

Check whether this is a polynomial in the given variables.

- virtual int `degree` (const `ex` &s) const

Return degree of highest power in object s.

- virtual int `ldegree` (const `ex` &s) const

Return degree of lowest power in object s.

- virtual `ex coeff` (const `ex` &s, int `n`=1) const

Return coefficient of degree n in object s.

- virtual `ex expand` (unsigned `options`=0) const

Expand expression, i.e.

- virtual `ex collect` (const `ex` &s, bool `distributed`=false) const

Sort expanded expression in terms of powers of some object(s).

- virtual `ex series` (const `relational` &r, int `order`, unsigned `options`=0) const

Default implementation of `ex::series()`.

- virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const

Default implementation of `ex::normal()`.

- virtual `ex to_rational` (`exmap` &repl) const

Default implementation of `ex::to_rational()`.

- virtual `ex to_polynomial` (`exmap` &repl) const

- virtual `numeric integer_content` () const

- virtual `ex smod` (const `numeric` &xi) const

Apply symmetric modular homomorphism to an expanded multivariate polynomial.

- virtual `numeric max_coefficient` () const

Implementation `ex::max_coefficient()`.

- virtual `exvector get_free_indices` () const

Return a vector containing the free indices of an expression.

- virtual `return_type_t return_type_tinfo` () const

- template<class T >

void `print_dispatch` (const `print_context` &c, unsigned level) const

Like `print()`, but dispatch to the specified class.

- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const

Like `print()`, but dispatch to the specified class.

- `ex subs_one_level` (const `exmap` &m, unsigned `options`) const

Helper function for `subs()`.

- `ex diff` (const `symbol` &s, unsigned `nth`=1) const

Default interface of `nth` derivative `ex::diff(s, n)`.

- int `compare` (const `basic` &other) const

Compare objects syntactically to establish canonical ordering.

- bool `is_equal` (const `basic` &other) const

Test for syntactic equality.

- const [basic](#) & [hold](#) () const

Stop further evaluation.

- unsigned [gethash](#) () const
- const [basic](#) & [setflag](#) (unsigned f) const

Set some [status_flags](#).

- const [basic](#) & [clearflag](#) (unsigned f) const

Clear some [status_flags](#).

Public Member Functions inherited from [GiNaC::refcounted](#)

- [refcounted](#) () noexcept
- unsigned int [add_reference](#) () noexcept
- unsigned int [remove_reference](#) () noexcept
- unsigned int [get_refcount](#) () const noexcept
- void [set_refcount](#) (unsigned int r) noexcept

Protected Member Functions

- [matrix](#) (unsigned r, unsigned c, const [exvector](#) &m2)
Ctor from representation, for internal use only.
- [matrix](#) (unsigned r, unsigned c, [exvector](#) &&m2)
- bool [match_same_type](#) (const [basic](#) &other) const override
Returns true if the attributes of two objects are similar enough for a match.
- unsigned [return_type](#) () const override
- [ex_determinant_minor](#) () const
Recursive determinant for small matrices having at least one symbolic entry.
- std::vector< unsigned > [echelon_form](#) (unsigned algo, int n)
- int [gauss_elimination](#) (const bool det=false)
Perform the steps of an ordinary Gaussian elimination to bring the m x n matrix into an upper echelon form.
- int [division_free_elimination](#) (const bool det=false)
Perform the steps of division free elimination to bring the m x n matrix into an upper echelon form.
- int [fraction_free_elimination](#) (const bool det=false)
Perform the steps of Bareiss' one-step fraction free elimination to bring the matrix into an upper echelon form.
- std::vector< unsigned > [markowitz_elimination](#) (unsigned n)
- int [pivot](#) (unsigned ro, unsigned co, bool symbolic=true)
Partial pivoting method for matrix elimination schemes.
- void [print_elements](#) (const [print_context](#) &c, const char *row_start, const char *row_end, const char *row↔_sep, const char *col_sep) const
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
- void [do_print_latex](#) (const [print_latex](#) &c, unsigned level) const
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual [ex derivative](#) (const [symbol](#) &s) const
Default implementation of [ex::diff\(\)](#).
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Protected Attributes

- unsigned [row](#)
number of rows
- unsigned [col](#)
number of columns
- [exvector](#) m
representation (cols indexed first)

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.91.1 Detailed Description

Symbolic matrices.

6.91.2 Constructor & Destructor Documentation

6.91.2.1 [matrix\(\)](#) [1/5]

```
GiNaC::matrix::matrix (
    unsigned r,
    unsigned c )
```

Very common ctor.

Initializes to r x c-dimensional zero-matrix.

Parameters

<i>r</i>	number of rows
<i>c</i>	number of cols

References [GiNaC::_ex0](#), [GiNaC::status_flags::not_shareable](#), and [GiNaC::basic::setflag\(\)](#).

6.91.2.2 matrix() [2/5]

```
GiNaC::matrix::matrix (
    unsigned r,
    unsigned c,
    const lst & l )
```

Construct matrix from (flat) list of elements.

If the list has fewer elements than the matrix, the remaining matrix elements are set to zero. If the list has more elements than the matrix, the excessive elements are thrown away.

References [c](#), [m](#), [GiNaC::status_flags::not_shareable](#), [r](#), [GiNaC::basic::setflag\(\)](#), and [x](#).

6.91.2.3 matrix() [3/5]

```
GiNaC::matrix::matrix (
    std::initializer_list< std::initializer_list< ex > > l )
```

Construct a matrix from an 2 dimensional initializer list.

Throws an exception if some row has a different length than all the others.

References [c](#), [col](#), [m](#), [GiNaC::status_flags::not_shareable](#), [r](#), [row](#), and [GiNaC::basic::setflag\(\)](#).

6.91.2.4 matrix() [4/5]

```
GiNaC::matrix::matrix (
    unsigned r,
    unsigned c,
    const exvector & m2 ) [protected]
```

Ctor from representation, for internal use only.

References [GiNaC::status_flags::not_shareable](#), and [GiNaC::basic::setflag\(\)](#).

6.91.2.5 matrix() [5/5]

```
GiNaC::matrix::matrix (
    unsigned r,
    unsigned c,
    exvector && m2 ) [protected]
```

References [GiNaC::status_flags::not_shareable](#), and [GiNaC::basic::setflag\(\)](#).

6.91.3 Member Function Documentation

6.91.3.1 nops()

```
size_t GiNaC::matrix::nops ( ) const [override], [virtual]
```

nops is defined to be rows x columns.

Reimplemented from [GiNaC::basic](#).

References [col](#), and [row](#).

Referenced by [let_op\(\)](#), and [op\(\)](#).

6.91.3.2 op()

```
ex GiNaC::matrix::op (
    size_t i ) const [override], [virtual]
```

returns matrix entry at position (i/col, icol).

Reimplemented from [GiNaC::basic](#).

References [GINAC_ASSERT](#), [m](#), and [nops\(\)](#).

Referenced by [contract_with\(\)](#).

6.91.3.3 let_op()

```
ex & GiNaC::matrix::let_op (
    size_t i ) [override], [virtual]
```

returns writable matrix entry at position (i/col, icol).

Reimplemented from [GiNaC::basic](#).

References [GiNaC::basic::ensure_if_modifiable\(\)](#), [GINAC_ASSERT](#), [m](#), and [nops\(\)](#).

6.91.3.4 evalm()

```
ex GiNaC::matrix::evalm ( ) const [inline], [override], [virtual]
```

Evaluate sums, products and integer powers of matrices.

Reimplemented from [GiNaC::basic](#).

6.91.3.5 subs()

```
ex GiNaC::matrix::subs (
    const exmap & m,
    unsigned options = 0 ) const [override], [virtual]
```

Substitute a set of objects by arbitrary expressions.

The ex returned will already be evaluated.

Reimplemented from [GiNaC::basic](#).

References [c](#), [col](#), [m](#), [options](#), [r](#), [row](#), [subs\(\)](#), and [GiNaC::basic::subs_one_level\(\)](#).

Referenced by [fraction_free_elimination\(\)](#), and [subs\(\)](#).

6.91.3.6 eval_indexed()

```
ex GiNaC::matrix::eval_indexed (
    const basic & i ) const [override], [virtual]
```

Automatic symbolic evaluation of an indexed matrix.

Reimplemented from [GiNaC::basic](#).

References [col](#), [GiNaC::idx::get_dim\(\)](#), [GiNaC::idx::get_value\(\)](#), [GINAC_ASSERT](#), [GiNaC::basic::hold\(\)](#), [GiNaC::is_dummy_pair\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::info_flags::nonnegint](#), [GiNaC::basic::nops\(\)](#), [GiNaC::basic::op\(\)](#), [row](#), and [trace\(\)](#).

6.91.3.7 add_indexed()

```
ex GiNaC::matrix::add_indexed (
    const ex & self,
    const ex & other ) const [override], [virtual]
```

Sum of two indexed matrices.

Reimplemented from [GiNaC::basic](#).

References [add\(\)](#), [col](#), [GINAC_ASSERT](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), [row](#), and [transpose\(\)](#).

6.91.3.8 scalar_mul_indexed()

```
ex GiNaC::matrix::scalar_mul_indexed (
    const ex & self,
    const numeric & other ) const [override], [virtual]
```

Product of an indexed matrix with a number.

Reimplemented from [GiNaC::basic](#).

References [GINAC_ASSERT](#), [mul\(\)](#), [GiNaC::ex::nops\(\)](#), and [GiNaC::ex::op\(\)](#).

6.91.3.9 contract_with()

```
bool GiNaC::matrix::contract_with (
    exvector::iterator self,
    exvector::iterator other,
    exvector & v ) const [override], [virtual]
```

Contraction of an indexed matrix with something else.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex1](#), [col](#), [GINAC_ASSERT](#), [GiNaC::is_dummy_pair\(\)](#), [mul\(\)](#), [GiNaC::ex::op\(\)](#), [op\(\)](#), [row](#), and [transpose\(\)](#).

6.91.3.10 conjugate()

```
ex GiNaC::matrix::conjugate ( ) const [override], [virtual]
```

Complex conjugate every matrix entry.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::are_ex_trivially_equal\(\)](#), [col](#), [GiNaC::ex::conjugate\(\)](#), [m](#), [row](#), and [x](#).

6.91.3.11 real_part()

```
ex GiNaC::matrix::real_part ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [col](#), [m](#), and [row](#).

6.91.3.12 imag_part()

```
ex GiNaC::matrix::imag_part ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [col](#), [m](#), and [row](#).

6.91.3.13 archive()

```
void GiNaC::matrix::archive (
    archive_node & n ) const [override], [virtual]
```

Save (a.k.a.

serialize) object into archive.

Reimplemented from [GiNaC::basic](#).

References [col](#), [m](#), [n](#), and [row](#).

6.91.3.14 read_archive()

```
void GiNaC::matrix::read_archive (
    const archive_node & n,
    lst & syms ) [override], [virtual]
```

Read (a.k.a.

deserialize) object from archive.

Reimplemented from [GiNaC::basic](#).

References [col](#), [m](#), [n](#), and [row](#).

6.91.3.15 match_same_type()

```
bool GiNaC::matrix::match_same_type (
    const basic & other ) const [override], [protected], [virtual]
```

Returns true if the attributes of two objects are similar enough for a match.

This function must not match subexpressions (this is already done by [basic::match\(\)](#)). Only attributes not accessible by [op\(\)](#) should be compared. This is also the reason why this function doesn't take the wildcard replacement list from [match\(\)](#) as an argument: only subexpressions are subject to wildcard matches. Also, this function only needs to be implemented for container classes because [is_equal_same_type\(\)](#) is automatically used instead of [match_same_type\(\)](#) if [nops\(\)](#) == 0.

See also

[basic::match](#)

Reimplemented from [GiNaC::basic](#).

References [col](#), [cols\(\)](#), [GINAC_ASSERT](#), [row](#), and [rows\(\)](#).

6.91.3.16 return_type()

```
unsigned GiNaC::matrix::return_type ( ) const [inline], [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::return_types::noncommutative](#).

6.91.3.17 rows()

```
unsigned GiNaC::matrix::rows ( ) const [inline]
```

Get number of rows.

References [row](#).

Referenced by [division_free_elimination\(\)](#), [fraction_free_elimination\(\)](#), [gauss_elimination\(\)](#), [GiNaC::lsolve\(\)](#), [match_same_type\(\)](#), [mul\(\)](#), [solve\(\)](#), and [transpose\(\)](#).

6.91.3.18 cols()

```
unsigned GiNaC::matrix::cols ( ) const [inline]
```

Get number of columns.

References [col](#).

Referenced by [determinant_minor\(\)](#), [division_free_elimination\(\)](#), [fraction_free_elimination\(\)](#), [gauss_elimination\(\)](#), [GiNaC::lsolve\(\)](#), [match_same_type\(\)](#), [mul\(\)](#), [solve\(\)](#), and [transpose\(\)](#).

6.91.3.19 add()

```
matrix GiNaC::matrix::add (
    const matrix & other ) const
```

Sum of matrices.

Exceptions

<i>logic_error</i>	(incompatible matrices)

References [col](#), [m](#), and [row](#).

Referenced by [add_indexed\(\)](#), and [GiNaC::add::evalm\(\)](#).

6.91.3.20 sub()

```
matrix GiNaC::matrix::sub (
    const matrix & other ) const
```

Difference of matrices.

Exceptions

<i>logic_error</i>	(incompatible matrices)

References [col](#), [m](#), and [row](#).

6.91.3.21 mul() [1/2]

```
matrix GiNaC::matrix::mul (
    const matrix & other ) const
```

Product of matrices.

Exceptions

<i>logic_error</i>	(incompatible matrices)

References [c](#), [col](#), [cols\(\)](#), [GiNaC::is_zero\(\)](#), [m](#), [row](#), and [rows\(\)](#).

Referenced by [contract_with\(\)](#), [GiNaC::ncmul::evalm\(\)](#), [GiNaC::Gt::findMoebiusTransform\(\)](#), [pow\(\)](#), and [scalar_mul_indexed\(\)](#).

6.91.3.22 mul() [2/2]

```
matrix GiNaC::matrix::mul (
    const numeric & other ) const
```

Product of matrix and scalar.

References [c](#), [col](#), [m](#), [r](#), and [row](#).

6.91.3.23 mul_scalar()

```
matrix GiNaC::matrix::mul_scalar (
    const ex & other ) const
```

Product of matrix and scalar expression.

References [c](#), [col](#), [GiNaC::return_types::commutative](#), [m](#), [r](#), [GiNaC::ex::return_type\(\)](#), and [row](#).

6.91.3.24 pow()

```
matrix GiNaC::matrix::pow (
    const ex & expn ) const
```

Power of a matrix.

Currently handles integer exponents only.

References [GiNaC::_ex1](#), [GiNaC::_num1_p](#), [GiNaC::_num2_p](#), [col](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [inverse\(\)](#), [GiNaC::numeric::is_odd\(\)](#), [GiNaC::numeric::is_zero\(\)](#), [mul\(\)](#), [GiNaC::info_flags::negative](#), [r](#), and [row](#).

6.91.3.25 operator>() [1/2]

```
const ex & GiNaC::matrix::operator() (
    unsigned ro,
    unsigned co ) const
```

[operator\(\)](#) to access elements for reading.

Parameters

<i>ro</i>	row of element
<i>co</i>	column of element

Exceptions

<i>range_error</i>	(index out of range)

References [col](#), [m](#), and [row](#).

6.91.3.26 operator>() [2/2]

```
ex & GiNaC::matrix::operator() (
    unsigned ro,
    unsigned co )
```

operator() to access elements for writing.

Parameters

<i>ro</i>	row of element
<i>co</i>	column of element

Exceptions

<i>range_error</i>	(index out of range)

References [col](#), [GiNaC::basic::ensure_if_modifiable\(\)](#), [m](#), and [row](#).

6.91.3.27 set()

```
matrix & GiNaC::matrix::set (
    unsigned ro,
    unsigned co,
    const ex & value ) [inline]
```

References [value](#).

6.91.3.28 transpose()

```
matrix GiNaC::matrix::transpose ( ) const
```

Transposed of an m x n matrix, producing a new n x m matrix object that represents the transposed.

References [c](#), [cols\(\)](#), [m](#), [r](#), and [rows\(\)](#).

Referenced by [add_indexed\(\)](#), [contract_with\(\)](#), and [GiNaC::transpose\(\)](#).

6.91.3.29 determinant()

```
ex GiNaC::matrix::determinant (
    unsigned algo = determinant\_algo::automatic ) const
```

Determinant of square matrix.

This routine doesn't actually calculate the determinant, it only implements some heuristics about which algorithm to run. If all the elements of the matrix are elements of an integral domain the determinant is also in that integral domain and the result is expanded only. If one or more elements are from a quotient field the determinant is usually also in that quotient field and the result is normalized before it is returned. This implies that the determinant of the symbolic 2x2 matrix $\begin{bmatrix} a/(a-b), 1 \\ b/(a-b), 1 \end{bmatrix}$ is returned as unity. (In this respect, it behaves like MapleV and unlike Mathematica.)

Parameters

<i>algo</i>	allows to chose an algorithm

Returns

the determinant as a new expression

Exceptions

<i>logic_error</i>	(matrix not square)

See also

[determinant_algo](#)

References [GiNaC::ex0](#), [GiNaC::determinant_algo::automatic](#), [GiNaC::determinant_algo::bareiss](#), [c](#), [col](#), [GiNaC::info_flags::crational_polynomial](#), [determinant_minor\(\)](#), [GiNaC::determinant_algo::divfree](#), [division_free_elimination\(\)](#), [GiNaC::ex::expand\(\)](#), [fraction_free_elimination\(\)](#), [GiNaC::determinant_algo::gauss](#), [gauss_elimination\(\)](#), [GINAC_ASSERT](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::is_zero\(\)](#), [GiNaC::determinant_algo::laplace](#), [m](#), [GiNaC::ex::normal\(\)](#), [GiNaC::basic::normal\(\)](#), [GiNaC::info_flags::numeric](#), [GiNaC::permutation_sign\(\)](#), [r](#), [GiNaC::info_flags::rational_function](#), [row](#), and [GiNaC::ex::to_rational\(\)](#).

Referenced by [charpoly\(\)](#), and [GiNaC::tensepsilon::contract_with\(\)](#).

6.91.3.30 trace()

```
ex GiNaC::matrix::trace ( ) const
```

Trace of a matrix.

The result is normalized if it is in some quotient field and expanded only otherwise. This implies that the trace of the symbolic 2x2 matrix $\begin{bmatrix} a/(a-b), x \\ y, b/(b-a) \end{bmatrix}$ is recognized to be unity.

Returns

the sum of diagonal elements

Exceptions

<i>logic_error</i>	(matrix not square)

References [col](#), [GiNaC::info_flags::crational_polynomial](#), [GiNaC::ex::expand\(\)](#), [GiNaC::ex::info\(\)](#), [m](#), [GiNaC::ex::normal\(\)](#), [r](#), [GiNaC::info_flags::rational_function](#), and [row](#).

Referenced by [charpoly\(\)](#), and [eval_indexed\(\)](#).

6.91.3.31 charpoly()

```
ex GiNaC::matrix::charpoly (
    const ex & lambda ) const
```

Characteristic Polynomial.

Following mathematica notation the characteristic polynomial of a matrix M is defined as the determinant of (M - lambda * 1) where 1 stands for the unit matrix of the same dimension as M. Note that some CASs define it with a sign inside the determinant which gives rise to an overall sign if the dimension is odd. This method returns the characteristic polynomial collected in powers of lambda as a new expression.

Returns

characteristic polynomial as new expression

Exceptions

<i>logic_error</i>	(matrix not square)

See also

[matrix::determinant\(\)](#)

References [c](#), [col](#), [GiNaC::ex::collect\(\)](#), [determinant\(\)](#), [GiNaC::basic::ex](#), [m](#), [GiNaC::info_flags::numeric](#), [poly](#), [r](#), [row](#), and [trace\(\)](#).

6.91.3.32 inverse() [1/2]

```
matrix GiNaC::matrix::inverse ( ) const
```

Inverse of this matrix, with automatic algorithm selection.

References [GiNaC::solve_algo::automatic](#), and [inverse\(\)](#).

Referenced by [inverse\(\)](#), [GiNaC::inverse\(\)](#), [GiNaC::inverse\(\)](#), and [pow\(\)](#).

6.91.3.33 inverse() [2/2]

```
matrix GiNaC::matrix::inverse (
    unsigned algo ) const
```

Inverse of this matrix.

Parameters

<i>algo</i>	selects the algorithm (one of solve_algo)

Returns

the inverted matrix

Exceptions

<i>logic_error</i>	(matrix not square)
<i>runtime_error</i>	(singular matrix)

References [GiNaC::_ex1](#), [c](#), [col](#), [r](#), [row](#), and [solve\(\)](#).

6.91.3.34 solve()

```
matrix GiNaC::matrix::solve (
    const matrix & vars,
    const matrix & rhs,
    unsigned algo = solve\_algo::automatic ) const
```

Solve a linear system consisting of a m x n matrix and a m x p right hand side by applying an elimination scheme to the augmented matrix.

Parameters

<i>vars</i>	n x p matrix, all elements must be symbols
<i>rhs</i>	m x p matrix
<i>algo</i>	selects the solving algorithm

Returns

n x p solution matrix

Exceptions

<i>logic_error</i>	(incompatible matrices)
<i>invalid_argument</i>	(1st argument must be matrix of symbols)
<i>runtime_error</i>	(inconsistent linear system)

See also

[solve_algo](#)

References [c](#), [cols\(\)](#), [echelon_form\(\)](#), [GiNaC::basic::info\(\)](#), [m](#), [n](#), [GiNaC::basic::normal\(\)](#), [r](#), [GiNaC::rhs\(\)](#), [rows\(\)](#), and [GiNaC::info_flags::symbol](#).

Referenced by [inverse\(\)](#), [GiNaC::lsolve\(\)](#), and [GiNaC::sqrfree_parfrac\(\)](#).

6.91.3.35 rank() [1/2]

```
unsigned GiNaC::matrix::rank ( ) const
```

Compute the rank of this matrix.

References [GiNaC::solve_algo::automatic](#), and [rank\(\)](#).

Referenced by [rank\(\)](#), and [GiNaC::rank\(\)](#).

6.91.3.36 rank() [2/2]

```
unsigned GiNaC::matrix::rank (
    unsigned solve_algo ) const
```

Compute the rank of this matrix using the given algorithm, which should be a member of enum [solve_algo](#).

References [col](#), [echelon_form\(\)](#), [GINAC_ASSERT](#), [m](#), [r](#), and [row](#).

6.91.3.37 is_zero_matrix()

```
bool GiNaC::matrix::is_zero_matrix ( ) const
```

Function to check that all elements of the matrix are zero.

References [m](#).

6.91.3.38 determinant_minor()

```
ex GiNaC::matrix::determinant_minor ( ) const [protected]
```

Recursive determinant for small matrices having at least one symbolic entry.

The basic algorithm, known as Laplace-expansion, is enhanced by some bookkeeping to avoid calculation of the same submatrices ("minors") more than once. According to W.M.Gentleman and S.C.Johnson this algorithm is better than elimination schemes for matrices of sparse multivariate polynomials and also for matrices of dense univariate polynomials if the matrix' dimension is larger than 7.

Returns

the determinant as a new expression (in expanded form)

See also

[matrix::determinant\(\)](#)

References [GiNaC::_ex0](#), [GiNaC::_ex1](#), [c](#), [cols\(\)](#), [GiNaC::ex::expand\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::is_zero\(\)](#), [m](#), [n](#), and [r](#).

Referenced by [determinant\(\)](#).

6.91.3.39 echelon_form()

```
std::vector< unsigned > GiNaC::matrix::echelon_form (
    unsigned algo,
    int n ) [protected]
```

References [GiNaC::solve_algo::automatic](#), [GiNaC::solve_algo::bareiss](#), [c](#), [col](#), [GiNaC::solve_algo::divfree](#), [division_free_elimination\(\)](#), [fraction_free_elimination\(\)](#), [GiNaC::solve_algo::gauss](#), [gauss_elimination\(\)](#), [GiNaC::ex::is_zero\(\)](#), [m](#), [GiNaC::solve_algo::markowitz](#), [markowitz_elimination\(\)](#), [n](#), [GiNaC::info_flags::numeric](#), [r](#), and [row](#).

Referenced by [rank\(\)](#), and [solve\(\)](#).

6.91.3.40 gauss_elimination()

```
int GiNaC::matrix::gauss_elimination (
    const bool det = false ) [protected]
```

Perform the steps of an ordinary Gaussian elimination to bring the m x n matrix into an upper echelon form.

The algorithm is ok for matrices with numeric coefficients but quite unsuited for symbolic matrices.

Parameters

<i>det</i>	may be set to true to save a lot of space if one is only interested in the diagonal elements (i.e. for calculating determinants). The others are set to zero in this case.

Returns

sign is 1 if an even number of rows was swapped, -1 if an odd number of rows was swapped and 0 if the matrix is singular.

References [GiNaC::ex0](#), [c](#), [cols\(\)](#), [GiNaC::basic::ensure_if_modifiable\(\)](#), [GINAC_ASSERT](#), [GiNaC::is_zero\(\)](#), [m](#), [n](#), [GiNaC::basic::normal\(\)](#), [pivot\(\)](#), [r](#), and [rows\(\)](#).

Referenced by [determinant\(\)](#), and [echelon_form\(\)](#).

6.91.3.41 division_free_elimination()

```
int GiNaC::matrix::division_free_elimination (
    const bool det = false ) [protected]
```

Perform the steps of division free elimination to bring the m x n matrix into an upper echelon form.

Parameters

<i>det</i>	may be set to true to save a lot of space if one is only interested in the diagonal elements (i.e. for calculating determinants). The others are set to zero in this case.

Returns

sign is 1 if an even number of rows was swapped, -1 if an odd number of rows was swapped and 0 if the matrix is singular.

References [GiNaC::_ex0](#), [c](#), [cols\(\)](#), [GiNaC::basic::ensure_if_modifiable\(\)](#), [GINAC_ASSERT](#), [m](#), [n](#), [pivot\(\)](#), [r](#), and [rows\(\)](#).

Referenced by [determinant\(\)](#), and [echelon_form\(\)](#).

6.91.3.42 fraction_free_elimination()

```
int GiNaC::matrix::fraction_free_elimination (
    const bool det = false ) [protected]
```

Perform the steps of Bareiss' one-step fraction free elimination to bring the matrix into an upper echelon form.

Fraction free elimination means that divide is used straightforwardly, without computing GCDs first. This is possible, since we know the divisor at each step.

Parameters

<i>det</i>	may be set to true to save a lot of space if one is only interested in the last element (i.e. for calculating determinants). The others are set to zero in this case.

Returns

sign is 1 if an even number of rows was swapped, -1 if an odd number of rows was swapped and 0 if the matrix is singular.

References [GiNaC::_ex0](#), [GiNaC::_ex1](#), [c](#), [cols\(\)](#), [GiNaC::divide\(\)](#), [GiNaC::basic::ensure_if_modifiable\(\)](#), [GiNaC::basic::expand\(\)](#), [GiNaC::ex::expand\(\)](#), [GINAC_ASSERT](#), [GiNaC::ex::is_zero\(\)](#), [m](#), [n](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::ex::normal\(\)](#), [GiNaC::ex::numer_denom\(\)](#), [GiNaC::ex::op\(\)](#), [r](#), [rows\(\)](#), [subs\(\)](#), and [GiNaC::ex::to_rational\(\)](#).

Referenced by [determinant\(\)](#), and [echelon_form\(\)](#).

6.91.3.43 markowitz_elimination()

```
std::vector< unsigned > GiNaC::matrix::markowitz_elimination (
    unsigned n ) [protected]
```

References [GiNaC::_ex0](#), [c](#), [col](#), [GINAC_ASSERT](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::is_zero\(\)](#), [k](#), [m](#), [n](#), [GiNaC::basic::normal\(\)](#), [r](#), [row](#), and [std::swap\(\)](#).

Referenced by [echelon_form\(\)](#).

6.91.3.44 pivot()

```
int GiNaC::matrix::pivot (
    unsigned ro,
    unsigned co,
    bool symbolic = true ) [protected]
```

Partial pivoting method for matrix elimination schemes.

Usual pivoting (symbolic==false) returns the index to the element with the largest absolute value in column ro and swaps the current row with the one where the element was found. With (symbolic==true) it does the same thing with the first non-zero element.

Parameters

<i>ro</i>	is the row from where to begin
<i>co</i>	is the column to be inspected
<i>symbolic</i>	signal if we want the first non-zero element to be pivoted (true) or the one with the largest absolute value (false).

Returns

0 if no interchange occurred, -1 if all are zero (usually signaling a degeneracy) and positive integer k means that rows *ro* and *k* were swapped.

References [GiNaC::abs\(\)](#), [c](#), [col](#), [GiNaC::basic::ensure_if_modifiable\(\)](#), [GiNaC::basic::expand\(\)](#), [GINAC_ASSERT](#), [GiNaC::numeric::is_zero\(\)](#), [GiNaC::is_zero\(\)](#), [k](#), [m](#), [row](#), and [GiNaC::swap\(\)](#).

Referenced by [division_free_elimination\(\)](#), and [gauss_elimination\(\)](#).

6.91.3.45 print_elements()

```
void GiNaC::matrix::print_elements (
    const print_context & c,
    const char * row_start,
    const char * row_end,
    const char * row_sep,
    const char * col_sep ) const [protected]
```

References [c](#), [col](#), [m](#), and [row](#).

Referenced by [do_print\(\)](#), [do_print_latex\(\)](#), and [do_print_python_repr\(\)](#).

6.91.3.46 do_print()

```
void GiNaC::matrix::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

References [c](#), and [print_elements\(\)](#).

6.91.3.47 do_print_latex()

```
void GiNaC::matrix::do_print_latex (
    const print_latex & c,
    unsigned level ) const [protected]
```

References [c](#), [col](#), and [print_elements\(\)](#).

6.91.3.48 do_print_python_repr()

```
void GiNaC::matrix::do_print_python_repr (
    const print\_python\_repr & c,
    unsigned level ) const [protected]
```

References [c](#), and [print_elements\(\)](#).

6.91.4 Member Data Documentation

6.91.4.1 row

```
unsigned GiNaC::matrix::row [protected]
```

number of rows

Referenced by [add\(\)](#), [add_indexed\(\)](#), [archive\(\)](#), [charpoly\(\)](#), [conjugate\(\)](#), [contract_with\(\)](#), [determinant\(\)](#), [echelon_form\(\)](#), [eval_indexed\(\)](#), [imag_part\(\)](#), [inverse\(\)](#), [markowitz_elimination\(\)](#), [match_same_type\(\)](#), [matrix\(\)](#), [mul\(\)](#), [mul\(\)](#), [mul_scalar\(\)](#), [nops\(\)](#), [operator\(\)](#), [operator\(\)](#), [pivot\(\)](#), [pow\(\)](#), [print_elements\(\)](#), [rank\(\)](#), [read_archive\(\)](#), [real_part\(\)](#), [rows\(\)](#), [sub\(\)](#), [subs\(\)](#), and [trace\(\)](#).

6.91.4.2 col

```
unsigned GiNaC::matrix::col [protected]
```

number of columns

Referenced by [add\(\)](#), [add_indexed\(\)](#), [archive\(\)](#), [charpoly\(\)](#), [cols\(\)](#), [conjugate\(\)](#), [contract_with\(\)](#), [determinant\(\)](#), [do_print_latex\(\)](#), [echelon_form\(\)](#), [eval_indexed\(\)](#), [imag_part\(\)](#), [inverse\(\)](#), [markowitz_elimination\(\)](#), [match_same_type\(\)](#), [matrix\(\)](#), [mul\(\)](#), [mul\(\)](#), [mul_scalar\(\)](#), [nops\(\)](#), [operator\(\)](#), [operator\(\)](#), [pivot\(\)](#), [pow\(\)](#), [print_elements\(\)](#), [rank\(\)](#), [read_archive\(\)](#), [real_part\(\)](#), [sub\(\)](#), [subs\(\)](#), and [trace\(\)](#).

6.91.4.3 m

```
exvector GiNaC::matrix::m [protected]
```

representation (cols indexed first)

Referenced by [add\(\)](#), [archive\(\)](#), [charpoly\(\)](#), [conjugate\(\)](#), [determinant\(\)](#), [determinant_minor\(\)](#), [division_free_elimination\(\)](#), [echelon_form\(\)](#), [fraction_free_elimination\(\)](#), [gauss_elimination\(\)](#), [imag_part\(\)](#), [is_zero_matrix\(\)](#), [let_op\(\)](#), [markowitz_elimination\(\)](#), [matrix\(\)](#), [matrix\(\)](#), [mul\(\)](#), [mul\(\)](#), [mul_scalar\(\)](#), [op\(\)](#), [operator\(\)](#), [operator\(\)](#), [pivot\(\)](#), [print_elements\(\)](#), [rank\(\)](#), [read_archive\(\)](#), [real_part\(\)](#), [solve\(\)](#), [sub\(\)](#), [subs\(\)](#), [trace\(\)](#), and [transpose\(\)](#).

The documentation for this class was generated from the following files:

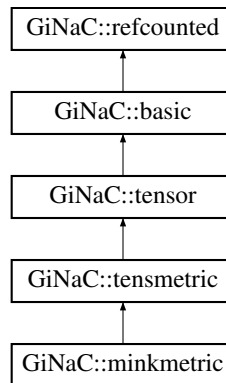
- [matrix.h](#)
- [matrix.cpp](#)

6.92 GiNaC::minkmetric Class Reference

This class represents a Minkowski metric tensor.

```
#include <tensor.h>
```

Inheritance diagram for GiNaC::minkmetric:



Public Member Functions

- [minkmetric](#) (bool [pos_sig](#))
Construct Lorentz metric tensor with given signature.
- bool [info](#) (unsigned int) const override
Information about the object.
- [ex eval_indexed](#) (const [basic](#) &i) const override
Automatic symbolic evaluation of an indexed Lorentz metric tensor.
- void [archive](#) ([archive_node](#) &n) const override
Save (a.k.a.
- void [read_archive](#) (const [archive_node](#) &n, [lst](#) &syms) override
Read (a.k.a.

Public Member Functions inherited from [GiNaC::tensmetric](#)

- bool [info](#) (unsigned int) const override
Information about the object.
- [ex eval_indexed](#) (const [basic](#) &i) const override
Automatic symbolic evaluation of an indexed metric tensor.
- bool [contract_with](#) (exvector::iterator self, exvector::iterator other, [exvector](#) &v) const override
Contraction of an indexed metric tensor with something else.

Public Member Functions inherited from [GiNaC::tensor](#)

- bool [replace_contr_index](#) (exvector::iterator self, exvector::iterator other) const
Replace dummy index in contracted-with object by the contracting object's second index (used internally for delta and metric tensor contractions.

Public Member Functions inherited from GiNaC::basic

- virtual `~basic` ()
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic` (const `basic` &other)
- const `basic` & `operator=` (const `basic` &other)
basic assignment operator: the other object might be of a derived class.
- virtual `basic` * `duplicate` () const
Create a clone of this object on the heap.
- virtual `ex eval` () const
Perform automatic non-interruptive term rewriting rules.
- virtual `ex evalf` () const
Evaluate object numerically.
- virtual `ex evalm` () const
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ` () const
Evaluate integrals, if result is known.
- virtual void `print` (const `print_context` &c, unsigned level=0) const
Output to stream.
- virtual void `dbgprint` () const
Little wrapper around print to be called within a debugger.
- virtual void `dbgprinttree` () const
Little wrapper around printtree to be called within a debugger.
- virtual unsigned `precedence` () const
Return relative operator precedence (for parenthezing output).
- virtual size_t `nops` () const
Number of operands/members.
- virtual `ex op` (size_t i) const
Return operand/member at position i.
- virtual `ex operator[]` (const `ex` &index) const
- virtual `ex operator[]` (size_t i) const
- virtual `ex` & `let_op` (size_t i)
Return modifiable operand/member at position i.
- virtual `ex` & `operator[]` (const `ex` &index)
- virtual `ex` & `operator[]` (size_t i)
- virtual bool `has` (const `ex` &other, unsigned `options`=0) const
Test for occurrence of a pattern.
- virtual bool `match` (const `ex` &pattern, `exmap` &repls) const
Check whether the expression matches a given pattern.
- virtual `ex subs` (const `exmap` &m, unsigned `options`=0) const
Substitute a set of objects by arbitrary expressions.
- virtual `ex map` (`map_function` &f) const
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void `accept` (`GiNaC::visitor` &v) const
- virtual bool `is_polynomial` (const `ex` &var) const
Check whether this is a polynomial in the given variables.
- virtual int `degree` (const `ex` &s) const
Return degree of highest power in object s.
- virtual int `ldegree` (const `ex` &s) const
Return degree of lowest power in object s.
- virtual `ex coeff` (const `ex` &s, int n=1) const

- Return coefficient of degree n in object s .*

 - virtual `ex expand` (unsigned `options=0`) const

Expand expression, i.e.
 - virtual `ex collect` (const `ex` & s , bool `distributed=false`) const

Sort expanded expression in terms of powers of some object(s).
 - virtual `ex series` (const `relational` & r , int `order`, unsigned `options=0`) const

Default implementation of `ex::series()`.
 - virtual `ex normal` (`exmap` & $repl$, `exmap` & rev_lookup , `lst` & $modifier$) const

Default implementation of `ex::normal()`.
 - virtual `ex to_rational` (`exmap` & $repl$) const

Default implementation of `ex::to_rational()`.
 - virtual `ex to_polynomial` (`exmap` & $repl$) const
 - virtual `numeric integer_content` () const
 - virtual `ex smod` (const `numeric` & xi) const
- Apply symmetric modular homomorphism to an expanded multivariate polynomial.*
- virtual `numeric max_coefficient` () const
- Implementation `ex::max_coefficient()`.*
- virtual `exvector get_free_indices` () const
- Return a vector containing the free indices of an expression.*
- virtual `ex add_indexed` (const `ex` & $self$, const `ex` & $other$) const
- Add two indexed expressions.*
- virtual `ex scalar_mul_indexed` (const `ex` & $self$, const `numeric` & $other$) const
- Multiply an indexed expression with a scalar.*
- virtual `return_type_t return_type_info` () const
 - virtual `ex conjugate` () const
 - virtual `ex real_part` () const
 - virtual `ex imag_part` () const
- template<class T >
 - void `print_dispatch` (const `print_context` & c , unsigned `level`) const
- Like `print()`, but dispatch to the specified class.*
- void `print_dispatch` (const `registered_class_info` & ri , const `print_context` & c , unsigned `level`) const
- Like `print()`, but dispatch to the specified class.*
- `ex subs_one_level` (const `exmap` & m , unsigned `options`) const
- Helper function for `subs()`.*
- `ex diff` (const `symbol` & s , unsigned `nth=1`) const
- Default interface of n th derivative `ex::diff(s, n)`.*
- int `compare` (const `basic` & $other$) const
- Compare objects syntactically to establish canonical ordering.*
- bool `is_equal` (const `basic` & $other$) const
- Test for syntactic equality.*
- const `basic` & `hold` () const
- Stop further evaluation.*
- unsigned `gethash` () const
 - const `basic` & `setflag` (unsigned `f`) const
- Set some `status_flags`.*
- const `basic` & `clearflag` (unsigned `f`) const
- Clear some `status_flags`.*

Public Member Functions inherited from GiNaC::refcounted

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Protected Member Functions

- unsigned `return_type` () const override
- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_latex` (const `print_latex` &c, unsigned level) const

Protected Member Functions inherited from GiNaC::tensmetric

- unsigned `return_type` () const override
- void `do_print` (const `print_context` &c, unsigned level) const

Protected Member Functions inherited from GiNaC::tensor

- unsigned `return_type` () const override

Protected Member Functions inherited from GiNaC::basic

- `basic` ()
- virtual `ex eval_ncmul` (const `exvector` &v) const
- virtual bool `match_same_type` (const `basic` &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual `ex derivative` (const `symbol` &s) const
Default implementation of `ex::diff()`.
- virtual int `compare_same_type` (const `basic` &other) const
Returns order relation between two objects of same type.
- virtual bool `is_equal_same_type` (const `basic` &other) const
Returns true if two objects of same type are equal.
- virtual unsigned `calchash` () const
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- void `ensure_if_modifiable` () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void `do_print` (const `print_context` &c, unsigned level) const
Default output to stream.
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
Tree output to stream.
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
Python parsable output to stream.

Private Attributes

- bool `pos_sig`
If true, the metric is $\text{diag}(-1, 1, 1, \dots)$.

Additional Inherited Members

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.92.1 Detailed Description

This class represents a Minkowski metric tensor.

It has all the properties of a metric tensor and is (as a matrix) equal to $\text{diag}(1,-1,-1,\dots)$ or $\text{diag}(-1,1,1,\dots)$.

6.92.2 Constructor & Destructor Documentation

6.92.2.1 `minkmetric()`

```
GiNaC::minkmetric::minkmetric (
    bool pos_sig )
```

Construct Lorentz metric tensor with given signature.

6.92.3 Member Function Documentation

6.92.3.1 `info()`

```
bool GiNaC::minkmetric::info (
    unsigned int ) const [override], [virtual]
```

Information about the object.

See also

class [info_flags](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::info_flags::real](#).

6.92.3.2 `eval_indexed()`

```
ex GiNaC::minkmetric::eval_indexed (
    const basic & i ) const [override], [virtual]
```

Automatic symbolic evaluation of an indexed Lorentz metric tensor.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex0](#), [GiNaC::_ex1](#), [GiNaC::_ex_1](#), [GiNaC::idx::get_value\(\)](#), [GINAC_ASSERT](#), [GiNaC::info_flags::nonnegint](#), [GiNaC::basic::nops\(\)](#), [GiNaC::basic::op\(\)](#), and [pos_sig](#).

6.92.3.3 archive()

```
void GiNaC::minkmetric::archive (
    archive_node & n ) const [override], [virtual]
```

Save (a.k.a.

serialize) object into archive.

Reimplemented from [GiNaC::basic](#).

References [n](#), and [pos_sig](#).

6.92.3.4 read_archive()

```
void GiNaC::minkmetric::read_archive (
    const archive_node & n,
    lst & syms ) [override], [virtual]
```

Read (a.k.a.

deserialize) object from archive.

Reimplemented from [GiNaC::basic](#).

References [n](#), and [pos_sig](#).

6.92.3.5 return_type()

```
unsigned GiNaC::minkmetric::return_type ( ) const [inline], [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::return_types::commutative](#).

6.92.3.6 do_print()

```
void GiNaC::minkmetric::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

6.92.3.7 do_print_latex()

```
void GiNaC::minkmetric::do_print_latex (
    const print_latex & c,
    unsigned level ) const [protected]
```

6.92.4 Member Data Documentation

6.92.4.1 pos_sig

```
bool GiNaC::minkmetric::pos_sig [private]
```

If true, the metric is $\text{diag}(-1,1,1,\dots)$.

Otherwise it is $\text{diag}(1,-1,-1,\dots)$.

Referenced by [archive\(\)](#), [eval_indexed\(\)](#), and [read_archive\(\)](#).

The documentation for this class was generated from the following files:

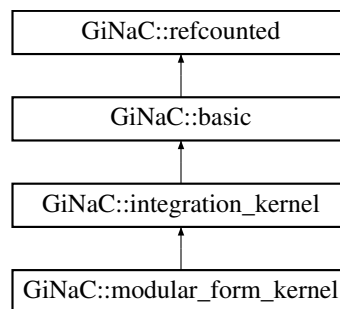
- [tensor.h](#)
- [tensor.cpp](#)

6.93 GiNaC::modular_form_kernel Class Reference

A kernel corresponding to a polynomial in Eisenstein series.

```
#include <integration_kernel.h>
```

Inheritance diagram for GiNaC::modular_form_kernel:



Public Member Functions

- [modular_form_kernel](#) (const [ex](#) &k, const [ex](#) &P, const [ex](#) &C_norm=numeric(1))
- [ex series](#) (const [relational](#) &r, int [order](#), unsigned [options](#)=0) const override
The series method for this class returns the qbar-expansion of the modular form, without an additional factor of C_norm/qbar.
- [size_t nops](#) () const override
Number of operands/members.
- [ex op](#) (size_t i) const override
Return operand/member at position i.
- [ex & let_op](#) (size_t i) override
Return modifiable operand/member at position i.
- [bool is_numeric](#) (void) const override
This routine returns true, if the integration kernel can be evaluated numerically.
- [ex Laurent_series](#) (const [ex](#) &qbar, int [order](#)) const override
Returns the Laurent series, starting possibly with the pole term.
- [ex get_numerical_value](#) (const [ex](#) &qbar, int N_trunc=0) const override
Returns the value of the modular form.
- [ex q_expansion_modular_form](#) (const [ex](#) &q, int [order](#)) const

Public Member Functions inherited from GiNaC::integration_kernel

- virtual bool [has_trailing_zero](#) (void) const
This routine returns true, if the integration kernel has a trailing zero.
- size_t [get_cache_size](#) (void) const
Returns the current size of the cache.
- void [set_cache_step](#) (int cache_steps) const
Sets the step size by which the cache is increased.
- [ex](#) [get_series_coeff](#) (int i) const
Wrapper around series_coeff(i), converts cl_N to numeric.
- cln::cl_N [series_coeff](#) (int i) const
Subclasses have either to implement series_coeff_impl or the two methods Laurent_series and uses_Laurent_series.

Public Member Functions inherited from GiNaC::basic

- virtual [~basic](#) ()
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- [basic](#) (const [basic](#) &other)
- const [basic](#) & [operator=](#) (const [basic](#) &other)
basic assignment operator: the other object might be of a derived class.
- virtual [basic](#) * [duplicate](#) () const
Create a clone of this object on the heap.
- virtual [ex](#) [eval](#) () const
Perform automatic non-interruptive term rewriting rules.
- virtual [ex](#) [evalf](#) () const
Evaluate object numerically.
- virtual [ex](#) [evalm](#) () const
Evaluate sums, products and integer powers of matrices.
- virtual [ex](#) [eval_integ](#) () const
Evaluate integrals, if result is known.
- virtual [ex](#) [eval_indexed](#) (const [basic](#) &i) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void [print](#) (const [print_context](#) &c, unsigned level=0) const
Output to stream.
- virtual void [dbgprint](#) () const
Little wrapper around print to be called within a debugger.
- virtual void [dbgprinttree](#) () const
Little wrapper around printtree to be called within a debugger.
- virtual unsigned [precedence](#) () const
Return relative operator precedence (for parenthezing output).
- virtual bool [info](#) (unsigned inf) const
Information about the object.
- virtual [ex](#) [operator\[\]](#) (const [ex](#) &index) const
- virtual [ex](#) [operator\[\]](#) (size_t i) const
- virtual [ex](#) & [operator\[\]](#) (const [ex](#) &index)
- virtual [ex](#) & [operator\[\]](#) (size_t i)
- virtual bool [has](#) (const [ex](#) &other, unsigned [options](#)=0) const
Test for occurrence of a pattern.
- virtual bool [match](#) (const [ex](#) &pattern, [exmap](#) &repls) const
Check whether the expression matches a given pattern.
- virtual [ex](#) [subs](#) (const [exmap](#) &m, unsigned [options](#)=0) const

- Substitute a set of objects by arbitrary expressions.*

 - virtual `ex map` (`map_function` &`f`) const

Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).

 - virtual void `accept` (`GiNaC::visitor` &`v`) const
 - virtual bool `is_polynomial` (const `ex` &`var`) const

Check whether this is a polynomial in the given variables.

 - virtual int `degree` (const `ex` &`s`) const

Return degree of highest power in object `s`.

 - virtual int `ldegree` (const `ex` &`s`) const

Return degree of lowest power in object `s`.

 - virtual `ex coeff` (const `ex` &`s`, int `n`=1) const

Return coefficient of degree `n` in object `s`.

 - virtual `ex expand` (unsigned `options`=0) const

Expand expression, i.e.

 - virtual `ex collect` (const `ex` &`s`, bool `distributed`=false) const

Sort expanded expression in terms of powers of some object(s).

 - virtual `ex normal` (`exmap` &`repl`, `exmap` &`rev_lookup`, `lst` &`modifier`) const

Default implementation of `ex::normal()`.

 - virtual `ex to_rational` (`exmap` &`repl`) const

Default implementation of `ex::to_rational()`.

 - virtual `ex to_polynomial` (`exmap` &`repl`) const
 - virtual `numeric integer_content` () const
 - virtual `ex smod` (const `numeric` &`xi`) const

Apply symmetric modular homomorphism to an expanded multivariate polynomial.

 - virtual `numeric max_coefficient` () const

Implementation `ex::max_coefficient()`.

 - virtual `exvector get_free_indices` () const

Return a vector containing the free indices of an expression.

 - virtual `ex add_indexed` (const `ex` &`self`, const `ex` &`other`) const

Add two indexed expressions.

 - virtual `ex scalar_mul_indexed` (const `ex` &`self`, const `numeric` &`other`) const

Multiply an indexed expression with a scalar.

 - virtual bool `contract_with` (`exvector::iterator` `self`, `exvector::iterator` `other`, `exvector` &`v`) const

Try to contract two indexed expressions that appear in the same product.

 - virtual unsigned `return_type` () const
 - virtual `return_type_t return_type_tinfo` () const
 - virtual `ex conjugate` () const
 - virtual `ex real_part` () const
 - virtual `ex imag_part` () const
 - template<class `T` >
 - void `print_dispatch` (const `print_context` &`c`, unsigned `level`) const

Like `print()`, but dispatch to the specified class.

 - void `print_dispatch` (const `registered_class_info` &`ri`, const `print_context` &`c`, unsigned `level`) const

Like `print()`, but dispatch to the specified class.

 - virtual void `archive` (`archive_node` &`n`) const

Save (serialize) the object into archive node.

 - virtual void `read_archive` (const `archive_node` &`n`, `lst` &`syms`)

Load (deserialize) the object from an archive node.

 - `ex subs_one_level` (const `exmap` &`m`, unsigned `options`) const

Helper function for `subs()`.

 - `ex diff` (const `symbol` &`s`, unsigned `nth`=1) const

Default interface of *nth* derivative *ex::diff(s, n)*.

- int [compare](#) (const [basic](#) &other) const
Compare objects syntactically to establish canonical ordering.
- bool [is_equal](#) (const [basic](#) &other) const
Test for syntactic equality.
- const [basic](#) & [hold](#) () const
Stop further evaluation.
- unsigned [gethash](#) () const
- const [basic](#) & [setflag](#) (unsigned f) const
Set some *status_flags*.
- const [basic](#) & [clearflag](#) (unsigned f) const
Clear some *status_flags*.

Public Member Functions inherited from [GiNaC::refcounted](#)

- [refcounted](#) () noexcept
- unsigned int [add_reference](#) () noexcept
- unsigned int [remove_reference](#) () noexcept
- unsigned int [get_refcount](#) () const noexcept
- void [set_refcount](#) (unsigned int r) noexcept

Protected Member Functions

- bool [uses_Laurent_series](#) () const override
Returns true, if the coefficients are computed from the Laurent series (in which case the method *Laurent_series* needs to be implemented).
- void [do_print](#) (const [print_context](#) &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::integration_kernel](#)

- virtual [cln::cl_N](#) [series_coeff_impl](#) (int i) const
For $\omega = d\lambda$ only the coefficient of λ^0 is non-zero.
- [ex](#) [get_numerical_value_impl](#) (const [ex](#) &lambda, const [ex](#) &pre, int shift, int N_trunc) const
The actual implementation for computing a numerical value for the integrand.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex](#) [eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual [ex](#) [derivative](#) (const [symbol](#) &s) const
Default implementation of *ex::diff()*.
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- virtual unsigned [calchash](#) () const

Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.

- void [ensure_if_modifiable](#) () const

Ensure the object may be modified without hurting others, throws if this is not the case.

- void [do_print](#) (const [print_context](#) &c, unsigned level) const

Default output to stream.

- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const

Tree output to stream.

- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const

Python parsable output to stream.

Protected Attributes

- [ex k](#)
- [ex P](#)
- [ex C_norm](#)

Protected Attributes inherited from [GiNaC::integration_kernel](#)

- int [cache_step_size](#)
- std::vector< cln::cl_N > [series_vec](#)

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.93.1 Detailed Description

A kernel corresponding to a polynomial in Eisenstein series.

This class represents the differential one-form

$$\omega^{\text{modular}}(P_k(\eta_{k_1}^{(1)}, \dots, \eta_{k_r}^{(r)})) = C_k P_k(\eta_{k_1}^{(1)}, \dots, \eta_{k_r}^{(r)}) \frac{d\bar{q}_N}{\bar{q}_N}.$$

6.93.2 Constructor & Destructor Documentation

6.93.2.1 modular_form_kernel()

```
GiNaC::modular_form_kernel::modular_form_kernel (
    const ex & k,
    const ex & P,
    const ex & C_norm = numeric(1) )
```

6.93.3 Member Function Documentation

6.93.3.1 series()

```
ex GiNaC::modular_form_kernel::series (
    const relational & r,
    int order,
    unsigned options = 0 ) const [override], [virtual]
```

The series method for this class returns the qbar-expansion of the modular form, without an additional factor of $C\_norm/qbar$.

Reimplemented from [GiNaC::integration_kernel](#).

References [GiNaC::ex::lhs\(\)](#), [order](#), [P](#), [GiNaC::pow\(\)](#), [qbar](#), [r](#), and [GiNaC::ex::series\(\)](#).

Referenced by [q_expansion_modular_form\(\)](#).

6.93.3.2 nops()

```
size_t GiNaC::modular_form_kernel::nops ( ) const [override], [virtual]
```

Number of operands/members.

Reimplemented from [GiNaC::basic](#).

6.93.3.3 op()

```
ex & GiNaC::modular_form_kernel::op (
    size_t i ) const [override], [virtual]
```

Return operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [C_norm](#), [k](#), and [P](#).

6.93.3.4 let_op()

```
ex & GiNaC::modular_form_kernel::let_op (
    size_t i ) [override], [virtual]
```

Return modifiable operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [C_norm](#), [GiNaC::basic::ensure_if_modifiable\(\)](#), [k](#), and [P](#).

6.93.3.5 is_numeric()

```
bool GiNaC::modular_form_kernel::is_numeric (
    void ) const [override], [virtual]
```

This routine returns true, if the integration kernel can be evaluated numerically.

Reimplemented from [GiNaC::integration_kernel](#).

References [C_norm](#), [GiNaC::ex::evalf\(\)](#), [GiNaC::ex::info\(\)](#), [k](#), [GiNaC::info_flags::nonnegint](#), [GiNaC::info_flags::numeric](#), [q_expansion_modular_form\(\)](#), [qbar](#), [GiNaC::series_to_poly\(\)](#), and [GiNaC::ex::subs\(\)](#).

6.93.3.6 Laurent_series()

```
ex GiNaC::modular_form_kernel::Laurent_series (
    const ex & x,
    int order ) const [override], [virtual]
```

Returns the Laurent series, starting possibly with the pole term.

Neglected terms are of order $O(x^{order})$.

Reimplemented from [GiNaC::integration_kernel](#).

References [C_norm](#), [order](#), [q_expansion_modular_form\(\)](#), [qbar](#), [GiNaC::ex::series\(\)](#), and [GiNaC::series_to_poly\(\)](#).

6.93.3.7 get_numerical_value()

```
ex GiNaC::modular_form_kernel::get_numerical_value (
    const ex & qbar,
    int N_trunc = 0 ) const [override], [virtual]
```

Returns the value of the modular form.

Reimplemented from [GiNaC::integration_kernel](#).

References [C_norm](#), [GiNaC::integration_kernel::get_numerical_value_impl\(\)](#), and [qbar](#).

6.93.3.8 uses_Laurent_series()

```
bool GiNaC::modular_form_kernel::uses_Laurent_series ( ) const [override], [protected], [virtual]
```

Returns true, if the coefficients are computed from the Laurent series (in which case the method `Laurent_series` needs to be implemented).

Returns false if this is not the case (and the class has an implementation of `series_coeff_impl`).

Reimplemented from [GiNaC::integration_kernel](#).

6.93.3.9 q_expansion_modular_form()

```
ex GiNaC::modular_form_kernel::q_expansion_modular_form (
    const ex & q,
    int order ) const
```

References [series\(\)](#).

Referenced by [is_numeric\(\)](#), and [Laurent_series\(\)](#).

6.93.3.10 do_print()

```
void GiNaC::modular_form_kernel::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

References [c](#), [C_norm](#), [k](#), [P](#), and [GiNaC::ex::print\(\)](#).

6.93.4 Member Data Documentation

6.93.4.1 k

```
ex GiNaC::modular_form_kernel::k [protected]
```

Referenced by [do_print\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), and [op\(\)](#).

6.93.4.2 P

```
ex GiNaC::modular_form_kernel::P [protected]
```

Referenced by [do_print\(\)](#), [let_op\(\)](#), [op\(\)](#), and [series\(\)](#).

6.93.4.3 C_norm

```
ex GiNaC::modular_form_kernel::C_norm [protected]
```

Referenced by [do_print\(\)](#), [get_numerical_value\(\)](#), [is_numeric\(\)](#), [Laurent_series\(\)](#), [let_op\(\)](#), and [op\(\)](#).

The documentation for this class was generated from the following files:

- [integration_kernel.h](#)
- [integration_kernel.cpp](#)

6.94 GiNaC::basic_partition_generator::mpartition2 Struct Reference

```
#include <utils.h>
```

Public Member Functions

- [mpartition2](#) (unsigned *n_*, unsigned *m_*)
- bool [next_partition](#) ()

Public Attributes

- std::vector< unsigned > [x](#)
- unsigned [n](#)
- unsigned [m](#)

6.94.1 Constructor & Destructor Documentation

6.94.1.1 mpartition2()

```
GiNaC::basic_partition_generator::mpartition2::mpartition2 (
    unsigned n_,
    unsigned m_ ) [inline]
```

References [k](#), [m](#), [n](#), and [x](#).

6.94.2 Member Function Documentation

6.94.2.1 next_partition()

```
bool GiNaC::basic_partition_generator::mpartition2::next_partition ( ) [inline]
```

References [k](#), [m](#), and [x](#).

Referenced by [GiNaC::partition_with_zero_parts_generator::next\(\)](#), and [GiNaC::partition_generator::next\(\)](#).

6.94.3 Member Data Documentation

6.94.3.1 x

```
std::vector<unsigned> GiNaC::basic_partition_generator::mpartition2::x
```

Referenced by [GiNaC::partition_with_zero_parts_generator::get\(\)](#), [GiNaC::partition_generator::get\(\)](#), [mpartition2\(\)](#), and [next_partition\(\)](#).

6.94.3.2 n

```
unsigned GiNaC::basic_partition_generator::mpartition2::n
```

Referenced by [mpartition2\(\)](#), and [GiNaC::partition_with_zero_parts_generator::next\(\)](#).

6.94.3.3 m

```
unsigned GiNaC::basic_partition_generator::mpartition2::m
```

Referenced by [GiNaC::partition_with_zero_parts_generator::get\(\)](#), [GiNaC::partition_generator::get\(\)](#), [mpartition2\(\)](#), [GiNaC::partition_with_zero_parts_generator::next\(\)](#), and [next_partition\(\)](#).

The documentation for this struct was generated from the following file:

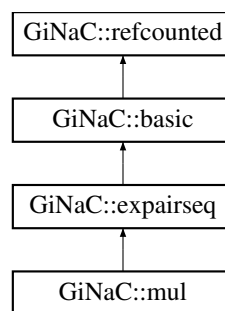
- [utils.h](#)

6.95 GiNaC::mul Class Reference

Product of expressions.

```
#include <mul.h>
```

Inheritance diagram for GiNaC::mul:



Public Member Functions

- [mul](#) (const [ex](#) &lh, const [ex](#) &rh)
- [mul](#) (const [exvector](#) &v)
- [mul](#) (const [epvector](#) &v)
- [mul](#) (const [epvector](#) &v, const [ex](#) &oc, bool do_index_renaming=false)
- [mul](#) ([epvector](#) &&vp)
- [mul](#) ([epvector](#) &&vp, const [ex](#) &oc, bool do_index_renaming=false)
- [mul](#) (const [ex](#) &lh, const [ex](#) &mh, const [ex](#) &rh)
- unsigned [precedence](#) () const override
Return relative operator precedence (for parenthezing output).
- bool [info](#) (unsigned inf) const override
Information about the object.
- bool [is_polynomial](#) (const [ex](#) &var) const override
Check whether this is a polynomial in the given variables.
- int [degree](#) (const [ex](#) &s) const override
Return degree of highest power in object s.
- int [ldegree](#) (const [ex](#) &s) const override
Return degree of lowest power in object s.
- [ex coeff](#) (const [ex](#) &s, int n=1) const override
Return coefficient of degree n in object s.

- `bool has` (const `ex` &other, unsigned `options`=0) const override
Test for occurrence of a pattern.
- `ex eval` () const override
Perform automatic term rewriting rules in this class.
- `ex evalf` () const override
Evaluate object numerically.
- `ex real_part` () const override
- `ex imag_part` () const override
- `ex evalm` () const override
Evaluate sums, products and integer powers of matrices.
- `ex series` (const `relational` &s, int `order`, unsigned `options`=0) const override
Implementation of `ex::series()` for product.
- `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const override
Implementation of `ex::normal()` for a product.
- `numeric integer_content` () const override
- `ex smod` (const `numeric` &xi) const override
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- `numeric max_coefficient` () const override
Implementation `ex::max_coefficient()`.
- `exvector get_free_indices` () const override
Return a vector containing the free indices of an expression.
- `ex conjugate` () const override
- `ex algebraic_subs_mul` (const `exmap` &m, unsigned `options`) const

Public Member Functions inherited from `GiNaC::expairseq`

- `expairseq` (const `ex` &lh, const `ex` &rh)
- `expairseq` (const `exvector` &v)
- `expairseq` (const `epvector` &v, const `ex` &oc, bool do_index_renaming=false)
- `expairseq` (`epvector` &&vp, const `ex` &oc, bool do_index_renaming=false)
- `size_t nops` () const override
Number of operands/members.
- `ex op` (size_t i) const override
Return operand/member at position i.
- `ex map` (`map_function` &f) const override
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- `ex to_rational` (`exmap` &repl) const override
Implementation of `ex::to_rational()` for expairseqs.
- `ex to_polynomial` (`exmap` &repl) const override
Implementation of `ex::to_polynomial()` for expairseqs.
- `bool match` (const `ex` &pattern, `exmap` &repl_lst) const override
Check whether the expression matches a given pattern.
- `ex subs` (const `exmap` &m, unsigned `options`=0) const override
Substitute a set of objects by arbitrary expressions.
- `void archive` (`archive_node` &n) const override
Save (serialize) the object into archive node.
- `void read_archive` (const `archive_node` &n, `lst` &syms) override
Load (deserialize) the object from an archive node.

Public Member Functions inherited from GiNaC::basic

- virtual `~basic` ()
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic` (const `basic` &other)
- const `basic` & `operator=` (const `basic` &other)
basic assignment operator: the other object might be of a derived class.
- virtual `basic` * `duplicate` () const
Create a clone of this object on the heap.
- virtual `ex eval_integ` () const
Evaluate integrals, if result is known.
- virtual `ex eval_indexed` (const `basic` &i) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void `print` (const `print_context` &c, unsigned level=0) const
Output to stream.
- virtual void `dbgprint` () const
Little wrapper around print to be called within a debugger.
- virtual void `dbgprinttree` () const
Little wrapper around printtree to be called within a debugger.
- virtual `ex operator[]` (const `ex` &index) const
- virtual `ex operator[]` (size_t i) const
- virtual `ex` & `let_op` (size_t i)
Return modifiable operand/member at position i.
- virtual `ex` & `operator[]` (const `ex` &index)
- virtual `ex` & `operator[]` (size_t i)
- virtual void `accept` (GiNaC::visitor &v) const
- virtual `ex collect` (const `ex` &s, bool distributed=false) const
Sort expanded expression in terms of powers of some object(s).
- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
Add two indexed expressions.
- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
Multiply an indexed expression with a scalar.
- virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const
Try to contract two indexed expressions that appear in the same product.
- template<class T >
void `print_dispatch` (const `print_context` &c, unsigned level) const
Like print(), but dispatch to the specified class.
- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const
Like print(), but dispatch to the specified class.
- `ex subs_one_level` (const `exmap` &m, unsigned `options`) const
Helper function for subs().
- `ex diff` (const `symbol` &s, unsigned nth=1) const
Default interface of nth derivative ex::diff(s, n).
- int `compare` (const `basic` &other) const
Compare objects syntactically to establish canonical ordering.
- bool `is_equal` (const `basic` &other) const
Test for syntactic equality.
- const `basic` & `hold` () const
Stop further evaluation.
- unsigned `gethash` () const
- const `basic` & `setflag` (unsigned f) const
Set some status_flags.
- const `basic` & `clearflag` (unsigned f) const
Clear some status_flags.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Protected Member Functions

- `ex derivative` (const `symbol` &s) const override
Implementation of `ex::diff()` for a product.
- `ex eval_ncmul` (const `exvector` &v) const override
- unsigned `return_type` () const override
- `return_type_t return_type_tinfo` () const override
- `ex thisexpairseq` (const `epvector` &v, const `ex` &oc, bool do_index_renaming=false) const override
Create an object of this type.
- `ex thisexpairseq` (`epvector` &&vp, const `ex` &oc, bool do_index_renaming=false) const override
- `expair split_ex_to_pair` (const `ex` &e) const override
Form an expair from an ex, using the corresponding semantics.
- `expair combine_ex_with_coeff_to_pair` (const `ex` &e, const `ex` &c) const override
- `expair combine_pair_with_coeff_to_pair` (const `expair` &p, const `ex` &c) const override
- `ex recombine_pair_to_ex` (const `expair` &p) const override
Form an ex out of an expair, using the corresponding semantics.
- bool `expair_needs_further_processing` (`ep` it) override
- `ex default_overall_coeff` () const override
- void `combine_overall_coeff` (const `ex` &c) override
- void `combine_overall_coeff` (const `ex` &c1, const `ex` &c2) override
- bool `can_make_flat` (const `expair` &p) const override
- `ex expand` (unsigned `options`=0) const override
Expand expression, i.e.
- void `find_real_imag` (`ex` &, `ex` &) const
- void `print_overall_coeff` (const `print_context` &c, const char *mul_sym) const
- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_latex` (const `print_latex` &c, unsigned level) const
- void `do_print_csrc` (const `print_csrc` &c, unsigned level) const
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
- `epvector expandchildren` (unsigned `options`) const
Member-wise expand the expairs representing this sequence.

Protected Member Functions inherited from `GiNaC::expairseq`

- bool `is_equal_same_type` (const `basic` &other) const override
Returns true if two objects of same type are equal.
- unsigned `calchash` () const override
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- virtual void `printseq` (const `print_context` &c, char delim, unsigned this_precedence, unsigned upper_precedence) const
- virtual void `printpair` (const `print_context` &c, const `expair` &p, unsigned upper_precedence) const
- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
- void `construct_from_2_ex` (const `ex` &lh, const `ex` &rh)

- void `construct_from_2_expireseq` (const `expireseq` &s1, const `expireseq` &s2)
- void `construct_from_expireseq_ex` (const `expireseq` &s, const `ex` &e)
- void `construct_from_exvector` (const `exvector` &v)
- void `construct_from_epvector` (const `epvector` &v, bool do_index_renaming=false)
- void `construct_from_epvector` (`epvector` &&v, bool do_index_renaming=false)
- void `make_flat` (const `exvector` &v)
Combine this `expireseq` with argument `exvector`.
- void `make_flat` (const `epvector` &v, bool do_index_renaming=false)
Combine this `expireseq` with argument `epvector`.
- void `canonicalize` ()
Brings this `expireseq` into a sorted (canonical) form.
- void `combine_same_terms_sorted_seq` ()
Compact a presorted `expireseq` by combining all matching `expires` to one each.
- bool `is_canonical` () const
Check if this `expireseq` is in sorted (canonical) form.
- `epvector` `expandchildren` (unsigned `options`) const
Member-wise expand the `expires` in this sequence.
- `epvector` `evalchildren` () const
Member-wise evaluate the `expires` in this sequence.
- `epvector` `subchildren` (const `exmap` &m, unsigned `options`=0) const
Member-wise substitute in this sequence.

Protected Member Functions inherited from `GiNaC::basic`

- `basic` ()
- virtual bool `match_same_type` (const `basic` &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual int `compare_same_type` (const `basic` &other) const
Returns order relation between two objects of same type.
- void `ensure_if_modifiable` () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void `do_print` (const `print_context` &c, unsigned level) const
Default output to stream.
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
Tree output to stream.
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
Python parsable output to stream.

Static Protected Member Functions

- static bool `can_be_further_expanded` (const `ex` &e)

Friends

- class `add`
- class `ncmul`
- class `power`

Additional Inherited Members

Protected Attributes inherited from [GiNaC::expairseq](#)

- [epvector seq](#)
- [ex overall_coeff](#)

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.95.1 Detailed Description

Product of expressions.

6.95.2 Constructor & Destructor Documentation

6.95.2.1 [mul\(\)](#) [1/7]

```
GiNaC::mul::mul (
    const ex & lh,
    const ex & rh )
```

References [GiNaC::_ex1](#), [GiNaC::expairseq::construct_from_2_ex\(\)](#), [GINAC_ASSERT](#), [GiNaC::expairseq::is_canonical\(\)](#), and [GiNaC::expairseq::overall_coeff](#).

6.95.2.2 [mul\(\)](#) [2/7]

```
GiNaC::mul::mul (
    const exvector & v )
```

References [GiNaC::_ex1](#), [GiNaC::expairseq::construct_from_exvector\(\)](#), [GINAC_ASSERT](#), [GiNaC::expairseq::is_canonical\(\)](#), and [GiNaC::expairseq::overall_coeff](#).

6.95.2.3 [mul\(\)](#) [3/7]

```
GiNaC::mul::mul (
    const epvector & v )
```

References [GiNaC::_ex1](#), [GiNaC::expairseq::construct_from_epvector\(\)](#), [GINAC_ASSERT](#), [GiNaC::expairseq::is_canonical\(\)](#), and [GiNaC::expairseq::overall_coeff](#).

6.95.2.4 mul() [4/7]

```
GiNaC::mul::mul (
    const epvector & v,
    const ex & oc,
    bool do_index_renaming = false )
```

References [GiNaC::expairseq::construct_from_epvector\(\)](#), [GINAC_ASSERT](#), [GiNaC::expairseq::is_canonical\(\)](#), and [GiNaC::expairseq::overall_coeff](#).

6.95.2.5 mul() [5/7]

```
GiNaC::mul::mul (
    epvector && vp )
```

References [GiNaC::_ex1](#), [GiNaC::expairseq::construct_from_epvector\(\)](#), [GINAC_ASSERT](#), [GiNaC::expairseq::is_canonical\(\)](#), and [GiNaC::expairseq::overall_coeff](#).

6.95.2.6 mul() [6/7]

```
GiNaC::mul::mul (
    epvector && vp,
    const ex & oc,
    bool do_index_renaming = false )
```

References [GiNaC::expairseq::construct_from_epvector\(\)](#), [GINAC_ASSERT](#), [GiNaC::expairseq::is_canonical\(\)](#), and [GiNaC::expairseq::overall_coeff](#).

6.95.2.7 mul() [7/7]

```
GiNaC::mul::mul (
    const ex & lh,
    const ex & mh,
    const ex & rh )
```

References [GiNaC::_ex1](#), [GiNaC::expairseq::construct_from_exvector\(\)](#), [factors](#), [GINAC_ASSERT](#), [GiNaC::expairseq::is_canonical\(\)](#), and [GiNaC::expairseq::overall_coeff](#).

6.95.3 Member Function Documentation**6.95.3.1 precedence()**

```
unsigned GiNaC::mul::precedence ( ) const [inline], [override], [virtual]
```

Return relative operator precedence (for parenthezing output).

Reimplemented from [GiNaC::expairseq](#).

Referenced by [do_print\(\)](#), [do_print_csrc\(\)](#), [do_print_latex\(\)](#), and [print_overall_coeff\(\)](#).

6.95.3.2 info()

```
bool GiNaC::mul::info (
    unsigned int ) const [override], [virtual]
```

Information about the object.

See also

class [info_flags](#)

Reimplemented from [GiNaC::expairseq](#).

References [GiNaC::_num1_p](#), [GiNaC::info_flags::cinteger](#), [GiNaC::info_flags::cinteger_polynomial](#), [GiNaC::info_flags::crational](#), [GiNaC::info_flags::crational_polynomial](#), [GiNaC::info_flags::even](#), [GiNaC::factor\(\)](#), [GiNaC::basic::flags](#), [GiNaC::info_flags::indefinite](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::info_flags::integer_polynomial](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::status_flags::is_negative](#), [GiNaC::status_flags::is_positive](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::negint](#), [GiNaC::info_flags::nonnegative](#), [GiNaC::info_flags::nonnegint](#), [GiNaC::expairseq::overall_coeff](#), [GiNaC::info_flags::polynomial](#), [GiNaC::info_flags::posint](#), [GiNaC::info_flags::positive](#), [GiNaC::status_flags::purely_indefinite](#), [GiNaC::info_flags::rational](#), [GiNaC::info_flags::rational_function](#), [GiNaC::info_flags::rational_polynomial](#), [GiNaC::info_flags::real](#), [recombine_pair_to_ex\(\)](#), [GiNaC::expairseq::seq](#), and [GiNaC::basic::setflag\(\)](#).

Referenced by [expand\(\)](#).

6.95.3.3 is_polynomial()

```
bool GiNaC::mul::is_polynomial (
    const ex & var ) const [override], [virtual]
```

Check whether this is a polynomial in the given variables.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::info_flags::nonnegint](#), and [GiNaC::expairseq::seq](#).

6.95.3.4 degree()

```
int GiNaC::mul::degree (
    const ex & s ) const [override], [virtual]
```

Return degree of highest power in object s.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::degree\(\)](#), [recombine_pair_to_ex\(\)](#), and [GiNaC::expairseq::seq](#).

6.95.3.5 ldegree()

```
int GiNaC::mul::ldegree (
    const ex & s ) const [override], [virtual]
```

Return degree of lowest power in object s.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::ldegree\(\)](#), [recombine_pair_to_ex\(\)](#), and [GiNaC::expairseq::seq](#).

6.95.3.6 coeff()

```
ex GiNaC::mul::coeff (
    const ex & s,
    int n = 1 ) const [override], [virtual]
```

Return coefficient of degree n in object s.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex0](#), [c](#), [GiNaC::ex::coeff\(\)](#), [n](#), [GiNaC::expairseq::overall_coeff](#), [recombine_pair_to_ex\(\)](#), and [GiNaC::expairseq::seq](#).

Referenced by [print_overall_coeff\(\)](#).

6.95.3.7 has()

```
bool GiNaC::mul::has (
    const ex & pattern,
    unsigned options = 0 ) const [override], [virtual]
```

Test for occurrence of a pattern.

An object 'has' a pattern if it matches the pattern itself or one of the children 'has' it. As a consequence (according to the definition of children) given $e=x+y+z$, $e.has(x)$ is true but $e.has(x+y)$ is false.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::has_options::algebraic](#), [GiNaC::algebraic_match_mul_with_mul\(\)](#), [GiNaC::basic::has\(\)](#), [GiNaC::expairseq::nops\(\)](#), and [options](#).

6.95.3.8 eval()

```
ex GiNaC::mul::eval ( ) const [override], [virtual]
```

Perform automatic term rewriting rules in this class.

In the following x , x_1 , $x_2, \dots$ stand for a symbolic variables of type `ex` and c , c_1 , $c_2, \dots$ stand for such expressions that contain a plain number.

- $*(\dots, x; 0) \rightarrow 0$
- $*(+(x_1, x_2, \dots); c) \rightarrow *(*(x_1, c), *(x_2, c), \dots)$
- $*(x; 1) \rightarrow x$
- $*(; c) \rightarrow c$

Reimplemented from [GiNaC::expairseq](#).

References [GiNaC::_ex0](#), [GiNaC::_ex1](#), [GiNaC::_num1_p](#), [GiNaC::_num_1_p](#), [c](#), [GiNaC::basic::clearflag\(\)](#), [GiNaC::ex::coeff\(\)](#), [GiNaC::add::combine_pair_with_coeff_to_pair\(\)](#), [GiNaC::numeric::div\(\)](#), [GiNaC::expairseq::evalchildren\(\)](#), [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::expanded](#), [GiNaC::basic::flags](#), [GINAC_ASSERT](#), [GiNaC::status_flags::hash_c](#), [GiNaC::basic::hold\(\)](#), [GiNaC::numeric::integer_content\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::numeric::is_integer\(\)](#), [GiNaC::numeric::is_pos_integer\(\)](#), [GiNaC::ex::is_zero\(\)](#), [last](#), [likely](#), [GiNaC::numeric::mul\(\)](#), [GiNaC::expairseq::overall_coeff](#), [recombine_pair_to_ex\(\)](#), [GiNaC::expairseq::seq](#), and [unlikely](#).

Referenced by [do_print_latex\(\)](#).

6.95.3.9 evalf()

```
ex GiNaC::mul::evalf ( ) const [override], [virtual]
```

Evaluate object numerically.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::evalf\(\)](#), [GiNaC::expairseq::overall_coeff](#), and [GiNaC::expairseq::seq](#).

6.95.3.10 real_part()

```
ex GiNaC::mul::real_part ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [find_real_imag\(\)](#).

6.95.3.11 imag_part()

```
ex GiNaC::mul::imag_part ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [find_real_imag\(\)](#).

6.95.3.12 evalm()

```
ex GiNaC::mul::evalm ( ) const [override], [virtual]
```

Evaluate sums, products and integer powers of matrices.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex1](#), [GiNaC::ex::end\(\)](#), [GiNaC::ex::evalm\(\)](#), [m](#), [GiNaC::expairseq::overall_coeff](#), [recombine_pair_to_ex\(\)](#), [GiNaC::expairseq::seq](#), and [split_ex_to_pair\(\)](#).

6.95.3.13 series()

```
ex GiNaC::mul::series (
    const relational & r,
    int order,
    unsigned options = 0 ) const [override], [virtual]
```

Implementation of [ex::series\(\)](#) for product.

This performs series multiplication when multiplying series.

See also

[ex::series](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex1](#), [GiNaC::ex::coeff\(\)](#), [GiNaC::ex::expand\(\)](#), [GiNaC::factor\(\)](#), [GINAC_ASSERT](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::ex::lddegree\(\)](#), [GiNaC::ex::lhs\(\)](#), [GiNaC::pseries::mul_const\(\)](#), [GiNaC::pseries::mul_series\(\)](#), [GiNaC::expairseq::op\(\)](#), [options](#), [order](#), [GiNaC::expairseq::overall_coeff](#), [r](#), [recombine_pair_to_ex\(\)](#), [GiNaC::ex::rhs\(\)](#), [GiNaC::expairseq::seq](#), and [GiNaC::ex::series\(\)](#).

6.95.3.14 normal()

```
ex GiNaC::mul::normal (
    exmap & repl,
    exmap & rev_lookup,
    lst & modifier ) const [override], [virtual]
```

Implementation of [ex::normal\(\)](#) for a product.

It cancels common factors from fractions.

See also

[ex::normal\(\)](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::frac_cancel\(\)](#), [n](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::container< C >::nops\(\)](#), [GiNaC::container< C >::op\(\)](#), [GiNaC::expairseq::overall_coeff](#), [recombine_pair_to_ex\(\)](#), and [GiNaC::expairseq::seq](#).

6.95.3.15 integer_content()

```
numeric GiNaC::mul::integer_content ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::abs\(\)](#), [GINAC_ASSERT](#), [GiNaC::expairseq::overall_coeff](#), [recombine_pair_to_ex\(\)](#), and [GiNaC::expairseq::seq](#).

6.95.3.16 smod()

```
ex GiNaC::mul::smod (
    const numeric & xi ) const [override], [virtual]
```

Apply symmetric modular homomorphism to an expanded multivariate polynomial.

This function is usually used internally by [heur_gcd\(\)](#).

Parameters

xi	modulus

Returns

mapped polynomial

See also

[heur_gcd](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::basic::clearflag\(\)](#), [GiNaC::status_flags::evaluated](#), [GINAC_ASSERT](#), [GiNaC::status_flags::hash_calculated](#), [GiNaC::expairseq::overall_coeff](#), [recombine_pair_to_ex\(\)](#), [GiNaC::expairseq::seq](#), and [GiNaC::smod\(\)](#).

6.95.3.17 max_coefficient()

```
numeric GiNaC::mul::max_coefficient ( ) const [override], [virtual]
```

Implementation [ex::max_coefficient\(\)](#).

See also

[heur_gcd](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::abs\(\)](#), [GINAC_ASSERT](#), [GiNaC::expairseq::overall_coeff](#), [recombine_pair_to_ex\(\)](#), and [GiNaC::expairseq::seq](#).

6.95.3.18 get_free_indices()

```
exvector GiNaC::mul::get_free_indices ( ) const [override], [virtual]
```

Return a vector containing the free indices of an expression.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::find_free_and_dummy\(\)](#), [GiNaC::ex::get_free_indices\(\)](#), [GiNaC::expairseq::nops\(\)](#), and [GiNaC::expairseq::op\(\)](#).

6.95.3.19 conjugate()

```
ex GiNaC::mul::conjugate ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::expairseq](#).

References [GiNaC::are_ex_trivially_equal\(\)](#), [c](#), [GiNaC::ex::conjugate\(\)](#), [GiNaC::expairseq::overall_coeff](#), [recombine_pair_to_ex\(\)](#), [GiNaC::expairseq::seq](#), [split_ex_to_pair\(\)](#), [thisexpairseq\(\)](#), and [x](#).

6.95.3.20 derivative()

```
ex GiNaC::mul::derivative (
    const symbol & s ) const [override], [protected], [virtual]
```

Implementation of [ex::diff\(\)](#) for a product.

It applies the product rule.

See also

[ex::diff](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex1](#), [GiNaC::ex::begin\(\)](#), [GiNaC::ex::coeff\(\)](#), [GiNaC::basic::diff\(\)](#), [GiNaC::expairseq::overall_coeff](#), [GiNaC::pow\(\)](#), [GiNaC::expairseq::seq](#), [split_ex_to_pair\(\)](#), and [GiNaC::expair::swap\(\)](#).

6.95.3.21 eval_ncmul()

```
ex GiNaC::mul::eval_ncmul (
    const exvector & v ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::return_types::noncommutative](#), and [GiNaC::expairseq::seq](#).

6.95.3.22 return_type()

```
unsigned GiNaC::mul::return_type ( ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::expairseq](#).

References [GiNaC::return_types::commutative](#), [GiNaC::return_types::noncommutative](#), [GiNaC::return_types::noncommutative_comp](#) and [GiNaC::expairseq::seq](#).

6.95.3.23 return_type_tinfo()

```
return\_type\_t GiNaC::mul::return_type_tinfo ( ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::return_types::noncommutative](#), and [GiNaC::expairseq::seq](#).

6.95.3.24 thisexpairseq() [1/2]

```
ex GiNaC::mul::thisexpairseq (
    const epvector & v,
    const ex & oc,
    bool do_index_renaming = false ) const [override], [protected], [virtual]
```

Create an object of this type.

This method works similar to a constructor. It is useful because expairseq has (at least) two possible different semantics but we want to inherit methods thus avoiding code duplication. Sometimes a method in expairseq has to create a new one of the same semantics, which cannot be done by a ctor because the name (add, mul,...) is unknown on the expairseq level. In order for this trick to work a derived class must of course override this definition.

Reimplemented from [GiNaC::expairseq](#).

Referenced by [conjugate\(\)](#).

6.95.3.25 thisexpairseq() [2/2]

```
ex GiNaC::mul::thisexpairseq (
    epvector && vp,
    const ex & oc,
    bool do_index_renaming = false ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::expairseq](#).

6.95.3.26 split_ex_to_pair()

```
expair GiNaC::mul::split_ex_to_pair (
    const ex & e ) const [override], [protected], [virtual]
```

Form an expair from an ex, using the corresponding semantics.

See also

[expairseq::recombine_pair_to_ex\(\)](#)

Reimplemented from [GiNaC::expairseq](#).

References [GiNaC::_ex1](#), [GiNaC::power::basis](#), and [GiNaC::power::exponent](#).

Referenced by [combine_ex_with_coeff_to_pair\(\)](#), [combine_pair_with_coeff_to_pair\(\)](#), [conjugate\(\)](#), [derivative\(\)](#), [evalm\(\)](#), [expair_needs_further_processing\(\)](#), [expand\(\)](#), and [expandchildren\(\)](#).

6.95.3.27 combine_ex_with_coeff_to_pair()

```
expair GiNaC::mul::combine_ex_with_coeff_to_pair (
    const ex & e,
    const ex & c ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::expairseq](#).

References [GiNaC::_ex1](#), [c](#), [GINAC_ASSERT](#), [GiNaC::pow\(\)](#), and [split_ex_to_pair\(\)](#).

6.95.3.28 combine_pair_with_coeff_to_pair()

```
expair GiNaC::mul::combine_pair_with_coeff_to_pair (
    const expair & p,
    const ex & c ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::expairseq](#).

References [GiNaC::_ex1](#), [c](#), [GiNaC::expair::coeff](#), [GINAC_ASSERT](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::pow\(\)](#), [recombine_pair_to_ex\(\)](#), [GiNaC::expair::rest](#), and [split_ex_to_pair\(\)](#).

6.95.3.29 recombine_pair_to_ex()

```
ex GiNaC::mul::recombine_pair_to_ex (
    const expair & p ) const [override], [protected], [virtual]
```

Form an ex out of an expair, using the corresponding semantics.

See also

[expairseq::split_ex_to_pair\(\)](#)

Reimplemented from [GiNaC::expairseq](#).

References [GiNaC::_ex1](#), [GiNaC::expair::coeff](#), [GiNaC::ex::is_equal\(\)](#), and [GiNaC::expair::rest](#).

Referenced by [coeff\(\)](#), [combine_pair_with_coeff_to_pair\(\)](#), [conjugate\(\)](#), [degree\(\)](#), [do_print\(\)](#), [do_print_latex\(\)](#), [eval\(\)](#), [evalm\(\)](#), [expair_needs_further_processing\(\)](#), [expandchildren\(\)](#), [find_real_imag\(\)](#), [info\(\)](#), [integer_content\(\)](#), [ldegree\(\)](#), [max_coefficient\(\)](#), [normal\(\)](#), [series\(\)](#), and [smod\(\)](#).

6.95.3.30 expair_needs_further_processing()

```
bool GiNaC::mul::expair_needs_further_processing (
    expair it ) [override], [protected], [virtual]
```

Reimplemented from [GiNaC::expairseq](#).

References [GiNaC::_ex1](#), [GiNaC::expair::is_equal\(\)](#), [recombine_pair_to_ex\(\)](#), and [split_ex_to_pair\(\)](#).

6.95.3.31 default_overall_coeff()

```
ex GiNaC::mul::default_overall_coeff ( ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::expairseq](#).

References [GiNaC::_ex1](#).

6.95.3.32 combine_overall_coeff() [1/2]

```
void GiNaC::mul::combine_overall_coeff (
    const ex & c ) [override], [protected], [virtual]
```

Reimplemented from [GiNaC::expairseq](#).

References [c](#), [GINAC_ASSERT](#), and [GiNaC::expairseq::overall_coeff](#).

6.95.3.33 combine_overall_coeff() [2/2]

```
void GiNaC::mul::combine_overall_coeff (
    const ex & c1,
    const ex & c2 ) [override], [protected], [virtual]
```

Reimplemented from [GiNaC::expairseq](#).

References [GINAC_ASSERT](#), [GiNaC::expairseq::overall_coeff](#), and [power](#).

6.95.3.34 can_make_flat()

```
bool GiNaC::mul::can_make_flat (
    const expair & p ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::expairseq](#).

References [GiNaC::expair::coeff](#), [GINAC_ASSERT](#), [GiNaC::ex::info\(\)](#), and [GiNaC::info_flags::integer](#).

6.95.3.35 `expand()`

```
ex GiNaC::mul::expand (
    unsigned options = 0 ) const [override], [protected], [virtual]
```

Expand expression, i.e.

multiply it out and return the result as a new expression.

Reimplemented from [GiNaC::expairseq](#).

References [GiNaC::_ex1](#), [GiNaC::_num0_p](#), [can_be_further_expanded\(\)](#), [GiNaC::ex::expand\(\)](#), [GiNaC::expand_options::expand_ren](#), [expandchildren\(\)](#), [GiNaC::status_flags::expanded](#), [factors](#), [GiNaC::get_all_dummy_indices_safely\(\)](#), [GiNaC::info_flags::has_indices](#), [info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::numeric::mul\(\)](#), [n](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), [GiNaC::container< C >::op\(\)](#), [options](#), [GiNaC::expairseq::overall_coeff](#), [GiNaC::rename_dummy_indices_uniquely\(\)](#), [GiNaC::expairseq::seq](#), [GiNaC::basic::setflag\(\)](#), and [split_ex_to_pair\(\)](#).

6.95.3.36 `algebraic_subs_mul()`

```
ex GiNaC::mul::algebraic_subs_mul (
    const exmap & m,
    unsigned options ) const
```

References [GiNaC::subs_options::algebraic](#), [GiNaC::algebraic_match_mul_with_mul\(\)](#), [m](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::expairseq::nops\(\)](#), [GiNaC::expairseq::op\(\)](#), [options](#), [GiNaC::pow\(\)](#), [GiNaC::ex::subs\(\)](#), [GiNaC::basic::subs_one_level\(\)](#), and [GiNaC::tryfactsubs\(\)](#).

6.95.3.37 `find_real_imag()`

```
void GiNaC::mul::find_real_imag (
    ex & rp,
    ex & ip ) const [protected]
```

References [GiNaC::ex::expand\(\)](#), [GiNaC::factor\(\)](#), [GiNaC::ex::imag_part\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::expairseq::overall_coeff](#), [GiNaC::ex::real_part\(\)](#), [recombine_pair_to_ex\(\)](#), and [GiNaC::expairseq::seq](#).

Referenced by [imag_part\(\)](#), and [real_part\(\)](#).

6.95.3.38 `print_overall_coeff()`

```
void GiNaC::mul::print_overall_coeff (
    const print_context & c,
    const char * mul_sym ) const [protected]
```

References [GiNaC::_num1_p](#), [GiNaC::_num_1_p](#), [c](#), [coeff\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::expairseq::overall_coeff](#), [precedence\(\)](#), [GiNaC::basic::print\(\)](#), and [GiNaC::ex::print\(\)](#).

Referenced by [do_print\(\)](#), and [do_print_latex\(\)](#).

6.95.3.39 do_print()

```
void GiNaC::mul::do_print (
    const print\_context & c,
    unsigned level ) const [protected]
```

References [c](#), [precedence\(\)](#), [GiNaC::ex::print\(\)](#), [print_overall_coeff\(\)](#), [recombine_pair_to_ex\(\)](#), and [GiNaC::expairseq::seq](#).

6.95.3.40 do_print_latex()

```
void GiNaC::mul::do_print_latex (
    const print\_latex & c,
    unsigned level ) const [protected]
```

References [c](#), [eval\(\)](#), [GINAC_ASSERT](#), [precedence\(\)](#), [GiNaC::ex::print\(\)](#), [print_overall_coeff\(\)](#), [recombine_pair_to_ex\(\)](#), and [GiNaC::expairseq::seq](#).

6.95.3.41 do_print_csrc()

```
void GiNaC::mul::do_print_csrc (
    const print\_csrc & c,
    unsigned level ) const [protected]
```

References [GiNaC::_ex1](#), [GiNaC::_ex_1](#), [c](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::info_flags::negint](#), [GiNaC::expairseq::overall_coeff](#), [power](#), [precedence\(\)](#), [GiNaC::ex::print\(\)](#), and [GiNaC::expairseq::seq](#).

6.95.3.42 do_print_python_repr()

```
void GiNaC::mul::do_print_python_repr (
    const print\_python\_repr & c,
    unsigned level ) const [protected]
```

References [c](#), [GiNaC::expairseq::nops\(\)](#), [GiNaC::expairseq::op\(\)](#), and [GiNaC::ex::print\(\)](#).

6.95.3.43 can_be_further_expanded()

```
bool GiNaC::mul::can_be_further_expanded (
    const ex & e ) [static], [protected]
```

References [GiNaC::ex::info\(\)](#), [GiNaC::ex::op\(\)](#), and [GiNaC::info_flags::posint](#).

Referenced by [expand\(\)](#).

6.95.3.44 `expandchildren()`

```
epvector GiNaC::mul::expandchildren (
    unsigned options ) const [protected]
```

Member-wise expand the expairs representing this sequence.

This must be overridden from `expairseq::expandchildren()` and done iteratively in order to allow for early cancellations and thus save memory.

See also

[mul::expand\(\)](#)

Returns

epvector containing expanded pairs, empty if no members had to be changed.

References [GiNaC::are_ex_trivially_equal\(\)](#), [GiNaC::ex::begin\(\)](#), [GiNaC::ex::expand\(\)](#), [GiNaC::factor\(\)](#), [last](#), [options](#), [recombine_pair_to_ex\(\)](#), [GiNaC::expairseq::seq](#), and [split_ex_to_pair\(\)](#).

Referenced by [expand\(\)](#).

6.95.4 Friends And Related Symbol Documentation

6.95.4.1 `add`

```
friend class add [friend]
```

6.95.4.2 `ncmul`

```
friend class ncmul [friend]
```

6.95.4.3 `power`

```
friend class power [friend]
```

Referenced by [combine_overall_coeff\(\)](#), and [do_print_csrc\(\)](#).

The documentation for this class was generated from the following files:

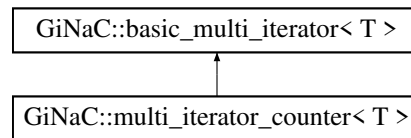
- [mul.h](#)
- [indexed.cpp](#)
- [mul.cpp](#)
- [normal.cpp](#)
- [pseries.cpp](#)

6.96 GiNaC::multi_iterator_counter< T > Class Template Reference

The class `multi_iterator_counter` defines a multi_iterator $(i_1, i_2, \dots, i_k)$, such that.

```
#include <utils_multi_iterator.h>
```

Inheritance diagram for GiNaC::multi_iterator_counter< T >:



Public Member Functions

- `multi_iterator_counter` (void)
Default constructor.
- `multi_iterator_counter` (T B, T N, size_t k)
Construct a multi_iterator with upper limit N and size k .
- `multi_iterator_counter` (T B, T N, const std::vector< T > &vv)
Construct from a vector.
- `basic_multi_iterator< T > & init` (void)
Initialize the multi-index to.
- `basic_multi_iterator< T > & operator++` (int)
The postfix increment operator allows to write for a multi-index n++, which will update n to the next configuration.

Public Member Functions inherited from `GiNaC::basic_multi_iterator< T >`

- `basic_multi_iterator` (void)
Default constructor.
- `basic_multi_iterator` (T B, T N, size_t k)
Construct a multi_iterator with upper limit N, lower limit B and size k .
- `basic_multi_iterator` (T B, T N, const std::vector< T > &vv)
Construct from a vector.
- virtual `~basic_multi_iterator` ()
Destructor.
- size_t `size` (void) const
Returns the size of a multi_iterator.
- bool `overflow` (void) const
Return the overflow flag.
- const std::vector< T > & `get_vector` (void) const
Returns a reference to the vector v.
- T `operator[]` (size_t i) const
Subscription via [].
- T & `operator[]` (size_t i)
Subscription via [].
- T `operator()` (size_t i) const
Subscription via ().
- T & `operator()` (size_t i)
Subscription via ().

Friends

- `template<class TT >`
`std::ostream & operator<< (std::ostream &os, const multi_iterator_counter< TT > &v)`

Additional Inherited Members

Protected Attributes inherited from `GiNaC::basic_multi_iterator< T >`

- `T N`
- `T B`
- `std::vector< T > v`
- `bool flag_overflow`

6.96.1 Detailed Description

```
template<class T>
class GiNaC::multi_iterator_counter< T >
```

The class `multi_iterator_counter` defines a `multi_iterator` $(i_1, i_2, \dots, i_k)$, such that.

$$B \leq i_j < N$$

6.96.2 Constructor & Destructor Documentation

6.96.2.1 `multi_iterator_counter()` [1/3]

```
template<class T >
GiNaC::multi_iterator_counter< T >::multi_iterator_counter (
    void ) [inline]
```

Default constructor.

6.96.2.2 `multi_iterator_counter()` [2/3]

```
template<class T >
GiNaC::multi_iterator_counter< T >::multi_iterator_counter (
    T B,
    T N,
    size_t k ) [inline], [explicit]
```

Construct a `multi_iterator` with upper limit N and size k .

6.96.2.3 multi_iterator_counter() [3/3]

```
template<class T >
GiNaC::multi_iterator_counter< T >::multi_iterator_counter (
    T B,
    T N,
    const std::vector< T > & vv ) [inline], [explicit]
```

Construct from a vector.

6.96.3 Member Function Documentation**6.96.3.1 init()**

```
template<class T >
basic_multi_iterator< T > & GiNaC::multi_iterator_counter< T >::init (
    void ) [inline], [virtual]
```

Initialize the multi-index to.

$$(n_1, n_2, n_3, \dots, n_k) = (B, B, \dots, B)$$

Reimplemented from [GiNaC::basic_multi_iterator< T >](#).

6.96.3.2 operator++()

```
template<class T >
basic_multi_iterator< T > & GiNaC::multi_iterator_counter< T >::operator++ (
    int ) [inline], [virtual]
```

The postfix increment operator allows to write for a multi-index $n++$, which will update n to the next configuration.

If n is in the last configuration and the increment operator $++$ is applied to n , the overflow flag will be raised.

Reimplemented from [GiNaC::basic_multi_iterator< T >](#).

References [k](#).

6.96.4 Friends And Related Symbol Documentation**6.96.4.1 operator<<**

```
template<class T >
template<class TT >
std::ostream & operator<< (
    std::ostream & os,
    const multi_iterator_counter< TT > & v ) [friend]
```

The documentation for this class was generated from the following file:

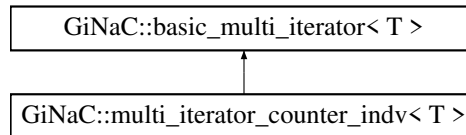
- [utils_multi_iterator.h](#)

6.97 GiNaC::multi_iterator_counter_indv< T > Class Template Reference

The class `multi_iterator_counter_indv` defines a `multi_iterator` ($i_1, i_2, \dots, i_k$), such that.

```
#include <utils_multi_iterator.h>
```

Inheritance diagram for `GiNaC::multi_iterator_counter_indv< T >`:



Public Member Functions

- `multi_iterator_counter_indv` (void)
Default constructor.
- `multi_iterator_counter_indv` (T B, const std::vector< T > &Nv, size_t k)
Construct a multi_iterator with upper limit N and size k .
- `multi_iterator_counter_indv` (T B, const std::vector< T > &Nv, const std::vector< T > &vv)
Construct from a vector.
- `basic_multi_iterator< T > &init` (void)
Initialize the multi-index to.
- `basic_multi_iterator< T > &operator++` (int)
The postfix increment operator allows to write for a multi-index n++, which will update n to the next configuration.

Public Member Functions inherited from `GiNaC::basic_multi_iterator< T >`

- `basic_multi_iterator` (void)
Default constructor.
- `basic_multi_iterator` (T B, T N, size_t k)
Construct a multi_iterator with upper limit N, lower limit B and size k .
- `basic_multi_iterator` (T B, T N, const std::vector< T > &vv)
Construct from a vector.
- virtual `~basic_multi_iterator` ()
Destructor.
- size_t `size` (void) const
Returns the size of a multi_iterator.
- bool `overflow` (void) const
Return the overflow flag.
- const std::vector< T > &`get_vector` (void) const
Returns a reference to the vector v.
- T `operator[]` (size_t i) const
Subscription via [].
- T &`operator[]` (size_t i)
Subscription via [].
- T `operator()` (size_t i) const
Subscription via ().
- T &`operator()` (size_t i)
Subscription via ().

Protected Attributes

- `std::vector< T > Nv`

Protected Attributes inherited from [GiNaC::basic_multi_iterator< T >](#)

- `T N`
- `T B`
- `std::vector< T > v`
- `bool flag_overflow`

Friends

- `template<class TT >`
`std::ostream & operator<< (std::ostream &os, const multi\_iterator\_counter\_indv< TT > &v)`

6.97.1 Detailed Description

`template<class T>`
class [GiNaC::multi_iterator_counter_indv< T >](#)

The class [multi_iterator_counter_indv](#) defines a multi_iterator $(i_1, i_2, \dots, i_k)$, such that.

$$B \leq i_j < N_j$$

6.97.2 Constructor & Destructor Documentation

6.97.2.1 [multi_iterator_counter_indv\(\)](#) [1/3]

```
template<class T >
GiNaC::multi\_iterator\_counter\_indv< T >::multi\_iterator\_counter\_indv (
    void ) [inline]
```

Default constructor.

6.97.2.2 [multi_iterator_counter_indv\(\)](#) [2/3]

```
template<class T >
GiNaC::multi\_iterator\_counter\_indv< T >::multi\_iterator\_counter\_indv (
    T B,
    const std::vector< T > & Nv,
    size_t k ) [inline], [explicit]
```

Construct a multi_iterator with upper limit N and size k .

6.97.2.3 multi_iterator_counter_indv() [3/3]

```
template<class T >
GiNaC::multi_iterator_counter_indv< T >::multi_iterator_counter_indv (
    T B,
    const std::vector< T > & Nv,
    const std::vector< T > & vv ) [inline], [explicit]
```

Construct from a vector.

6.97.3 Member Function Documentation

6.97.3.1 init()

```
template<class T >
basic_multi_iterator< T > & GiNaC::multi_iterator_counter_indv< T >::init (
    void ) [inline], [virtual]
```

Initialize the multi-index to.

$$(n_1, n_2, n_3, \dots, n_k) = (B, B, \dots, B)$$

Reimplemented from [GiNaC::basic_multi_iterator< T >](#).

6.97.3.2 operator++()

```
template<class T >
basic_multi_iterator< T > & GiNaC::multi_iterator_counter_indv< T >::operator++ (
    int ) [inline], [virtual]
```

The postfix increment operator allows to write for a multi-index $n++$, which will update n to the next configuration.

If n is in the last configuration and the increment operator $++$ is applied to n , the overflow flag will be raised.

Reimplemented from [GiNaC::basic_multi_iterator< T >](#).

References [k](#).

6.97.4 Friends And Related Symbol Documentation

6.97.4.1 operator<<

```
template<class T >
template<class TT >
std::ostream & operator<< (
    std::ostream & os,
    const multi_iterator_counter_indv< TT > & v ) [friend]
```

6.97.5 Member Data Documentation

6.97.5.1 Nv

```
template<class T >
std::vector<T> GiNaC::multi_iterator_counter_indv< T >::Nv [protected]
```

The documentation for this class was generated from the following file:

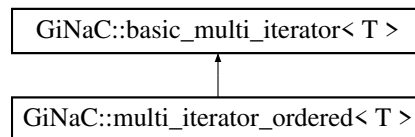
- [utils_multi_iterator.h](#)

6.98 GiNaC::multi_iterator_ordered< T > Class Template Reference

The class [multi_iterator_ordered](#) defines a multi_iterator $(i_1, i_2, \dots, i_k)$, such that.

```
#include <utils_multi_iterator.h>
```

Inheritance diagram for GiNaC::multi_iterator_ordered< T >:



Public Member Functions

- [multi_iterator_ordered](#) (void)
Default constructor.
- [multi_iterator_ordered](#) (T B, T N, size_t k)
Construct a multi_iterator with upper limit N and size k .
- [multi_iterator_ordered](#) (T B, T N, const std::vector< T > &vv)
Construct from a vector.
- [basic_multi_iterator](#)< T > & [init](#) (void)
Initialize the multi-index to.
- [basic_multi_iterator](#)< T > & [operator++](#) (int)
The postfix increment operator allows to write for a multi-index n++, which will update n to the next configuration.

Public Member Functions inherited from [GiNaC::basic_multi_iterator< T >](#)

- [basic_multi_iterator](#) (void)
Default constructor.
- [basic_multi_iterator](#) (T [B](#), T [N](#), size_t [k](#))
Construct a multi_iterator with upper limit N, lower limit B and size k .
- [basic_multi_iterator](#) (T [B](#), T [N](#), const std::vector< T > &vv)
Construct from a vector.
- virtual [~basic_multi_iterator](#) ()
Destructor.
- size_t [size](#) (void) const
Returns the size of a multi_iterator.
- bool [overflow](#) (void) const
Return the overflow flag.
- const std::vector< T > & [get_vector](#) (void) const
Returns a reference to the vector v.
- T [operator\[\]](#) (size_t i) const
Subscription via [].
- T & [operator\[\]](#) (size_t i)
Subscription via [].
- T [operator\(\)](#) (size_t i) const
Subscription via ().
- T & [operator\(\)](#) (size_t i)
Subscription via ().

Friends

- template<class TT >
std::ostream & [operator<<](#) (std::ostream &os, const [multi_iterator_ordered](#)< TT > &v)

Additional Inherited Members

Protected Attributes inherited from [GiNaC::basic_multi_iterator< T >](#)

- T [N](#)
- T [B](#)
- std::vector< T > [v](#)
- bool [flag_overflow](#)

6.98.1 Detailed Description

template<class T>
class [GiNaC::multi_iterator_ordered](#)< T >

The class [multi_iterator_ordered](#) defines a multi_iterator $(i_1, i_2, \dots, i_k)$, such that.

$$B \leq i_j < N$$

and

$$i_j < i_{j+1}.$$

It is assumed that $k > 0$ and $N - B \geq k$.

6.98.2 Constructor & Destructor Documentation

6.98.2.1 multi_iterator_ordered() [1/3]

```
template<class T >
GiNaC::multi_iterator_ordered< T >::multi_iterator_ordered (
    void ) [inline]
```

Default constructor.

6.98.2.2 multi_iterator_ordered() [2/3]

```
template<class T >
GiNaC::multi_iterator_ordered< T >::multi_iterator_ordered (
    T B,
    T N,
    size_t k ) [inline], [explicit]
```

Construct a multi_iterator with upper limit N and size k .

6.98.2.3 multi_iterator_ordered() [3/3]

```
template<class T >
GiNaC::multi_iterator_ordered< T >::multi_iterator_ordered (
    T B,
    T N,
    const std::vector< T > & vv ) [inline], [explicit]
```

Construct from a vector.

6.98.3 Member Function Documentation

6.98.3.1 init()

```
template<class T >
basic_multi_iterator< T > & GiNaC::multi_iterator_ordered< T >::init (
    void ) [inline], [virtual]
```

Initialize the multi-index to.

$$(n_1, n_2, n_3, \dots, n_k) = (B + 0, B + 1, B + 2, \dots, B + k - 1)$$

Reimplemented from [GiNaC::basic_multi_iterator< T >](#).

6.98.3.2 operator++()

```
template<class T >
basic_multi_iterator< T > & GiNaC::multi_iterator_ordered< T >::operator++ (
    int ) [inline], [virtual]
```

The postfix increment operator allows to write for a multi-index $n++$, which will update n to the next configuration.

If n is in the last configuration and the increment operator $++$ is applied to n , the overflow flag will be raised.

Reimplemented from [GiNaC::basic_multi_iterator< T >](#).

References [k](#).

6.98.4 Friends And Related Symbol Documentation

6.98.4.1 operator<<

```
template<class T >
template<class TT >
std::ostream & operator<< (
    std::ostream & os,
    const multi_iterator_ordered< TT > & v ) [friend]
```

The documentation for this class was generated from the following file:

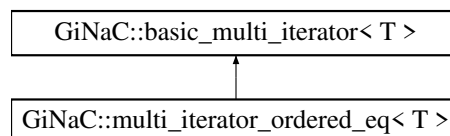
- [utils_multi_iterator.h](#)

6.99 GiNaC::multi_iterator_ordered_eq< T > Class Template Reference

The class [multi_iterator_ordered_eq](#) defines a multi_iterator $(i_1, i_2, \dots, i_k)$, such that.

```
#include <utils_multi_iterator.h>
```

Inheritance diagram for [GiNaC::multi_iterator_ordered_eq< T >](#):



Public Member Functions

- [multi_iterator_ordered_eq](#) (void)
Default constructor.
- [multi_iterator_ordered_eq](#) (T B, T N, size_t k)
Construct a multi_iterator with upper limit N and size k.
- [multi_iterator_ordered_eq](#) (T B, T N, const std::vector< T > &vv)
Construct from a vector.
- [basic_multi_iterator< T > & init](#) (void)
Initialize the multi-index to.
- [basic_multi_iterator< T > & operator++](#) (int)
The postfix increment operator allows to write for a multi-index $n++$, which will update n to the next configuration.

Public Member Functions inherited from [GiNaC::basic_multi_iterator< T >](#)

- [basic_multi_iterator](#) (void)
Default constructor.
- [basic_multi_iterator](#) (T B, T N, size_t k)
Construct a multi_iterator with upper limit N, lower limit B and size k .
- [basic_multi_iterator](#) (T B, T N, const std::vector< T > &vv)
Construct from a vector.
- virtual [~basic_multi_iterator](#) ()
Destructor.
- size_t [size](#) (void) const
Returns the size of a multi_iterator.
- bool [overflow](#) (void) const
Return the overflow flag.
- const std::vector< T > & [get_vector](#) (void) const
Returns a reference to the vector v.
- T [operator\[\]](#) (size_t i) const
Subscription via [].
- T & [operator\[\]](#) (size_t i)
Subscription via [].
- T [operator\(\)](#) (size_t i) const
Subscription via ().
- T & [operator\(\)](#) (size_t i)
Subscription via ().

Friends

- template<class TT >
std::ostream & [operator<<](#) (std::ostream &os, const [multi_iterator_ordered_eq](#)< TT > &v)

Additional Inherited Members**Protected Attributes inherited from [GiNaC::basic_multi_iterator< T >](#)**

- T N
- T B
- std::vector< T > v
- bool [flag_overflow](#)

6.99.1 Detailed Description

```
template<class T>
class GiNaC::multi_iterator_ordered_eq< T >
```

The class [multi_iterator_ordered_eq](#) defines a multi_iterator $(i_1, i_2, \dots, i_k)$, such that.

$$B \leq i_j < N$$

and

$$i_j \leq i_{j+1}.$$

It is assumed that $k > 0$.

6.99.2 Constructor & Destructor Documentation

6.99.2.1 multi_iterator_ordered_eq() [1/3]

```
template<class T >
GiNaC::multi_iterator_ordered_eq< T >::multi_iterator_ordered_eq (
    void ) [inline]
```

Default constructor.

6.99.2.2 multi_iterator_ordered_eq() [2/3]

```
template<class T >
GiNaC::multi_iterator_ordered_eq< T >::multi_iterator_ordered_eq (
    T B,
    T N,
    size_t k ) [inline], [explicit]
```

Construct a multi_iterator with upper limit N and size k .

6.99.2.3 multi_iterator_ordered_eq() [3/3]

```
template<class T >
GiNaC::multi_iterator_ordered_eq< T >::multi_iterator_ordered_eq (
    T B,
    T N,
    const std::vector< T > & vv ) [inline], [explicit]
```

Construct from a vector.

6.99.3 Member Function Documentation

6.99.3.1 init()

```
template<class T >
basic_multi_iterator< T > & GiNaC::multi_iterator_ordered_eq< T >::init (
    void ) [inline], [virtual]
```

Initialize the multi-index to.

$$(n_1, n_2, \dots, n_k) = (B, B, \dots, B)$$

Reimplemented from [GiNaC::basic_multi_iterator< T >](#).

6.99.3.2 operator++()

```
template<class T >
basic_multi_iterator< T > & GiNaC::multi_iterator_ordered_eq< T >::operator++ (
    int ) [inline], [virtual]
```

The postfix increment operator allows to write for a multi-index $n++$, which will update n to the next configuration.

If n is in the last configuration and the increment operator $++$ is applied to n , the overflow flag will be raised.

Reimplemented from [GiNaC::basic_multi_iterator< T >](#).

References [k](#).

6.99.4 Friends And Related Symbol Documentation

6.99.4.1 operator<<

```
template<class T >
template<class TT >
std::ostream & operator<< (
    std::ostream & os,
    const multi_iterator_ordered_eq< TT > & v ) [friend]
```

The documentation for this class was generated from the following file:

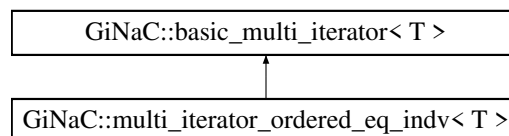
- [utils_multi_iterator.h](#)

6.100 GiNaC::multi_iterator_ordered_eq_indv< T > Class Template Reference

The class [multi_iterator_ordered_eq_indv](#) defines a multi_iterator $(i_1, i_2, \dots, i_k)$, such that.

```
#include <utils_multi_iterator.h>
```

Inheritance diagram for [GiNaC::multi_iterator_ordered_eq_indv< T >](#):



Public Member Functions

- [multi_iterator_ordered_eq_indv](#) (void)
Default constructor.
- [multi_iterator_ordered_eq_indv](#) (T B, const std::vector< T > &Nv, size_t k)
Construct a multi_iterator with upper limit N and size k.
- [multi_iterator_ordered_eq_indv](#) (T B, const std::vector< T > &Nv, const std::vector< T > &vv)
Construct from a vector.
- [basic_multi_iterator< T > & init](#) (void)
Initialize the multi-index to.
- [basic_multi_iterator< T > & operator++](#) (int)
The postfix increment operator allows to write for a multi-index $n++$, which will update n to the next configuration.

Public Member Functions inherited from [GiNaC::basic_multi_iterator< T >](#)

- [basic_multi_iterator](#) (void)
Default constructor.
- [basic_multi_iterator](#) (T [B](#), T [N](#), size_t [k](#))
Construct a multi_iterator with upper limit N, lower limit B and size k.
- [basic_multi_iterator](#) (T [B](#), T [N](#), const std::vector< T > &vv)
Construct from a vector.
- virtual [~basic_multi_iterator](#) ()
Destructor.
- size_t [size](#) (void) const
Returns the size of a multi_iterator.
- bool [overflow](#) (void) const
Return the overflow flag.
- const std::vector< T > & [get_vector](#) (void) const
Returns a reference to the vector v.
- T [operator\[\]](#) (size_t i) const
Subscription via [].
- T & [operator\[\]](#) (size_t i)
Subscription via [].
- T [operator\(\)](#) (size_t i) const
Subscription via ().
- T & [operator\(\)](#) (size_t i)
Subscription via ().

Protected Attributes

- std::vector< T > [Nv](#)

Protected Attributes inherited from [GiNaC::basic_multi_iterator< T >](#)

- T [N](#)
- T [B](#)
- std::vector< T > [v](#)
- bool [flag_overflow](#)

Friends

- template<class TT >
std::ostream & [operator<<](#) (std::ostream &os, const [multi_iterator_ordered_eq_indv](#)< TT > &v)

6.100.1 Detailed Description

template<class T>
class [GiNaC::multi_iterator_ordered_eq_indv](#)< T >

The class [multi_iterator_ordered_eq_indv](#) defines a multi_iterator $(i_1, i_2, \dots, i_k)$, such that.

$$B \leq i_j < N_j$$

and

$$i_j \leq i_{j+1}.$$

6.100.2 Constructor & Destructor Documentation

6.100.2.1 multi_iterator_ordered_eq_indv() [1/3]

```
template<class T >
GiNaC::multi_iterator_ordered_eq_indv< T >::multi_iterator_ordered_eq_indv (
    void ) [inline]
```

Default constructor.

6.100.2.2 multi_iterator_ordered_eq_indv() [2/3]

```
template<class T >
GiNaC::multi_iterator_ordered_eq_indv< T >::multi_iterator_ordered_eq_indv (
    T B,
    const std::vector< T > & Nv,
    size_t k ) [inline], [explicit]
```

Construct a multi_iterator with upper limit N and size k .

6.100.2.3 multi_iterator_ordered_eq_indv() [3/3]

```
template<class T >
GiNaC::multi_iterator_ordered_eq_indv< T >::multi_iterator_ordered_eq_indv (
    T B,
    const std::vector< T > & Nv,
    const std::vector< T > & vv ) [inline], [explicit]
```

Construct from a vector.

6.100.3 Member Function Documentation

6.100.3.1 init()

```
template<class T >
basic_multi_iterator< T > & GiNaC::multi_iterator_ordered_eq_indv< T >::init (
    void ) [inline], [virtual]
```

Initialize the multi-index to.

$$(n_1, n_2, n_3, \dots, n_k) = (B, B, B, \dots, B)$$

Reimplemented from [GiNaC::basic_multi_iterator< T >](#).

6.100.3.2 operator++()

```
template<class T >
basic_multi_iterator< T > & GiNaC::multi_iterator_ordered_eq_indv< T >::operator++ (
    int ) [inline], [virtual]
```

The postfix increment operator allows to write for a multi-index $n++$, which will update n to the next configuration.

If n is in the last configuration and the increment operator $++$ is applied to n , the overflow flag will be raised.

Reimplemented from [GiNaC::basic_multi_iterator< T >](#).

References [k](#).

6.100.4 Friends And Related Symbol Documentation

6.100.4.1 operator<<

```
template<class T >
template<class TT >
std::ostream & operator<< (
    std::ostream & os,
    const multi_iterator_ordered_eq_indv< TT > & v ) [friend]
```

6.100.5 Member Data Documentation

6.100.5.1 Nv

```
template<class T >
std::vector<T> GiNaC::multi_iterator_ordered_eq_indv< T >::Nv [protected]
```

The documentation for this class was generated from the following file:

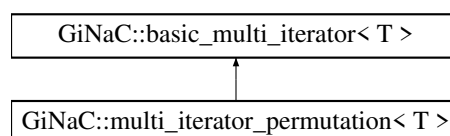
- [utils_multi_iterator.h](#)

6.101 GiNaC::multi_iterator_permutation< T > Class Template Reference

The class [multi_iterator_permutation](#) defines a multi_iterator $(i_1, i_2, \dots, i_k)$, for which.

```
#include <utils_multi_iterator.h>
```

Inheritance diagram for [GiNaC::multi_iterator_permutation< T >](#):



Public Member Functions

- [multi_iterator_permutation](#) (void)
Default constructor.
- [multi_iterator_permutation](#) (T B, T N, size_t k)
Construct a multi_iterator with upper limit N and size k .
- [multi_iterator_permutation](#) (T B, T N, const std::vector< T > &vv)
Construct from a vector.
- [basic_multi_iterator](#)< T > & [init](#) (void)
Initialize the multi-index to.
- [basic_multi_iterator](#)< T > & [operator++](#) (int)
The postfix increment operator allows to write for a multi-index n++, which will update n to the next configuration.
- int [get_sign](#) (void) const
Returns the sign of the permutation, defined by.

Public Member Functions inherited from [GiNaC::basic_multi_iterator](#)< T >

- [basic_multi_iterator](#) (void)
Default constructor.
- [basic_multi_iterator](#) (T B, T N, size_t k)
Construct a multi_iterator with upper limit N, lower limit B and size k .
- [basic_multi_iterator](#) (T B, T N, const std::vector< T > &vv)
Construct from a vector.
- virtual [~basic_multi_iterator](#) ()
Destructor.
- size_t [size](#) (void) const
Returns the size of a multi_iterator.
- bool [overflow](#) (void) const
Return the overflow flag.
- const std::vector< T > & [get_vector](#) (void) const
Returns a reference to the vector v.
- T [operator\[\]](#) (size_t i) const
Subscription via [].
- T & [operator\[\]](#) (size_t i)
Subscription via [].
- T [operator\(\)](#) (size_t i) const
Subscription via ().
- T & [operator\(\)](#) (size_t i)
Subscription via ().

Friends

- template<class TT >
std::ostream & [operator<<](#) (std::ostream &os, const [multi_iterator_permutation](#)< TT > &v)

Additional Inherited Members

Protected Attributes inherited from [GiNaC::basic_multi_iterator](#)< T >

- T N
- T B
- std::vector< T > v
- bool [flag_overflow](#)

6.101.1 Detailed Description

```
template<class T>
class GiNaC::multi_iterator_permutation< T >
```

The class [multi_iterator_permutation](#) defines a multi_iterator $(i_1, i_2, \dots, i_k)$, for which.

$$B \leq i_j < N$$

and

$$i_i \neq i_j$$

In particular, if $N - B = k$, [multi_iterator_permutation](#) loops over all permutations of k elements.

6.101.2 Constructor & Destructor Documentation

6.101.2.1 multi_iterator_permutation() [1/3]

```
template<class T >
GiNaC::multi_iterator_permutation< T >::multi_iterator_permutation (
    void ) [inline]
```

Default constructor.

6.101.2.2 multi_iterator_permutation() [2/3]

```
template<class T >
GiNaC::multi_iterator_permutation< T >::multi_iterator_permutation (
    T B,
    T N,
    size_t k ) [inline], [explicit]
```

Construct a multi_iterator with upper limit N and size k .

6.101.2.3 multi_iterator_permutation() [3/3]

```
template<class T >
GiNaC::multi_iterator_permutation< T >::multi_iterator_permutation (
    T B,
    T N,
    const std::vector< T > & vv ) [inline], [explicit]
```

Construct from a vector.

6.101.3 Member Function Documentation

6.101.3.1 init()

```
template<class T >
basic_multi_iterator< T > & GiNaC::multi_iterator_permutation< T >::init (
    void ) [inline], [virtual]
```

Initialize the multi-index to.

$$(n_1, n_2, n_3, \dots, n_k) = (B + 0, B + 1, B + 2, \dots, B + k - 1)$$

Reimplemented from [GiNaC::basic_multi_iterator< T >](#).

6.101.3.2 operator++()

```
template<class T >
basic_multi_iterator< T > & GiNaC::multi_iterator_permutation< T >::operator++ (
    int ) [inline], [virtual]
```

The postfix increment operator allows to write for a multi-index $n++$, which will update n to the next configuration.

If n is in the last configuration and the increment operator $++$ is applied to n , the overflow flag will be raised.

Reimplemented from [GiNaC::basic_multi_iterator< T >](#).

References [k](#).

6.101.3.3 get_sign()

```
template<class T >
int GiNaC::multi_iterator_permutation< T >::get_sign (
    void ) const [inline]
```

Returns the sign of the permutation, defined by.

$$(-1)^{n_{inv}},$$

where n_{inv} is the number of inversions, e.g. the number of pairs $i < j$ for which

$$n_i > n_j.$$

References [k](#).

6.101.4 Friends And Related Symbol Documentation

6.101.4.1 operator<<

```
template<class T >
template<class TT >
std::ostream & operator<< (
    std::ostream & os,
    const multi_iterator_permutation< TT > & v ) [friend]
```

The documentation for this class was generated from the following file:

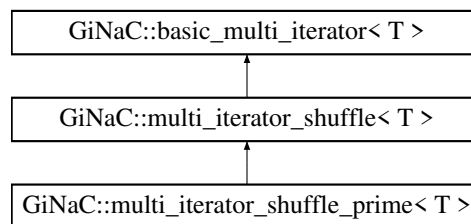
- [utils_multi_iterator.h](#)

6.102 GiNaC::multi_iterator_shuffle< T > Class Template Reference

The class [multi_iterator_shuffle](#) defines a multi_iterator, which runs over all shuffles of a and b.

```
#include <utils_multi_iterator.h>
```

Inheritance diagram for GiNaC::multi_iterator_shuffle< T >:



Public Member Functions

- [multi_iterator_shuffle](#) (void)
Default constructor.
- [multi_iterator_shuffle](#) (const std::vector< T > &a, const std::vector< T > &b)
Construct from a vector.
- [basic_multi_iterator](#)< T > & [init](#) (void)
Initialize the multi-index to the first shuffle.
- [basic_multi_iterator](#)< T > & [operator++](#) (int)
The postfix increment operator allows to write for a multi-index n++, which will update n to the next configuration.

Public Member Functions inherited from [GiNaC::basic_multi_iterator< T >](#)

- [basic_multi_iterator](#) (void)
Default constructor.
- [basic_multi_iterator](#) (T [B](#), T [N](#), size_t [k](#))
Construct a multi_iterator with upper limit N, lower limit B and size k .
- [basic_multi_iterator](#) (T [B](#), T [N](#), const std::vector< T > &vv)
Construct from a vector.
- virtual [~basic_multi_iterator](#) ()
Destructor.
- size_t [size](#) (void) const
Returns the size of a multi_iterator.
- bool [overflow](#) (void) const
Return the overflow flag.
- const std::vector< T > & [get_vector](#) (void) const
Returns a reference to the vector v.
- T [operator\[\]](#) (size_t i) const
Subscription via [].
- T & [operator\[\]](#) (size_t i)
Subscription via [].
- T [operator\(\)](#) (size_t i) const
Subscription via ().
- T & [operator\(\)](#) (size_t i)
Subscription via ().

Protected Attributes

- size_t [N_internal](#)
- std::vector< size_t > [v_internal](#)
- std::vector< T > [v_orig](#)

Protected Attributes inherited from [GiNaC::basic_multi_iterator< T >](#)

- T [N](#)
- T [B](#)
- std::vector< T > [v](#)
- bool [flag_overflow](#)

Friends

- template<class TT >
std::ostream & [operator<<](#) (std::ostream &os, const [multi_iterator_shuffle](#)< TT > &v)

6.102.1 Detailed Description

template<class T>
class [GiNaC::multi_iterator_shuffle](#)< T >

The class [multi_iterator_shuffle](#) defines a multi_iterator, which runs over all shuffles of a and b.

6.102.2 Constructor & Destructor Documentation

6.102.2.1 `multi_iterator_shuffle()` [1/2]

```
template<class T >
GiNaC::multi_iterator_shuffle< T >::multi_iterator_shuffle (
    void ) [inline]
```

Default constructor.

6.102.2.2 `multi_iterator_shuffle()` [2/2]

```
template<class T >
GiNaC::multi_iterator_shuffle< T >::multi_iterator_shuffle (
    const std::vector< T > & a,
    const std::vector< T > & b ) [inline], [explicit]
```

Construct from a vector.

References [GiNaC::basic_multi_iterator< T >::B](#), [GiNaC::multi_iterator_shuffle< T >::N_internal](#), [GiNaC::basic_multi_iterator< T >::v](#), [GiNaC::multi_iterator_shuffle< T >::v_internal](#), and [GiNaC::multi_iterator_shuffle< T >::v_orig](#).

6.102.3 Member Function Documentation

6.102.3.1 `init()`

```
template<class T >
basic_multi_iterator< T > & GiNaC::multi_iterator_shuffle< T >::init (
    void ) [inline], [virtual]
```

Initialize the multi-index to the first shuffle.

Reimplemented from [GiNaC::basic_multi_iterator< T >](#).

Reimplemented in [GiNaC::multi_iterator_shuffle_prime< T >](#).

6.102.3.2 `operator++()`

```
template<class T >
basic_multi_iterator< T > & GiNaC::multi_iterator_shuffle< T >::operator++ (
    int ) [inline], [virtual]
```

The postfix increment operator allows to write for a multi-index `n++`, which will update `n` to the next configuration.

If `n` is in the last configuration and the increment operator `++` is applied to `n`, the overflow flag will be raised.

Reimplemented from [GiNaC::basic_multi_iterator< T >](#).

References [k](#), and [GiNaC::basic_multi_iterator< T >::size\(\)](#).

6.102.4 Friends And Related Symbol Documentation

6.102.4.1 operator<<

```
template<class T >
template<class TT >
std::ostream & operator<< (
    std::ostream & os,
    const multi_iterator_shuffle< TT > & v ) [friend]
```

6.102.5 Member Data Documentation

6.102.5.1 N_internal

```
template<class T >
size_t GiNaC::multi_iterator_shuffle< T >::N_internal [protected]
```

Referenced by [GiNaC::multi_iterator_shuffle< T >::multi_iterator_shuffle\(\)](#).

6.102.5.2 v_internal

```
template<class T >
std::vector<size_t> GiNaC::multi_iterator_shuffle< T >::v_internal [protected]
```

Referenced by [GiNaC::multi_iterator_shuffle< T >::multi_iterator_shuffle\(\)](#).

6.102.5.3 v_orig

```
template<class T >
std::vector<T> GiNaC::multi_iterator_shuffle< T >::v_orig [protected]
```

Referenced by [GiNaC::multi_iterator_shuffle< T >::multi_iterator_shuffle\(\)](#).

The documentation for this class was generated from the following file:

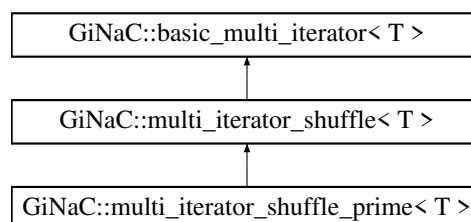
- [utils_multi_iterator.h](#)

6.103 GiNaC::multi_iterator_shuffle_prime< T > Class Template Reference

The class [multi_iterator_shuffle_prime](#) defines a multi_iterator, which runs over all shuffles of a and b, excluding the first one (a,b).

```
#include <utils_multi_iterator.h>
```

Inheritance diagram for GiNaC::multi_iterator_shuffle_prime< T >:



Public Member Functions

- [multi_iterator_shuffle_prime](#) (void)
Default constructor.
- [multi_iterator_shuffle_prime](#) (const std::vector< T > &a, const std::vector< T > &b)
Construct from a vector.
- [basic_multi_iterator](#)< T > & [init](#) (void)
Initialize the multi-index to the first shuffle.

Public Member Functions inherited from [GiNaC::multi_iterator_shuffle< T >](#)

- [multi_iterator_shuffle](#) (void)
Default constructor.
- [multi_iterator_shuffle](#) (const std::vector< T > &a, const std::vector< T > &b)
Construct from a vector.
- [basic_multi_iterator](#)< T > & [operator++](#) (int)
The postfix increment operator allows to write for a multi-index n++, which will update n to the next configuration.

Public Member Functions inherited from [GiNaC::basic_multi_iterator< T >](#)

- [basic_multi_iterator](#) (void)
Default constructor.
- [basic_multi_iterator](#) (T B, T N, size_t k)
Construct a multi_iterator with upper limit N, lower limit B and size k .
- [basic_multi_iterator](#) (T B, T N, const std::vector< T > &vv)
Construct from a vector.
- virtual [~basic_multi_iterator](#) ()
Destructor.
- size_t [size](#) (void) const
Returns the size of a multi_iterator.
- bool [overflow](#) (void) const
Return the overflow flag.
- const std::vector< T > & [get_vector](#) (void) const
Returns a reference to the vector v.
- T [operator\[\]](#) (size_t i) const
Subscription via [].
- T & [operator\[\]](#) (size_t i)
Subscription via [].
- T [operator\(\)](#) (size_t i) const
Subscription via ().
- T & [operator\(\)](#) (size_t i)
Subscription via ().

Friends

- template<class TT >
std::ostream & [operator<<](#) (std::ostream &os, const [multi_iterator_shuffle_prime](#)< TT > &v)

Additional Inherited Members

Protected Attributes inherited from [GiNaC::multi_iterator_shuffle< T >](#)

- [size_t N_internal](#)
- [std::vector< size_t > v_internal](#)
- [std::vector< T > v_orig](#)

Protected Attributes inherited from [GiNaC::basic_multi_iterator< T >](#)

- [T N](#)
- [T B](#)
- [std::vector< T > v](#)
- [bool flag_overflow](#)

6.103.1 Detailed Description

```
template<class T>
class GiNaC::multi_iterator_shuffle_prime< T >
```

The class [multi_iterator_shuffle_prime](#) defines a multi_iterator, which runs over all shuffles of a and b, excluding the first one (a,b).

6.103.2 Constructor & Destructor Documentation

6.103.2.1 multi_iterator_shuffle_prime() [1/2]

```
template<class T >
GiNaC::multi_iterator_shuffle_prime< T >::multi_iterator_shuffle_prime (
    void ) [inline]
```

Default constructor.

6.103.2.2 multi_iterator_shuffle_prime() [2/2]

```
template<class T >
GiNaC::multi_iterator_shuffle_prime< T >::multi_iterator_shuffle_prime (
    const std::vector< T > & a,
    const std::vector< T > & b ) [inline], [explicit]
```

Construct from a vector.

6.103.3 Member Function Documentation

6.103.3.1 init()

```
template<class T >
basic_multi_iterator< T > & GiNaC::multi_iterator_shuffle_prime< T >::init (
    void ) [inline], [virtual]
```

Initialize the multi-index to the first shuffle.

Reimplemented from [GiNaC::multi_iterator_shuffle< T >](#).

6.103.4 Friends And Related Symbol Documentation

6.103.4.1 operator<<

```
template<class T >
template<class TT >
std::ostream & operator<< (
    std::ostream & os,
    const multi_iterator_shuffle_prime< TT > & v ) [friend]
```

The documentation for this class was generated from the following file:

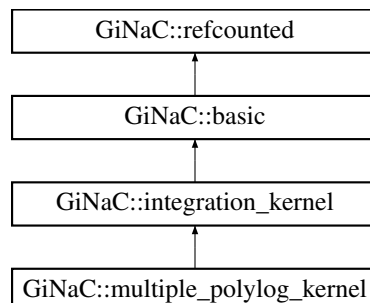
- [utils_multi_iterator.h](#)

6.104 GiNaC::multiple_polylog_kernel Class Reference

The integration kernel for multiple polylogarithms.

```
#include <integration_kernel.h>
```

Inheritance diagram for GiNaC::multiple_polylog_kernel:



Public Member Functions

- [multiple_polylog_kernel](#) (const [ex](#) &z)
 - [size_t nops](#) () const override

Number of operands/members.
 - [ex op](#) (size_t i) const override

Return operand/member at position i.
 - [ex & let_op](#) (size_t i) override

Return modifiable operand/member at position i.
 - bool [is_numeric](#) (void) const override

This routine returns true, if the integration kernel can be evaluated numerically.

Public Member Functions inherited from [GiNaC::integration_kernel](#)

- [ex series](#) (const [relational](#) &r, int [order](#), unsigned [options](#)=0) const override

Default implementation of [ex::series\(\)](#).
- virtual bool [has_trailing_zero](#) (void) const

This routine returns true, if the integration kernel has a trailing zero.
- virtual [ex Laurent_series](#) (const [ex](#) &x, int [order](#)) const

Returns the Laurent series, starting possibly with the pole term.
- virtual [ex get_numerical_value](#) (const [ex](#) &lambda, int N_trunc=0) const

Evaluates the integrand at lambda.
- [size_t get_cache_size](#) (void) const

Returns the current size of the cache.
- void [set_cache_step](#) (int cache_steps) const

Sets the step size by which the cache is increased.
- [ex get_series_coeff](#) (int i) const

Wrapper around [series_coeff\(i\)](#), converts cl_N to numeric.
- [cln::cl_N series_coeff](#) (int i) const

Subclasses have either to implement [series_coeff_impl](#) or the two methods [Laurent_series](#) and [uses_Laurent_series](#).

Public Member Functions inherited from [GiNaC::basic](#)

- virtual [~basic](#) ()

basic destructor, virtual because class [ex](#) will delete objects of derived classes via a [basic](#).*
- [basic](#) (const [basic](#) &other)
- const [basic](#) & [operator=](#) (const [basic](#) &other)

basic assignment operator: the other object might be of a derived class.
- virtual [basic * duplicate](#) () const

Create a clone of this object on the heap.
- virtual [ex eval](#) () const

Perform automatic non-interruptive term rewriting rules.
- virtual [ex evalf](#) () const

Evaluate object numerically.
- virtual [ex evalm](#) () const

Evaluate sums, products and integer powers of matrices.
- virtual [ex eval_integ](#) () const

Evaluate integrals, if result is known.
- virtual [ex eval_indexed](#) (const [basic](#) &i) const

Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.

- virtual void `print` (const `print_context` &c, unsigned level=0) const
Output to stream.
- virtual void `dbgprint` () const
Little wrapper around print to be called within a debugger.
- virtual void `dbgprnttree` () const
Little wrapper around prnttree to be called within a debugger.
- virtual unsigned `precedence` () const
Return relative operator precedence (for parenthezing output).
- virtual bool `info` (unsigned inf) const
Information about the object.
- virtual `ex operator[]` (const `ex` &index) const
- virtual `ex operator[]` (size_t i) const
- virtual `ex & operator[]` (const `ex` &index)
- virtual `ex & operator[]` (size_t i)
- virtual bool `has` (const `ex` &other, unsigned `options`=0) const
Test for occurrence of a pattern.
- virtual bool `match` (const `ex` &pattern, `exmap` &repls) const
Check whether the expression matches a given pattern.
- virtual `ex subs` (const `exmap` &m, unsigned `options`=0) const
Substitute a set of objects by arbitrary expressions.
- virtual `ex map` (`map_function` &f) const
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void `accept` (`GiNaC::visitor` &v) const
- virtual bool `is_polynomial` (const `ex` &var) const
Check whether this is a polynomial in the given variables.
- virtual int `degree` (const `ex` &s) const
Return degree of highest power in object s.
- virtual int `ldegree` (const `ex` &s) const
Return degree of lowest power in object s.
- virtual `ex coeff` (const `ex` &s, int n=1) const
Return coefficient of degree n in object s.
- virtual `ex expand` (unsigned `options`=0) const
Expand expression, i.e.
- virtual `ex collect` (const `ex` &s, bool distributed=false) const
Sort expanded expression in terms of powers of some object(s).
- virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const
Default implementation of `ex::normal()`.
- virtual `ex to_rational` (`exmap` &repl) const
Default implementation of `ex::to_rational()`.
- virtual `ex to_polynomial` (`exmap` &repl) const
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &xi) const
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient` () const
Implementation `ex::max_coefficient()`.
- virtual `exvector get_free_indices` () const
Return a vector containing the free indices of an expression.
- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
Add two indexed expressions.
- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
Multiply an indexed expression with a scalar.

- virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const
Try to contract two indexed expressions that appear in the same product.
- virtual unsigned `return_type` () const
- virtual `return_type_t` `return_type_tinfo` () const
- virtual `ex` `conjugate` () const
- virtual `ex` `real_part` () const
- virtual `ex` `imag_part` () const
- template<class T >
void `print_dispatch` (const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- virtual void `archive` (`archive_node` &n) const
Save (serialize) the object into archive node.
- virtual void `read_archive` (const `archive_node` &n, `lst` &syms)
Load (deserialize) the object from an archive node.
- `ex` `subs_one_level` (const `exmap` &m, unsigned `options`) const
Helper function for `subs()`.
- `ex` `diff` (const `symbol` &s, unsigned nth=1) const
Default interface of nth derivative `ex::diff(s, n)`.
- int `compare` (const `basic` &other) const
Compare objects syntactically to establish canonical ordering.
- bool `is_equal` (const `basic` &other) const
Test for syntactic equality.
- const `basic` & `hold` () const
Stop further evaluation.
- unsigned `gethash` () const
- const `basic` & `setflag` (unsigned f) const
Set some `status_flags`.
- const `basic` & `clearflag` (unsigned f) const
Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Protected Member Functions

- cln::cl_N `series_coeff_impl` (int i) const override
For $\omega = d\lambda$ only the coefficient of λ^0 is non-zero.
- void `do_print` (const `print_context` &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::integration_kernel](#)

- virtual bool [uses_Laurent_series](#) () const
Returns true, if the coefficients are computed from the Laurent series (in which case the method `Laurent_series` needs to be implemented).
- [ex](#) [get_numerical_value_impl](#) (const [ex](#) &lambda, const [ex](#) &pre, int shift, int N_trunc) const
The actual implementation for computing a numerical value for the integrand.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex](#) [eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual [ex](#) [derivative](#) (const [symbol](#) &s) const
Default implementation of `ex::diff()`.
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Protected Attributes

- [ex](#) [z](#)

Protected Attributes inherited from [GiNaC::integration_kernel](#)

- int [cache_step_size](#)
- std::vector< [cln::cl_N](#) > [series_vec](#)

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type `status_flags`
- unsigned [hashvalue](#)
hash value

6.104.1 Detailed Description

The integration kernel for multiple polylogarithms.

This class represents the differential one-form

$$\omega^{\text{mpl}}(z) = \frac{d\lambda}{\lambda - z}$$

For the case $z = 0$ the class [basic_log_kernel](#) should be used.

6.104.2 Constructor & Destructor Documentation

6.104.2.1 multiple_polylog_kernel()

```
GiNaC::multiple_polylog_kernel::multiple_polylog_kernel (
    const ex & z )
```

6.104.3 Member Function Documentation

6.104.3.1 nops()

```
size_t GiNaC::multiple_polylog_kernel::nops ( ) const [override], [virtual]
```

Number of operands/members.

Reimplemented from [GiNaC::basic](#).

6.104.3.2 op()

```
ex GiNaC::multiple_polylog_kernel::op (
    size_t i ) const [override], [virtual]
```

Return operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [z](#).

6.104.3.3 let_op()

```
ex & GiNaC::multiple_polylog_kernel::let_op (
    size_t i ) [override], [virtual]
```

Return modifiable operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::basic::ensure_if_modifiable\(\)](#), and [z](#).

6.104.3.4 `is_numeric()`

```
bool GiNaC::multiple_polylog_kernel::is_numeric (
    void ) const [override], [virtual]
```

This routine returns true, if the integration kernel can be evaluated numerically.

Reimplemented from [GiNaC::integration_kernel](#).

References [GiNaC::ex::evalf\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::numeric](#), and [z](#).

6.104.3.5 `series_coeff_impl()`

```
cln::cl_N GiNaC::multiple_polylog_kernel::series_coeff_impl (
    int i ) const [override], [protected], [virtual]
```

For $\omega = d\lambda$ only the coefficient of λ^0 is non-zero.

The i -th coefficient corresponds to the power λ^{i-1} .

Reimplemented from [GiNaC::integration_kernel](#).

References [GiNaC::ex::evalf\(\)](#), and [z](#).

6.104.3.6 `do_print()`

```
void GiNaC::multiple_polylog_kernel::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

References [c](#), [GiNaC::ex::print\(\)](#), and [z](#).

6.104.4 Member Data Documentation

6.104.4.1 `z`

```
ex GiNaC::multiple_polylog_kernel::z [protected]
```

Referenced by [do_print\(\)](#), [is_numeric\(\)](#), [let_op\(\)](#), [op\(\)](#), and [series_coeff_impl\(\)](#).

The documentation for this class was generated from the following files:

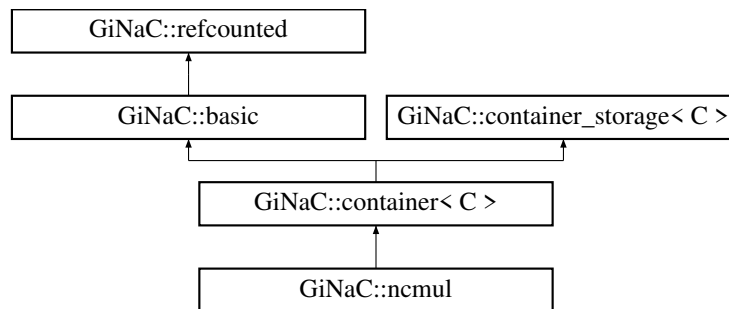
- [integration_kernel.h](#)
- [integration_kernel.cpp](#)

6.105 GiNaC::ncmul Class Reference

Non-commutative product of expressions.

```
#include <ncmul.h>
```

Inheritance diagram for GiNaC::ncmul:



Public Member Functions

- `ncmul` (const `ex` &lh, const `ex` &rh)
- `ncmul` (const `ex` &f1, const `ex` &f2, const `ex` &f3)
- `ncmul` (const `ex` &f1, const `ex` &f2, const `ex` &f3, const `ex` &f4)
- `ncmul` (const `ex` &f1, const `ex` &f2, const `ex` &f3, const `ex` &f4, const `ex` &f5)
- `ncmul` (const `ex` &f1, const `ex` &f2, const `ex` &f3, const `ex` &f4, const `ex` &f5, const `ex` &f6)
- `ncmul` (const `exvector` &v)
- `ncmul` (`exvector` &&v)
- unsigned `precedence` () const override
Return relative operator precedence (for parenthezing output).
- bool `info` (unsigned inf) const override
Information about the object.
- int `degree` (const `ex` &s) const override
Return degree of highest power in object s.
- int `ldegree` (const `ex` &s) const override
Return degree of lowest power in object s.
- `ex expand` (unsigned `options`=0) const override
Expand expression, i.e.
- `ex coeff` (const `ex` &s, int `n`=1) const override
Return coefficient of degree n in object s.
- `ex eval` () const override
Perform automatic term rewriting rules in this class.
- `ex evalm` () const override
Evaluate sums, products and integer powers of matrices.
- `exvector get_free_indices` () const override
Return a vector containing the free indices of an expression.
- `ex thiscontainer` (const `exvector` &v) const override
- `ex thiscontainer` (`exvector` &&v) const override
- `ex conjugate` () const override
- `ex real_part` () const override
- `ex imag_part` () const override
- const `exvector` & `get_factors` () const

Public Member Functions inherited from `GiNaC::container< C >`

- `container` (STLT const &s)
- `container` (STLT &&v)
- `container` (exvector::const_iterator b, exvector::const_iterator e)
- `container` (std::initializer_list< ex > il)
- `size_t nops` () const override
Number of operands/members.
- `ex op` (size_t i) const override
Return operand/member at position i.
- `ex & let_op` (size_t i) override
Return modifiable operand/member at position i.
- `ex subs` (const exmap &m, unsigned options=0) const override
Substitute a set of objects by arbitrary expressions.
- void `read_archive` (const archive_node &n, lst &sym_lst) override
Load (deserialize) the object from an archive node.
- void `archive` (archive_node &n) const override
Archive the object.
- `container & prepend` (const ex &b)
Add element at front.
- `container & append` (const ex &b)
Add element at back.
- `container & remove_first` ()
Remove first element.
- `container & remove_last` ()
Remove last element.
- `container & remove_all` ()
Remove all elements.
- `container & sort` ()
Sort elements.
- `container & unique` ()
Remove adjacent duplicate elements.
- `const_iterator begin` () const
- `const_iterator end` () const
- `const_reverse_iterator rbegin` () const
- `const_reverse_iterator rend` () const

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic` ()
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic` (const basic &other)
- const `basic & operator=` (const basic &other)
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate` () const
Create a clone of this object on the heap.
- virtual `ex evalf` () const
Evaluate object numerically.
- virtual `ex eval_integ` () const
Evaluate integrals, if result is known.
- virtual `ex eval_indexed` (const basic &i) const

- Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.*
- virtual void `print` (const `print_context` &c, unsigned level=0) const
Output to stream.
- virtual void `dbgprint` () const
Little wrapper around print to be called within a debugger.
- virtual void `dbgprinttree` () const
Little wrapper around printtree to be called within a debugger.
- virtual `ex operator[]` (const `ex` &index) const
- virtual `ex operator[]` (size_t i) const
- virtual `ex & operator[]` (const `ex` &index)
- virtual `ex & operator[]` (size_t i)
- virtual bool `has` (const `ex` &other, unsigned options=0) const
Test for occurrence of a pattern.
- virtual bool `match` (const `ex` &pattern, `exmap` &repls) const
Check whether the expression matches a given pattern.
- virtual `ex map` (`map_function` &f) const
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void `accept` (`GiNaC::visitor` &v) const
- virtual bool `is_polynomial` (const `ex` &var) const
Check whether this is a polynomial in the given variables.
- virtual `ex collect` (const `ex` &s, bool distributed=false) const
Sort expanded expression in terms of powers of some object(s).
- virtual `ex series` (const `relational` &r, int order, unsigned options=0) const
Default implementation of `ex::series()`.
- virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const
Default implementation of `ex::normal()`.
- virtual `ex to_rational` (`exmap` &repl) const
Default implementation of `ex::to_rational()`.
- virtual `ex to_polynomial` (`exmap` &repl) const
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &xi) const
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient` () const
Implementation `ex::max_coefficient()`.
- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
Add two indexed expressions.
- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
Multiply an indexed expression with a scalar.
- virtual bool `contract_with` (`exvector::iterator` self, `exvector::iterator` other, `exvector` &v) const
Try to contract two indexed expressions that appear in the same product.
- template<class T >
void `print_dispatch` (const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- `ex subs_one_level` (const `exmap` &m, unsigned options) const
Helper function for `subs()`.
- `ex diff` (const `symbol` &s, unsigned nth=1) const
Default interface of nth derivative `ex::diff(s, n)`.
- int `compare` (const `basic` &other) const
Compare objects syntactically to establish canonical ordering.

- bool `is_equal` (const `basic` &other) const
Test for syntactic equality.
- const `basic` & `hold` () const
Stop further evaluation.
- unsigned `gethash` () const
- const `basic` & `setflag` (unsigned f) const
Set some `status_flags`.
- const `basic` & `clearflag` (unsigned f) const
Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Protected Member Functions

- `ex derivative` (const `symbol` &s) const override
Implementation of `ex::diff()` for a non-commutative product.
- unsigned `return_type` () const override
- `return_type_t` `return_type_tinfo` () const override
- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_csrf` (const `print_context` &c, unsigned level) const
- size_t `count_factors` (const `ex` &e) const
- void `append_factors` (`exvector` &v, const `ex` &e) const
- `exvector` `expandchildren` (unsigned `options`) const

Protected Member Functions inherited from `GiNaC::container< C >`

- bool `is_equal_same_type` (const `basic` &other) const override
Returns true if two objects of same type are equal.
- virtual `ex thiscontainer` (const `STLT` &v) const
Similar to `duplicate()`, but with a preset sequence.
- virtual `ex thiscontainer` (`STLT` &&v) const
Similar to `duplicate()`, but with a preset sequence (which gets pilfered).
- virtual void `printseq` (const `print_context` &c, char openbracket, char delim, char closebracket, unsigned this←_precedence, unsigned upper_precedence=0) const
Print sequence of contained elements.
- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
- void `do_print_python` (const `print_python` &c, unsigned level) const
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
- `STLT` `subchildren` (const `exmap` &m, unsigned `options`=0) const

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Protected Member Functions inherited from [GiNaC::container_storage< C >](#)

- [container_storage](#) ()
- [container_storage](#) (size_t n, const [ex](#) &e)
- [container_storage](#) (std::initializer_list< [ex](#) > il)
- template<class In >
 [container_storage](#) (In b, In e)
- void [reserve](#) (size_t)
- [~container_storage](#) ()
- void [reserve](#) (size_t n)
- void [reserve](#) (std::vector< [ex](#) > &v, size_t n)

Friends

- class [power](#)
- [ex reeval_ncmul](#) (const [exvector](#) &v)
- [ex hold_ncmul](#) (const [exvector](#) &v)

Additional Inherited Members

Public Types inherited from [GiNaC::container< C >](#)

- typedef STLT::const_iterator [const_iterator](#)
- typedef STLT::const_reverse_iterator [const_reverse_iterator](#)

Protected Types inherited from [GiNaC::container< C >](#)

- typedef [container_storage< C >::STLT](#) [STLT](#)

Protected Types inherited from [GiNaC::container_storage< C >](#)

- typedef [C< ex >](#) [STLT](#)

Static Protected Member Functions inherited from [GiNaC::container< C >](#)

- static unsigned [get_default_flags](#) ()
Specialization of [container::get_default_flags\(\)](#) for [lst](#).
- static char [get_open_delim](#) ()
Specialization of [container::get_open_delim\(\)](#) for [lst](#).
- static char [get_close_delim](#) ()
Specialization of [container::get_close_delim\(\)](#) for [lst](#).

Static Protected Member Functions inherited from [GiNaC::container_storage< C >](#)

- static void [reserve](#) ([STLT](#) &, size_t)

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

Protected Attributes inherited from [GiNaC::container_storage< C >](#)

- [STLT seq](#)

6.105.1 Detailed Description

Non-commutative product of expressions.

6.105.2 Constructor & Destructor Documentation

6.105.2.1 [ncmul\(\)](#) [1/7]

```
GiNaC::ncmul::ncmul (
    const ex & lh,
    const ex & rh )
```

6.105.2.2 [ncmul\(\)](#) [2/7]

```
GiNaC::ncmul::ncmul (
    const ex & f1,
    const ex & f2,
    const ex & f3 )
```

6.105.2.3 ncmul() [3/7]

```
GiNaC::ncmul::ncmul (
    const ex & f1,
    const ex & f2,
    const ex & f3,
    const ex & f4 )
```

6.105.2.4 ncmul() [4/7]

```
GiNaC::ncmul::ncmul (
    const ex & f1,
    const ex & f2,
    const ex & f3,
    const ex & f4,
    const ex & f5 )
```

6.105.2.5 ncmul() [5/7]

```
GiNaC::ncmul::ncmul (
    const ex & f1,
    const ex & f2,
    const ex & f3,
    const ex & f4,
    const ex & f5,
    const ex & f6 )
```

6.105.2.6 ncmul() [6/7]

```
GiNaC::ncmul::ncmul (
    const exvector & v )
```

6.105.2.7 ncmul() [7/7]

```
GiNaC::ncmul::ncmul (
    exvector && v )
```

6.105.3 Member Function Documentation**6.105.3.1 precedence()**

```
unsigned GiNaC::ncmul::precedence ( ) const [inline], [override], [virtual]
```

Return relative operator precedence (for parenthezing output).

Reimplemented from [GiNaC::container< C >](#).

Referenced by [do_print\(\)](#), and [do_print_csrc\(\)](#).

6.105.3.2 info()

```
bool GiNaC::ncmul::info (
    unsigned inf ) const [override], [virtual]
```

Information about the object.

See also

class [info_flags](#)

Reimplemented from [GiNaC::container< C >](#).

6.105.3.3 degree()

```
int GiNaC::ncmul::degree (
    const ex & s ) const [override], [virtual]
```

Return degree of highest power in object *s*.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::basic::is_equal\(\)](#), and [GiNaC::container_storage< C >::seq](#).

6.105.3.4 ldegree()

```
int GiNaC::ncmul::ldegree (
    const ex & s ) const [override], [virtual]
```

Return degree of lowest power in object *s*.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::basic::is_equal\(\)](#), and [GiNaC::container_storage< C >::seq](#).

6.105.3.5 expand()

```
ex GiNaC::ncmul::expand (
    unsigned options = 0 ) const [override], [virtual]
```

Expand expression, i.e.

multiply it out and return the result as a new expression.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex1](#), [expandchildren\(\)](#), [GiNaC::status_flags::expanded](#), [k](#), [GiNaC::container< C >::op\(\)](#), [options](#), [GiNaC::rename_dummy_indices_uniquely\(\)](#), [GiNaC::container_storage< C >::seq](#), and [GiNaC::basic::setflag\(\)](#).

6.105.3.6 coeff()

```
ex GiNaC::ncmul::coeff (
    const ex & s,
    int n = 1 ) const [override], [virtual]
```

Return coefficient of degree n in object s.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex0](#), [GiNaC::_ex1](#), [c](#), [GiNaC::ex::coeff\(\)](#), [GiNaC::basic::is_equal\(\)](#), [n](#), and [GiNaC::container_storage< C >::seq](#).

Referenced by [GiNaC::add::coeff\(\)](#).

6.105.3.7 eval()

```
ex GiNaC::ncmul::eval ( ) const [override], [virtual]
```

Perform automatic term rewriting rules in this class.

In the following x, x1, x2,... stand for a symbolic variables of type ex and c, c1, c2... stand for such expressions that contain a plain number.

- `ncmul(...,(x1,x2),...,ncmul(x3,x4),...)` -> `ncmul(...,x1,x2,...,x3,x4,...)` (associativity)
- `ncmul(x)` -> `x`
- `ncmul()` -> `1`
- `ncmul(...,c1,...,c2,...)` -> `*(c1,c2,ncmul(...))` (pull out commutative elements)
- `ncmul(x1,y1,x2,y2)` -> `*(ncmul(x1,x2),ncmul(y1,y2))` (collect elements of same type)
- `ncmul(x1,x2,x3,...)` -> `x::eval_ncmul(x1,x2,x3,...)`

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex1](#), [append_factors\(\)](#), [GiNaC::return_types::commutative](#), [count_factors\(\)](#), [GiNaC::status_flags::evaluated](#), [GiNaC::factor\(\)](#), [factors](#), [GiNaC::basic::flags](#), [GINAC_ASSERT](#), [GiNaC::make_flat_inserter::handle_factor\(\)](#), [GiNaC::return_types::noncommutative](#), [GiNaC::return_types::noncommutative_composite](#), [GiNaC::container_storage< C >::reserve](#) and [GiNaC::container_storage< C >::seq](#).

6.105.3.8 evalm()

```
ex GiNaC::ncmul::evalm ( ) const [override], [virtual]
```

Evaluate sums, products and integer powers of matrices.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::begin\(\)](#), [GiNaC::matrix::mul\(\)](#), and [GiNaC::container_storage< C >::seq](#).

6.105.3.9 get_free_indices()

```
exvector GiNaC::ncmul::get_free_indices ( ) const [override], [virtual]
```

Return a vector containing the free indices of an expression.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::find_free_and_dummy\(\)](#), [GiNaC::ex::get_free_indices\(\)](#), [GiNaC::container< C >::nops\(\)](#), and [GiNaC::container< C >::op\(\)](#).

6.105.3.10 thiscontainer() [1/2]

```
ex GiNaC::ncmul::thiscontainer (
    const exvector & v ) const [override]
```

6.105.3.11 thiscontainer() [2/2]

```
ex GiNaC::ncmul::thiscontainer (
    exvector && v ) const [override]
```

6.105.3.12 conjugate()

```
ex GiNaC::ncmul::conjugate ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::container< C >](#).

References [GiNaC::container< C >::begin\(\)](#), [GiNaC::container< C >::conjugate\(\)](#), [GiNaC::container< C >::end\(\)](#), [GiNaC::is_clifford_tinfo\(\)](#), [GiNaC::return_types::noncommutative](#), [GiNaC::container< C >::nops\(\)](#), [return_type\(\)](#), and [return_type_tinfo\(\)](#).

6.105.3.13 real_part()

```
ex GiNaC::ncmul::real_part ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::container< C >](#).

References [GiNaC::basic::real_part\(\)](#).

6.105.3.14 imag_part()

```
ex GiNaC::ncmul::imag_part ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::container< C >](#).

References [GiNaC::basic::imag_part\(\)](#).

6.105.3.15 derivative()

```
ex GiNaC::ncmul::derivative (
    const symbol & s ) const [override], [protected], [virtual]
```

Implementation of [ex::diff\(\)](#) for a non-commutative product.

It applies the product rule.

See also

[ex::diff](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::diff\(\)](#), [GiNaC::container_storage< C >::seq](#), and [GiNaC::ex::swap\(\)](#).

6.105.3.16 return_type()

```
unsigned GiNaC::ncmul::return_type ( ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::return_types::commutative](#), [GiNaC::container< C >::end\(\)](#), [GINAC_ASSERT](#), [GiNaC::return_types::noncommutative](#), [GiNaC::return_types::noncommutative_composite](#), and [GiNaC::container_storage< C >::seq](#).

Referenced by [conjugate\(\)](#).

6.105.3.17 return_type_tinfo()

```
return\_type\_t GiNaC::ncmul::return_type_tinfo ( ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::return_types::noncommutative](#), and [GiNaC::container_storage< C >::seq](#).

Referenced by [conjugate\(\)](#).

6.105.3.18 do_print()

```
void GiNaC::ncmul::do_print (
    const print\_context & c,
    unsigned level ) const [protected]
```

References [c](#), [precedence\(\)](#), and [GiNaC::container< C >::printseq\(\)](#).

6.105.3.19 do_print_csrc()

```
void GiNaC::ncmul::do_print_csrc (
    const print\_context & c,
    unsigned level ) const [protected]
```

References [c](#), [precedence\(\)](#), and [GiNaC::container< C >::printseq\(\)](#).

6.105.3.20 count_factors()

```
size_t GiNaC::ncmul::count_factors (
    const ex & e ) const [protected]
```

References [GiNaC::return_types::commutative](#), [count_factors\(\)](#), [factors](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), and [GiNaC::ex::return_type\(\)](#).

Referenced by [count_factors\(\)](#), and [eval\(\)](#).

6.105.3.21 append_factors()

```
void GiNaC::ncmul::append_factors (
    exvector & v,
    const ex & e ) const [protected]
```

References [append_factors\(\)](#), [GiNaC::return_types::commutative](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), and [GiNaC::ex::return_type\(\)](#).

Referenced by [append_factors\(\)](#), and [eval\(\)](#).

6.105.3.22 expandchildren()

```
exvector GiNaC::ncmul::expandchildren (
    unsigned options ) const [protected]
```

References [GiNaC::are_ex_trivially_equal\(\)](#), [GiNaC::container< C >::end\(\)](#), [GiNaC::ex::expand\(\)](#), [options](#), and [GiNaC::container_storage< C >::seq](#).

Referenced by [expand\(\)](#).

6.105.3.23 get_factors()

```
const exvector & GiNaC::ncmul::get_factors ( ) const
```

References [GiNaC::container_storage< C >::seq](#).

6.105.4 Friends And Related Symbol Documentation**6.105.4.1 power**

```
friend class power [friend]
```

6.105.4.2 reeval_ncmul

```
ex reeval_ncmul (
    const exvector & v ) [friend]
```

6.105.4.3 hold_ncmul

```
ex hold_ncmul (
    const exvector & v ) [friend]
```

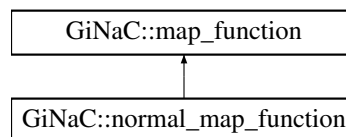
The documentation for this class was generated from the following files:

- [ncmul.h](#)
- [indexed.cpp](#)
- [ncmul.cpp](#)

6.106 GiNaC::normal_map_function Struct Reference

Function object to be applied by [basic::normal\(\)](#).

Inheritance diagram for GiNaC::normal_map_function:



Public Member Functions

- [ex operator\(\)](#) (const [ex](#) &e) override

Public Member Functions inherited from [GiNaC::map_function](#)

- virtual [~map_function](#) ()

Additional Inherited Members

Public Types inherited from [GiNaC::map_function](#)

- typedef const [ex](#) & [argument_type](#)
- typedef [ex](#) [result_type](#)

6.106.1 Detailed Description

Function object to be applied by [basic::normal\(\)](#).

6.106.2 Member Function Documentation

6.106.2.1 operator>()

```
ex GiNaC::normal_map_function::operator() (
    const ex & e ) [inline], [override], [virtual]
```

Implements [GiNaC::map_function](#).

References [GiNaC::normal\(\)](#).

The documentation for this struct was generated from the following file:

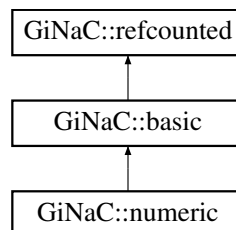
- [normal.cpp](#)

6.107 GiNaC::numeric Class Reference

This class is a wrapper around CLN-numbers within the [GiNaC](#) class hierarchy.

```
#include <numeric.h>
```

Inheritance diagram for GiNaC::numeric:



Public Member Functions

- [numeric](#) (int i)
- [numeric](#) (unsigned int i)
- [numeric](#) (long i)
- [numeric](#) (unsigned long i)
- [numeric](#) (long long i)
- [numeric](#) (unsigned long long i)
- [numeric](#) (long [numer](#), long [denom](#))
Constructor for rational numerics a/b.
- [numeric](#) (double d)
- [numeric](#) (const char *)
ctor from C-style string.
- unsigned [precedence](#) () const override
Return relative operator precedence (for parenthezing output).
- bool [info](#) (unsigned inf) const override
Information about the object.
- bool [is_polynomial](#) (const [ex](#) &var) const override
Check whether this is a polynomial in the given variables.

- `int degree` (const `ex` &`s`) const override
Return degree of highest power in object s.
- `int ldegree` (const `ex` &`s`) const override
Return degree of lowest power in object s.
- `ex coeff` (const `ex` &`s`, int `n=1`) const override
Return coefficient of degree n in object s.
- `bool has` (const `ex` &`other`, unsigned `options=0`) const override
Disassemble real part and imaginary part to scan for the occurrence of a single number.
- `ex eval` () const override
Evaluation of numbers doesn't do anything at all.
- `ex evalf` () const override
Cast numeric into a floating-point object.
- `ex subs` (const `exmap` &`m`, unsigned `options=0`) const override
Substitute a set of objects by arbitrary expressions.
- `ex normal` (`exmap` &`repl`, `exmap` &`rev_lookup`, `lst` &`modifier`) const override
Implementation of `ex::normal()` for a numeric.
- `ex to_rational` (`exmap` &`repl`) const override
Implementation of `ex::to_rational()` for a numeric.
- `ex to_polynomial` (`exmap` &`repl`) const override
Implementation of `ex::to_polynomial()` for a numeric.
- `numeric integer_content` () const override
- `ex smod` (const `numeric` &`xi`) const override
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- `numeric max_coefficient` () const override
Implementation `ex::max_coefficient()`.
- `ex conjugate` () const override
- `ex real_part` () const override
- `ex imag_part` () const override
- `void archive` (`archive_node` &`n`) const override
Save (a.k.a.
- `void read_archive` (const `archive_node` &`n`, `lst` &`syms`) override
Read (a.k.a.
- `const numeric add` (const `numeric` &`other`) const
Numerical addition method.
- `const numeric sub` (const `numeric` &`other`) const
Numerical subtraction method.
- `const numeric mul` (const `numeric` &`other`) const
Numerical multiplication method.
- `const numeric div` (const `numeric` &`other`) const
Numerical division method.
- `const numeric power` (const `numeric` &`other`) const
Numerical exponentiation.
- `const numeric & add_dyn` (const `numeric` &`other`) const
Numerical addition method.
- `const numeric & sub_dyn` (const `numeric` &`other`) const
Numerical subtraction method.
- `const numeric & mul_dyn` (const `numeric` &`other`) const
Numerical multiplication method.
- `const numeric & div_dyn` (const `numeric` &`other`) const
Numerical division method.
- `const numeric & power_dyn` (const `numeric` &`other`) const

Numerical exponentiation.

- const `numeric & operator=` (int i)
- const `numeric & operator=` (unsigned int i)
- const `numeric & operator=` (long i)
- const `numeric & operator=` (unsigned long i)
- const `numeric & operator=` (double d)
- const `numeric & operator=` (const char *s)
- const `numeric inverse` () const

Inverse of a number.

- `numeric step` () const

Return the step function of a numeric.

- int `csgn` () const

Return the complex half-plane (left or right) in which the number lies.

- int `compare` (const `numeric` &other) const

This method establishes a canonical order on all numbers.

- bool `is_equal` (const `numeric` &other) const
- bool `is_zero` () const

True if object is zero.

- bool `is_positive` () const

True if object is not complex and greater than zero.

- bool `is_negative` () const

True if object is not complex and less than zero.

- bool `is_integer` () const

True if object is a non-complex integer.

- bool `is_pos_integer` () const

True if object is an exact integer greater than zero.

- bool `is_nonneg_integer` () const

True if object is an exact integer greater or equal zero.

- bool `is_even` () const

True if object is an exact even integer.

- bool `is_odd` () const

True if object is an exact odd integer.

- bool `is_prime` () const

Probabilistic primality test.

- bool `is_rational` () const

True if object is an exact rational number, may even be complex (denominator may be unity).

- bool `is_real` () const

True if object is a real integer, rational or float (but not complex).

- bool `is_cinteger` () const

True if object is element of the domain of integers extended by I, i.e.

- bool `is_crational` () const

True if object is an exact rational number, may even be complex (denominator may be unity).

- bool `operator==` (const `numeric` &other) const
- bool `operator!=` (const `numeric` &other) const
- bool `operator<` (const `numeric` &other) const

Numerical comparison: less.

- bool `operator<=` (const `numeric` &other) const

Numerical comparison: less or equal.

- bool `operator>` (const `numeric` &other) const

Numerical comparison: greater.

- bool `operator>=` (const `numeric` &other) const

- Numerical comparison: greater or equal.*

 - int `to_int` () const

Converts numeric types to machine's int.
- long `to_long` () const

Converts numeric types to machine's long.
- double `to_double` () const

Converts numeric types to machine's double.
- `cln::cl_N to_cl_N` () const

Returns a new CLN object of type `cl_N`, representing the value of `*this`.
- const `numeric real` () const

Real part of a number.
- const `numeric imag` () const

Imaginary part of a number.
- const `numeric numer` () const

Numerator.
- const `numeric denom` () const

Denominator.
- int `int_length` () const

Size in binary notation.
- `numeric` (const `cln::cl_N &z`)

Ctor from CLN types.

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic` ()

basic destructor, virtual because class `ex` will delete objects of derived classes via a `basic*`.
- `basic` (const `basic &other`)
- const `basic & operator=` (const `basic &other`)

basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate` () const

Create a clone of this object on the heap.
- virtual `ex evalm` () const

Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ` () const

Evaluate integrals, if result is known.
- virtual `ex eval_indexed` (const `basic &i`) const

Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void `print` (const `print_context &c`, unsigned level=0) const

Output to stream.
- virtual void `dbgprint` () const

Little wrapper around `print` to be called within a debugger.
- virtual void `dbgprinttree` () const

Little wrapper around `printtree` to be called within a debugger.
- virtual size_t `nops` () const

Number of operands/members.
- virtual `ex op` (size_t i) const

Return operand/member at position `i`.
- virtual `ex operator[]` (const `ex &index`) const
- virtual `ex operator[]` (size_t i) const
- virtual `ex & let_op` (size_t i)

- *Return modifiable operand/member at position i.*
- virtual `ex & operator[]` (const `ex` &index)
- virtual `ex & operator[]` (size_t i)
- virtual bool `match` (const `ex` &pattern, `exmap` &repls) const
- *Check whether the expression matches a given pattern.*
- virtual `ex map` (map_function &f) const
- *Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).*
- virtual void `accept` (GiNaC::visitor &v) const
- virtual `ex expand` (unsigned options=0) const
- *Expand expression, i.e.*
- virtual `ex collect` (const `ex` &s, bool distributed=false) const
- *Sort expanded expression in terms of powers of some object(s).*
- virtual `ex series` (const relational &r, int order, unsigned options=0) const
- *Default implementation of ex::series().*
- virtual `exvector get_free_indices` () const
- *Return a vector containing the free indices of an expression.*
- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
- *Add two indexed expressions.*
- virtual `ex scalar_mul_indexed` (const `ex` &self, const numeric &other) const
- *Multiply an indexed expression with a scalar.*
- virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const
- *Try to contract two indexed expressions that appear in the same product.*
- virtual unsigned `return_type` () const
- virtual `return_type_t return_type_tinfo` () const
- template<class T >
- void `print_dispatch` (const print_context &c, unsigned level) const
- *Like print(), but dispatch to the specified class.*
- void `print_dispatch` (const registered_class_info &ri, const print_context &c, unsigned level) const
- *Like print(), but dispatch to the specified class.*
- `ex subs_one_level` (const exmap &m, unsigned options) const
- *Helper function for subs().*
- `ex diff` (const symbol &s, unsigned nth=1) const
- *Default interface of nth derivative ex::diff(s, n).*
- int `compare` (const basic &other) const
- *Compare objects syntactically to establish canonical ordering.*
- bool `is_equal` (const basic &other) const
- *Test for syntactic equality.*
- const basic & `hold` () const
- *Stop further evaluation.*
- unsigned `gethash` () const
- const basic & `setflag` (unsigned f) const
- *Set some status_flags.*
- const basic & `clearflag` (unsigned f) const
- *Clear some status_flags.*

Public Member Functions inherited from GiNaC::refcounted

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Protected Member Functions

- `ex_derivative` (const `symbol` &s) const override
Implementation of `ex::diff` for a numeric always returns 0.
- bool `is_equal_same_type` (const `basic` &other) const override
Returns true if two objects of same type are equal.
- unsigned `calchash` () const override
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- void `print_numeric` (const `print_context` &c, const char *par_open, const char *par_close, const char *imag←_sym, const char *mul_sym, unsigned level) const
- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_latex` (const `print_latex` &c, unsigned level) const
- void `do_print_csrc` (const `print_csrc` &c, unsigned level) const
- void `do_print_csrc_cl_N` (const `print_csrc_cl_N` &c, unsigned level) const
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const

Protected Member Functions inherited from `GiNaC::basic`

- `basic` ()
- virtual `ex eval_ncmul` (const `exvector` &v) const
- virtual bool `match_same_type` (const `basic` &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual int `compare_same_type` (const `basic` &other) const
Returns order relation between two objects of same type.
- void `ensure_if_modifiable` () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void `do_print` (const `print_context` &c, unsigned level) const
Default output to stream.
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
Tree output to stream.
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
Python parsable output to stream.

Protected Attributes

- `cln::cl_N` `value`

Protected Attributes inherited from `GiNaC::basic`

- unsigned `flags`
of type `status_flags`
- unsigned `hashvalue`
hash value

6.107.1 Detailed Description

This class is a wrapper around CLN-numbers within the `GiNaC` class hierarchy.

Objects of this type may directly be created by the user.

6.107.2 Constructor & Destructor Documentation

6.107.2.1 `numeric()` [1/10]

```
GiNaC::numeric::numeric (
    int i )
```

References [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::expanded](#), [GiNaC::basic::setflag\(\)](#), and [value](#).

6.107.2.2 `numeric()` [2/10]

```
GiNaC::numeric::numeric (
    unsigned int i )
```

References [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::expanded](#), [GiNaC::basic::setflag\(\)](#), and [value](#).

6.107.2.3 `numeric()` [3/10]

```
GiNaC::numeric::numeric (
    long i )
```

References [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::expanded](#), [GiNaC::basic::setflag\(\)](#), and [value](#).

6.107.2.4 `numeric()` [4/10]

```
GiNaC::numeric::numeric (
    unsigned long i )
```

References [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::expanded](#), [GiNaC::basic::setflag\(\)](#), and [value](#).

6.107.2.5 `numeric()` [5/10]

```
GiNaC::numeric::numeric (
    long long i )
```

References [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::expanded](#), [GiNaC::basic::setflag\(\)](#), and [value](#).

6.107.2.6 `numeric()` [6/10]

```
GiNaC::numeric::numeric (
    unsigned long long i )
```

References [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::expanded](#), [GiNaC::basic::setflag\(\)](#), and [value](#).

6.107.2.7 `numeric()` [7/10]

```
GiNaC::numeric::numeric (
    long numer,
    long denom )
```

Constructor for rational numerics a/b.

Exceptions

<code>overflow_error</code>	(division by zero)

References [denom\(\)](#), [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::expanded](#), [numer\(\)](#), [GiNaC::basic::setflag\(\)](#), and [value](#).

6.107.2.8 numeric() [8/10]

```
GiNaC::numeric::numeric (
    double d )
```

References [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::expanded](#), [GiNaC::basic::setflag\(\)](#), and [value](#).

6.107.2.9 numeric() [9/10]

```
GiNaC::numeric::numeric (
    const char * s )
```

ctor from C-style string.

It also accepts complex numbers in [GiNaC](#) notation like "2+5*I".

References [GiNaC::Digits](#), [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::expanded](#), [GiNaC::basic::setflag\(\)](#), and [value](#).

6.107.2.10 numeric() [10/10]

```
GiNaC::numeric::numeric (
    const cln::cl_N & z ) [explicit]
```

Ctor from CLN types.

This is for the initiated user or internal use only.

References [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::expanded](#), [GiNaC::basic::setflag\(\)](#), and [value](#).

6.107.3 Member Function Documentation**6.107.3.1 precedence()**

```
unsigned GiNaC::numeric::precedence ( ) const [inline], [override], [virtual]
```

Return relative operator precedence (for parenthezing output).

Reimplemented from [GiNaC::basic](#).

Referenced by [print_numeric\(\)](#).

6.107.3.2 info()

```
bool GiNaC::numeric::info (
    unsigned int ) const [override], [virtual]
```

Information about the object.

See also

class [info_flags](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::info_flags::cinteger](#), [GiNaC::info_flags::cinteger_polynomial](#), [GiNaC::info_flags::crational](#), [GiNaC::info_flags::crational_polynomial](#), [GiNaC::info_flags::even](#), [GiNaC::info_flags::expanded](#), [GiNaC::info_flags::integer](#), [GiNaC::info_flags::integer_polynomial](#), [is_cinteger\(\)](#), [is_crational\(\)](#), [is_even\(\)](#), [is_integer\(\)](#), [is_negative\(\)](#), [is_nonneg_integer\(\)](#), [is_odd\(\)](#), [is_pos_integer\(\)](#), [is_positive\(\)](#), [is_prime\(\)](#), [is_rational\(\)](#), [is_real\(\)](#), [is_zero\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::negint](#), [GiNaC::info_flags::nonnegative](#), [GiNaC::info_flags::nonnegint](#), [GiNaC::info_flags::numeric](#), [GiNaC::info_flags::odd](#), [GiNaC::info_flags::polynomial](#), [GiNaC::info_flags::posint](#), [GiNaC::info_flags::positive](#), [GiNaC::info_flags::prime](#), [GiNaC::info_flags::rational](#), [GiNaC::info_flags::rational_function](#), [GiNaC::info_flags::rational_polynomial](#), and [GiNaC::info_flags::real](#).

Referenced by [GiNaC::abs_power\(\)](#), [GiNaC::csgn_power\(\)](#), and [GiNaC::zeta1_eval\(\)](#).

6.107.3.3 is_polynomial()

```
bool GiNaC::numeric::is_polynomial (
    const ex & var ) const [override], [virtual]
```

Check whether this is a polynomial in the given variables.

Reimplemented from [GiNaC::basic](#).

6.107.3.4 degree()

```
int GiNaC::numeric::degree (
    const ex & s ) const [override], [virtual]
```

Return degree of highest power in object s.

Reimplemented from [GiNaC::basic](#).

6.107.3.5 ldegree()

```
int GiNaC::numeric::ldegree (
    const ex & s ) const [override], [virtual]
```

Return degree of lowest power in object s.

Reimplemented from [GiNaC::basic](#).

6.107.3.6 `coeff()`

```
ex GiNaC::numeric::coeff (
    const ex & s,
    int n = 1 ) const [override], [virtual]
```

Return coefficient of degree n in object s .

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex0](#), and [n](#).

Referenced by [GiNaC::pseries::mul_const\(\)](#).

6.107.3.7 has()

```
bool GiNaC::numeric::has (
    const ex & other,
    unsigned options = 0 ) const    [override], [virtual]
```

Disassemble real part and imaginary part to scan for the occurrence of a single number.

Also handles the imaginary unit. It ignores the sign on both this and the argument, which may lead to what might appear as funny results: $(2+i).has(-2) \rightarrow \text{true}$. But this is consistent, since we also would like to have $(-2+i).has(2) \rightarrow \text{true}$ and we want to think about the sign as a multiplicative factor.

Reimplemented from [GiNaC::basic](#).

References [GiNaC:: num0_p](#), [GiNaC::l](#), [imag\(\)](#), [is_equal\(\)](#), [is_real\(\)](#), [is_zero\(\)](#), and [real\(\)](#).

6.107.3.8 eval()

```
ex GiNaC::numeric::eval ( ) const [override], [virtual]
```

Evaluation of numbers doesn't do anything at all.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::basic::hold\(\)](#).

6.107.3.9 evalf()

```
ex GiNaC::numeric::evalf ( ) const [override], [virtual]
```

Cast numeric into a floating-point object.

For example `exact.numeric(1)` is returned as a `1.00000000000000000000` and so on according to how Digits is currently set. In case the object already was a floating point number the precision is trimmed to match the currently set default.

Returns

an ex-handle to a numeric.

Reimplemented from [GiNaC::basic](#).

References [value](#).

6.107.3.10 subs()

```
ex GiNaC::numeric::subs (
    const exmap & m,
    unsigned options = 0 ) const [inline], [override], [virtual]
```

Substitute a set of objects by arbitrary expressions.

The ex returned will already be evaluated.

Reimplemented from [GiNaC::basic](#).

References [m](#), [options](#), and [GiNaC::basic::subs_one_level\(\)](#).

6.107.3.11 normal()

```
ex GiNaC::numeric::normal (
    exmap & repl,
    exmap & rev_lookup,
    lst & modifier ) const [override], [virtual]
```

Implementation of [ex::normal\(\)](#) for a numeric.

It splits complex numbers into $re+I*im$ and replaces I and non-rational real numbers with a temporary symbol.

See also

[ex::normal](#)

Reimplemented from [GiNaC::basic](#).

References [denom\(\)](#), [GiNaC::I](#), [imag\(\)](#), [is_integer\(\)](#), [is_rational\(\)](#), [is_real\(\)](#), [numer\(\)](#), [real\(\)](#), and [GiNaC::replace_with_symbol\(\)](#).

6.107.3.12 to_rational()

```
ex GiNaC::numeric::to_rational (
    exmap & repl ) const [override], [virtual]
```

Implementation of [ex::to_rational\(\)](#) for a numeric.

It splits complex numbers into $re+I*im$ and replaces I and non-rational real numbers with a temporary symbol.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::I](#), [imag\(\)](#), [is_rational\(\)](#), [is_real\(\)](#), [real\(\)](#), and [GiNaC::replace_with_symbol\(\)](#).

6.107.3.13 to_polynomial()

```
ex GiNaC::numeric::to_polynomial (
    exmap & repl ) const [override], [virtual]
```

Implementation of [ex::to_polynomial\(\)](#) for a numeric.

It splits complex numbers into $re+I*im$ and replaces I and non-integer real numbers with a temporary symbol.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::I](#), [imag\(\)](#), [is_integer\(\)](#), [is_real\(\)](#), [real\(\)](#), and [GiNaC::replace_with_symbol\(\)](#).

6.107.3.14 integer_content()

```
numeric GiNaC::numeric::integer_content ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::abs\(\)](#).

Referenced by [GiNaC::mul::eval\(\)](#), and [GiNaC::ex::integer_content\(\)](#).

6.107.3.15 smod()

```
ex GiNaC::numeric::smod (
    const numeric & xi ) const [override], [virtual]
```

Apply symmetric modular homomorphism to an expanded multivariate polynomial.

This function is usually used internally by [heur_gcd\(\)](#).

Parameters

xi	modulus

Returns

mapped polynomial

See also

[heur_gcd](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::smod\(\)](#).

6.107.3.16 max_coefficient()

```
numeric GiNaC::numeric::max_coefficient ( ) const [override], [virtual]
```

Implementation [ex::max_coefficient\(\)](#).

See also

[heur_gcd](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::abs\(\)](#).

Referenced by [GiNaC::ex::max_coefficient\(\)](#).

6.107.3.17 conjugate()

```
ex GiNaC::numeric::conjugate ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [is_real\(\)](#), and [value](#).

6.107.3.18 real_part()

```
ex GiNaC::numeric::real_part ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [value](#).

6.107.3.19 imag_part()

```
ex GiNaC::numeric::imag_part ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [value](#).

6.107.3.20 archive()

```
void GiNaC::numeric::archive (
    archive_node & n ) const [override], [virtual]
```

Save (a.k.a.

serialize) object into archive.

Reimplemented from [GiNaC::basic](#).

References [n](#), [value](#), and [GiNaC::write_real_float\(\)](#).

6.107.3.21 read_archive()

```
void GiNaC::numeric::read_archive (
    const archive_node & n,
    lst & syms ) [override], [virtual]
```

Read (a.k.a.

deserialize) object from archive.

Reimplemented from [GiNaC::basic](#).

References [c](#), [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::expanded](#), [n](#), [GiNaC::read_real_float\(\)](#), [GiNaC::basic::setflag\(\)](#), and [value](#).

6.107.3.22 derivative()

```
ex GiNaC::numeric::derivative (
    const symbol & s ) const [inline], [override], [protected], [virtual]
```

Implementation of [ex::diff](#) for a numeric always returns 0.

See also

[ex::diff](#)

Reimplemented from [GiNaC::basic](#).

6.107.3.23 is_equal_same_type()

```
bool GiNaC::numeric::is_equal_same_type (
    const basic & other ) const [override], [protected], [virtual]
```

Returns true if two objects of same type are equal.

Normally needs not be reimplemented as long as it wasn't overwritten by some parent class, since it just calls [compare_same_type\(\)](#). The reason why this function exists is that sometimes it is easier to determine equality than an order relation and then it can be overridden.

Reimplemented from [GiNaC::basic](#).

References [GINAC_ASSERT](#), and [is_equal\(\)](#).

6.107.3.24 calchash()

```
unsigned GiNaC::numeric::calchash ( ) const [override], [protected], [virtual]
```

Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.

The method inherited from class basic computes a hash value based on the type and hash values of possible members. For this reason it is well suited for container classes but atomic classes should override this implementation because otherwise they would all end up with the same hashvalue.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::golden_ratio_hash\(\)](#), [GiNaC::status_flags::hash_calculated](#), [GiNaC::basic::hashvalue](#), [GiNaC::basic::setflag\(\)](#), and [value](#).

6.107.3.25 add()

```
const numeric GiNaC::numeric::add (
    const numeric & other ) const
```

Numerical addition method.

Adds argument to *this and returns result as a numeric object.

References [value](#).

Referenced by [GiNaC::multinomial_coefficient\(\)](#), [GiNaC::operator+\(\)](#), [GiNaC::operator++\(\)](#), [GiNaC::operator++\(\)](#), [GiNaC::operator+=\(\)](#), [GiNaC::operator--\(\)](#), and [GiNaC::operator--\(\)](#).

6.107.3.26 sub()

```
const numeric GiNaC::numeric::sub (
    const numeric & other ) const
```

Numerical subtraction method.

Subtracts argument from *this and returns result as a numeric object.

References [value](#).

Referenced by [GiNaC::operator-\(\)](#), [GiNaC::operator-=\(\)](#), and [GiNaC::tgamma_eval\(\)](#).

6.107.3.27 mul()

```
const numeric GiNaC::numeric::mul (
    const numeric & other ) const
```

Numerical multiplication method.

Multiplies *this and argument and returns result as a numeric object.

References [value](#).

Referenced by [GiNaC::mul::eval\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::mul::expand\(\)](#), [GiNaC::power::expand_add_2\(\)](#), [GiNaC::Gt::findMoebiusTransform\(\)](#), [GiNaC::lcm\(\)](#), [GiNaC::multinomial_coefficient\(\)](#), [GiNaC::operator*\(\)](#), [GiNaC::operator*=\(\)](#), and [GiNaC::operator-\(\)](#).

6.107.3.28 div()

```
const numeric GiNaC::numeric::div (
    const numeric & other ) const
```

Numerical division method.

Divides *this by argument and returns result as a numeric object.

Exceptions

overflow_error	(division by zero)

References [value](#).

Referenced by [GiNaC::mul::eval\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::multinomial_coefficient\(\)](#), [GiNaC::operator/\(\)](#), [GiNaC::operator/=\(\)](#), [GiNaC::basic::series\(\)](#), and [GiNaC::tgamma_eval\(\)](#).

6.107.3.29 power()

```
const numeric GiNaC::numeric::power (
    const numeric & other ) const
```

Numerical exponentiation.

Raises *this to the power given as argument and returns result as a numeric object.

References [GiNaC::_num0_p](#), [GiNaC::_num1_p](#), and [value](#).

Referenced by [GiNaC::binomial\(\)](#), and [GiNaC::power::eval\(\)](#).

6.107.3.30 add_dyn()

```
const numeric & GiNaC::numeric::add_dyn (
    const numeric & other ) const
```

Numerical addition method.

Adds argument to *this and returns result as a numeric object on the heap. Use internally only for direct wrapping into an ex object, where the result would end up on the heap anyways.

References [GiNaC::_num0_p](#), and [value](#).

6.107.3.31 sub_dyn()

```
const numeric & GiNaC::numeric::sub_dyn (
    const numeric & other ) const
```

Numerical subtraction method.

Subtracts argument from *this and returns result as a numeric object on the heap. Use internally only for direct wrapping into an ex object, where the result would end up on the heap anyways.

References [GiNaC::_num0_p](#), and [value](#).

6.107.3.32 mul_dyn()

```
const numeric & GiNaC::numeric::mul_dyn (
    const numeric & other ) const
```

Numerical multiplication method.

Multiplies *this and argument and returns result as a numeric object on the heap. Use internally only for direct wrapping into an ex object, where the result would end up on the heap anyways.

References [GiNaC::_num1_p](#), and [value](#).

Referenced by [GiNaC::power::expand_add_2\(\)](#).

6.107.3.33 div_dyn()

```
const numeric & GiNaC::numeric::div_dyn (
    const numeric & other ) const
```

Numerical division method.

Divides *this by argument and returns result as a numeric object on the heap. Use internally only for direct wrapping into an ex object, where the result would end up on the heap anyways.

Exceptions

<code>overflow_error</code>	(division by zero)

References [GiNaC::_num1_p](#), and [value](#).

6.107.3.34 power_dyn()

```
const numeric & GiNaC::numeric::power_dyn (
    const numeric & other ) const
```

Numerical exponentiation.

Raises *this to the power given as argument and returns result as a numeric object on the heap. Use internally only for direct wrapping into an ex object, where the result would end up on the heap anyways.

References [GiNaC::_num0_p](#), [GiNaC::_num1_p](#), and [value](#).

6.107.3.35 operator=() [1/6]

```
const numeric & GiNaC::numeric::operator= (
    int i )
```

References [operator=\(\)](#).

Referenced by [operator=\(\)](#), [operator=\(\)](#), [operator=\(\)](#), [operator=\(\)](#), [operator=\(\)](#), and [operator=\(\)](#).

6.107.3.36 operator=() [2/6]

```
const numeric & GiNaC::numeric::operator= (
    unsigned int i )
```

References [operator=\(\)](#).

6.107.3.37 operator=() [3/6]

```
const numeric & GiNaC::numeric::operator= (
    long i )
```

References [operator=\(\)](#).

6.107.3.38 operator=() [4/6]

```
const numeric & GiNaC::numeric::operator= (
    unsigned long i )
```

References [operator=\(\)](#).

6.107.3.39 operator=() [5/6]

```
const numeric & GiNaC::numeric::operator= (
    double d )
```

References [operator=\(\)](#).

6.107.3.40 operator=() [6/6]

```
const numeric & GiNaC::numeric::operator= (
    const char * s )
```

References [operator=\(\)](#).

6.107.3.41 inverse()

```
const numeric GiNaC::numeric::inverse ( ) const
```

Inverse of a number.

References [value](#).

Referenced by [GiNaC::divide_in_z\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::heur_gcd_z\(\)](#), [GiNaC::interpolate\(\)](#), and [GiNaC::psi1_eval\(\)](#).

6.107.3.42 step()

```
numeric GiNaC::numeric::step ( ) const
```

Return the step function of a numeric.

The imaginary part of it is ignored because the step function is generally considered real but a numeric may develop a small imaginary part due to rounding errors.

References [r](#), and [value](#).

6.107.3.43 csgn()

```
int GiNaC::numeric::csgn ( ) const
```

Return the complex half-plane (left or right) in which the number lies.

$\text{csgn}(x) == 0$ for $x == 0$, $\text{csgn}(x) == 1$ for $\text{Re}(x) > 0$ or $\text{Re}(x) = 0$ and $\text{Im}(x) > 0$, $\text{csgn}(x) == -1$ for $\text{Re}(x) < 0$ or $\text{Re}(x) = 0$ and $\text{Im}(x) < 0$.

See also

`numeric::compare(const numeric &other)`

References [r](#), and [value](#).

6.107.3.44 compare()

```
int GiNaC::numeric::compare (
    const numeric & other ) const
```

This method establishes a canonical order on all numbers.

For complex numbers this is not possible in a mathematically consistent way but we need to establish some order and it ought to be fast. So we simply define it to be compatible with our method `csgn`.

Returns

`csgn(*this-other)`

See also

[numeric::csgn\(\)](#)

References [value](#).

Referenced by [GiNaC::power::eval\(\)](#).

6.107.3.45 is_equal()

```
bool GiNaC::numeric::is_equal (
    const numeric & other ) const
```

References [value](#).

Referenced by [GiNaC::atan\(\)](#), [GiNaC::cos_eval\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::exp_eval\(\)](#), [has\(\)](#), [is_equal_same_type\(\)](#), [GiNaC::psi2_eval\(\)](#), [GiNaC::sin_eval\(\)](#), [GiNaC::tan_eval\(\)](#), and [GiNaC::zeta1_eval\(\)](#).

6.107.3.46 is_zero()

```
bool GiNaC::numeric::is_zero ( ) const
```

True if object is zero.

References [value](#).

Referenced by [GiNaC::asinh_conjugate\(\)](#), [GiNaC::atan\(\)](#), [GiNaC::atan_conjugate\(\)](#), [GiNaC::expairseq::construct_from_2_expairseq\(\)](#), [GiNaC::expairseq::construct_from_expairseq_ex\(\)](#), [GiNaC::csgn_eval\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::fsolve\(\)](#), [GiNaC::gcd\(\)](#), [has\(\)](#), [info\(\)](#), [GiNaC::iquo\(\)](#), [GiNaC::iquo\(\)](#), [GiNaC::irem\(\)](#), [GiNaC::irem\(\)](#), [GiNaC::matrix::pivot\(\)](#), [GiNaC::matrix::pow\(\)](#), [GiNaC::pseries::power_const\(\)](#), [GiNaC::step_eval\(\)](#), [GiNaC::Gt::tauToFundamental\(\)](#), and [GiNaC::zeta1_eval\(\)](#).

6.107.3.47 is_positive()

```
bool GiNaC::numeric::is_positive ( ) const
```

True if object is not complex and greater than zero.

References [value](#).

Referenced by [GiNaC::power::eval\(\)](#), [GiNaC::Gt::findMoebiusTransform\(\)](#), [info\(\)](#), [GiNaC::psi1_eval\(\)](#), [GiNaC::psi2_eval\(\)](#), [GiNaC::Gt::qExpand\(\)](#), and [GiNaC::tgamma_eval\(\)](#).

6.107.3.48 is_negative()

```
bool GiNaC::numeric::is_negative ( ) const
```

True if object is not complex and less than zero.

References [value](#).

Referenced by [GiNaC::bernoulli\(\)](#), [GiNaC::beta_eval\(\)](#), [GiNaC::eta_eval\(\)](#), [GiNaC::eta_evalf\(\)](#), [info\(\)](#), [GiNaC::pseries::power_const\(\)](#), and [GiNaC::Gt::qExpand\(\)](#).

6.107.3.49 is_integer()

```
bool GiNaC::numeric::is_integer ( ) const
```

True if object is a non-complex integer.

References [value](#).

Referenced by [GiNaC::bernoulli\(\)](#), [GiNaC::beta_eval\(\)](#), [GiNaC::binomial_sym\(\)](#), [GiNaC::Gt::decomposeIntegrationPath\(\)](#), [GiNaC::mul::eval\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::gcd\(\)](#), [info\(\)](#), [GiNaC::iquo\(\)](#), [GiNaC::iquo\(\)](#), [GiNaC::irem\(\)](#), [GiNaC::irem\(\)](#), [GiNaC::lcm\(\)](#), [GiNaC::mod\(\)](#), [normal\(\)](#), [GiNaC::psi1_eval\(\)](#), [GiNaC::psi2_eval\(\)](#), [GiNaC::smod\(\)](#), [GiNaC::tgamma_eval\(\)](#), [to_int\(\)](#), [to_long\(\)](#), [to_polynomial\(\)](#), and [GiNaC::zeta1_eval\(\)](#).

6.107.3.50 is_pos_integer()

```
bool GiNaC::numeric::is_pos_integer ( ) const
```

True if object is an exact integer greater than zero.

References [value](#).

Referenced by [GiNaC::mul::eval\(\)](#), [GiNaC::power::eval\(\)](#), [info\(\)](#), and [GiNaC::pseries::power_const\(\)](#).

6.107.3.51 is_nonneg_integer()

```
bool GiNaC::numeric::is_nonneg_integer ( ) const
```

True if object is an exact integer greater or equal zero.

References [value](#).

Referenced by [GiNaC::binomial_sym\(\)](#), [info\(\)](#), and [print_numeric\(\)](#).

6.107.3.52 is_even()

```
bool GiNaC::numeric::is_even ( ) const
```

True if object is an exact even integer.

References [value](#).

Referenced by [info\(\)](#), [GiNaC::kronecker_symbol\(\)](#), and [GiNaC::tgamma_eval\(\)](#).

6.107.3.53 is_odd()

```
bool GiNaC::numeric::is_odd ( ) const
```

True if object is an exact odd integer.

References [value](#).

Referenced by [info\(\)](#), [GiNaC::is_discriminant_of_quadratic_number_field\(\)](#), and [GiNaC::matrix::pow\(\)](#).

6.107.3.54 is_prime()

```
bool GiNaC::numeric::is_prime ( ) const
```

Probabilistic primality test.

Returns

true if object is exact integer and prime.

References [value](#).

Referenced by [info\(\)](#).

6.107.3.55 is_rational()

```
bool GiNaC::numeric::is_rational ( ) const
```

True if object is an exact rational number, may even be complex (denominator may be unity).

References [value](#).

Referenced by [GiNaC::power::eval\(\)](#), [info\(\)](#), [GiNaC::multiply_lcm\(\)](#), [normal\(\)](#), and [to_rational\(\)](#).

6.107.3.56 is_real()

```
bool GiNaC::numeric::is_real ( ) const
```

True if object is a real integer, rational or float (but not complex).

References [value](#).

Referenced by [GiNaC::atan\(\)](#), [GiNaC::beta_eval\(\)](#), [conjugate\(\)](#), [GiNaC::csgn_eval\(\)](#), [denom\(\)](#), [do_print_csrc\(\)](#), [do_print_csrc_cl_N\(\)](#), [GiNaC::eta_eval\(\)](#), [GiNaC::eta_evalf\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::fsolve\(\)](#), [has\(\)](#), [info\(\)](#), [is_cinteger\(\)](#), [is_crational\(\)](#), [normal\(\)](#), [numer\(\)](#), [operator<\(\)](#), [operator<=\(\)](#), [operator>\(\)](#), [operator>=\(\)](#), [GiNaC::step_eval\(\)](#), [to_double\(\)](#), [to_polynomial\(\)](#), and [to_rational\(\)](#).

6.107.3.57 is_cinteger()

```
bool GiNaC::numeric::is_cinteger ( ) const
```

True if object is element of the domain of integers extended by i , i.e.

is of the form $a+bi$, where a and b are integers.

References [is_real\(\)](#), and [value](#).

Referenced by [info\(\)](#).

6.107.3.58 is_crational()

```
bool GiNaC::numeric::is_crational ( ) const
```

True if object is an exact rational number, may even be complex (denominator may be unity).

References [is_real\(\)](#), and [value](#).

Referenced by [GiNaC::power::eval\(\)](#), and [info\(\)](#).

6.107.3.59 operator==()

```
bool GiNaC::numeric::operator== (
    const numeric & other ) const
```

References [value](#).

6.107.3.60 operator"!="()

```
bool GiNaC::numeric::operator!= (
    const numeric & other ) const
```

References [value](#).

6.107.3.61 operator<()

```
bool GiNaC::numeric::operator< (
    const numeric & other ) const
```

Numerical comparison: less.

Exceptions

<i>invalid_argument</i>	(complex inequality)

References [is_real\(\)](#), and [value](#).

6.107.3.62 operator<=()

```
bool GiNaC::numeric::operator<= (
    const numeric & other ) const
```

Numerical comparison: less or equal.

Exceptions

<i>invalid_argument</i>	(complex inequality)

References [is_real\(\)](#), and [value](#).

6.107.3.63 operator>()

```
bool GiNaC::numeric::operator> (
    const numeric & other ) const
```

Numerical comparison: greater.

Exceptions

<i>invalid_argument</i>	(complex inequality)

References [is_real\(\)](#), and [value](#).

6.107.3.64 operator>=()

```
bool GiNaC::numeric::operator>= (
    const numeric & other ) const
```

Numerical comparison: greater or equal.

Exceptions

<i>invalid_argument</i>	(complex inequality)

References [is_real\(\)](#), and [value](#).

6.107.3.65 to_int()

```
int GiNaC::numeric::to_int ( ) const
```

Converts numeric types to machine's int.

You should check with [is_integer\(\)](#) if the number is really an integer before calling this method. You may also consider checking the range first.

References [GINAC_ASSERT](#), [is_integer\(\)](#), and [value](#).

Referenced by [GiNaC::bernoulli\(\)](#), [GiNaC::binomial_sym\(\)](#), [GiNaC::power::do_print_csrc\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::power::expand\(\)](#), [GiNaC::generalised_Bernoulli_number\(\)](#), [GiNaC::Li_eval\(\)](#), and [GiNaC::Kronecker_dtau_kernel::series_coe](#)

6.107.3.66 to_long()

```
long GiNaC::numeric::to_long ( ) const
```

Converts numeric types to machine's long.

You should check with [is_integer\(\)](#) if the number is really an integer before calling this method. You may also consider checking the range first.

References [GINAC_ASSERT](#), [is_integer\(\)](#), and [value](#).

Referenced by [GiNaC::power::expand\(\)](#), and [GiNaC::power::expand_add\(\)](#).

6.107.3.67 to_double()

```
double GiNaC::numeric::to_double ( ) const
```

Converts numeric types to machine's double.

You should check with [is_real\(\)](#) if the number is really not complex before calling this method.

References [GINAC_ASSERT](#), [is_real\(\)](#), and [value](#).

Referenced by [GiNaC::Gt::decomposeIntegrationPath\(\)](#), and [GiNaC::Gt::qExpand\(\)](#).

6.107.3.68 to_cl_N()

```
cln::cl_N GiNaC::numeric::to_cl_N ( ) const
```

Returns a new CLN object of type `cl_N`, representing the value of `*this`.

This method may be used when mixing [GiNaC](#) and CLN in one project.

References [value](#).

Referenced by [GiNaC::atan\(\)](#), [GiNaC::Gt::findMoebiusTransform\(\)](#), [GiNaC::gcd\(\)](#), [GiNaC::iquo\(\)](#), [GiNaC::iquo\(\)](#), [GiNaC::irem\(\)](#), [GiNaC::irem\(\)](#), [GiNaC::lcm\(\)](#), [GiNaC::mod\(\)](#), [GiNaC::Kronecker_dtau_kernel::series_coeff_impl\(\)](#), and [GiNaC::smod\(\)](#).

6.107.3.69 real()

```
const numeric GiNaC::numeric::real ( ) const
```

Real part of a number.

References [value](#).

Referenced by [GiNaC::csgn_eval\(\)](#), [GiNaC::Gt::decomposeIntegrationPath\(\)](#), [GiNaC::power::eval\(\)](#), [has\(\)](#), [normal\(\)](#), [GiNaC::pseries::power_const\(\)](#), [GiNaC::Gt::qExpand\(\)](#), [GiNaC::step_eval\(\)](#), [to_polynomial\(\)](#), [to_rational\(\)](#), and [GiNaC::Gt::zisToFundamental\(\)](#).

6.107.3.70 imag()

```
const numeric GiNaC::numeric::imag ( ) const
```

Imaginary part of a number.

References [value](#).

Referenced by [GiNaC::csgn_eval\(\)](#), [GiNaC::Gt::decomposeIntegrationPath\(\)](#), [GiNaC::Gt::findMoebiusTransform\(\)](#), [has\(\)](#), [normal\(\)](#), [GiNaC::Gt::qExpand\(\)](#), [GiNaC::step_eval\(\)](#), [to_polynomial\(\)](#), [to_rational\(\)](#), and [GiNaC::Gt::zisToFundamental\(\)](#).

6.107.3.71 numer()

```
const numeric GiNaC::numeric::numer ( ) const
```

Numerator.

Computes the numerator of rational numbers, rationalized numerator of complex if real and imaginary part are both rational numbers (i.e `numer(4/3+5/6*I) == 8+5*I`), the number carrying the sign in all other cases.

References [is_real\(\)](#), [r](#), and [value](#).

Referenced by [GiNaC::power::eval\(\)](#), [GiNaC::frac_cancel\(\)](#), [normal\(\)](#), and [numeric\(\)](#).

6.107.3.72 denom()

```
const numeric GiNaC::numeric::denom ( ) const
```

Denominator.

Computes the denominator of rational numbers, common integer denominator of complex if real and imaginary part are both rational numbers (i.e `denom(4/3+5/6*I) == 6`), one in all other cases.

References [GiNaC::_num1_p](#), [is_real\(\)](#), [r](#), and [value](#).

Referenced by [GiNaC::power::eval\(\)](#), [GiNaC::frac_cancel\(\)](#), [normal\(\)](#), and [numeric\(\)](#).

6.107.3.73 int_length()

```
int GiNaC::numeric::int_length ( ) const
```

Size in binary notation.

For integers, this is the smallest $n \geq 0$ such that $-2^n \leq x < 2^n$. If $x > 0$, this is the unique $n > 0$ such that $2^{n-1} \leq x < 2^n$.

Returns

number of bits (excluding sign) needed to represent that number in two's complement if it is an integer, 0 otherwise.

References [value](#).

Referenced by [GiNaC::heur_gcd_z\(\)](#).

6.107.3.74 print_numeric()

```
void GiNaC::numeric::print_numeric (
    const print\_context & c,
    const char * par_open,
    const char * par_close,
    const char * imag_sym,
    const char * mul_sym,
    unsigned level ) const [protected]
```

References [c](#), [is_nonneg_integer\(\)](#), [precedence\(\)](#), [GiNaC::print_real_number\(\)](#), [r](#), [GiNaC::print_context::s](#), and [value](#).

Referenced by [do_print\(\)](#), [do_print_latex\(\)](#), and [do_print_python_repr\(\)](#).

6.107.3.75 do_print()

```
void GiNaC::numeric::do_print (
    const print\_context & c,
    unsigned level ) const [protected]
```

References [c](#), and [print_numeric\(\)](#).

6.107.3.76 do_print_latex()

```
void GiNaC::numeric::do_print_latex (
    const print\_latex & c,
    unsigned level ) const [protected]
```

References [c](#), and [print_numeric\(\)](#).

6.107.3.77 do_print_csrc()

```
void GiNaC::numeric::do_print_csrc (
    const print\_csrc & c,
    unsigned level ) const [protected]
```

References [c](#), [is_real\(\)](#), [GiNaC::print_real_csrc\(\)](#), and [value](#).

6.107.3.78 do_print_csrc_cl_N()

```
void GiNaC::numeric::do_print_csrc_cl_N (
    const print\_csrc\_cl\_N & c,
    unsigned level ) const [protected]
```

References [c](#), [is_real\(\)](#), [GiNaC::print_real_cl_N\(\)](#), and [value](#).

6.107.3.79 do_print_tree()

```
void GiNaC::numeric::do_print_tree (
    const print\_tree & c,
    unsigned level ) const [protected]
```

References [c](#), [GiNaC::basic::flags](#), [GiNaC::basic::hashvalue](#), and [value](#).

6.107.3.80 do_print_python_repr()

```
void GiNaC::numeric::do_print_python_repr (
    const print\_python\_repr & c,
    unsigned level ) const [protected]
```

References [c](#), and [print_numeric\(\)](#).

6.107.4 Member Data Documentation

6.107.4.1 value

```
cln::cl_N GiNaC::numeric::value [protected]
```

Referenced by [add\(\)](#), [add_dyn\(\)](#), [archive\(\)](#), [calchash\(\)](#), [compare\(\)](#), [conjugate\(\)](#), [csgn\(\)](#), [denom\(\)](#), [div\(\)](#), [div_dyn\(\)](#), [do_print_csrc\(\)](#), [do_print_csrc_cl_N\(\)](#), [do_print_tree\(\)](#), [evalf\(\)](#), [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT\(\)](#), [imag\(\)](#), [imag_part\(\)](#), [int_length\(\)](#), [inverse\(\)](#), [is_cinteger\(\)](#), [is_crational\(\)](#), [is_equal\(\)](#), [is_even\(\)](#), [is_integer\(\)](#), [is_negative\(\)](#), [is_nonneg_integer\(\)](#), [is_odd\(\)](#), [is_pos_integer\(\)](#), [is_positive\(\)](#), [is_prime\(\)](#), [is_rational\(\)](#), [is_real\(\)](#), [is_zero\(\)](#), [mul\(\)](#), [mul_dyn\(\)](#), [numer\(\)](#), [numeric\(\)](#), [numeric\(\)](#), [numeric\(\)](#), [numeric\(\)](#), [numeric\(\)](#), [numeric\(\)](#), [numeric\(\)](#), [numeric\(\)](#), [numeric\(\)](#), [operator!=\(\)](#), [operator<\(\)](#), [operator<=\(\)](#), [operator==\(\)](#), [operator>\(\)](#), [operator>=\(\)](#), [power\(\)](#), [power_dyn\(\)](#), [print_numeric\(\)](#), [read_archive\(\)](#), [real\(\)](#), [real_part\(\)](#), [step\(\)](#), [sub\(\)](#), [sub_dyn\(\)](#), [to_cl_N\(\)](#), [to_double\(\)](#), [to_int\(\)](#), and [to_long\(\)](#).

The documentation for this class was generated from the following files:

- [numeric.h](#)
- [normal.cpp](#)
- [numeric.cpp](#)

6.108 GiNaC::op0_is_equal Struct Reference

```
#include <ex.h>
```

Public Member Functions

- bool [operator\(\)](#) (const [ex](#) &lh, const [ex](#) &rh) const

6.108.1 Member Function Documentation

6.108.1.1 operator>()()

```
bool GiNaC::op0_is_equal::operator() (
    const ex & lh,
    const ex & rh ) const [inline]
```

References [GiNaC::ex::is_equal\(\)](#), and [GiNaC::ex::op\(\)](#).

The documentation for this struct was generated from the following file:

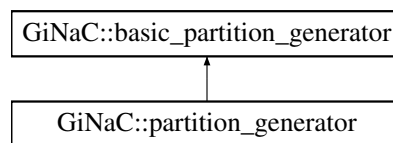
- [ex.h](#)

6.109 GiNaC::partition_generator Class Reference

Generate all bounded combinatorial partitions of an integer n with exactly m parts (not including zero parts) in non-decreasing order.

```
#include <utils.h>
```

Inheritance diagram for GiNaC::partition_generator:



Public Member Functions

- [partition_generator](#) (unsigned n_, unsigned m_)
- const std::vector< unsigned > & [get](#) () const
- bool [next](#) ()

Private Attributes

- std::vector< unsigned > [partition](#)
- bool [current_updated](#)

Additional Inherited Members

Protected Member Functions inherited from [GiNaC::basic_partition_generator](#)

- [basic_partition_generator](#) (unsigned n_, unsigned m_)

Protected Attributes inherited from [GiNaC::basic_partition_generator](#)

- [mpartition2](#) [mpgen](#)

6.109.1 Detailed Description

Generate all bounded combinatorial partitions of an integer n with exactly m parts (not including zero parts) in non-decreasing order.

6.109.2 Constructor & Destructor Documentation

6.109.2.1 [partition_generator\(\)](#)

```
GiNaC::partition_generator::partition_generator (
    unsigned n_,
    unsigned m_ ) [inline]
```

6.109.3 Member Function Documentation

6.109.3.1 [get\(\)](#)

```
const std::vector< unsigned > & GiNaC::partition_generator::get ( ) const [inline]
```

References [current_updated](#), [GiNaC::basic_partition_generator::mpartition2::m](#), [GiNaC::basic_partition_generator::mpgen](#), [partition](#), and [GiNaC::basic_partition_generator::mpartition2::x](#).

6.109.3.2 [next\(\)](#)

```
bool GiNaC::partition_generator::next ( ) [inline]
```

References [current_updated](#), [GiNaC::basic_partition_generator::mpgen](#), and [GiNaC::basic_partition_generator::mpartition2::next_pa](#)

6.109.4 Member Data Documentation

6.109.4.1 [partition](#)

```
std::vector<unsigned> GiNaC::partition_generator::partition [mutable], [private]
```

Referenced by [get\(\)](#).

6.109.4.2 [current_updated](#)

```
bool GiNaC::partition_generator::current_updated [mutable], [private]
```

Referenced by [get\(\)](#), and [next\(\)](#).

The documentation for this class was generated from the following file:

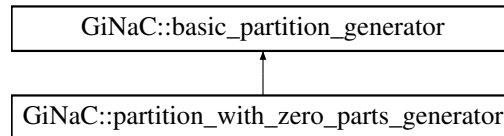
- [utils.h](#)

6.110 GiNaC::partition_with_zero_parts_generator Class Reference

Generate all bounded combinatorial partitions of an integer n with exactly m parts (including zero parts) in non-decreasing order.

```
#include <utils.h>
```

Inheritance diagram for GiNaC::partition_with_zero_parts_generator:



Public Member Functions

- [partition_with_zero_parts_generator](#) (unsigned $n_$, unsigned $m_$)
- const std::vector< unsigned > & [get](#) () const
- bool [next](#) ()

Private Attributes

- unsigned [m](#)
- std::vector< unsigned > [partition](#)
- bool [current_updated](#)

Additional Inherited Members

Protected Member Functions inherited from [GiNaC::basic_partition_generator](#)

- [basic_partition_generator](#) (unsigned $n_$, unsigned $m_$)

Protected Attributes inherited from [GiNaC::basic_partition_generator](#)

- [mpartition2](#) [mpgen](#)

6.110.1 Detailed Description

Generate all bounded combinatorial partitions of an integer n with exactly m parts (including zero parts) in non-decreasing order.

6.110.2 Constructor & Destructor Documentation

6.110.2.1 partition_with_zero_parts_generator()

```

GiNaC::partition_with_zero_parts_generator::partition_with_zero_parts_generator (
    unsigned  $n_$ ,
    unsigned  $m_$  ) [inline]
  
```

6.110.3 Member Function Documentation

6.110.3.1 `get()`

```
const std::vector< unsigned > & GiNaC::partition_with_zero_parts_generator::get ( ) const
[inline]
```

References [current_updated](#), [GiNaC::basic_partition_generator::mpartition2::m](#), [m](#), [GiNaC::basic_partition_generator::mpgen](#), [partition](#), and [GiNaC::basic_partition_generator::mpartition2::x](#).

Referenced by [GiNaC::power::expand_add\(\)](#).

6.110.3.2 `next()`

```
bool GiNaC::partition_with_zero_parts_generator::next ( ) [inline]
```

References [current_updated](#), [GiNaC::basic_partition_generator::mpartition2::m](#), [m](#), [GiNaC::basic_partition_generator::mpgen](#), [GiNaC::basic_partition_generator::mpartition2::n](#), and [GiNaC::basic_partition_generator::mpartition2::next_partition\(\)](#).

Referenced by [GiNaC::power::expand_add\(\)](#).

6.110.4 Member Data Documentation

6.110.4.1 `m`

```
unsigned GiNaC::partition_with_zero_parts_generator::m [private]
```

Referenced by [get\(\)](#), and [next\(\)](#).

6.110.4.2 `partition`

```
std::vector<unsigned> GiNaC::partition_with_zero_parts_generator::partition [mutable], [private]
```

Referenced by [get\(\)](#).

6.110.4.3 `current_updated`

```
bool GiNaC::partition_with_zero_parts_generator::current_updated [mutable], [private]
```

Referenced by [get\(\)](#), and [next\(\)](#).

The documentation for this class was generated from the following file:

- [utils.h](#)

6.111 GiNaC::Gt_detail::pathintegral_term Class Reference

```
#include <Gt_helpers.h>
```

Public Member Functions

- [pathintegral_term](#) ([ex](#) *prefactor*, int *power*, const std::vector< [ex](#) > &*polylog*={}, [ex](#) *denom*=0)
- [pathintegral_term](#) (const [pathintegral_term](#) &*a*, const [pathintegral_term](#) &*b*, int *power_offset*=0)
- void [integrate](#) (const [ex](#) &*start*, [pathintegral_term_list](#) &*integrand*, [pathintegral_term_list](#) &*result*) const
- [ex](#) [evaluate](#) (const [ex](#) &*upper_bound*, const std::vector< [ex](#) > &*path*) const

Static Private Member Functions

- static [ex](#) [G_path](#) (const std::vector< [ex](#) > &*args*, const std::vector< [ex](#) > &*path*)
- static [ex](#) [G_path](#) (const std::vector< [ex](#) > &*args*, const [ex](#) &*start*, const [ex](#) &*end*)

Private Attributes

- [ex](#) *prefactor*
- int *power*
- std::vector< [ex](#) > *polylog*
- [ex](#) *denom*

Friends

- class [pathintegral_term_list](#)
- std::ostream & [operator<<](#) (std::ostream &*os*, const [pathintegral_term](#) &*term*)
- bool [operator<](#) (const [pathintegral_term](#) &*lh*, const [pathintegral_term](#) &*rh*)

6.111.1 Constructor & Destructor Documentation

6.111.1.1 [pathintegral_term](#)() [1/2]

```
GiNaC::Gt_detail::pathintegral_term::pathintegral_term (
    ex prefactor,
    int power,
    const std::vector< ex > & polylog = {},
    ex denom = 0 ) [inline]
```

6.111.1.2 [pathintegral_term](#)() [2/2]

```
GiNaC::Gt_detail::pathintegral_term::pathintegral_term (
    const pathintegral\_term & a,
    const pathintegral\_term & b,
    int power_offset = 0 )
```

References [denom](#), [GiNaC::ex::is_zero\(\)](#), and [polylog](#).

6.111.2 Member Function Documentation

6.111.2.1 `integrate()`

```
void GiNaC::Gt_detail::pathintegral_term::integrate (
    const ex & start,
    pathintegral_term_list & integrand,
    pathintegral_term_list & result ) const
```

References [GiNaC::Gt_detail::pathintegral_term_list::add\(\)](#), [denom](#), [GiNaC::ex::is_zero\(\)](#), [polylog](#), [GiNaC::pow\(\)](#), and [prefactor](#).

6.111.2.2 `evaluate()`

```
ex GiNaC::Gt_detail::pathintegral_term::evaluate (
    const ex & upper_bound,
    const std::vector< ex > & path ) const
```

References [denom](#), [G_path\(\)](#), [GiNaC::ex::is_zero\(\)](#), [polylog](#), [GiNaC::pow\(\)](#), and [prefactor](#).

6.111.2.3 `G_path()` [1/2]

```
ex GiNaC::Gt_detail::pathintegral_term::G_path (
    const std::vector< ex > & args,
    const std::vector< ex > & path ) [static], [private]
```

References [GiNaC::Gt_detail::deconcatenate_path< *ex* >\(\)](#), and [G_path\(\)](#).

Referenced by [evaluate\(\)](#), and [G_path\(\)](#).

6.111.2.4 `G_path()` [2/2]

```
ex GiNaC::Gt_detail::pathintegral_term::G_path (
    const std::vector< ex > & args,
    const ex & start,
    const ex & end ) [static], [private]
```

References [GiNaC::container< *C* >::append\(\)](#), [GiNaC::factorial\(\)](#), [GiNaC::G\(\)](#), [GiNaC::basic::hold\(\)](#), and [GiNaC::pow\(\)](#).

6.111.3 Friends And Related Symbol Documentation

6.111.3.1 `pathintegral_term_list`

```
friend class pathintegral_term_list [friend]
```

6.111.3.2 operator<<

```
std::ostream & operator<< (
    std::ostream & os,
    const pathintegral_term & term ) [friend]
```

6.111.3.3 operator<

```
bool operator< (
    const pathintegral_term & lh,
    const pathintegral_term & rh ) [friend]
```

6.111.4 Member Data Documentation

6.111.4.1 prefactor

```
ex GiNaC::Gt_detail::pathintegral_term::prefactor [mutable], [private]
```

Referenced by [evaluate\(\)](#), and [integrate\(\)](#).

6.111.4.2 power

```
int GiNaC::Gt_detail::pathintegral_term::power [private]
```

6.111.4.3 polylog

```
std::vector<ex> GiNaC::Gt_detail::pathintegral_term::polylog [private]
```

Referenced by [evaluate\(\)](#), [integrate\(\)](#), and [pathintegral_term\(\)](#).

6.111.4.4 denom

```
ex GiNaC::Gt_detail::pathintegral_term::denom [private]
```

Referenced by [evaluate\(\)](#), [integrate\(\)](#), and [pathintegral_term\(\)](#).

The documentation for this class was generated from the following files:

- [Gt_helpers.h](#)
- [Gt_helpers.cpp](#)

6.112 GiNaC::Gt_detail::pathintegral_term_list Class Reference

```
#include <Gt_helpers.h>
```

Public Member Functions

- `std::set< pathintegral\_term >::const_iterator begin () const`
- `std::set< pathintegral\_term >::const_iterator end () const`
- `void clear ()`
- `bool empty () const`
- `size_t size () const`
- `void add (pathintegral\_term &&term)`
- `template<typename... Args, typename = typename std::enable_if<(sizeof...(Args)>1)>::type>
void add (Args &&...args)`
- `void add (const pathintegral\_term\_list &other)`
- `pathintegral\_term pop ()`

Private Attributes

- `std::set< pathintegral\_term > terms`

6.112.1 Member Function Documentation

6.112.1.1 [begin\(\)](#)

```
std::set< pathintegral\_term >::const_iterator GiNaC::Gt_detail::pathintegral_term_list::begin  
( ) const [inline]
```

References [terms](#).

Referenced by [pop\(\)](#).

6.112.1.2 [end\(\)](#)

```
std::set< pathintegral\_term >::const_iterator GiNaC::Gt_detail::pathintegral_term_list::end (  
) const [inline]
```

References [terms](#).

Referenced by [add\(\)](#).

6.112.1.3 [clear\(\)](#)

```
void GiNaC::Gt_detail::pathintegral_term_list::clear ( ) [inline]
```

References [terms](#).

6.112.1.4 [empty\(\)](#)

```
bool GiNaC::Gt_detail::pathintegral_term_list::empty ( ) const [inline]
```

References [terms](#).

6.112.1.5 size()

```
size_t GiNaC::Gt_detail::pathintegral_term_list::size ( ) const [inline]
```

References [terms](#).

6.112.1.6 add() [1/3]

```
void GiNaC::Gt_detail::pathintegral_term_list::add (
    pathintegral_term && term )
```

References [end\(\)](#), and [terms](#).

Referenced by [GiNaC::Gt_detail::pathintegral_term::integrate\(\)](#).

6.112.1.7 add() [2/3]

```
template<typename... Args, typename = typename std::enable_if<(sizeof...(Args)>1)>::type>
void GiNaC::Gt_detail::pathintegral_term_list::add (
    Args &&... args ) [inline]
```

6.112.1.8 add() [3/3]

```
void GiNaC::Gt_detail::pathintegral_term_list::add (
    const pathintegral_term_list & other )
```

6.112.1.9 pop()

```
pathintegral_term GiNaC::Gt_detail::pathintegral_term_list::pop ( )
```

References [begin\(\)](#), and [terms](#).

6.112.2 Member Data Documentation

6.112.2.1 terms

```
std::set<pathintegral_term> GiNaC::Gt_detail::pathintegral_term_list::terms [private]
```

Referenced by [add\(\)](#), [begin\(\)](#), [clear\(\)](#), [empty\(\)](#), [end\(\)](#), [pop\(\)](#), and [size\(\)](#).

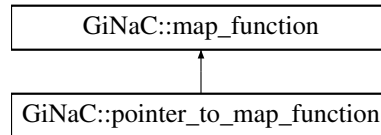
The documentation for this class was generated from the following files:

- [Gt_helpers.h](#)
- [Gt_helpers.cpp](#)

6.113 GiNaC::pointer_to_map_function Class Reference

```
#include <ex.h>
```

Inheritance diagram for GiNaC::pointer_to_map_function:



Public Member Functions

- [pointer_to_map_function](#) ([ex](#) x(const [ex](#) &))
- [ex operator\(\)](#) (const [ex](#) &e) override

Public Member Functions inherited from [GiNaC::map_function](#)

- virtual [~map_function](#) ()

Protected Attributes

- [ex](#)(* [ptr](#))(const [ex](#) &)

Additional Inherited Members

Public Types inherited from [GiNaC::map_function](#)

- typedef const [ex](#) & [argument_type](#)
- typedef [ex](#) [result_type](#)

6.113.1 Constructor & Destructor Documentation

6.113.1.1 [pointer_to_map_function\(\)](#)

```
GiNaC::pointer_to_map_function::pointer_to_map_function (
    ex xconst ex & ) [inline], [explicit]
```

6.113.2 Member Function Documentation

6.113.2.1 [operator\(\)](#)()

```
ex GiNaC::pointer_to_map_function::operator() (
    const ex & e ) [inline], [override], [virtual]
```

Implements [GiNaC::map_function](#).

References [ptr](#).

6.113.3 Member Data Documentation

6.113.3.1 ptr

`ex(* GiNaC::pointer_to_map_function::ptr) (const ex &) [protected]`

Referenced by `operator()()`.

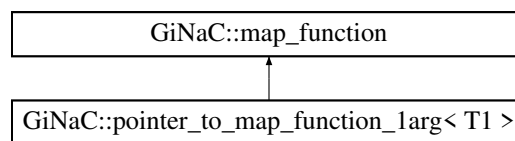
The documentation for this class was generated from the following file:

- `ex.h`

6.114 GiNaC::pointer_to_map_function_1arg< T1 > Class Template Reference

```
#include <ex.h>
```

Inheritance diagram for GiNaC::pointer_to_map_function_1arg< T1 >:



Public Member Functions

- `pointer_to_map_function_1arg` (`ex` x(const `ex` &, T1), T1 a1)
- `ex operator()` (const `ex` &e) override

Public Member Functions inherited from `GiNaC::map_function`

- virtual `~map_function` ()

Protected Attributes

- `ex(* ptr)` (const `ex` &, T1)
- T1 `arg1`

Additional Inherited Members

Public Types inherited from `GiNaC::map_function`

- typedef const `ex` & `argument_type`
- typedef `ex` `result_type`

6.114.1 Constructor & Destructor Documentation

6.114.1.1 `pointer_to_map_function_1arg()`

```
template<class T1 >
GiNaC::pointer_to_map_function_1arg< T1 >::pointer_to_map_function_1arg (
    ex xconst ex &, T1,
    T1 a1 ) [inline], [explicit]
```

6.114.2 Member Function Documentation

6.114.2.1 `operator>()`

```
template<class T1 >
ex GiNaC::pointer_to_map_function_1arg< T1 >::operator() (
    const ex & e ) [inline], [override], [virtual]
```

Implements [GiNaC::map_function](#).

References [GiNaC::pointer_to_map_function_1arg< T1 >::arg1](#), and [GiNaC::pointer_to_map_function_1arg< T1 >::ptr](#).

6.114.3 Member Data Documentation

6.114.3.1 `ptr`

```
template<class T1 >
ex(* GiNaC::pointer_to_map_function_1arg< T1 >::ptr) (const ex &, T1) [protected]
```

Referenced by [GiNaC::pointer_to_map_function_1arg< T1 >::operator>\(\)](#).

6.114.3.2 `arg1`

```
template<class T1 >
T1 GiNaC::pointer_to_map_function_1arg< T1 >::arg1 [protected]
```

Referenced by [GiNaC::pointer_to_map_function_1arg< T1 >::operator>\(\)](#).

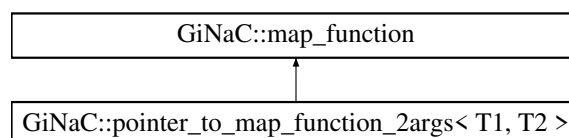
The documentation for this class was generated from the following file:

- [ex.h](#)

6.115 `GiNaC::pointer_to_map_function_2args< T1, T2 >` Class Template Reference

```
#include <ex.h>
```

Inheritance diagram for `GiNaC::pointer_to_map_function_2args< T1, T2 >`:



Public Member Functions

- [pointer_to_map_function_2args](#) ([ex](#) x(const [ex](#) &, T1, T2), T1 a1, T2 a2)
- [ex operator\(\)](#) (const [ex](#) &e) override

Public Member Functions inherited from [GiNaC::map_function](#)

- virtual [~map_function](#) ()

Protected Attributes

- [ex](#)(* [ptr](#))(const [ex](#) &, T1, T2)
- T1 [arg1](#)
- T2 [arg2](#)

Additional Inherited Members

Public Types inherited from [GiNaC::map_function](#)

- typedef const [ex](#) & [argument_type](#)
- typedef [ex](#) [result_type](#)

6.115.1 Constructor & Destructor Documentation

6.115.1.1 [pointer_to_map_function_2args](#)()

```
template<class T1 , class T2 >
GiNaC::pointer_to_map_function_2args< T1, T2 >::pointer_to_map_function_2args (
    ex xconst ex &, T1, T2,
    T1 a1,
    T2 a2 ) [inline], [explicit]
```

6.115.2 Member Function Documentation

6.115.2.1 [operator](#)()()

```
template<class T1 , class T2 >
ex GiNaC::pointer_to_map_function_2args< T1, T2 >::operator() (
    const ex & e ) [inline], [override], [virtual]
```

Implements [GiNaC::map_function](#).

References [GiNaC::pointer_to_map_function_2args< T1, T2 >::arg1](#), [GiNaC::pointer_to_map_function_2args< T1, T2 >::arg2](#), and [GiNaC::pointer_to_map_function_2args< T1, T2 >::ptr](#).

6.115.3 Member Data Documentation

6.115.3.1 ptr

```
template<class T1 , class T2 >
ex(* GiNaC::pointer_to_map_function_2args< T1, T2 >::ptr) (const ex &, T1, T2) [protected]
```

Referenced by [GiNaC::pointer_to_map_function_2args< T1, T2 >::operator\(\)\(\)](#).

6.115.3.2 arg1

```
template<class T1 , class T2 >
T1 GiNaC::pointer_to_map_function_2args< T1, T2 >::arg1 [protected]
```

Referenced by [GiNaC::pointer_to_map_function_2args< T1, T2 >::operator\(\)\(\)](#).

6.115.3.3 arg2

```
template<class T1 , class T2 >
T2 GiNaC::pointer_to_map_function_2args< T1, T2 >::arg2 [protected]
```

Referenced by [GiNaC::pointer_to_map_function_2args< T1, T2 >::operator\(\)\(\)](#).

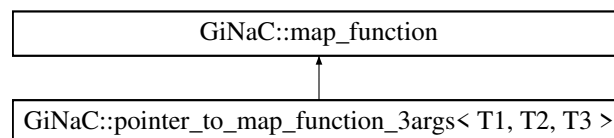
The documentation for this class was generated from the following file:

- [ex.h](#)

6.116 GiNaC::pointer_to_map_function_3args< T1, T2, T3 > Class Template Reference

```
#include <ex.h>
```

Inheritance diagram for [GiNaC::pointer_to_map_function_3args< T1, T2, T3 >](#):



Public Member Functions

- [pointer_to_map_function_3args](#) ([ex](#) x(const [ex](#) &, T1, T2, T3), T1 a1, T2 a2, T3 a3)
- [ex operator\(\)](#) (const [ex](#) &e) override

Public Member Functions inherited from [GiNaC::map_function](#)

- virtual [~map_function](#) ()

Protected Attributes

- [ex\(* ptr\)](#)(const [ex](#) &, T1, T2, T3)
- T1 [arg1](#)
- T2 [arg2](#)
- T3 [arg3](#)

Additional Inherited Members

Public Types inherited from [GiNaC::map_function](#)

- typedef const [ex](#) & [argument_type](#)
- typedef [ex](#) [result_type](#)

6.116.1 Constructor & Destructor Documentation

6.116.1.1 [pointer_to_map_function_3args\(\)](#)

```
template<class T1 , class T2 , class T3 >
GiNaC::pointer\_to\_map\_function\_3args< T1, T2, T3 >::pointer_to_map_function_3args (
    ex xconst ex &, T1, T2, T3,
    T1 a1,
    T2 a2,
    T3 a3 ) [inline], [explicit]
```

6.116.2 Member Function Documentation

6.116.2.1 [operator>\(\)](#)

```
template<class T1 , class T2 , class T3 >
ex GiNaC::pointer\_to\_map\_function\_3args< T1, T2, T3 >::operator() (
    const ex & e ) [inline], [override], [virtual]
```

Implements [GiNaC::map_function](#).

References [GiNaC::pointer_to_map_function_3args< T1, T2, T3 >::arg1](#), [GiNaC::pointer_to_map_function_3args< T1, T2, T3 >::a](#), [GiNaC::pointer_to_map_function_3args< T1, T2, T3 >::arg3](#), and [GiNaC::pointer_to_map_function_3args< T1, T2, T3 >::ptr](#).

6.116.3 Member Data Documentation

6.116.3.1 [ptr](#)

```
template<class T1 , class T2 , class T3 >
ex(* GiNaC::pointer\_to\_map\_function\_3args< T1, T2, T3 >::ptr) (const ex &, T1, T2, T3) [protected]
```

Referenced by [GiNaC::pointer_to_map_function_3args< T1, T2, T3 >::operator>\(\)](#).

6.116.3.2 `arg1`

```
template<class T1 , class T2 , class T3 >
T1 GiNaC::pointer_to_map_function_3args< T1, T2, T3 >::arg1 [protected]
```

Referenced by [GiNaC::pointer_to_map_function_3args< T1, T2, T3 >::operator\(\)](#).

6.116.3.3 `arg2`

```
template<class T1 , class T2 , class T3 >
T2 GiNaC::pointer_to_map_function_3args< T1, T2, T3 >::arg2 [protected]
```

Referenced by [GiNaC::pointer_to_map_function_3args< T1, T2, T3 >::operator\(\)](#).

6.116.3.4 `arg3`

```
template<class T1 , class T2 , class T3 >
T3 GiNaC::pointer_to_map_function_3args< T1, T2, T3 >::arg3 [protected]
```

Referenced by [GiNaC::pointer_to_map_function_3args< T1, T2, T3 >::operator\(\)](#).

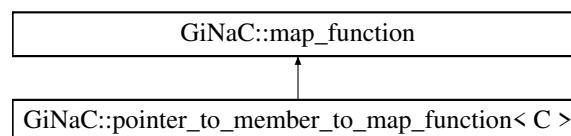
The documentation for this class was generated from the following file:

- [ex.h](#)

6.117 `GiNaC::pointer_to_member_to_map_function< C >` Class Template Reference

```
#include <ex.h>
```

Inheritance diagram for `GiNaC::pointer_to_member_to_map_function< C >`:



Public Member Functions

- [pointer_to_member_to_map_function](#) ([ex](#)(C::*member)(const [ex](#) &), C &obj)
- [ex operator\(\)](#) (const [ex](#) &e) override

Public Member Functions inherited from [GiNaC::map_function](#)

- virtual [~map_function](#) ()

Protected Attributes

- [ex](#)(C::[* ptr](#))(const [ex](#) &)
- C & [c](#)

Additional Inherited Members

Public Types inherited from [GiNaC::map_function](#)

- typedef const [ex](#) & [argument_type](#)
- typedef [ex](#) [result_type](#)

6.117.1 Constructor & Destructor Documentation

6.117.1.1 [pointer_to_member_to_map_function](#)()

```
template<class C >
GiNaC::pointer\_to\_member\_to\_map\_function< C >::pointer_to_member_to_map_function (
    ex(C::\*)(const ex &) member,
    C & obj ) [inline], [explicit]
```

6.117.2 Member Function Documentation

6.117.2.1 [operator](#)()()

```
template<class C >
ex GiNaC::pointer\_to\_member\_to\_map\_function< C >::operator() (
    const ex & e ) [inline], [override], [virtual]
```

Implements [GiNaC::map_function](#).

References [GiNaC::pointer_to_member_to_map_function< C >::c](#).

6.117.3 Member Data Documentation

6.117.3.1 [ptr](#)

```
template<class C >
ex(C::\* GiNaC::pointer\_to\_member\_to\_map\_function< C >::ptr) (const ex &) [protected]
```

6.117.3.2 [c](#)

```
template<class C >
C& GiNaC::pointer\_to\_member\_to\_map\_function< C >::c [protected]
```

Referenced by [GiNaC::pointer_to_member_to_map_function< C >::operator](#)()().

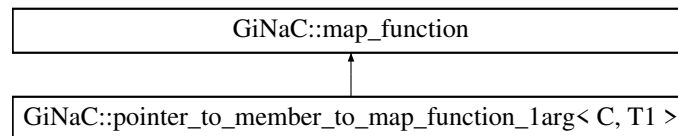
The documentation for this class was generated from the following file:

- [ex.h](#)

6.118 GiNaC::pointer_to_member_to_map_function_1arg< C, T1 > Class Template Reference

```
#include <ex.h>
```

Inheritance diagram for GiNaC::pointer_to_member_to_map_function_1arg< C, T1 >:



Public Member Functions

- [pointer_to_member_to_map_function_1arg](#) ([ex](#)(C::*member)(const [ex](#) &, T1), C &obj, T1 a1)
- [ex operator\(\)](#) (const [ex](#) &e) override

Public Member Functions inherited from [GiNaC::map_function](#)

- virtual [~map_function](#) ()

Protected Attributes

- [ex](#)(C::* [ptr](#))(const [ex](#) &, T1)
- C & [c](#)
- T1 [arg1](#)

Additional Inherited Members

Public Types inherited from [GiNaC::map_function](#)

- typedef const [ex](#) & [argument_type](#)
- typedef [ex](#) [result_type](#)

6.118.1 Constructor & Destructor Documentation

6.118.1.1 pointer_to_member_to_map_function_1arg()

```

template<class C , class T1 >
GiNaC::pointer_to_member_to_map_function_1arg< C, T1 >::pointer_to_member_to_map_function_1arg
(
    ex(C::*)(const ex &, T1) member,
    C & obj,
    T1 a1 ) [inline], [explicit]
  
```

6.118.2 Member Function Documentation

6.118.2.1 operator>()

```
template<class C , class T1 >
ex GiNaC::pointer_to_member_to_map_function_1arg< C, T1 >::operator() (
    const ex & e ) [inline], [override], [virtual]
```

Implements [GiNaC::map_function](#).

References [GiNaC::pointer_to_member_to_map_function_1arg< C, T1 >::arg1](#), and [GiNaC::pointer_to_member_to_map_function_1arg< C, T1 >::c](#).

6.118.3 Member Data Documentation

6.118.3.1 ptr

```
template<class C , class T1 >
ex(C::* GiNaC::pointer_to_member_to_map_function_1arg< C, T1 >::ptr) (const ex &, T1) [protected]
```

6.118.3.2 c

```
template<class C , class T1 >
C& GiNaC::pointer_to_member_to_map_function_1arg< C, T1 >::c [protected]
```

Referenced by [GiNaC::pointer_to_member_to_map_function_1arg< C, T1 >::operator>\(\)](#).

6.118.3.3 arg1

```
template<class C , class T1 >
T1 GiNaC::pointer_to_member_to_map_function_1arg< C, T1 >::arg1 [protected]
```

Referenced by [GiNaC::pointer_to_member_to_map_function_1arg< C, T1 >::operator>\(\)](#).

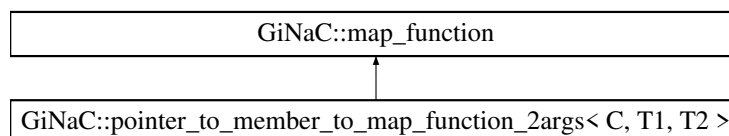
The documentation for this class was generated from the following file:

- [ex.h](#)

6.119 GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 > Class Template Reference

```
#include <ex.h>
```

Inheritance diagram for [GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 >](#):



Public Member Functions

- [pointer_to_member_to_map_function_2args](#) ([ex](#)(C::[*member](#))(const [ex](#) &, T1, T2), C &obj, T1 a1, T2 a2)
- [ex operator\(\)](#) (const [ex](#) &e) override

Public Member Functions inherited from [GiNaC::map_function](#)

- virtual [~map_function](#) ()

Protected Attributes

- [ex](#)(C::[* ptr](#))(const [ex](#) &, T1, T2)
- C & [c](#)
- T1 [arg1](#)
- T2 [arg2](#)

Additional Inherited Members

Public Types inherited from [GiNaC::map_function](#)

- typedef const [ex](#) & [argument_type](#)
- typedef [ex](#) [result_type](#)

6.119.1 Constructor & Destructor Documentation

6.119.1.1 [pointer_to_member_to_map_function_2args](#)()

```
template<class C , class T1 , class T2 >
GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 >::pointer_to_member_to_map_function_2args (
    ex(C::\*member)(const ex &, T1, T2) member,
    C & obj,
    T1 a1,
    T2 a2 ) [inline], [explicit]
```

6.119.2 Member Function Documentation

6.119.2.1 [operator](#)()()

```
template<class C , class T1 , class T2 >
ex GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 >::operator() (
    const ex & e ) [inline], [override], [virtual]
```

Implements [GiNaC::map_function](#).

References [GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 >::arg1](#), [GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 >::c](#).

6.119.3 Member Data Documentation

6.119.3.1 ptr

```
template<class C , class T1 , class T2 >
ex(C::* GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 >::ptr) (const ex &, T1, T2)
[protected]
```

6.119.3.2 c

```
template<class C , class T1 , class T2 >
C& GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 >::c [protected]
```

Referenced by [GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 >::operator\(\)](#).

6.119.3.3 arg1

```
template<class C , class T1 , class T2 >
T1 GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 >::arg1 [protected]
```

Referenced by [GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 >::operator\(\)](#).

6.119.3.4 arg2

```
template<class C , class T1 , class T2 >
T2 GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 >::arg2 [protected]
```

Referenced by [GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 >::operator\(\)](#).

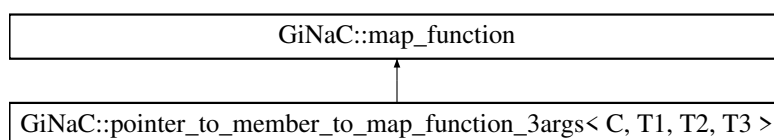
The documentation for this class was generated from the following file:

- [ex.h](#)

6.120 GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 > Class Template Reference

```
#include <ex.h>
```

Inheritance diagram for GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >:



Public Member Functions

- [pointer_to_member_to_map_function_3args](#) ([ex](#)(C::[*member](#))(const [ex](#) &, T1, T2, T3), C &obj, T1 a1, T2 a2, T3 a3)
- [ex operator\(\)](#) (const [ex](#) &e) override

Public Member Functions inherited from [GiNaC::map_function](#)

- virtual [~map_function](#) ()

Protected Attributes

- [ex](#)(C::[* ptr](#))(const [ex](#) &, T1, T2, T3)
- C & [c](#)
- T1 [arg1](#)
- T2 [arg2](#)
- T3 [arg3](#)

Additional Inherited Members

Public Types inherited from [GiNaC::map_function](#)

- typedef const [ex](#) & [argument_type](#)
- typedef [ex](#) [result_type](#)

6.120.1 Constructor & Destructor Documentation

6.120.1.1 [pointer_to_member_to_map_function_3args\(\)](#)

```
template<class C , class T1 , class T2 , class T3 >
GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >::pointer_to_member_to_map_↵
function_3args (
    ex(C::\*)(const ex &, T1, T2, T3) member,
    C & obj,
    T1 a1,
    T2 a2,
    T3 a3 ) [inline], [explicit]
```

6.120.2 Member Function Documentation

6.120.2.1 [operator\(\)](#)()

```
template<class C , class T1 , class T2 , class T3 >
ex GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >::operator() (
    const ex & e ) [inline], [override], [virtual]
```

Implements [GiNaC::map_function](#).

References [GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >::arg1](#), [GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >::arg2](#), [GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >::arg3](#), and [GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >::arg4](#).

6.120.3 Member Data Documentation

6.120.3.1 ptr

```
template<class C , class T1 , class T2 , class T3 >
ex(C::* GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >::ptr) (const ex &, T1,
T2, T3) [protected]
```

6.120.3.2 c

```
template<class C , class T1 , class T2 , class T3 >
C& GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >::c [protected]
```

Referenced by [GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >::operator\(\)\(\)](#).

6.120.3.3 arg1

```
template<class C , class T1 , class T2 , class T3 >
T1 GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >::arg1 [protected]
```

Referenced by [GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >::operator\(\)\(\)](#).

6.120.3.4 arg2

```
template<class C , class T1 , class T2 , class T3 >
T2 GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >::arg2 [protected]
```

Referenced by [GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >::operator\(\)\(\)](#).

6.120.3.5 arg3

```
template<class C , class T1 , class T2 , class T3 >
T3 GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >::arg3 [protected]
```

Referenced by [GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >::operator\(\)\(\)](#).

The documentation for this class was generated from the following file:

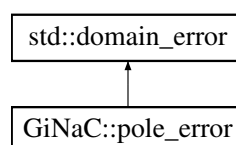
- [ex.h](#)

6.121 GiNaC::pole_error Class Reference

Exception class thrown when a singularity is encountered.

```
#include <numeric.h>
```

Inheritance diagram for GiNaC::pole_error:



Public Member Functions

- [pole_error](#) (const std::string &what_arg, int [degree](#))
ctor for [pole_error](#) exception class.
- int [degree](#) () const
Return the degree of the [pole_error](#) exception class.

Private Attributes

- int [deg](#)

6.121.1 Detailed Description

Exception class thrown when a singularity is encountered.

6.121.2 Constructor & Destructor Documentation

6.121.2.1 pole_error()

```
GiNaC::pole_error::pole_error (  
    const std::string & what_arg,  
    int degree ) [explicit]
```

ctor for [pole_error](#) exception class.

6.121.3 Member Function Documentation

6.121.3.1 degree()

```
int GiNaC::pole_error::degree ( ) const
```

Return the degree of the [pole_error](#) exception class.

References [deg](#).

6.121.4 Member Data Documentation

6.121.4.1 deg

```
int GiNaC::pole_error::deg [private]
```

Referenced by [degree\(\)](#).

The documentation for this class was generated from the following files:

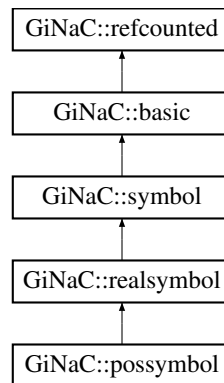
- [numeric.h](#)
- [utils.cpp](#)

6.122 GiNaC::possymbol Class Reference

Specialization of symbol to real positive domain.

```
#include <symbol.h>
```

Inheritance diagram for GiNaC::possymbol:



Public Member Functions

- [possymbol](#) ()
- [possymbol](#) (const std::string &initname)
- [possymbol](#) (const std::string &initname, const std::string &texname)
- unsigned [get_domain](#) () const override
- [possymbol](#) * [duplicate](#) () const override

Create a clone of this object on the heap.

Public Member Functions inherited from [GiNaC::realsymbol](#)

- [realsymbol](#) ()
- [realsymbol](#) (const std::string &initname)
- [realsymbol](#) (const std::string &initname, const std::string &texname)
- unsigned [get_domain](#) () const override
- [ex conjugate](#) () const override
- [ex real_part](#) () const override
- [ex imag_part](#) () const override
- [realsymbol](#) * [duplicate](#) () const override

Create a clone of this object on the heap.

Public Member Functions inherited from [GiNaC::symbol](#)

- [symbol](#) (const std::string &initname)
- [symbol](#) (const std::string &initname, const std::string &texname)
- bool [info](#) (unsigned inf) const override

Information about the object.

- [ex eval](#) () const override

Perform automatic non-interruptive term rewriting rules.

- [ex evalf](#) () const override

- Evaluate object numerically.*

 - **ex series** (const **relational** &s, int **order**, unsigned **options**=0) const override

*Implementation of **ex::series()** for symbols.*
 - **ex subs** (const **exmap** &m, unsigned **options**=0) const override

Substitute a set of objects by arbitrary expressions.
 - **ex normal** (**exmap** &repl, **exmap** &rev_lookup, **lst** &modifier) const override

*Implementation of **ex::normal()** for symbols.*
 - **ex to_rational** (**exmap** &repl) const override

*Implementation of **ex::to_rational()** for symbols.*
 - **ex to_polynomial** (**exmap** &repl) const override

*Implementation of **ex::to_polynomial()** for symbols.*
 - **ex conjugate** () const override
 - **ex real_part** () const override
 - **ex imag_part** () const override
 - bool **is_polynomial** (const **ex** &var) const override
- Check whether this is a polynomial in the given variables.*
- void **archive** (**archive_node** &n) const override
- Save (a.k.a.*
- void **read_archive** (const **archive_node** &n, **lst** &syms) override
- Read (a.k.a.*
- void **set_name** (const std::string &n)
 - void **set_TeX_name** (const std::string &n)
 - std::string **get_name** () const
 - std::string **get_TeX_name** () const

Public Member Functions inherited from **GiNaC::basic**

- virtual **~basic** ()
- basic destructor, virtual because class ex will delete objects of derived classes via a basic*.*
- **basic** (const **basic** &other)
 - const **basic** & **operator=** (const **basic** &other)
- basic assignment operator: the other object might be of a derived class.*
- virtual **ex evalm** () const
- Evaluate sums, products and integer powers of matrices.*
- virtual **ex eval_integ** () const
- Evaluate integrals, if result is known.*
- virtual **ex eval_indexed** (const **basic** &i) const
- Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.*
- virtual void **print** (const **print_context** &c, unsigned level=0) const
- Output to stream.*
- virtual void **dbgprint** () const
- Little wrapper around print to be called within a debugger.*
- virtual void **dbgprinttree** () const
- Little wrapper around printtree to be called within a debugger.*
- virtual unsigned **precedence** () const
- Return relative operator precedence (for parenthezing output).*
- virtual size_t **nops** () const
- Number of operands/members.*
- virtual **ex op** (size_t i) const
- Return operand/member at position i.*
- virtual **ex operator[]** (const **ex** &index) const

- virtual `ex operator[]` (size_t i) const
- virtual `ex & let_op` (size_t i)
Return modifiable operand/member at position i.
- virtual `ex & operator[]` (const `ex` &index)
- virtual `ex & operator[]` (size_t i)
- virtual bool `has` (const `ex` &other, unsigned `options`=0) const
Test for occurrence of a pattern.
- virtual bool `match` (const `ex` &pattern, `exmap` &repls) const
Check whether the expression matches a given pattern.
- virtual `ex map` (`map_function` &f) const
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void `accept` (GiNaC::visitor &v) const
- virtual int `degree` (const `ex` &s) const
Return degree of highest power in object s.
- virtual int `ldegree` (const `ex` &s) const
Return degree of lowest power in object s.
- virtual `ex coeff` (const `ex` &s, int n=1) const
Return coefficient of degree n in object s.
- virtual `ex expand` (unsigned `options`=0) const
Expand expression, i.e.
- virtual `ex collect` (const `ex` &s, bool distributed=false) const
Sort expanded expression in terms of powers of some object(s).
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &xi) const
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient` () const
Implementation `ex::max_coefficient()`.
- virtual `exvector get_free_indices` () const
Return a vector containing the free indices of an expression.
- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
Add two indexed expressions.
- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
Multiply an indexed expression with a scalar.
- virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const
Try to contract two indexed expressions that appear in the same product.
- virtual unsigned `return_type` () const
- virtual `return_type_t return_type_tinfo` () const
- template<class T >
void `print_dispatch` (const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- `ex subs_one_level` (const `exmap` &m, unsigned `options`) const
Helper function for `subs()`.
- `ex diff` (const `symbol` &s, unsigned nth=1) const
Default interface of nth derivative `ex::diff(s, n)`.
- int `compare` (const `basic` &other) const
Compare objects syntactically to establish canonical ordering.
- bool `is_equal` (const `basic` &other) const
Test for syntactic equality.
- const `basic` & `hold` () const

Stop further evaluation.

- unsigned [gethash](#) () const
- const [basic](#) & [setflag](#) (unsigned f) const

Set some [status_flags](#).

- const [basic](#) & [clearflag](#) (unsigned f) const

Clear some [status_flags](#).

Public Member Functions inherited from [GiNaC::refcounted](#)

- [refcounted](#) () noexcept
- unsigned int [add_reference](#) () noexcept
- unsigned int [remove_reference](#) () noexcept
- unsigned int [get_refcount](#) () const noexcept
- void [set_refcount](#) (unsigned int r) noexcept

Additional Inherited Members

Protected Member Functions inherited from [GiNaC::symbol](#)

- [ex_derivative](#) (const [symbol](#) &s) const override
Implementation of [ex::diff\(\)](#) for single differentiation of a symbol.
- bool [is_equal_same_type](#) (const [basic](#) &other) const override
Returns true if two objects of same type are equal.
- unsigned [calchash](#) () const override
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
- void [do_print_latex](#) (const [print_latex](#) &c, unsigned level) const
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Protected Attributes inherited from [GiNaC::symbol](#)

- unsigned [serial](#)
unique serial number for comparison
- std::string [name](#)
printname of this symbol
- std::string [TeX_name](#)
LaTeX name of this symbol.

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.122.1 Detailed Description

Specialization of symbol to real positive domain.

6.122.2 Constructor & Destructor Documentation

6.122.2.1 `possymbol()` [1/3]

```
GiNaC::possymbol::possymbol ( )
```

Referenced by [duplicate\(\)](#).

6.122.2.2 `possymbol()` [2/3]

```
GiNaC::possymbol::possymbol (
    const std::string & initname ) [explicit]
```

6.122.2.3 `possymbol()` [3/3]

```
GiNaC::possymbol::possymbol (
    const std::string & initname,
    const std::string & texname )
```

6.122.3 Member Function Documentation

6.122.3.1 `get_domain()`

```
unsigned GiNaC::possymbol::get_domain ( ) const [inline], [override], [virtual]
```

Reimplemented from [GiNaC::symbol](#).

References [GiNaC::domain::positive](#).

6.122.3.2 duplicate()

```
possymbol * GiNaC::possymbol::duplicate ( ) const [inline], [override], [virtual]
```

Create a clone of this object on the heap.

One can think of this as simulating a virtual copy constructor which is needed for instance by the refcounted construction of an ex from a basic.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::status_flags::dynallocated](#), [possymbol\(\)](#), and [GiNaC::basic::setflag\(\)](#).

The documentation for this class was generated from the following files:

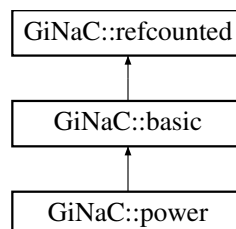
- [symbol.h](#)
- [symbol.cpp](#)

6.123 GiNaC::power Class Reference

This class holds a two-component object, a basis and an exponent representing exponentiation.

```
#include <power.h>
```

Inheritance diagram for GiNaC::power:



Public Member Functions

- [power](#) (const [ex](#) &lh, const [ex](#) &rh)
- `template<typename T >`
[power](#) (const [ex](#) &lh, const T &rh)
- unsigned [precedence](#) () const override
Return relative operator precedence (for parenthesizing output).
- bool [info](#) (unsigned inf) const override
Information about the object.
- size_t [nops](#) () const override
Number of operands/members.
- [ex](#) [op](#) (size_t i) const override
Return operand/member at position i.
- [ex](#) [map](#) ([map_function](#) &f) const override
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- bool [is_polynomial](#) (const [ex](#) &var) const override

- *Check whether this is a polynomial in the given variables.*
- `int degree (const ex &s) const` override
Return degree of highest power in object s.
- `int ldegree (const ex &s) const` override
Return degree of lowest power in object s.
- `ex coeff (const ex &s, int n=1) const` override
Return coefficient of degree n in object s.
- `ex eval () const` override
Perform automatic term rewriting rules in this class.
- `ex evalf () const` override
Evaluate object numerically.
- `ex evalm () const` override
Evaluate sums, products and integer powers of matrices.
- `ex series (const relational &s, int order, unsigned options=0) const` override
Implementation of `ex::series()` for powers.
- `ex subs (const exmap &m, unsigned options=0) const` override
Substitute a set of objects by arbitrary expressions.
- `bool has (const ex &other, unsigned options=0) const` override
Test for occurrence of a pattern.
- `ex normal (exmap &repl, exmap &rev_lookup, lst &modifier) const` override
Implementation of `ex::normal()` for powers.
- `ex to_rational (exmap &repl) const` override
Implementation of `ex::to_rational()` for powers.
- `ex to_polynomial (exmap &repl) const` override
Implementation of `ex::to_polynomial()` for powers.
- `ex conjugate () const` override
- `ex real_part () const` override
- `ex imag_part () const` override
- `void archive (archive_node &n) const` override
Save (a.k.a.
- `void read_archive (const archive_node &n, lst &syms) const` override
Read (a.k.a.

Public Member Functions inherited from GiNaC::basic

- `virtual ~basic ()`
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic (const basic &other)`
- `const basic & operator= (const basic &other)`
basic assignment operator: the other object might be of a derived class.
- `virtual basic * duplicate () const`
Create a clone of this object on the heap.
- `virtual ex eval_integ () const`
Evaluate integrals, if result is known.
- `virtual ex eval_indexed (const basic &i) const`
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- `virtual void print (const print_context &c, unsigned level=0) const`
Output to stream.
- `virtual void dbgprint () const`
Little wrapper around print to be called within a debugger.

- virtual void `dbgprinttree` () const
Little wrapper around `printtree` to be called within a debugger.
- virtual `ex operator[]` (const `ex` &index) const
- virtual `ex operator[]` (size_t i) const
- virtual `ex & let_op` (size_t i)
Return modifiable operand/member at position i.
- virtual `ex & operator[]` (const `ex` &index)
- virtual `ex & operator[]` (size_t i)
- virtual bool `match` (const `ex` &pattern, `exmap` &repls) const
Check whether the expression matches a given pattern.
- virtual void `accept` (`GiNaC::visitor` &v) const
- virtual `ex collect` (const `ex` &s, bool distributed=false) const
Sort expanded expression in terms of powers of some object(s).
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &xi) const
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient` () const
Implementation `ex::max_coefficient()`.
- virtual `exvector get_free_indices` () const
Return a vector containing the free indices of an expression.
- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
Add two indexed expressions.
- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
Multiply an indexed expression with a scalar.
- virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const
Try to contract two indexed expressions that appear in the same product.
- template<class T >
void `print_dispatch` (const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- `ex subs_one_level` (const `exmap` &m, unsigned options) const
Helper function for `subs()`.
- `ex diff` (const `symbol` &s, unsigned nth=1) const
Default interface of nth derivative `ex::diff(s, n)`.
- int `compare` (const `basic` &other) const
Compare objects syntactically to establish canonical ordering.
- bool `is_equal` (const `basic` &other) const
Test for syntactic equality.
- const `basic & hold` () const
Stop further evaluation.
- unsigned `gethash` () const
- const `basic & setflag` (unsigned f) const
Set some `status_flags`.
- const `basic & clearflag` (unsigned f) const
Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Protected Member Functions

- `ex derivative` (const `symbol` &s) const override
Implementation of `ex::diff()` for a power.
- `ex eval_ncmul` (const `exvector` &v) const override
- unsigned `return_type` () const override
- `return_type_t return_type_tinfo` () const override
- `ex expand` (unsigned `options`=0) const override
Expand expression, i.e.
- void `print_power` (const `print_context` &c, const char *powersymbol, const char *openbrace, const char *closebrace, unsigned level) const
- void `do_print_dflt` (const `print_dflt` &c, unsigned level) const
- void `do_print_latex` (const `print_latex` &c, unsigned level) const
- void `do_print_csrc` (const `print_csrc` &c, unsigned level) const
- void `do_print_python` (const `print_python` &c, unsigned level) const
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
- void `do_print_csrc_cl_N` (const `print_csrc_cl_N` &c, unsigned level) const

Protected Member Functions inherited from `GiNaC::basic`

- `basic` ()
- virtual bool `match_same_type` (const `basic` &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual int `compare_same_type` (const `basic` &other) const
Returns order relation between two objects of same type.
- virtual bool `is_equal_same_type` (const `basic` &other) const
Returns true if two objects of same type are equal.
- virtual unsigned `calchash` () const
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- void `ensure_if_modifiable` () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void `do_print` (const `print_context` &c, unsigned level) const
Default output to stream.
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
Tree output to stream.
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
Python parsable output to stream.

Static Protected Member Functions

- static `ex expand_add` (const `add` &a, long n, unsigned `options`)
expand a^n where a is an add and n is a positive integer.
- static `ex expand_add_2` (const `add` &a, unsigned `options`)
Special case of `power::expand_add`.
- static `ex expand_mul` (const `mul` &m, const `numeric` &n, unsigned `options`, bool from_expand=false)
Expand factors of m in m^n where m is a mul and n is an integer.

Protected Attributes

- `ex basis`
- `ex exponent`

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

Friends

- class [mul](#)

6.123.1 Detailed Description

This class holds a two-component object, a basis and an exponent representing exponentiation.

6.123.2 Constructor & Destructor Documentation

6.123.2.1 `power()` [1/2]

```
GiNaC::power::power (
    const ex & lh,
    const ex & rh ) [inline]
```

6.123.2.2 `power()` [2/2]

```
template<typename T >
GiNaC::power::power (
    const ex & lh,
    const T & rh ) [inline]
```

6.123.3 Member Function Documentation

6.123.3.1 `precedence()`

```
unsigned GiNaC::power::precedence ( ) const [inline], [override], [virtual]
```

Return relative operator precedence (for parenthezing output).

Reimplemented from [GiNaC::basic](#).

Referenced by [print_power\(\)](#).

6.123.3.2 info()

```
bool GiNaC::power::info (
    unsigned int ) const [override], [virtual]
```

Information about the object.

See also

class [info_flags](#)

Reimplemented from [GiNaC::basic](#).

References [basis](#), [GiNaC::info_flags::cinteger_polynomial](#), [GiNaC::basic::clearflag\(\)](#), [GiNaC::info_flags::crational_polynomial](#), [GiNaC::info_flags::even](#), [GiNaC::status_flags::expanded](#), [GiNaC::info_flags::expanded](#), [exponent](#), [GiNaC::basic::flags](#), [GiNaC::status_flags::has_indices](#), [GiNaC::info_flags::has_indices](#), [GiNaC::status_flags::has_no_indices](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::info_flags::integer_polynomial](#), [GiNaC::info_flags::nonnegative](#), [GiNaC::info_flags::nonnegint](#), [GiNaC::info_flags::polynomial](#), [GiNaC::info_flags::positive](#), [GiNaC::info_flags::rational_function](#), [GiNaC::info_flags::rational_polynomial](#), [GiNaC::info_flags::real](#), and [GiNaC::basic::setflag\(\)](#).

6.123.3.3 nops()

```
size_t GiNaC::power::nops ( ) const [override], [virtual]
```

Number of operands/members.

Reimplemented from [GiNaC::basic](#).

6.123.3.4 op()

```
ex GiNaC::power::op (
    size_t i ) const [override], [virtual]
```

Return operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [basis](#), [exponent](#), and [GINAC_ASSERT](#).

6.123.3.5 map()

```
ex GiNaC::power::map (
    map_function & f ) const [override], [virtual]
```

Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).

Reimplemented from [GiNaC::basic](#).

References [GiNaC::are_ex_trivially_equal\(\)](#), [basis](#), and [exponent](#).

6.123.3.6 is_polynomial()

```
bool GiNaC::power::is_polynomial (
    const ex & var ) const [override], [virtual]
```

Check whether this is a polynomial in the given variables.

Reimplemented from [GiNaC::basic](#).

References [basis](#), [exponent](#), [GiNaC::ex::has\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_polynomial\(\)](#), and [GiNaC::info_flags::nonnegint](#).

6.123.3.7 degree()

```
int GiNaC::power::degree (
    const ex & s ) const [override], [virtual]
```

Return degree of highest power in object s.

Reimplemented from [GiNaC::basic](#).

References [basis](#), [GiNaC::ex::degree\(\)](#), [exponent](#), [GiNaC::ex::has\(\)](#), [GiNaC::basic::is_equal\(\)](#), [GiNaC::ex::is_equal\(\)](#), and [GiNaC::is_integer\(\)](#).

6.123.3.8 ldegree()

```
int GiNaC::power::ldegree (
    const ex & s ) const [override], [virtual]
```

Return degree of lowest power in object s.

Reimplemented from [GiNaC::basic](#).

References [basis](#), [exponent](#), [GiNaC::ex::has\(\)](#), [GiNaC::basic::is_equal\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::is_integer\(\)](#), and [GiNaC::ex::ldegree\(\)](#).

6.123.3.9 coeff()

```
ex GiNaC::power::coeff (
    const ex & s,
    int n = 1 ) const [override], [virtual]
```

Return coefficient of degree n in object s.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex0](#), [GiNaC::_ex1](#), [basis](#), [exponent](#), [GiNaC::basic::is_equal\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::is_integer\(\)](#), and [n](#).

Referenced by [expand\(\)](#), and [expand_add\(\)](#).

6.123.3.10 eval()

```
ex GiNaC::power::eval ( ) const [override], [virtual]
```

Perform automatic term rewriting rules in this class.

In the following $x, x_1, x_2, \dots$ stand for a symbolic variables of type `ex` and $c, c_1, c_2, \dots$ stand for such expressions that contain a plain number.

- $x^0 \rightarrow 1$ (also handles 0^0)
- $x^1 \rightarrow x$
- $0^c \rightarrow 0$ or exception (depending on the real part of c)
- $1^x \rightarrow 1$
- $c_1^{c_2} \rightarrow c_1^n c_1^{c_2-n}$ (so that $0 < c_2-n < 1$, try to evaluate roots, possibly in numerator and denominator of c_1)
- $x^{(x, c_1), c_2} \rightarrow x^{(x, c_1 * c_2)}$ if x is positive and c_1 is real.
- $x^{(x, c_1), c_2} \rightarrow x^{(x, c_1 * c_2)}$ (c_2 integer or $-1 < c_1 \leq 1$ or ($c_1 = -1$ and $c_2 > 0$), case $c_1 = 1$ should not happen, see below!)
- $x^{(x, y, z), c} \rightarrow x^c y^c z^c$ (if c integer)
- $x^{(x, c_1), c_2} \rightarrow x^{(x, c_2)} c_1^{c_2}$ ($c_1 > 0$)
- $x^{(x, c_1), c_2} \rightarrow x^{(-x, c_2)} c_1^{c_2}$ ($c_1 < 0$)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex0](#), [GiNaC::ex1](#), [GiNaC::ex_1](#), [GiNaC::num0_p](#), [GiNaC::num1_p](#), [GiNaC::num_1_p](#), [GiNaC::abs\(\)](#), [basis](#), [c](#), [GiNaC::basic::clearflag\(\)](#), [GiNaC::ex::coeff\(\)](#), [GiNaC::return_types::commutative](#), [GiNaC::numeric::compare\(\)](#), [GiNaC::numeric::denom\(\)](#), [GiNaC::numeric::div\(\)](#), [GiNaC::status_flags::evaluated](#), [expand_mul\(\)](#), [exponent](#), [GiNaC::basic::flags](#), [GINAC_ASSERT](#), [GiNaC::status_flags::hash_calculated](#), [GiNaC::basic::hold\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::integer_content\(\)](#), [GiNaC::numeric::inverse\(\)](#), [GiNaC::iquo\(\)](#), [GiNaC::numeric::is_crational\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::numeric::is_equal\(\)](#), [GiNaC::numeric::is_integer\(\)](#), [GiNaC::is_negative\(\)](#), [GiNaC::numeric::is_pos_integ](#), [GiNaC::numeric::is_positive\(\)](#), [GiNaC::numeric::is_rational\(\)](#), [GiNaC::numeric::is_real\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::numeric::is_zero\(\)](#), [likely](#), [m](#), [GiNaC::numeric::mul\(\)](#), [n](#), [GiNaC::numeric::numer\(\)](#), [GiNaC::ex::op\(\)](#), [GiNaC::expairseq::overall_coeff](#), [GiNaC::info_flags::positive](#), [GiNaC::pow\(\)](#), [GiNaC::numeric::power\(\)](#), [r](#), [GiNaC::info_flags::real](#), [GiNaC::numeric::real\(\)](#), [GiNaC::ex::return_type\(\)](#), [GiNaC::expairseq::seq](#), [GiNaC::basic::setflag\(\)](#), and [GiNaC::numeric::to_int\(\)](#).

Referenced by [do_print_latex\(\)](#).

6.123.3.11 evalf()

```
ex GiNaC::power::evalf ( ) const [override], [virtual]
```

Evaluate object numerically.

Reimplemented from [GiNaC::basic](#).

References [basis](#), [GiNaC::ex::evalf\(\)](#), and [exponent](#).

6.123.3.12 evalm()

```
ex GiNaC::power::evalm ( ) const [override], [virtual]
```

Evaluate sums, products and integer powers of matrices.

Reimplemented from [GiNaC::basic](#).

References [basis](#), [GiNaC::ex::evalm\(\)](#), [exponent](#), and [GiNaC::pow\(\)](#).

6.123.3.13 series()

```
ex GiNaC::power::series (
    const relational & r,
    int order,
    unsigned options = 0 ) const [override], [virtual]
```

Implementation of [ex::series\(\)](#) for powers.

This performs Laurent expansion of reciprocals of series at singularities.

See also

[ex::series](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex1](#), [basis](#), [GiNaC::exp\(\)](#), [GiNaC::ex::expand\(\)](#), [exponent](#), [GiNaC::ex::info\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::is_integer\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::ex::lddegree\(\)](#), [GiNaC::ex::lhs\(\)](#), [GiNaC::log\(\)](#), [GiNaC::info_flags::negint](#), [GiNaC::subs_options::no_pattern](#), [options](#), [order](#), [r](#), [GiNaC::info_flags::rational_function](#), [GiNaC::ex::rhs\(\)](#), [GiNaC::ex::series\(\)](#), [GiNaC::basic::series\(\)](#), [GiNaC::ex::subs\(\)](#), and [GiNaC::to_int\(\)](#).

Referenced by [GiNaC::psi1_series\(\)](#).

6.123.3.14 subs()

```
ex GiNaC::power::subs (
    const exmap & m,
    unsigned options = 0 ) const [override], [virtual]
```

Substitute a set of objects by arbitrary expressions.

The ex returned will already be evaluated.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::subs_options::algebraic](#), [GiNaC::are_ex_trivially_equal\(\)](#), [basis](#), [exponent](#), [m](#), [GiNaC::subs_options::no_pattern](#), [options](#), [GiNaC::pow\(\)](#), [GiNaC::ex::subs\(\)](#), [GiNaC::basic::subs_one_level\(\)](#), and [GiNaC::tryfactsubs\(\)](#).

6.123.3.15 has()

```
bool GiNaC::power::has (
    const ex & pattern,
    unsigned options = 0 ) const [override], [virtual]
```

Test for occurrence of a pattern.

An object 'has' a pattern if it matches the pattern itself or one of the children 'has' it. As a consequence (according to the definition of children) given $e=x+y+z$, $e.has(x)$ is true but $e.has(x+y)$ is false.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::has_options::algebraic](#), [basis](#), [exponent](#), [GiNaC::basic::has\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::ex::match\(\)](#), [GiNaC::info_flags::negint](#), [GiNaC::ex::op\(\)](#), [options](#), and [GiNaC::info_flags::posint](#).

6.123.3.16 normal()

```
ex GiNaC::power::normal (
    exmap & repl,
    exmap & rev_lookup,
    lst & modifier ) const [override], [virtual]
```

Implementation of [ex::normal\(\)](#) for powers.

It normalizes the basis, distributes integer exponents to numerator and denominator, and replaces non-integer powers by temporary symbols.

See also

[ex::normal](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex1](#), [basis](#), [exponent](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::container< C >::nops\(\)](#), [GiNaC::ex::normal\(\)](#), [GiNaC::ex::op\(\)](#), [GiNaC::container< C >::op\(\)](#), [GiNaC::info_flags::positive](#), [GiNaC::pow\(\)](#), [GiNaC::replace_with_symbol\(\)](#), and [GiNaC::ex::subs\(\)](#).

6.123.3.17 to_rational()

```
ex GiNaC::power::to_rational (
    exmap & repl ) const [override], [virtual]
```

Implementation of [ex::to_rational\(\)](#) for powers.

It replaces non-integer powers by temporary symbols.

Reimplemented from [GiNaC::basic](#).

References [basis](#), [exponent](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::pow\(\)](#), [GiNaC::replace_with_symbol\(\)](#), and [GiNaC::ex::to_rational\(\)](#).

6.123.3.18 to_polynomial()

```
ex GiNaC::power::to_polynomial (
    exmap & repl ) const [override], [virtual]
```

Implementation of [ex::to_polynomial\(\)](#) for powers.

It replaces non-posint powers by temporary symbols.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex_1](#), [basis](#), [GiNaC::collect_common_factors\(\)](#), [exponent](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::negint](#), [GiNaC::info_flags::posint](#), [GiNaC::pow\(\)](#), [GiNaC::replace_with_symbol\(\)](#), and [GiNaC::ex::to_polynomial\(\)](#).

6.123.3.19 conjugate()

```
ex GiNaC::power::conjugate ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::are_ex_trivially_equal\(\)](#), [basis](#), [GiNaC::ex::conjugate\(\)](#), [exponent](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), and [GiNaC::info_flags::positive](#).

6.123.3.20 real_part()

```
ex GiNaC::power::real_part ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex1](#), [GiNaC::abs\(\)](#), [basis](#), [GiNaC::binomial\(\)](#), [c](#), [GiNaC::cos\(\)](#), [GiNaC::exp\(\)](#), [exponent](#), [GiNaC::ex::imag_part\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::log\(\)](#), [n](#), [GiNaC::info_flags::nonnegative](#), [GiNaC::numer\(\)](#), [GiNaC::pow\(\)](#), and [GiNaC::ex::real_part\(\)](#).

6.123.3.21 imag_part()

```
ex GiNaC::power::imag_part ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex1](#), [GiNaC::abs\(\)](#), [basis](#), [GiNaC::binomial\(\)](#), [c](#), [GiNaC::exp\(\)](#), [exponent](#), [GiNaC::ex::imag_part\(\)](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::log\(\)](#), [n](#), [GiNaC::info_flags::nonnegative](#), [GiNaC::numer\(\)](#), [GiNaC::pow\(\)](#), [GiNaC::ex::real_part\(\)](#), and [GiNaC::sin\(\)](#).

6.123.3.22 archive()

```
void GiNaC::power::archive (
    archive_node & n ) const [override], [virtual]
```

Save (a.k.a.

serialize) object into archive.

Reimplemented from [GiNaC::basic](#).

References [basis](#), [exponent](#), and [n](#).

6.123.3.23 read_archive()

```
void GiNaC::power::read_archive (
    const archive_node & n,
    lst & syms ) [override], [virtual]
```

Read (a.k.a.

deserialize) object from archive.

Reimplemented from [GiNaC::basic](#).

References [basis](#), [exponent](#), and [n](#).

6.123.3.24 derivative()

```
ex GiNaC::power::derivative (
    const symbol & s ) const [override], [protected], [virtual]
```

Implementation of [ex::diff\(\)](#) for a power.

See also

[ex::diff](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex1](#), [GiNaC::_ex_1](#), [basis](#), [GiNaC::ex::diff\(\)](#), [exponent](#), [GiNaC::log\(\)](#), and [GiNaC::pow\(\)](#).

6.123.3.25 eval_ncmul()

```
ex GiNaC::power::eval_ncmul (
    const exvector & v ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

6.123.3.26 return_type()

```
unsigned GiNaC::power::return_type ( ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [basis](#), and [GiNaC::ex::return_type\(\)](#).

6.123.3.27 return_type_tinfo()

```
return_type_t GiNaC::power::return_type_tinfo ( ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [basis](#), and [GiNaC::ex::return_type_tinfo\(\)](#).

6.123.3.28 expand()

```
ex GiNaC::power::expand (
    unsigned options = 0 ) const [override], [protected], [virtual]
```

Expand expression, i.e.

multiply it out and return the result as a new expression.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex1](#), [GiNaC::_ex_1](#), [GiNaC::are_ex_trivially_equal\(\)](#), [basis](#), [coeff\(\)](#), [GiNaC::basic::expand\(\)](#), [GiNaC::ex::expand\(\)](#), [expand_add\(\)](#), [expand_mul\(\)](#), [GiNaC::status_flags::expanded](#), [exponent](#), [GiNaC::basic::hold\(\)](#), [GiNaC::info_flags::indefinite](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::is_integer\(\)](#), [m](#), [GiNaC::info_flags::negative](#), [options](#), [GiNaC::expairseq::overall_coeff](#), [GiNaC::info_flags::positive](#), [GiNaC::pow\(\)](#), [GiNaC::status_flags::purely_indefinite](#), [r](#), [GiNaC::add::recombine_pair_to_ex\(\)](#), [GiNaC::expairseq::seq](#), [GiNaC::basic::setflag\(\)](#), [GiNaC::numeric::to_int\(\)](#), and [GiNaC::numeric::to_long\(\)](#).

Referenced by [expand_add\(\)](#), and [expand_add_2\(\)](#).

6.123.3.29 print_power()

```
void GiNaC::power::print_power (
    const print_context & c,
    const char * powersymbol,
    const char * openbrace,
    const char * closebrace,
    unsigned level ) const [protected]
```

References [basis](#), [c](#), [exponent](#), [precedence\(\)](#), and [GiNaC::ex::print\(\)](#).

Referenced by [do_print_dflt\(\)](#), [do_print_latex\(\)](#), and [do_print_python\(\)](#).

6.123.3.30 do_print_dflt()

```
void GiNaC::power::do_print_dflt (
    const print_dflt & c,
    unsigned level ) const [protected]
```

References [GiNaC::_ex1_2](#), [basis](#), [c](#), [exponent](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::print\(\)](#), and [print_power\(\)](#).

6.123.3.31 do_print_latex()

```
void GiNaC::power::do_print_latex (
    const print_latex & c,
    unsigned level ) const [protected]
```

References [GiNaC::_ex1_2](#), [basis](#), [c](#), [eval\(\)](#), [exponent](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::is_negative\(\)](#), [GiNaC::ex::print\(\)](#), and [print_power\(\)](#).

6.123.3.32 do_print_csrc()

```
void GiNaC::power::do_print_csrc (
    const print\_csrc & c,
    unsigned level ) const [protected]
```

References [GiNaC::_ex_1](#), [basis](#), [c](#), [GiNaC::exp\(\)](#), [exponent](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::integer](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::ex::print\(\)](#), [GiNaC::print_sym_pow\(\)](#), and [GiNaC::numeric::to_int\(\)](#).

6.123.3.33 do_print_python()

```
void GiNaC::power::do_print_python (
    const print\_python & c,
    unsigned level ) const [protected]
```

References [c](#), and [print_power\(\)](#).

6.123.3.34 do_print_python_repr()

```
void GiNaC::power::do_print_python_repr (
    const print\_python\_repr & c,
    unsigned level ) const [protected]
```

References [basis](#), [c](#), [exponent](#), and [GiNaC::ex::print\(\)](#).

6.123.3.35 do_print_csrc_cl_N()

```
void GiNaC::power::do_print_csrc_cl_N (
    const print\_csrc\_cl\_N & c,
    unsigned level ) const [protected]
```

References [GiNaC::_ex_1](#), [basis](#), [c](#), [exponent](#), [GiNaC::ex::is_equal\(\)](#), and [GiNaC::ex::print\(\)](#).

6.123.3.36 expand_add()

```
ex GiNaC::power::expand_add (
    const add & a,
    long n,
    unsigned options ) [static], [protected]
```

expand a^n where a is an [add](#) and n is a positive integer.

See also

[power::expand](#)

References [GiNaC::_ex1](#), [GiNaC::_num1_p](#), [basis](#), [GiNaC::binomial\(\)](#), [c](#), [coeff\(\)](#), [expand\(\)](#), [expand_add_2\(\)](#), [GiNaC::status_flags::expanded](#), [exponent](#), [GiNaC::factor\(\)](#), [GiNaC::partition_with_zero_parts_generator::get\(\)](#), [GiNaC::composition_generator::get\(\)](#), [GINAC_ASSERT](#), [GiNaC::is_pos_integer\(\)](#), [GiNaC::ex::is_zero\(\)](#), [k](#), [mul](#), [GiNaC::multinomial_coefficient\(\)](#), [n](#), [GiNaC::partition_with_zero_parts_generator::next\(\)](#), [GiNaC::composition_generator::next\(\)](#), [GiNaC::expairseq::nops\(\)](#), [options](#), [GiNaC::expairseq::overall_coeff](#), [GiNaC::pow\(\)](#), [r](#), [GiNaC::expairseq::seq](#), and [GiNaC::numeric::to_long\(\)](#).

Referenced by [expand\(\)](#).

6.123.3.37 `expand_add_2()`

```
ex GiNaC::power::expand_add_2 (
    const add & a,
    unsigned options ) [static], [protected]
```

Special case of [power::expand_add](#).

Expands a^2 where a is an `add`.

See also

[power::expand_add](#)

References [GiNaC::_ex1](#), [GiNaC::_ex2](#), [GiNaC::_num2_p](#), [basis](#), [c](#), [GiNaC::ex::coeff\(\)](#), [GiNaC::add::combine_pair_with_coeff_to_pair](#), [expand\(\)](#), [expand_mul\(\)](#), [GiNaC::status_flags::expanded](#), [exponent](#), [GINAC_ASSERT](#), [GiNaC::is_pos_integer\(\)](#), [GiNaC::ex::is_zero\(\)](#), [last](#), [GiNaC::numeric::mul\(\)](#), [mul](#), [GiNaC::numeric::mul_dyn\(\)](#), [GiNaC::expairseq::nops\(\)](#), [options](#), [GiNaC::expairseq::overall_coeff](#), [r](#), and [GiNaC::expairseq::seq](#).

Referenced by [expand_add\(\)](#).

6.123.3.38 `expand_mul()`

```
ex GiNaC::power::expand_mul (
    const mul & m,
    const numeric & n,
    unsigned options,
    bool from_expand = false ) [static], [protected]
```

Expand factors of m in m^n where m is a `mul` and n is an integer.

See also

[power::expand](#)

References [GiNaC::_ex1](#), [GiNaC::expair::coeff](#), [GiNaC::basic::ex](#), [GiNaC::expand_options::expand_rename_idx](#), [GiNaC::status_flags::expanded](#), [GiNaC::get_all_dummy_indices\(\)](#), [GINAC_ASSERT](#), [GiNaC::info_flags::has_indices](#), [m](#), [n](#), [options](#), [GiNaC::rename_dummy_indices_uniquely\(\)](#), and [GiNaC::basic::setflag\(\)](#).

Referenced by [eval\(\)](#), [expand\(\)](#), and [expand_add_2\(\)](#).

6.123.4 Friends And Related Symbol Documentation

6.123.4.1 `mul`

```
friend class mul [friend]
```

Referenced by [expand_add\(\)](#), and [expand_add_2\(\)](#).

6.123.5 Member Data Documentation

6.123.5.1 basis

`ex` `GiNaC::power::basis` [protected]

Referenced by [archive\(\)](#), [coeff\(\)](#), [conjugate\(\)](#), [degree\(\)](#), [derivative\(\)](#), [do_print_csrc\(\)](#), [do_print_csrc_cl_N\(\)](#), [do_print_dflt\(\)](#), [do_print_latex\(\)](#), [do_print_python_repr\(\)](#), [eval\(\)](#), [evalf\(\)](#), [evalm\(\)](#), [expand\(\)](#), [expand_add\(\)](#), [expand_add_2\(\)](#), [has\(\)](#), [imag_part\(\)](#), [info\(\)](#), [is_polynomial\(\)](#), [ldegree\(\)](#), [map\(\)](#), [normal\(\)](#), [op\(\)](#), [print_power\(\)](#), [read_archive\(\)](#), [real_part\(\)](#), [return_type\(\)](#), [return_type_tinfo\(\)](#), [series\(\)](#), [GiNaC::mul::split_ex_to_pair\(\)](#), [subs\(\)](#), [to_polynomial\(\)](#), and [to_rational\(\)](#).

6.123.5.2 exponent

`ex` `GiNaC::power::exponent` [protected]

Referenced by [archive\(\)](#), [coeff\(\)](#), [conjugate\(\)](#), [degree\(\)](#), [derivative\(\)](#), [do_print_csrc\(\)](#), [do_print_csrc_cl_N\(\)](#), [do_print_dflt\(\)](#), [do_print_latex\(\)](#), [do_print_python_repr\(\)](#), [eval\(\)](#), [evalf\(\)](#), [evalm\(\)](#), [expand\(\)](#), [expand_add\(\)](#), [expand_add_2\(\)](#), [has\(\)](#), [imag_part\(\)](#), [info\(\)](#), [is_polynomial\(\)](#), [ldegree\(\)](#), [map\(\)](#), [normal\(\)](#), [op\(\)](#), [print_power\(\)](#), [read_archive\(\)](#), [real_part\(\)](#), [series\(\)](#), [GiNaC::mul::split_ex_to_pair\(\)](#), [subs\(\)](#), [to_polynomial\(\)](#), and [to_rational\(\)](#).

The documentation for this class was generated from the following files:

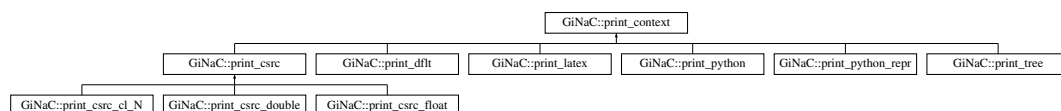
- [power.h](#)
- [normal.cpp](#)
- [power.cpp](#)
- [pseries.cpp](#)

6.124 GiNaC::print_context Class Reference

Base class for print_contexts.

```
#include <print.h>
```

Inheritance diagram for `GiNaC::print_context`:



Public Member Functions

- [print_context](#) (std::ostream &, unsigned [options](#)=0)
- virtual [~print_context](#) ()

Public Attributes

- `std::ostream & s`
stream to output to
- unsigned `options`
option flags

6.124.1 Detailed Description

Base class for `print_contexts`.

6.124.2 Constructor & Destructor Documentation

6.124.2.1 `print_context()`

```
GiNaC::print_context::print_context (
    std::ostream & os,
    unsigned options = 0 )
```

6.124.2.2 `~print_context()`

```
virtual GiNaC::print_context::~~print_context ( ) [inline], [virtual]
```

6.124.3 Member Data Documentation

6.124.3.1 `s`

```
std::ostream& GiNaC::print_context::s
```

stream to output to

Referenced by [GiNaC::function::print\(\)](#), and [GiNaC::numeric::print_numeric\(\)](#).

6.124.3.2 `options`

```
unsigned GiNaC::print_context::options
```

option flags

Referenced by [GiNaC::get_print_options\(\)](#), [GiNaC::set_print_context\(\)](#), and [GiNaC::set_print_options\(\)](#).

The documentation for this class was generated from the following files:

- [print.h](#)
- [print.cpp](#)

6.125 GiNaC::print_context_options Class Reference

This class stores information about a registered [print_context](#) class.

```
#include <print.h>
```

Public Member Functions

- [print_context_options](#) (const char **n*, const char **p*, unsigned *i*)
- const char * [get_name](#) () const
- const char * [get_parent_name](#) () const
- unsigned [get_id](#) () const

Private Attributes

- const char * [name](#)
Class name.
- const char * [parent_name](#)
Name of superclass.
- unsigned [id](#)
ID number (assigned automatically).

6.125.1 Detailed Description

This class stores information about a registered [print_context](#) class.

6.125.2 Constructor & Destructor Documentation

6.125.2.1 print_context_options()

```
GiNaC::print_context_options::print_context_options (  
    const char * n,  
    const char * p,  
    unsigned i ) [inline]
```

6.125.3 Member Function Documentation

6.125.3.1 get_name()

```
const char * GiNaC::print_context_options::get_name ( ) const [inline]
```

References [name](#).

6.125.3.2 `get_parent_name()`

```
const char * GiNaC::print_context_options::get_parent_name ( ) const [inline]
```

References [parent_name](#).

6.125.3.3 `get_id()`

```
unsigned GiNaC::print_context_options::get_id ( ) const [inline]
```

References [id](#).

6.125.4 Member Data Documentation

6.125.4.1 `name`

```
const char* GiNaC::print_context_options::name [private]
```

Class name.

Referenced by [get_name\(\)](#).

6.125.4.2 `parent_name`

```
const char* GiNaC::print_context_options::parent_name [private]
```

Name of superclass.

Referenced by [get_parent_name\(\)](#).

6.125.4.3 `id`

```
unsigned GiNaC::print_context_options::id [private]
```

ID number (assigned automatically).

Referenced by [get_id\(\)](#).

The documentation for this class was generated from the following file:

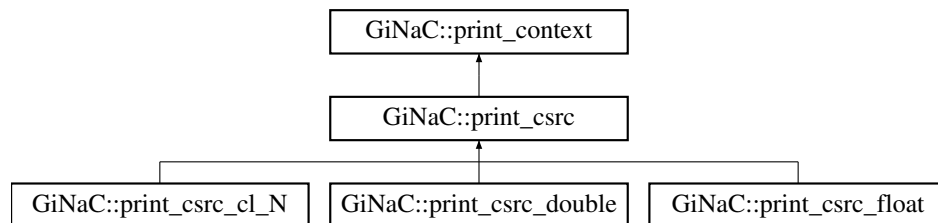
- [print.h](#)

6.126 GiNaC::print_csrc Class Reference

Base context for C source output.

```
#include <print.h>
```

Inheritance diagram for GiNaC::print_csrc:



Public Member Functions

- [print_csrc](#) (std::ostream &, unsigned [options](#)=0)

Public Member Functions inherited from [GiNaC::print_context](#)

- [print_context](#) (std::ostream &, unsigned [options](#)=0)
- virtual [~print_context](#) ()

Additional Inherited Members

Public Attributes inherited from [GiNaC::print_context](#)

- std::ostream & [s](#)
stream to output to
- unsigned [options](#)
option flags

6.126.1 Detailed Description

Base context for C source output.

6.126.2 Constructor & Destructor Documentation

6.126.2.1 [print_csrc\(\)](#)

```

GiNaC::print_csrc::print_csrc (
    std::ostream & os,
    unsigned options = 0 )

```

The documentation for this class was generated from the following files:

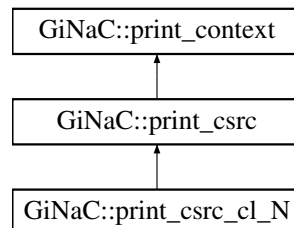
- [print.h](#)
- [print.cpp](#)

6.127 GiNaC::print_csrc_cl_N Class Reference

Context for C source output using CLN numbers.

```
#include <print.h>
```

Inheritance diagram for GiNaC::print_csrc_cl_N:



Public Member Functions

- [print_csrc_cl_N](#) (std::ostream &, unsigned [options](#)=0)

Public Member Functions inherited from [GiNaC::print_csrc](#)

- [print_csrc](#) (std::ostream &, unsigned [options](#)=0)

Public Member Functions inherited from [GiNaC::print_context](#)

- [print_context](#) (std::ostream &, unsigned [options](#)=0)
- virtual [~print_context](#) ()

Additional Inherited Members

Public Attributes inherited from [GiNaC::print_context](#)

- std::ostream & [s](#)
stream to output to
- unsigned [options](#)
option flags

6.127.1 Detailed Description

Context for C source output using CLN numbers.

6.127.2 Constructor & Destructor Documentation

6.127.2.1 print_csrc_cl_N()

```
GiNaC::print_csrc_cl_N::print_csrc_cl_N (
    std::ostream & os,
    unsigned options = 0 )
```

The documentation for this class was generated from the following files:

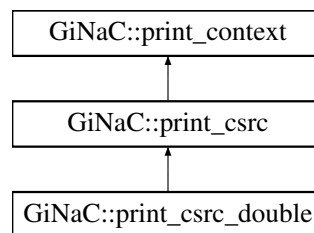
- [print.h](#)
- [print.cpp](#)

6.128 GiNaC::print_csrc_double Class Reference

Context for C source output using double precision.

```
#include <print.h>
```

Inheritance diagram for GiNaC::print_csrc_double:



Public Member Functions

- [print_csrc_double](#) (std::ostream &, unsigned options=0)

Public Member Functions inherited from [GiNaC::print_csrc](#)

- [print_csrc](#) (std::ostream &, unsigned options=0)

Public Member Functions inherited from [GiNaC::print_context](#)

- [print_context](#) (std::ostream &, unsigned options=0)
- virtual [~print_context](#) ()

Additional Inherited Members

Public Attributes inherited from [GiNaC::print_context](#)

- std::ostream & [s](#)
stream to output to
- unsigned [options](#)
option flags

6.128.1 Detailed Description

Context for C source output using double precision.

6.128.2 Constructor & Destructor Documentation

6.128.2.1 `print_csrc_double()`

```
GiNaC::print_csrc_double::print_csrc_double (
    std::ostream & os,
    unsigned options = 0 )
```

The documentation for this class was generated from the following files:

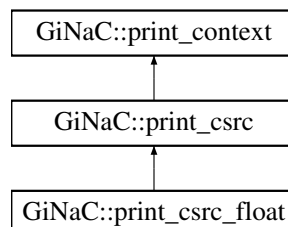
- [print.h](#)
- [print.cpp](#)

6.129 GiNaC::print_csrc_float Class Reference

Context for C source output using float precision.

```
#include <print.h>
```

Inheritance diagram for GiNaC::print_csrc_float:



Public Member Functions

- [print_csrc_float](#) (std::ostream &, unsigned options=0)

Public Member Functions inherited from [GiNaC::print_csrc](#)

- [print_csrc](#) (std::ostream &, unsigned options=0)

Public Member Functions inherited from [GiNaC::print_context](#)

- [print_context](#) (std::ostream &, unsigned options=0)
- virtual [~print_context](#) ()

Additional Inherited Members

Public Attributes inherited from [GiNaC::print_context](#)

- `std::ostream & s`
stream to output to
- unsigned [options](#)
option flags

6.129.1 Detailed Description

Context for C source output using float precision.

6.129.2 Constructor & Destructor Documentation

6.129.2.1 `print_csrc_float()`

```
GiNaC::print_csrc_float::print_csrc_float (
    std::ostream & os,
    unsigned options = 0 )
```

The documentation for this class was generated from the following files:

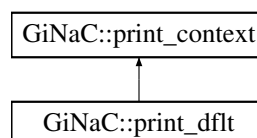
- [print.h](#)
- [print.cpp](#)

6.130 GiNaC::print_dflt Class Reference

Context for default (ginsh-parsable) output.

```
#include <print.h>
```

Inheritance diagram for `GiNaC::print_dflt`:



Public Member Functions

- [print_dflt](#) (`std::ostream &`, unsigned [options](#)=0)

Public Member Functions inherited from [GiNaC::print_context](#)

- [print_context](#) (std::ostream &, unsigned [options](#)=0)
- virtual [~print_context](#) ()

Additional Inherited Members

Public Attributes inherited from [GiNaC::print_context](#)

- std::ostream & [s](#)
stream to output to
- unsigned [options](#)
option flags

6.130.1 Detailed Description

Context for default (ginsh-parsable) output.

6.130.2 Constructor & Destructor Documentation

6.130.2.1 [print_dflt\(\)](#)

```
GiNaC::print_dflt::print_dflt (
    std::ostream & os,
    unsigned options = 0 )
```

The documentation for this class was generated from the following files:

- [print.h](#)
- [print.cpp](#)

6.131 GiNaC::print_functor Class Reference

This class represents a print method for a certain algebraic class and [print_context](#) type.

```
#include <print.h>
```

Public Member Functions

- [print_functor](#) ()
- [print_functor](#) (const [print_functor](#) &other)
- [print_functor](#) (std::unique_ptr< [print_functor_impl](#) > impl_)
- template<class T, class C >
[print_functor](#) (void f(const T &, const C &, unsigned))
- template<class T, class C >
[print_functor](#) (void(T::*f)(const C &, unsigned) const)
- [print_functor](#) & [operator=](#) (const [print_functor](#) &other)
- void [operator\(\)](#) (const [basic](#) &obj, const [print_context](#) &c, unsigned level) const
- bool [is_valid](#) () const

Private Attributes

- `std::unique_ptr< print\_functor\_impl > impl`

6.131.1 Detailed Description

This class represents a print method for a certain algebraic class and [print_context](#) type.

Its main purpose is to hide the difference between member functions and nonmember functions behind one unified `operator()` interface. [print_functor](#) has value semantics and acts as a smart pointer (with deep copy) to a class derived from [print_functor_impl](#) which implements the actual function call.

6.131.2 Constructor & Destructor Documentation

6.131.2.1 `print_functor()` [1/5]

```
GiNaC::print_functor::print_functor ( ) [inline]
```

6.131.2.2 `print_functor()` [2/5]

```
GiNaC::print_functor::print_functor (
    const print\_functor & other ) [inline]
```

6.131.2.3 `print_functor()` [3/5]

```
GiNaC::print_functor::print_functor (
    std::unique_ptr< print\_functor\_impl > impl_ ) [inline]
```

6.131.2.4 `print_functor()` [4/5]

```
template<class T , class C >
GiNaC::print_functor::print_functor (
    void fconst T &, const C &, unsigned ) [inline]
```

6.131.2.5 `print_functor()` [5/5]

```
template<class T , class C >
GiNaC::print_functor::print_functor (
    void(T::*)(const C &, unsigned) const f ) [inline]
```

6.131.3 Member Function Documentation

6.131.3.1 `operator=()`

```
print\_functor & GiNaC::print_functor::operator= (
    const print\_functor & other ) [inline]
```

References [impl](#).

6.131.3.2 operator>()

```
void GiNaC::print_functor::operator() (
    const basic & obj,
    const print\_context & c,
    unsigned level ) const [inline]
```

References [c](#).

6.131.3.3 is_valid()

```
bool GiNaC::print_functor::is_valid ( ) const [inline]
```

References [impl](#).

6.131.4 Member Data Documentation

6.131.4.1 impl

```
std::unique_ptr<print\_functor\_impl> GiNaC::print_functor::impl [private]
```

Referenced by [is_valid\(\)](#), and [operator=\(\)](#).

The documentation for this class was generated from the following file:

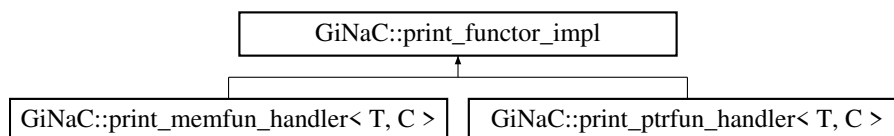
- [print.h](#)

6.132 GiNaC::print_functor_impl Class Reference

Base class for [print_functor](#) handlers.

```
#include <print.h>
```

Inheritance diagram for GiNaC::print_functor_impl:



Public Member Functions

- virtual [~print_functor_impl](#) ()
- virtual [print_functor_impl](#) * [duplicate](#) () const =0
- virtual void [operator\(\)](#) (const [basic](#) &obj, const [print_context](#) &c, unsigned level) const =0

6.132.1 Detailed Description

Base class for [print_functor](#) handlers.

6.132.2 Constructor & Destructor Documentation

6.132.2.1 ~print_functor_impl()

```
virtual GiNaC::print_functor_impl::~print_functor_impl ( ) [inline], [virtual]
```

6.132.3 Member Function Documentation

6.132.3.1 duplicate()

```
virtual print\_functor\_impl * GiNaC::print_functor_impl::duplicate ( ) const [pure virtual]
```

Implemented in [GiNaC::print_ptrfun_handler< T, C >](#), and [GiNaC::print_memfun_handler< T, C >](#).

6.132.3.2 operator>()()

```
virtual void GiNaC::print_functor_impl::operator() (
    const basic & obj,
    const print\_context & c,
    unsigned level ) const [pure virtual]
```

Implemented in [GiNaC::print_ptrfun_handler< T, C >](#), and [GiNaC::print_memfun_handler< T, C >](#).

The documentation for this class was generated from the following file:

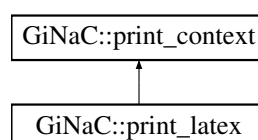
- [print.h](#)

6.133 GiNaC::print_latex Class Reference

Context for latex-parsable output.

```
#include <print.h>
```

Inheritance diagram for GiNaC::print_latex:



Public Member Functions

- [print_latex](#) (std::ostream &, unsigned [options](#)=0)

Public Member Functions inherited from [GiNaC::print_context](#)

- [print_context](#) (std::ostream &, unsigned [options](#)=0)
- virtual [~print_context](#) ()

Additional Inherited Members

Public Attributes inherited from [GiNaC::print_context](#)

- std::ostream & [s](#)
stream to output to
- unsigned [options](#)
option flags

6.133.1 Detailed Description

Context for latex-parsable output.

6.133.2 Constructor & Destructor Documentation

6.133.2.1 [print_latex\(\)](#)

```
GiNaC::print_latex::print_latex (
    std::ostream & os,
    unsigned options = 0 )
```

The documentation for this class was generated from the following files:

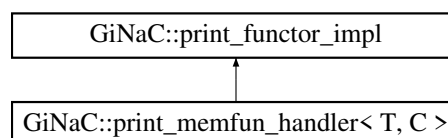
- [print.h](#)
- [print.cpp](#)

6.134 [GiNaC::print_memfun_handler< T, C >](#) Class Template Reference

[print_functor](#) handler for member functions of class T, context type C

```
#include <print.h>
```

Inheritance diagram for [GiNaC::print_memfun_handler< T, C >](#):



Public Types

- typedef void(T::* [F](#)) (const C &[c](#), unsigned level) const

Public Member Functions

- [print_memfun_handler](#) ([F](#) [f_](#))
- [print_memfun_handler](#) * [duplicate](#) () const override
- void [operator](#)() (const [basic](#) &obj, const [print_context](#) &[c](#), unsigned level) const override

Public Member Functions inherited from [GiNaC::print_funcutor_impl](#)

- virtual [~print_funcutor_impl](#) ()

Private Attributes

- [F](#) [f](#)

6.134.1 Detailed Description

```
template<class T, class C>
class GiNaC::print_memfun_handler< T, C >
```

[print_funcutor](#) handler for member functions of class T, context type C

6.134.2 Member Typedef Documentation

6.134.2.1 [F](#)

```
template<class T , class C >
typedef void(T::* GiNaC::print\_memfun\_handler< T, C >::F) (const C &c, unsigned level) const
```

6.134.3 Constructor & Destructor Documentation

6.134.3.1 [print_memfun_handler](#)()

```
template<class T , class C >
GiNaC::print\_memfun\_handler< T, C >::print_memfun_handler (
    F f\_ ) [inline]
```

6.134.4 Member Function Documentation

6.134.4.1 [duplicate](#)()

```
template<class T , class C >
print\_memfun\_handler * GiNaC::print\_memfun\_handler< T, C >::duplicate ( ) const [inline],
[override], [virtual]
```

Implements [GiNaC::print_funcutor_impl](#).

6.134.4.2 `operator>()()`

```
template<class T , class C >
void GiNaC::print_memfun_handler< T, C >::operator() (
    const basic & obj,
    const print_context & c,
    unsigned level ) const [inline], [override], [virtual]
```

Implements [GiNaC::print_functor_impl](#).

References [c](#), and [GiNaC::print_memfun_handler< T, C >::f](#).

6.134.5 Member Data Documentation

6.134.5.1 `f`

```
template<class T , class C >
F GiNaC::print_memfun_handler< T, C >::f [private]
```

Referenced by [GiNaC::print_memfun_handler< T, C >::operator>\(\)\(\)](#).

The documentation for this class was generated from the following file:

- [print.h](#)

6.135 GiNaC::print_options Class Reference

Flags to control the behavior of a [print_context](#).

```
#include <print.h>
```

Public Types

- enum { [print_index_dimensions](#) = 0x0001 }

6.135.1 Detailed Description

Flags to control the behavior of a [print_context](#).

6.135.2 Member Enumeration Documentation

6.135.2.1 anonymous enum

```
anonymous enum
```

Enumerator

print_index_dimensions	print the dimensions of indices

The documentation for this class was generated from the following file:

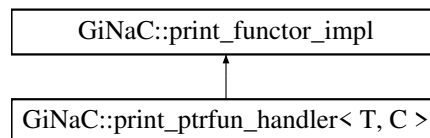
- [print.h](#)

6.136 GiNaC::print_ptrfun_handler< T, C > Class Template Reference

[print_functor](#) handler for pointer-to-functions of class T, context type C

```
#include <print.h>
```

Inheritance diagram for GiNaC::print_ptrfun_handler< T, C >:



Public Types

- typedef void(* [F](#)) (const T &, const C &, unsigned)

Public Member Functions

- [print_ptrfun_handler](#) (F f_)
- [print_ptrfun_handler](#) * [duplicate](#) () const override
- void [operator\(\)](#) (const [basic](#) &obj, const [print_context](#) &c, unsigned level) const override

Public Member Functions inherited from [GiNaC::print_functor_impl](#)

- virtual [~print_functor_impl](#) ()

Private Attributes

- [F f](#)

6.136.1 Detailed Description

```
template<class T, class C>
class GiNaC::print_ptrfun_handler< T, C >
```

[print_functor](#) handler for pointer-to-functions of class T, context type C

6.136.2 Member Typedef Documentation

6.136.2.1 F

```
template<class T , class C >
typedef void(* GiNaC::print\_ptrfun\_handler< T, C >::F) (const T &, const C &, unsigned)
```

6.136.3 Constructor & Destructor Documentation

6.136.3.1 [print_ptrfun_handler\(\)](#)

```
template<class T , class C >
GiNaC::print\_ptrfun\_handler< T, C >::print_ptrfun_handler (
    F f_ ) [inline]
```

6.136.4 Member Function Documentation

6.136.4.1 [duplicate\(\)](#)

```
template<class T , class C >
print\_ptrfun\_handler * GiNaC::print\_ptrfun\_handler< T, C >::duplicate ( ) const [inline],
[override], [virtual]
```

Implements [GiNaC::print_functor_impl](#).

6.136.4.2 [operator>\(\)\(\)](#)

```
template<class T , class C >
void GiNaC::print\_ptrfun\_handler< T, C >::operator() (
    const basic & obj,
    const print\_context & c,
    unsigned level ) const [inline], [override], [virtual]
```

Implements [GiNaC::print_functor_impl](#).

References [c](#), and [GiNaC::print_ptrfun_handler< T, C >::f](#).

6.136.5 Member Data Documentation

6.136.5.1 f

```
template<class T , class C >
F GiNaC::print\_ptrfun\_handler< T, C >::f [private]
```

Referenced by [GiNaC::print_ptrfun_handler< T, C >::operator>\(\)\(\)](#).

The documentation for this class was generated from the following file:

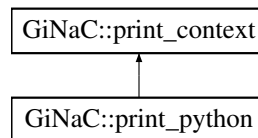
- [print.h](#)

6.137 GiNaC::print_python Class Reference

Context for python pretty-print output.

```
#include <print.h>
```

Inheritance diagram for GiNaC::print_python:



Public Member Functions

- [print_python](#) (std::ostream &, unsigned [options](#)=0)

Public Member Functions inherited from [GiNaC::print_context](#)

- [print_context](#) (std::ostream &, unsigned [options](#)=0)
- virtual [~print_context](#) ()

Additional Inherited Members

Public Attributes inherited from [GiNaC::print_context](#)

- std::ostream & [s](#)
stream to output to
- unsigned [options](#)
option flags

6.137.1 Detailed Description

Context for python pretty-print output.

6.137.2 Constructor & Destructor Documentation

6.137.2.1 print_python()

```
GiNaC::print_python::print_python (
    std::ostream & os,
    unsigned options = 0 )
```

The documentation for this class was generated from the following files:

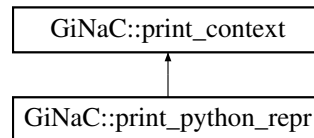
- [print.h](#)
- [print.cpp](#)

6.138 GiNaC::print_python_repr Class Reference

Context for python-parsable output.

```
#include <print.h>
```

Inheritance diagram for GiNaC::print_python_repr:



Public Member Functions

- [print_python_repr](#) (std::ostream &, unsigned [options](#)=0)

Public Member Functions inherited from [GiNaC::print_context](#)

- [print_context](#) (std::ostream &, unsigned [options](#)=0)
- virtual [~print_context](#) ()

Additional Inherited Members

Public Attributes inherited from [GiNaC::print_context](#)

- std::ostream & [s](#)
stream to output to
- unsigned [options](#)
option flags

6.138.1 Detailed Description

Context for python-parsable output.

6.138.2 Constructor & Destructor Documentation

6.138.2.1 print_python_repr()

```
GiNaC::print_python_repr::print_python_repr (
    std::ostream & os,
    unsigned options = 0 )
```

The documentation for this class was generated from the following files:

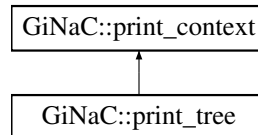
- [print.h](#)
- [print.cpp](#)

6.139 GiNaC::print_tree Class Reference

Context for tree-like output for debugging.

```
#include <print.h>
```

Inheritance diagram for GiNaC::print_tree:



Public Member Functions

- [print_tree](#) (unsigned d)
- [print_tree](#) (std::ostream &, unsigned [options](#)=0, unsigned d=4)

Public Member Functions inherited from [GiNaC::print_context](#)

- [print_context](#) (std::ostream &, unsigned [options](#)=0)
- virtual [~print_context](#) ()

Public Attributes

- const unsigned [delta_indent](#)
size of indentation step

Public Attributes inherited from [GiNaC::print_context](#)

- std::ostream & [s](#)
stream to output to
- unsigned [options](#)
option flags

6.139.1 Detailed Description

Context for tree-like output for debugging.

6.139.2 Constructor & Destructor Documentation

6.139.2.1 [print_tree\(\)](#) [1/2]

```
GiNaC::print_tree::print_tree (
    unsigned d )
```

6.139.2.2 `print_tree()` [2/2]

```
GiNaC::print_tree::print_tree (
    std::ostream & os,
    unsigned options = 0,
    unsigned d = 4 )
```

6.139.3 Member Data Documentation

6.139.3.1 `delta_indent`

```
const unsigned GiNaC::print_tree::delta_indent
```

size of indentation step

The documentation for this class was generated from the following files:

- [print.h](#)
- [print.cpp](#)

6.140 `GiNaC::archive_node::property` Struct Reference

Archived property (data type, name and associated data)

```
#include <archive.h>
```

Public Member Functions

- [property](#) ()
- [property](#) ([archive_atom](#) n, [property_type](#) t, unsigned v)

Public Attributes

- [property_type](#) type
Data type of property.
- [archive_atom](#) name
Name of property.
- unsigned [value](#)
Stored value.

6.140.1 Detailed Description

Archived property (data type, name and associated data)

6.140.2 Constructor & Destructor Documentation

6.140.2.1 property() [1/2]

```
GiNaC::archive_node::property::property ( ) [inline]
```

6.140.2.2 property() [2/2]

```
GiNaC::archive_node::property::property (
    archive_atom n,
    property_type t,
    unsigned v ) [inline]
```

6.140.3 Member Data Documentation

6.140.3.1 type

```
property_type GiNaC::archive_node::property::type
```

Data type of property.

6.140.3.2 name

```
archive_atom GiNaC::archive_node::property::name
```

Name of property.

6.140.3.3 value

```
unsigned GiNaC::archive_node::property::value
```

Stored value.

The documentation for this struct was generated from the following file:

- [archive.h](#)

6.141 GiNaC::archive_node::property_info Struct Reference

Information about a stored property.

```
#include <archive.h>
```

Public Member Functions

- [property_info](#) ()
- [property_info](#) ([property_type](#) t, const std::string &n, unsigned c=1)

Public Attributes

- [property_type](#) type
Data type of property.
- std::string name
Name of property.
- unsigned count
Number of occurrences.

6.141.1 Detailed Description

Information about a stored property.

A vector of these structures is returned by [get_properties\(\)](#).

See also

[get_properties](#)

6.141.2 Constructor & Destructor Documentation

6.141.2.1 [property_info](#)() [1/2]

```
GiNaC::archive_node::property_info::property_info ( ) [inline]
```

6.141.2.2 [property_info](#)() [2/2]

```
GiNaC::archive_node::property_info::property_info (
    property\_type t,
    const std::string & n,
    unsigned c = 1 ) [inline]
```

6.141.3 Member Data Documentation

6.141.3.1 [type](#)

[property_type](#) GiNaC::archive_node::property_info::type

Data type of property.

6.141.3.2 name

```
std::string GiNaC::archive_node::property_info::name
```

Name of property.

6.141.3.3 count

```
unsigned GiNaC::archive_node::property_info::count
```

Number of occurrences.

The documentation for this struct was generated from the following file:

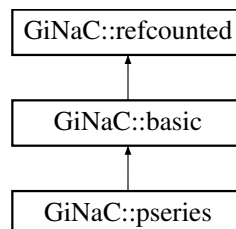
- [archive.h](#)

6.142 GiNaC::pseries Class Reference

This class holds a extended truncated power series (positive and negative integer powers).

```
#include <pseries.h>
```

Inheritance diagram for GiNaC::pseries:



Public Member Functions

- [pseries](#) (const [ex](#) &rel_, const [epvector](#) &ops_)
Construct pseries from a vector of coefficients and powers.
- [pseries](#) (const [ex](#) &rel_, [epvector](#) &&ops_)
- unsigned [precedence](#) () const override
Return relative operator precedence (for parenthezing output).
- size_t [nops](#) () const override
Return the number of operands including a possible order term.
- [ex](#) [op](#) (size_t i) const override
Return the ith term in the series when represented as a sum.
- int [degree](#) (const [ex](#) &s) const override
Return degree of highest power of the series.
- int [ldegree](#) (const [ex](#) &s) const override
Return degree of lowest power of the series.
- [ex](#) [coeff](#) (const [ex](#) &s, int n=1) const override

- Return coefficient of degree n in power series if s is the expansion variable.*

 - `ex collect` (const `ex` & s , bool distributed=false) const override

Does nothing.
- `ex eval` () const override

Perform coefficient-wise automatic term rewriting rules in this class.
- `ex evalf` () const override

Evaluate coefficients numerically.
- `ex series` (const `relational` & r , int `order`, unsigned `options`=0) const override

Re-expansion of a `pseries` object.
- `ex subs` (const `exmap` & m , unsigned `options`=0) const override

Substitute a set of objects by arbitrary expressions.
- `ex normal` (`exmap` & $repl$, `exmap` & rev_lookup , `lst` & $modifier$) const override

Implementation of `ex::normal()` for `pseries`.
- `ex expand` (unsigned `options`=0) const override

Implementation of `ex::expand()` for a power series.
- `ex conjugate` () const override
- `ex real_part` () const override
- `ex imag_part` () const override
- `ex eval_integ` () const override

Evaluate integrals, if result is known.
- `ex evalm` () const override

Evaluate sums, products and integer powers of matrices.
- void `archive` (`archive_node` & n) const override

Save (a.k.a.
- void `read_archive` (const `archive_node` & n , `lst` & $syms$) override

Read (a.k.a.
- `ex get_var` () const

Get the expansion variable.
- `ex get_point` () const

Get the expansion point.
- `ex convert_to_poly` (bool `no_order`=false) const

Convert the `pseries` object to an ordinary polynomial.
- bool `is_compatible_to` (const `pseries` & $other$) const

Check whether series is compatible to another series (expansion variable and point are the same.
- bool `is_zero` () const

Check whether series has the value zero.
- bool `is_terminating` () const

Returns true if there is no order term, i.e.
- `ex coeffop` (size_t i) const

Get coefficients and exponents.
- `ex exponop` (size_t i) const
- `ex add_series` (const `pseries` & $other$) const

Add one series object to another, producing a `pseries` object that represents the sum.
- `ex mul_const` (const `numeric` & $other$) const

Multiply a `pseries` object with a numeric constant, producing a `pseries` object that represents the product.
- `ex mul_series` (const `pseries` & $other$) const

Multiply one `pseries` object to another, producing a `pseries` object that represents the product.
- `ex power_const` (const `numeric` & p , int `deg`) const

Compute the p -th power of a series.
- `pseries shift_exponents` (int `deg`) const

Return a new `pseries` object with the powers shifted by `deg`.

Public Member Functions inherited from GiNaC::basic

- virtual `~basic ()`
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic (const basic &other)`
- `const basic & operator= (const basic &other)`
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate () const`
Create a clone of this object on the heap.
- virtual `ex eval_indexed (const basic &i) const`
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual `void print (const print_context &c, unsigned level=0) const`
Output to stream.
- virtual `void dbgprint () const`
Little wrapper around print to be called within a debugger.
- virtual `void dbgprinttree () const`
Little wrapper around printtree to be called within a debugger.
- virtual `bool info (unsigned inf) const`
Information about the object.
- virtual `ex operator[] (const ex &index) const`
- virtual `ex operator[] (size_t i) const`
- virtual `ex & let_op (size_t i)`
Return modifiable operand/member at position i.
- virtual `ex & operator[] (const ex &index)`
- virtual `ex & operator[] (size_t i)`
- virtual `bool has (const ex &other, unsigned options=0) const`
Test for occurrence of a pattern.
- virtual `bool match (const ex &pattern, exmap &repls) const`
Check whether the expression matches a given pattern.
- virtual `ex map (map_function &f) const`
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual `void accept (GiNaC::visitor &v) const`
- virtual `bool is_polynomial (const ex &var) const`
Check whether this is a polynomial in the given variables.
- virtual `ex to_rational (exmap &repl) const`
Default implementation of ex::to_rational().
- virtual `ex to_polynomial (exmap &repl) const`
- virtual `numeric integer_content () const`
- virtual `ex smod (const numeric &xi) const`
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient () const`
Implementation ex::max_coefficient().
- virtual `exvector get_free_indices () const`
Return a vector containing the free indices of an expression.
- virtual `ex add_indexed (const ex &self, const ex &other) const`
Add two indexed expressions.
- virtual `ex scalar_mul_indexed (const ex &self, const numeric &other) const`
Multiply an indexed expression with a scalar.
- virtual `bool contract_with (exvector::iterator self, exvector::iterator other, exvector &v) const`
Try to contract two indexed expressions that appear in the same product.
- virtual `unsigned return_type () const`
- virtual `return_type_t return_type_info () const`

- `template<class T >`
`void print_dispatch (const print_context &c, unsigned level) const`
Like `print()`, but dispatch to the specified class.
- `void print_dispatch (const registered_class_info &ri, const print_context &c, unsigned level) const`
Like `print()`, but dispatch to the specified class.
- `ex subs_one_level (const exmap &m, unsigned options) const`
Helper function for `subs()`.
- `ex diff (const symbol &s, unsigned nth=1) const`
Default interface of `nth` derivative `ex::diff(s, n)`.
- `int compare (const basic &other) const`
Compare objects syntactically to establish canonical ordering.
- `bool is_equal (const basic &other) const`
Test for syntactic equality.
- `const basic & hold () const`
Stop further evaluation.
- `unsigned gethash () const`
- `const basic & setflag (unsigned f) const`
Set some `status_flags`.
- `const basic & clearflag (unsigned f) const`
Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted ()` noexcept
- `unsigned int add_reference ()` noexcept
- `unsigned int remove_reference ()` noexcept
- `unsigned int get_refcount ()` const noexcept
- `void set_refcount (unsigned int r)` noexcept

Protected Member Functions

- `ex derivative (const symbol &s) const` override
Implementation of `ex::diff()` for a power series.
- `void print_series (const print_context &c, const char *openbrace, const char *closebrace, const char *mul←_sym, const char *pow_sym, unsigned level) const`
- `void do_print (const print_context &c, unsigned level) const`
- `void do_print_latex (const print_latex &c, unsigned level) const`
- `void do_print_tree (const print_tree &c, unsigned level) const`
- `void do_print_python (const print_python &c, unsigned level) const`
- `void do_print_python_repr (const print_python_repr &c, unsigned level) const`

Protected Member Functions inherited from `GiNaC::basic`

- `basic ()`
- `virtual ex eval_ncmul (const exvector &v) const`
- `virtual bool match_same_type (const basic &other) const`
Returns true if the attributes of two objects are similar enough for a match.
- `virtual int compare_same_type (const basic &other) const`
Returns order relation between two objects of same type.
- `virtual bool is_equal_same_type (const basic &other) const`

- Returns true if two objects of same type are equal.*
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Protected Attributes

- [epvector seq](#)
Vector of {coefficient, power} pairs.
- [ex var](#)
Series variable (holds a symbol)
- [ex point](#)
Expansion point.

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.142.1 Detailed Description

This class holds a extended truncated power series (positive and negative integer powers).

It consists of expression coefficients (only non-zero coefficients are stored), an expansion variable and an expansion point. Other classes must provide members to convert into this type.

6.142.2 Constructor & Destructor Documentation

6.142.2.1 [pseries\(\)](#) [1/2]

```
GiNaC::pseries::pseries (
    const ex & rel_,
    const epvector & ops_ )
```

Construct pseries from a vector of coefficients and powers.

[expair.rest](#) holds the coefficient, [expair.coeff](#) holds the power. The powers must be integers (positive or negative) and in ascending order; the last coefficient can be `Order(_ex1)` to represent a truncated, non-terminating series.

Parameters

$rel \leftrightarrow$ —	expansion variable and point (must hold a relational)
$ops \leftrightarrow$ —	vector of {coefficient, power} pairs (coefficient must not be zero)

References [GINAC_ASSERT](#), [GiNaC::is_order_function\(\)](#), [GiNaC::ex::lhs\(\)](#), [point](#), [GiNaC::ex::rhs\(\)](#), [seq](#), and [var](#).

6.142.2.2 pseries() [2/2]

```
GiNaC::pseries::pseries (
    const ex & rel_,
    epvector && ops_ )
```

References [GINAC_ASSERT](#), [GiNaC::is_order_function\(\)](#), [GiNaC::ex::lhs\(\)](#), [point](#), [GiNaC::ex::rhs\(\)](#), [seq](#), and [var](#).

6.142.3 Member Function Documentation**6.142.3.1 precedence()**

```
unsigned GiNaC::pseries::precedence ( ) const [inline], [override], [virtual]
```

Return relative operator precedence (for parenthezing output).

Reimplemented from [GiNaC::basic](#).

Referenced by [print_series\(\)](#).

6.142.3.2 nops()

```
size_t GiNaC::pseries::nops ( ) const [override], [virtual]
```

Return the number of operands including a possible order term.

Reimplemented from [GiNaC::basic](#).

References [seq](#).

Referenced by [coeffop\(\)](#), [exponop\(\)](#), and [GiNaC::log_series\(\)](#).

6.142.3.3 op()

```
ex GiNaC::pseries::op (
    size_t i ) const [override], [virtual]
```

Return the ith term in the series when represented as a sum.

Reimplemented from [GiNaC::basic](#).

References [coeff\(\)](#), [GiNaC::is_order_function\(\)](#), [point](#), [GiNaC::pow\(\)](#), [seq](#), and [var](#).

6.142.3.4 degree()

```
int GiNaC::pseries::degree (
    const ex & s ) const [override], [virtual]
```

Return degree of highest power of the series.

This is usually the exponent of the Order term. If s is not the expansion variable of the series, the series is examined termwise.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::is_equal\(\)](#), [seq](#), and [var](#).

Referenced by [mul_series\(\)](#), [power_const\(\)](#), and [series\(\)](#).

6.142.3.5 ldegree()

```
int GiNaC::pseries::ldegree (
    const ex & s ) const [override], [virtual]
```

Return degree of lowest power of the series.

This is usually the exponent of the leading term. If s is not the expansion variable of the series, the series is examined termwise. If s is the expansion variable but the expansion point is not zero the series is not expanded to find the degree. I.e.: $(1-x) + (1-x)^2 + \text{Order}((1-x)^3)$ has `ldegree(x)` 1, not 0.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::is_equal\(\)](#), [seq](#), and [var](#).

Referenced by [GiNaC::log_series\(\)](#), [mul_series\(\)](#), and [power_const\(\)](#).

6.142.3.6 coeff()

```
ex GiNaC::pseries::coeff (
    const ex & s,
    int n = 1 ) const [override], [virtual]
```

Return coefficient of degree n in power series if s is the expansion variable.

If the expansion point is nonzero, by definition the n=1 coefficient in s of $a+b*(s-z)+c*(s-z)^2+\text{Order}((s-z)^3)$ is b (assuming the expansion took place in the s in the first place). If s is not the expansion variable, an attempt is made to convert the series to a polynomial and return the corresponding coefficient from there.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex0](#), [GiNaC::ex::coeff\(\)](#), [coeff\(\)](#), [convert_to_poly\(\)](#), [GINAC_ASSERT](#), [GiNaC::ex::is_equal\(\)](#), [n](#), [seq](#), and [var](#).

Referenced by [coeff\(\)](#), [GiNaC::log_series\(\)](#), [mul_series\(\)](#), [op\(\)](#), [power_const\(\)](#), [read_archive\(\)](#), and [shift_exponents\(\)](#).

6.142.3.7 collect()

```
ex GiNaC::pseries::collect (
    const ex & s,
    bool distributed = false ) const [override], [virtual]
```

Does nothing.

Reimplemented from [GiNaC::basic](#).

6.142.3.8 eval()

```
ex GiNaC::pseries::eval ( ) const [override], [virtual]
```

Perform coefficient-wise automatic term rewriting rules in this class.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::basic::hold\(\)](#).

6.142.3.9 evalf()

```
ex GiNaC::pseries::evalf ( ) const [override], [virtual]
```

Evaluate coefficients numerically.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::status_flags::evaluated](#), [point](#), [seq](#), and [var](#).

6.142.3.10 series()

```
ex GiNaC::pseries::series (
    const relational & r,
    int order,
    unsigned options = 0 ) const [override], [virtual]
```

Re-expansion of a pseries object.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex1](#), [convert_to_poly\(\)](#), [degree\(\)](#), [GINAC_ASSERT](#), [GiNaC::ex::is_equal\(\)](#), [options](#), [order](#), [point](#), [r](#), [GiNaC::ex::rhs\(\)](#), [seq](#), [GiNaC::ex::series\(\)](#), and [var](#).

6.142.3.11 subs()

```
ex GiNaC::pseries::subs (
    const exmap & m,
    unsigned options = 0 ) const [override], [virtual]
```

Substitute a set of objects by arbitrary expressions.

The ex returned will already be evaluated.

Reimplemented from [GiNaC::basic](#).

References [convert_to_poly\(\)](#), [m](#), [options](#), [point](#), [seq](#), [GiNaC::ex::subs\(\)](#), and [var](#).

6.142.3.12 normal()

```
ex GiNaC::pseries::normal (
    exmap & repl,
    exmap & rev_lookup,
    lst & modifier ) const [override], [virtual]
```

Implementation of [ex::normal\(\)](#) for pseries.

It normalizes each coefficient and replaces the series by a temporary symbol.

See also

[ex::normal](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex1](#), [GiNaC::ex::coeff\(\)](#), [GiNaC::ex::is_zero\(\)](#), [n](#), [GiNaC::ex::normal\(\)](#), [point](#), [GiNaC::replace_with_symbol\(\)](#), [seq](#), and [var](#).

6.142.3.13 expand()

```
ex GiNaC::pseries::expand (
    unsigned options = 0 ) const [override], [virtual]
```

Implementation of [ex::expand\(\)](#) for a power series.

It expands all the terms individually and returns the resulting series as a new pseries.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::coeff\(\)](#), [GiNaC::ex::expand\(\)](#), [GiNaC::status_flags::expanded](#), [GiNaC::ex::is_zero\(\)](#), [options](#), [point](#), [seq](#), and [var](#).

6.142.3.14 conjugate()

```
ex GiNaC::pseries::conjugate ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::are_ex_trivially_equal\(\)](#), [GiNaC::ex::conjugate\(\)](#), [GiNaC::conjugateepvector\(\)](#), [GiNaC::ex::info\(\)](#), [point](#), [GiNaC::info_flags::real](#), [seq](#), and [var](#).

6.142.3.15 real_part()

```
ex GiNaC::pseries::real_part ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::info\(\)](#), [point](#), [GiNaC::info_flags::real](#), [GiNaC::ex::real_part\(\)](#), [seq](#), and [var](#).

6.142.3.16 imag_part()

```
ex GiNaC::pseries::imag_part ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::info\(\)](#), [point](#), [GiNaC::info_flags::real](#), [GiNaC::ex::real_part\(\)](#), [seq](#), and [var](#).

6.142.3.17 eval_integ()

```
ex GiNaC::pseries::eval_integ ( ) const [override], [virtual]
```

Evaluate integrals, if result is known.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::are_ex_trivially_equal\(\)](#), [GiNaC::ex::coeff\(\)](#), [GiNaC::ex::eval_integ\(\)](#), [point](#), [seq](#), and [var](#).

6.142.3.18 evalm()

```
ex GiNaC::pseries::evalm ( ) const [override], [virtual]
```

Evaluate sums, products and integer powers of matrices.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::are_ex_trivially_equal\(\)](#), [GiNaC::ex::coeff\(\)](#), [GiNaC::ex::evalm\(\)](#), [GiNaC::ex::is_zero\(\)](#), [point](#), [seq](#), and [var](#).

6.142.3.19 archive()

```
void GiNaC::pseries::archive (
    archive_node & n ) const [override], [virtual]
```

Save (a.k.a.

serialize) object into archive.

Reimplemented from [GiNaC::basic](#).

References [n](#), [point](#), [seq](#), and [var](#).

6.142.3.20 read_archive()

```
void GiNaC::pseries::read_archive (
    const archive_node & n,
    lst & syms ) [override], [virtual]
```

Read (a.k.a.

deserialize) object from archive.

Reimplemented from [GiNaC::basic](#).

References [coeff\(\)](#), [n](#), [point](#), [seq](#), and [var](#).

6.142.3.21 derivative()

```
ex GiNaC::pseries::derivative (
    const symbol & s ) const [override], [protected], [virtual]
```

Implementation of [ex::diff\(\)](#) for a power series.

See also

[ex::diff](#)

Reimplemented from [GiNaC::basic](#).

References [c](#), [GiNaC::ex::coeff\(\)](#), [GiNaC::ex::diff\(\)](#), [GiNaC::is_order_function\(\)](#), [point](#), [seq](#), and [var](#).

6.142.3.22 get_var()

```
ex GiNaC::pseries::get_var ( ) const [inline]
```

Get the expansion variable.

References [var](#).

6.142.3.23 get_point()

```
ex GiNaC::pseries::get_point ( ) const [inline]
```

Get the expansion point.

References [point](#).

6.142.3.24 convert_to_poly()

```
ex GiNaC::pseries::convert_to_poly (
    bool no_order = false ) const
```

Convert the pseries object to an ordinary polynomial.

Parameters

<i>no_order</i>	flag: discard higher order terms

References [GiNaC::ex::coeff\(\)](#), [GiNaC::is_order_function\(\)](#), [point](#), [GiNaC::pow\(\)](#), [seq](#), and [var](#).

Referenced by [coeff\(\)](#), [series\(\)](#), and [subs\(\)](#).

6.142.3.25 is_compatible_to()

```
bool GiNaC::pseries::is_compatible_to (
    const pseries & other ) const [inline]
```

Check whether series is compatible to another series (expansion variable and point are the same).

References [GiNaC::ex::is_equal\(\)](#), [point](#), and [var](#).

Referenced by [add_series\(\)](#), and [mul_series\(\)](#).

6.142.3.26 is_zero()

```
bool GiNaC::pseries::is_zero ( ) const [inline]
```

Check whether series has the value zero.

References [seq](#).

Referenced by [power_const\(\)](#).

6.142.3.27 is_terminating()

```
bool GiNaC::pseries::is_terminating ( ) const
```

Returns true if there is no order term, i.e.

the series terminates and false otherwise.

References [GiNaC::is_order_function\(\)](#), and [seq](#).

Referenced by [GiNaC::is_terminating\(\)](#), and [GiNaC::log_series\(\)](#).

6.142.3.28 coeffop()

```
ex GiNaC::pseries::coeffop (
    size_t i ) const
```

Get coefficients and exponents.

References [nops\(\)](#), and [seq](#).

6.142.3.29 exponop()

```
ex GiNaC::pseries::exponop (
    size_t i ) const
```

References [GiNaC::ex::coeff\(\)](#), [nops\(\)](#), and [seq](#).

6.142.3.30 add_series()

```
ex GiNaC::pseries::add_series (
    const pseries & other ) const
```

Add one series object to another, producing a pseries object that represents the sum.

Parameters

<i>other</i>	pseries object to add with

Returns

the sum as a pseries

References [GiNaC::_ex0](#), [GiNaC::_ex1](#), [is_compatible_to\(\)](#), [GiNaC::is_order_function\(\)](#), [GiNaC::ex::is_zero\(\)](#), [point](#), [seq](#), and [var](#).

Referenced by [GiNaC::log_series\(\)](#).

6.142.3.31 mul_const()

```
ex GiNaC::pseries::mul_const (
    const numeric & other ) const
```

Multiply a pseries object with a numeric constant, producing a pseries object that represents the product.

Parameters

<i>other</i>	constant to multiply with

Returns

the product as a pseries

References [GiNaC::numeric::coeff\(\)](#), [GiNaC::is_order_function\(\)](#), [point](#), [seq](#), and [var](#).

Referenced by [GiNaC::mul::series\(\)](#).

6.142.3.32 mul_series()

```
ex GiNaC::pseries::mul_series (
    const pseries & other ) const
```

Multiply one pseries object to another, producing a pseries object that represents the product.

Parameters

<i>other</i>	pseries object to multiply with

Returns

the product as a pseries

References [GiNaC::_ex0](#), [GiNaC::_ex1](#), [coeff\(\)](#), [degree\(\)](#), [GiNaC::ex::find\(\)](#), [is_compatible_to\(\)](#), [GiNaC::ex::is_equal\(\)](#), [GiNaC::is_order_function\(\)](#), [GiNaC::ex::is_zero\(\)](#), [ldegree\(\)](#), [point](#), [seq](#), and [var](#).

Referenced by [GiNaC::mul::series\(\)](#).

6.142.3.33 power_const()

```
ex GiNaC::pseries::power_const (
    const numeric & p,
    int deg ) const
```

Compute the p-th power of a series.

Parameters

<i>p</i>	power to compute
<i>deg</i>	truncation order of series calculation

References [GiNaC::_ex0](#), [GiNaC::_ex1](#), [c](#), [coeff\(\)](#), [degree\(\)](#), [GiNaC::is_integer\(\)](#), [GiNaC::numeric::is_negative\(\)](#), [GiNaC::is_order_function\(\)](#), [GiNaC::numeric::is_pos_integer\(\)](#), [GiNaC::numeric::is_zero\(\)](#), [is_zero\(\)](#), [ldegree\(\)](#), [point](#), [GiNaC::pow\(\)](#), [GiNaC::numeric::real\(\)](#), [seq](#), [GiNaC::to_int\(\)](#), and [var](#).

6.142.3.34 shift_exponents()

```
pseries GiNaC::pseries::shift_exponents (
    int deg ) const
```

Return a new pseries object with the powers shifted by deg.

References [coeff\(\)](#), [point](#), [seq](#), and [var](#).

6.142.3.35 print_series()

```
void GiNaC::pseries::print_series (
    const print_context & c,
    const char * openbrace,
    const char * closebrace,
    const char * mul_sym,
    const char * pow_sym,
    unsigned level ) const [protected]
```

References [GiNaC::_ex1](#), [c](#), [GiNaC::ex::coeff\(\)](#), [GiNaC::is_order_function\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::numeric](#), [point](#), [GiNaC::info_flags::positive](#), [GiNaC::pow\(\)](#), [precedence\(\)](#), [GiNaC::basic::print\(\)](#), [GiNaC::ex::print\(\)](#), [seq](#), and [var](#).

Referenced by [do_print\(\)](#), [do_print_latex\(\)](#), and [do_print_python\(\)](#).

6.142.3.36 do_print()

```
void GiNaC::pseries::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

References [c](#), and [print_series\(\)](#).

6.142.3.37 do_print_latex()

```
void GiNaC::pseries::do_print_latex (
    const print\_latex & c,
    unsigned level ) const [protected]
```

References [c](#), and [print_series\(\)](#).

6.142.3.38 do_print_tree()

```
void GiNaC::pseries::do_print_tree (
    const print\_tree & c,
    unsigned level ) const [protected]
```

References [c](#), [GiNaC::basic::flags](#), [GiNaC::basic::hashvalue](#), [point](#), [GiNaC::ex::print\(\)](#), [seq](#), and [var](#).

6.142.3.39 do_print_python()

```
void GiNaC::pseries::do_print_python (
    const print\_python & c,
    unsigned level ) const [protected]
```

References [c](#), and [print_series\(\)](#).

6.142.3.40 do_print_python_repr()

```
void GiNaC::pseries::do_print_python_repr (
    const print\_python\_repr & c,
    unsigned level ) const [protected]
```

References [c](#), [point](#), [GiNaC::ex::print\(\)](#), [seq](#), and [var](#).

6.142.4 Member Data Documentation**6.142.4.1 seq**

```
epvector GiNaC::pseries::seq [protected]
```

Vector of {coefficient, power} pairs.

Referenced by [add_series\(\)](#), [archive\(\)](#), [coeff\(\)](#), [coeffop\(\)](#), [conjugate\(\)](#), [convert_to_poly\(\)](#), [degree\(\)](#), [derivative\(\)](#), [do_print_python_repr\(\)](#), [do_print_tree\(\)](#), [eval_integ\(\)](#), [evalf\(\)](#), [evalm\(\)](#), [expand\(\)](#), [exponop\(\)](#), [imag_part\(\)](#), [is_terminating\(\)](#), [is_zero\(\)](#), [ldegree\(\)](#), [mul_const\(\)](#), [mul_series\(\)](#), [nops\(\)](#), [normal\(\)](#), [op\(\)](#), [power_const\(\)](#), [print_series\(\)](#), [pseries\(\)](#), [pseries\(\)](#), [read_archive\(\)](#), [real_part\(\)](#), [series\(\)](#), [shift_exponents\(\)](#), and [subs\(\)](#).

6.142.4.2 var

```
ex GiNaC::pseries::var [protected]
```

Series variable (holds a symbol)

Referenced by [add_series\(\)](#), [archive\(\)](#), [coeff\(\)](#), [conjugate\(\)](#), [convert_to_poly\(\)](#), [degree\(\)](#), [derivative\(\)](#), [do_print_python_repr\(\)](#), [do_print_tree\(\)](#), [eval_integ\(\)](#), [evalf\(\)](#), [evalm\(\)](#), [expand\(\)](#), [get_var\(\)](#), [imag_part\(\)](#), [is_compatible_to\(\)](#), [ldegree\(\)](#), [mul_const\(\)](#), [mul_series\(\)](#), [normal\(\)](#), [op\(\)](#), [power_const\(\)](#), [print_series\(\)](#), [pseries\(\)](#), [pseries\(\)](#), [read_archive\(\)](#), [real_part\(\)](#), [series\(\)](#), [shift_exponents\(\)](#), and [subs\(\)](#).

6.142.4.3 point

```
ex GiNaC::pseries::point [protected]
```

Expansion point.

Referenced by [add_series\(\)](#), [archive\(\)](#), [conjugate\(\)](#), [convert_to_poly\(\)](#), [derivative\(\)](#), [do_print_python_repr\(\)](#), [do_print_tree\(\)](#), [eval_integ\(\)](#), [evalf\(\)](#), [evalm\(\)](#), [expand\(\)](#), [get_point\(\)](#), [imag_part\(\)](#), [is_compatible_to\(\)](#), [mul_const\(\)](#), [mul_series\(\)](#), [normal\(\)](#), [op\(\)](#), [power_const\(\)](#), [print_series\(\)](#), [pseries\(\)](#), [pseries\(\)](#), [read_archive\(\)](#), [real_part\(\)](#), [series\(\)](#), [shift_exponents\(\)](#), and [subs\(\)](#).

The documentation for this class was generated from the following files:

- [pseries.h](#)
- [normal.cpp](#)
- [pseries.cpp](#)

6.143 GiNaC::psi1_SERIAL Class Reference

Polylogarithm and multiple polylogarithm.

```
#include <inifcns.h>
```

Static Public Attributes

- static unsigned [serial](#)

6.143.1 Detailed Description

Polylogarithm and multiple polylogarithm.

Nielsen's generalized polylogarithm. Harmonic polylogarithm. Gamma-function. Beta-function. Psi-function (aka digamma-function).

6.143.2 Member Data Documentation

6.143.2.1 serial

unsigned GiNaC::psi1_SERIAL::serial [static]

Initial value:

```
=
    function::register_new(function_options("psi", 1).
        eval_func(psi1_eval).
        evalf_func(psi1_evalf).
        derivative_func(psi1_deriv).
        series_func(psi1_series).
        latex_name("\\psi").
        overloaded(2))
```

Referenced by [GiNaC::psi\(\)](#).

The documentation for this class was generated from the following files:

- [inifcns.h](#)
- [inifcns_gamma.cpp](#)

6.144 GiNaC::psi2_SERIAL Class Reference

Derivatives of Psi-function (aka polygamma-functions).

```
#include <inifcns.h>
```

Static Public Attributes

- static unsigned [serial](#)

6.144.1 Detailed Description

Derivatives of Psi-function (aka polygamma-functions).

6.144.2 Member Data Documentation

6.144.2.1 serial

unsigned GiNaC::psi2_SERIAL::serial [static]

Initial value:

```
=
    function::register_new(function_options("psi", 2).
        eval_func(psi2_eval).
        evalf_func(psi2_evalf).
        derivative_func(psi2_deriv).
        series_func(psi2_series).
        latex_name("\\psi").
        overloaded(2))
```

Referenced by [GiNaC::psi\(\)](#).

The documentation for this class was generated from the following files:

- [inifcns.h](#)
- [inifcns_gamma.cpp](#)

6.145 GiNaC::ptr< T > Class Template Reference

Class of (intrusively) reference-counted pointers that support copy-on-write semantics.

```
#include <ptr.h>
```

Public Member Functions

- [ptr](#) (T *t) noexcept
Bind ptr to newly created object, start reference counting.
- [ptr](#) (T &t) noexcept
Bind ptr to existing reference-counted object.
- [ptr](#) (const [ptr](#) &other) noexcept
- [~ptr](#) ()
- [ptr](#) & [operator=](#) (const [ptr](#) &other)
- T & [operator*](#) () const noexcept
- T * [operator->](#) () const noexcept
- void [makewritable](#) ()
Announce your intention to modify the object bound to this ptr.
- void [swap](#) ([ptr](#) &other) noexcept
Swap the bound object of this ptr with another ptr.
- template<class U >
bool [operator==](#) (const [ptr](#)< U > &rhs) const noexcept
- template<class U >
bool [operator!=](#) (const [ptr](#)< U > &rhs) const noexcept

Private Attributes

- T * [p](#)

Friends

- struct [std::less](#)< [ptr](#)< T > >
- T * [get_pointer](#) (const [ptr](#) &x) noexcept
- template<class U >
bool [operator==](#) (const [ptr](#) &lhs, const U *rhs) noexcept
- template<class U >
bool [operator!=](#) (const [ptr](#) &lhs, const U *rhs) noexcept
- template<class U >
bool [operator==](#) (const U *lhs, const [ptr](#) &rhs) noexcept
- template<class U >
bool [operator!=](#) (const U *lhs, const [ptr](#) &rhs) noexcept
- std::ostream & [operator<<](#) (std::ostream &os, const [ptr](#)< T > &rhs)

6.145.1 Detailed Description

```
template<class T>
class GiNaC::ptr< T >
```

Class of (intrusively) reference-counted pointers that support copy-on-write semantics.

Requirements for T: must support the refcounted interface (usually by being derived from refcounted) T* T↔::duplicate() member function (only if makewritable() is used)

6.145.2 Constructor & Destructor Documentation

6.145.2.1 ptr() [1/3]

```
template<class T >
GiNaC::ptr< T >::ptr (
    T * t ) [inline], [noexcept]
```

Bind ptr to newly created object, start reference counting.

References [GINAC_ASSERT](#), and [GiNaC::ptr< T >::p](#).

6.145.2.2 ptr() [2/3]

```
template<class T >
GiNaC::ptr< T >::ptr (
    T & t ) [inline], [explicit], [noexcept]
```

Bind ptr to existing reference-counted object.

References [GiNaC::ptr< T >::p](#).

6.145.2.3 ptr() [3/3]

```
template<class T >
GiNaC::ptr< T >::ptr (
    const ptr< T > & other ) [inline], [noexcept]
```

References [GiNaC::ptr< T >::p](#).

6.145.2.4 ~ptr()

```
template<class T >
GiNaC::ptr< T >::~~ptr ( ) [inline]
```

References [GiNaC::ptr< T >::p](#).

6.145.3 Member Function Documentation

6.145.3.1 operator=()

```
template<class T >
ptr & GiNaC::ptr< T >::operator= (
    const ptr< T > & other ) [inline]
```

References [GiNaC::ptr< T >::p](#).

6.145.3.2 operator*()

```
template<class T >
T & GiNaC::ptr< T >::operator* ( ) const [inline], [noexcept]
```

References [GiNaC::ptr< T >::p](#).

6.145.3.3 operator->()

```
template<class T >
T * GiNaC::ptr< T >::operator-> ( ) const [inline], [noexcept]
```

References [GiNaC::ptr< T >::p](#).

6.145.3.4 makewritable()

```
template<class T >
void GiNaC::ptr< T >::makewritable ( ) [inline]
```

Announce your intention to modify the object bound to this ptr.

This ensures that the object is not shared by any other ptrs.

References [GiNaC::ptr< T >::p](#).

6.145.3.5 swap()

```
template<class T >
void GiNaC::ptr< T >::swap (
    ptr< T > & other ) [inline], [noexcept]
```

Swap the bound object of this ptr with another ptr.

References [GiNaC::ptr< T >::p](#).

6.145.3.6 operator==()

```
template<class T >
template<class U >
bool GiNaC::ptr< T >::operator== (
    const ptr< U > & rhs ) const [inline], [noexcept]
```

References [GiNaC::ptr< T >::get_pointer](#), [GiNaC::ptr< T >::p](#), and [GiNaC::rhs\(\)](#).

6.145.3.7 operator"!="()

```
template<class T >
template<class U >
bool GiNaC::ptr< T >::operator!= (
    const ptr< U > & rhs ) const [inline], [noexcept]
```

References [GiNaC::ptr< T >::get_pointer](#), [GiNaC::ptr< T >::p](#), and [GiNaC::rhs\(\)](#).

6.145.4 Friends And Related Symbol Documentation

6.145.4.1 std::less< ptr< T > >

```
template<class T >
friend struct std::less< ptr< T > > [friend]
```

6.145.4.2 get_pointer

```
template<class T >
T * get_pointer (
    const ptr< T > & x ) [friend]
```

Referenced by [GiNaC::ptr< T >::operator!=\(\)](#), and [GiNaC::ptr< T >::operator==\(\)](#).

6.145.4.3 operator== [1/2]

```
template<class T >
template<class U >
bool operator== (
    const ptr< T > & lhs,
    const U * rhs ) [friend]
```

6.145.4.4 operator"!= [1/2]

```
template<class T >
template<class U >
bool operator!= (
    const ptr< T > & lhs,
    const U * rhs ) [friend]
```

6.145.4.5 operator== [2/2]

```
template<class T >
template<class U >
bool operator== (
    const U * lhs,
    const ptr< T > & rhs ) [friend]
```

6.145.4.6 operator"!= [2/2]

```
template<class T >
template<class U >
bool operator!= (
    const U * lhs,
    const ptr< T > & rhs ) [friend]
```

6.145.4.7 operator<<

```
template<class T >
std::ostream & operator<< (
    std::ostream & os,
    const ptr< T > & rhs ) [friend]
```

6.145.5 Member Data Documentation

6.145.5.1 p

```
template<class T >
T* GiNaC::ptr< T >::p [private]
```

Referenced by [GiNaC::ptr< T >::makewritable\(\)](#), [GiNaC::ptr< T >::operator!=\(\)](#), [GiNaC::ptr< T >::operator*\(\)](#), [GiNaC::ptr< T >::operator->\(\)](#), [GiNaC::ptr< T >::operator=\(\)](#), [GiNaC::ptr< T >::operator==\(\)](#), [GiNaC::ptr< T >::ptr\(\)](#), [GiNaC::ptr< T >::ptr\(\)](#), [GiNaC::ptr< T >::ptr\(\)](#), [GiNaC::ptr< T >::swap\(\)](#), and [GiNaC::ptr< T >::~~ptr\(\)](#).

The documentation for this class was generated from the following file:

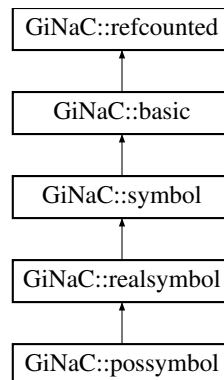
- [ptr.h](#)

6.146 GiNaC::realsymbol Class Reference

Specialization of symbol to real domain.

```
#include <symbol.h>
```

Inheritance diagram for GiNaC::realsymbol:



Public Member Functions

- [realsymbol](#) ()
- [realsymbol](#) (const std::string &initname)
- [realsymbol](#) (const std::string &initname, const std::string &texname)
- unsigned [get_domain](#) () const override
- [ex conjugate](#) () const override
- [ex real_part](#) () const override
- [ex imag_part](#) () const override
- [realsymbol](#) * [duplicate](#) () const override

Create a clone of this object on the heap.

Public Member Functions inherited from GiNaC::symbol

- [symbol](#) (const std::string &initname)
- [symbol](#) (const std::string &initname, const std::string &texname)
- bool [info](#) (unsigned inf) const override
Information about the object.
- [ex eval](#) () const override
Perform automatic non-interruptive term rewriting rules.
- [ex evalf](#) () const override
Evaluate object numerically.
- [ex series](#) (const [relational](#) &s, int [order](#), unsigned [options](#)=0) const override
Implementation of [ex::series\(\)](#) for symbols.
- [ex subs](#) (const [exmap](#) &m, unsigned [options](#)=0) const override
Substitute a set of objects by arbitrary expressions.
- [ex normal](#) ([exmap](#) &repl, [exmap](#) &rev_lookup, [lst](#) &modifier) const override
Implementation of [ex::normal\(\)](#) for symbols.
- [ex to_rational](#) ([exmap](#) &repl) const override
Implementation of [ex::to_rational\(\)](#) for symbols.
- [ex to_polynomial](#) ([exmap](#) &repl) const override
Implementation of [ex::to_polynomial\(\)](#) for symbols.
- [ex conjugate](#) () const override
- [ex real_part](#) () const override
- [ex imag_part](#) () const override
- bool [is_polynomial](#) (const [ex](#) &var) const override
Check whether this is a polynomial in the given variables.
- void [archive](#) ([archive_node](#) &n) const override
Save (a.k.a.
- void [read_archive](#) (const [archive_node](#) &n, [lst](#) &syms) override
Read (a.k.a.
- void [set_name](#) (const std::string &n)
- void [set_TeX_name](#) (const std::string &n)
- std::string [get_name](#) () const
- std::string [get_TeX_name](#) () const

Public Member Functions inherited from GiNaC::basic

- virtual [~basic](#) ()
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- [basic](#) (const [basic](#) &other)
- const [basic](#) & [operator=](#) (const [basic](#) &other)
basic assignment operator: the other object might be of a derived class.
- virtual [ex evalm](#) () const
Evaluate sums, products and integer powers of matrices.
- virtual [ex eval_integ](#) () const
Evaluate integrals, if result is known.
- virtual [ex eval_indexed](#) (const [basic](#) &i) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void [print](#) (const [print_context](#) &c, unsigned level=0) const
Output to stream.
- virtual void [dbgprint](#) () const
Little wrapper around print to be called within a debugger.

- virtual void `dbgprinttree` () const
Little wrapper around `printtree` to be called within a debugger.
- virtual unsigned `precedence` () const
Return relative operator precedence (for parenthezing output).
- virtual size_t `nops` () const
Number of operands/members.
- virtual `ex op` (size_t i) const
Return operand/member at position i.
- virtual `ex operator[]` (const `ex` &index) const
- virtual `ex operator[]` (size_t i) const
- virtual `ex & let_op` (size_t i)
Return modifiable operand/member at position i.
- virtual `ex & operator[]` (const `ex` &index)
- virtual `ex & operator[]` (size_t i)
- virtual bool `has` (const `ex` &other, unsigned `options`=0) const
Test for occurrence of a pattern.
- virtual bool `match` (const `ex` &pattern, `exmap` &repls) const
Check whether the expression matches a given pattern.
- virtual `ex map` (`map_function` &f) const
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void `accept` (`GiNaC::visitor` &v) const
- virtual int `degree` (const `ex` &s) const
Return degree of highest power in object s.
- virtual int `ldegree` (const `ex` &s) const
Return degree of lowest power in object s.
- virtual `ex coeff` (const `ex` &s, int n=1) const
Return coefficient of degree n in object s.
- virtual `ex expand` (unsigned `options`=0) const
Expand expression, i.e.
- virtual `ex collect` (const `ex` &s, bool distributed=false) const
Sort expanded expression in terms of powers of some object(s).
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &xi) const
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient` () const
Implementation `ex::max_coefficient()`.
- virtual `exvector get_free_indices` () const
Return a vector containing the free indices of an expression.
- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
Add two indexed expressions.
- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
Multiply an indexed expression with a scalar.
- virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const
Try to contract two indexed expressions that appear in the same product.
- virtual unsigned `return_type` () const
- virtual `return_type_t return_type_tinfo` () const
- template<class T >
void `print_dispatch` (const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.

- `ex subs_one_level` (const `exmap` &`m`, unsigned `options`) const
Helper function for `subs()`.
- `ex diff` (const `symbol` &`s`, unsigned `nth=1`) const
Default interface of `nth` derivative `ex::diff(s, n)`.
- `int compare` (const `basic` &`other`) const
Compare objects syntactically to establish canonical ordering.
- `bool is_equal` (const `basic` &`other`) const
Test for syntactic equality.
- `const basic & hold` () const
Stop further evaluation.
- `unsigned gethash` () const
- `const basic & setflag` (unsigned `f`) const
Set some `status_flags`.
- `const basic & clearflag` (unsigned `f`) const
Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- `unsigned int add_reference` () noexcept
- `unsigned int remove_reference` () noexcept
- `unsigned int get_refcount` () const noexcept
- `void set_refcount` (unsigned int `r`) noexcept

Additional Inherited Members

Protected Member Functions inherited from `GiNaC::symbol`

- `ex derivative` (const `symbol` &`s`) const override
Implementation of `ex::diff()` for single differentiation of a symbol.
- `bool is_equal_same_type` (const `basic` &`other`) const override
Returns true if two objects of same type are equal.
- `unsigned calchash` () const override
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- `void do_print` (const `print_context` &`c`, unsigned `level`) const
- `void do_print_latex` (const `print_latex` &`c`, unsigned `level`) const
- `void do_print_tree` (const `print_tree` &`c`, unsigned `level`) const
- `void do_print_python_repr` (const `print_python_repr` &`c`, unsigned `level`) const

Protected Member Functions inherited from `GiNaC::basic`

- `basic` ()
- `virtual ex eval_ncmul` (const `exvector` &`v`) const
- `virtual bool match_same_type` (const `basic` &`other`) const
Returns true if the attributes of two objects are similar enough for a match.
- `virtual int compare_same_type` (const `basic` &`other`) const
Returns order relation between two objects of same type.
- `void ensure_if_modifiable` () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- `void do_print` (const `print_context` &`c`, unsigned `level`) const
Default output to stream.
- `void do_print_tree` (const `print_tree` &`c`, unsigned `level`) const
Tree output to stream.
- `void do_print_python_repr` (const `print_python_repr` &`c`, unsigned `level`) const
Python parsable output to stream.

Protected Attributes inherited from [GiNaC::symbol](#)

- unsigned [serial](#)
unique serial number for comparison
- std::string [name](#)
printname of this symbol
- std::string [TeX_name](#)
LaTeX name of this symbol.

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.146.1 Detailed Description

Specialization of symbol to real domain.

6.146.2 Constructor & Destructor Documentation

6.146.2.1 [realsymbol\(\)](#) [1/3]

```
GiNaC::realsymbol::realsymbol ( )
```

Referenced by [duplicate\(\)](#).

6.146.2.2 [realsymbol\(\)](#) [2/3]

```
GiNaC::realsymbol::realsymbol (
    const std::string & initname ) [explicit]
```

6.146.2.3 [realsymbol\(\)](#) [3/3]

```
GiNaC::realsymbol::realsymbol (
    const std::string & initname,
    const std::string & texname )
```

6.146.3 Member Function Documentation

6.146.3.1 [get_domain\(\)](#)

```
unsigned GiNaC::realsymbol::get_domain ( ) const [inline], [override], [virtual]
```

Reimplemented from [GiNaC::symbol](#).

References [GiNaC::domain::real](#).

6.146.3.2 conjugate()

```
ex GiNaC::realsymbol::conjugate ( ) const [inline], [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

6.146.3.3 real_part()

```
ex GiNaC::realsymbol::real_part ( ) const [inline], [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

6.146.3.4 imag_part()

```
ex GiNaC::realsymbol::imag_part ( ) const [inline], [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

6.146.3.5 duplicate()

```
realsymbol * GiNaC::realsymbol::duplicate ( ) const [inline], [override], [virtual]
```

Create a clone of this object on the heap.

One can think of this as simulating a virtual copy constructor which is needed for instance by the recounted construction of an ex from a basic.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::status_flags::dynallocated](#), [realsymbol\(\)](#), and [GiNaC::basic::setflag\(\)](#).

The documentation for this class was generated from the following files:

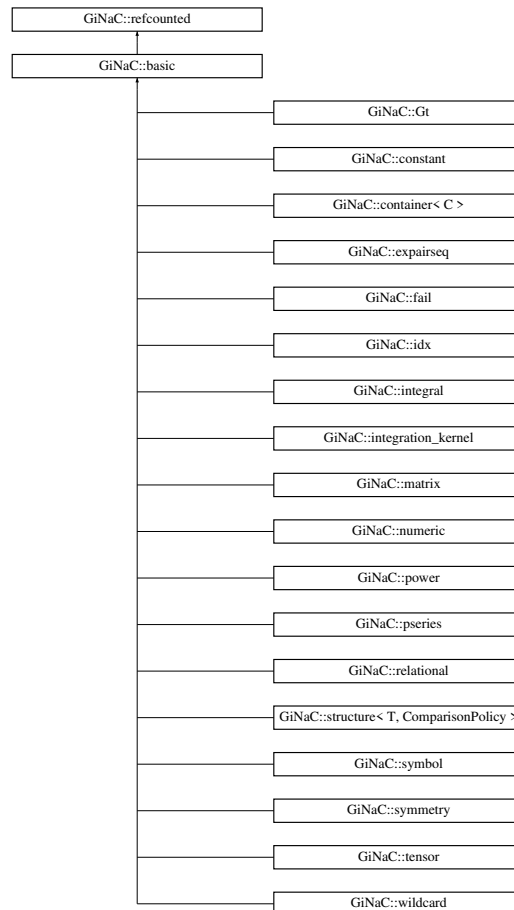
- [symbol.h](#)
- [symbol.cpp](#)

6.147 GiNaC::refcounted Class Reference

Base class for reference-counted objects.

```
#include <ptr.h>
```

Inheritance diagram for GiNaC::refcounted:



Public Member Functions

- [refcounted](#) () noexcept
- unsigned int [add_reference](#) () noexcept
- unsigned int [remove_reference](#) () noexcept
- unsigned int [get_refcount](#) () const noexcept
- void [set_refcount](#) (unsigned int r) noexcept

Private Attributes

- unsigned int [refcount](#)
reference counter

6.147.1 Detailed Description

Base class for reference-counted objects.

6.147.2 Constructor & Destructor Documentation

6.147.2.1 refcounted()

```
GiNaC::refcounted::refcounted ( ) [inline], [noexcept]
```

6.147.3 Member Function Documentation

6.147.3.1 add_reference()

```
unsigned int GiNaC::refcounted::add_reference ( ) [inline], [noexcept]
```

References [refcount](#).

6.147.3.2 remove_reference()

```
unsigned int GiNaC::refcounted::remove_reference ( ) [inline], [noexcept]
```

References [refcount](#).

6.147.3.3 get_refcount()

```
unsigned int GiNaC::refcounted::get_refcount ( ) const [inline], [noexcept]
```

References [refcount](#).

Referenced by [GiNaC::ex::construct_from_basic\(\)](#), and [GiNaC::basic::~~basic\(\)](#).

6.147.3.4 set_refcount()

```
void GiNaC::refcounted::set_refcount (
    unsigned int r ) [inline], [noexcept]
```

References [r](#), and [refcount](#).

6.147.4 Member Data Documentation

6.147.4.1 refcount

```
unsigned int GiNaC::refcounted::refcount [private]
```

reference counter

Referenced by [add_reference\(\)](#), [get_refcount\(\)](#), [remove_reference\(\)](#), and [set_refcount\(\)](#).

The documentation for this class was generated from the following file:

- [ptr.h](#)

6.148 GiNaC::registered_class_options Class Reference

This class stores information about a registered [GiNaC](#) class.

```
#include <registrar.h>
```

Public Member Functions

- [registered_class_options](#) ([const char](#) *n, [const char](#) *p, [const](#) [std::type_info](#) &ti)
- [const char](#) * [get_name](#) () [const](#)
- [const char](#) * [get_parent_name](#) () [const](#)
- [std::type_info](#) [const](#) * [get_id](#) () [const](#)
- [const](#) [std::vector](#)< [print_func](#) > & [get_print_dispatch_table](#) () [const](#)
- [template](#)<[class](#) Ctx , [class](#) T , [class](#) C >
[registered_class_options](#) & [print_func](#) ([void](#) f([const](#) T &, [const](#) C &c, [unsigned](#)))
- [template](#)<[class](#) Ctx , [class](#) T , [class](#) C >
[registered_class_options](#) & [print_func](#) ([void](#)(T::*f)([const](#) C &, [unsigned](#)))
- [template](#)<[class](#) Ctx >
[registered_class_options](#) & [print_func](#) ([const](#) [print_func](#) &f)
- [void](#) [set_print_func](#) ([unsigned](#) id, [const](#) [print_func](#) &f)

Private Attributes

- [const char](#) * [name](#)
Class name.
- [const char](#) * [parent_name](#)
Name of superclass.
- [std::type_info](#) [const](#) * [tinfo_key](#)
Type information key.
- [std::vector](#)< [print_func](#) > [print_dispatch_table](#)
Method table for print() dispatch.

6.148.1 Detailed Description

This class stores information about a registered [GiNaC](#) class.

6.148.2 Constructor & Destructor Documentation

6.148.2.1 registered_class_options()

```
GiNaC::registered_class_options::registered_class_options (
    const char * n,
    const char * p,
    const std::type\_info & ti ) [inline]
```

6.148.3 Member Function Documentation

6.148.3.1 get_name()

```
const char * GiNaC::registered_class_options::get_name ( ) const [inline]
```

References [name](#).

6.148.3.2 get_parent_name()

```
const char * GiNaC::registered_class_options::get_parent_name ( ) const [inline]
```

References [parent_name](#).

6.148.3.3 get_id()

```
std::type_info const * GiNaC::registered_class_options::get_id ( ) const [inline]
```

References [tinfo_key](#).

6.148.3.4 get_print_dispatch_table()

```
const std::vector< print_func_t > & GiNaC::registered_class_options::get_print_dispatch_table  
( ) const [inline]
```

References [print_dispatch_table](#).

6.148.3.5 print_func() [1/3]

```
template<class Ctx , class T , class C >  
registered_class_options & GiNaC::registered_class_options::print_func (   
    void f(const T &, const C &c, unsigned) ) [inline]
```

References [options](#), and [set_print_func\(\)](#).

6.148.3.6 print_func() [2/3]

```
template<class Ctx , class T , class C >  
registered_class_options & GiNaC::registered_class_options::print_func (   
    void(T::*)(const C &, unsigned) f ) [inline]
```

References [options](#), and [set_print_func\(\)](#).

6.148.3.7 `print_func()` [3/3]

```
template<class Ctx >
registered\_class\_options & GiNaC::registered_class_options::print_func (
    const print\_functor & f ) [inline]
```

References [options](#), and [set_print_func\(\)](#).

6.148.3.8 `set_print_func()`

```
void GiNaC::registered_class_options::set_print_func (
    unsigned id,
    const print\_functor & f ) [inline]
```

References [print_dispatch_table](#).

Referenced by [print_func\(\)](#), [print_func\(\)](#), and [print_func\(\)](#).

6.148.4 Member Data Documentation

6.148.4.1 `name`

```
const char* GiNaC::registered_class_options::name [private]
```

Class name.

Referenced by [get_name\(\)](#).

6.148.4.2 `parent_name`

```
const char* GiNaC::registered_class_options::parent_name [private]
```

Name of superclass.

Referenced by [get_parent_name\(\)](#).

6.148.4.3 `tinfo_key`

```
std::type_info const* GiNaC::registered_class_options::tinfo_key [private]
```

Type information key.

Referenced by [get_id\(\)](#).

6.148.4.4 print_dispatch_table

```
std::vector<print_functor> GiNaC::registered_class_options::print_dispatch_table [private]
```

Method table for print() dispatch.

Referenced by [get_print_dispatch_table\(\)](#), and [set_print_func\(\)](#).

The documentation for this class was generated from the following file:

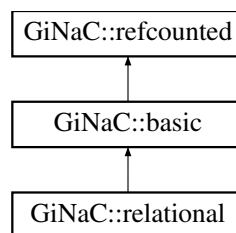
- [registrar.h](#)

6.149 GiNaC::relational Class Reference

This class holds a relation consisting of two expressions and a logical relation between them.

```
#include <relational.h>
```

Inheritance diagram for GiNaC::relational:



Classes

- struct [safe_bool_helper](#)

Public Types

- enum [operators](#) {
 [equal](#) , [not_equal](#) , [less](#) , [less_or_equal](#) ,
 [greater](#) , [greater_or_equal](#) }

Public Member Functions

- **relational** (const **ex** &lhs, const **ex** &rhs, **operators** oper=equal)
 unsigned **precedence** () const override
Return relative operator precedence (for parenthezing output).
- bool **info** (unsigned inf) const override
Information about the object.
- size_t **nops** () const override
Number of operands/members.
- **ex op** (size_t i) const override
Return operand/member at position i.
- **ex map** (**map_function** &f) const override
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- **ex subs** (const **exmap** &m, unsigned **options**=0) const override
Substitute a set of objects by arbitrary expressions.
- void **archive** (**archive_node** &n) const override
Save (a.k.a.
- void **read_archive** (const **archive_node** &n, **lst** &**syms**) override
Read (a.k.a.
- **ex canonical** () const
Returns an equivalent relational with zero right-hand side.
- **ex lhs** () const
- **ex rhs** () const
- **operator safe_bool** () const
Cast the relational into a Boolean, mainly for evaluation within an if-statement.
- **safe_bool operator!** () const

Public Member Functions inherited from **GiNaC::basic**

- virtual **~basic** ()
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- **basic** (const **basic** &other)
- const **basic** & **operator=** (const **basic** &other)
basic assignment operator: the other object might be of a derived class.
- virtual **basic** * **duplicate** () const
Create a clone of this object on the heap.
- virtual **ex eval** () const
Perform automatic non-interruptive term rewriting rules.
- virtual **ex evalf** () const
Evaluate object numerically.
- virtual **ex evalm** () const
Evaluate sums, products and integer powers of matrices.
- virtual **ex eval_integ** () const
Evaluate integrals, if result is known.
- virtual **ex eval_indexed** (const **basic** &i) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void **print** (const **print_context** &c, unsigned level=0) const
Output to stream.
- virtual void **dbgprint** () const
Little wrapper around print to be called within a debugger.
- virtual void **dbgprinttree** () const

Little wrapper around printtree to be called within a debugger.

- virtual `ex operator[]` (const `ex` &index) const
- virtual `ex operator[]` (size_t i) const
- virtual `ex & let_op` (size_t i)

Return modifiable operand/member at position i.

- virtual `ex & operator[]` (const `ex` &index)
- virtual `ex & operator[]` (size_t i)
- virtual bool `has` (const `ex` &other, unsigned `options`=0) const

Test for occurrence of a pattern.

- virtual bool `match` (const `ex` &pattern, `exmap` &repls) const

Check whether the expression matches a given pattern.

- virtual void `accept` (GiNaC::visitor &v) const
- virtual bool `is_polynomial` (const `ex` &var) const

Check whether this is a polynomial in the given variables.

- virtual int `degree` (const `ex` &s) const

Return degree of highest power in object s.

- virtual int `ldegree` (const `ex` &s) const

Return degree of lowest power in object s.

- virtual `ex coeff` (const `ex` &s, int `n`=1) const

Return coefficient of degree n in object s.

- virtual `ex expand` (unsigned `options`=0) const

Expand expression, i.e.

- virtual `ex collect` (const `ex` &s, bool `distributed`=false) const

Sort expanded expression in terms of powers of some object(s).

- virtual `ex series` (const `relational` &r, int `order`, unsigned `options`=0) const

Default implementation of `ex::series()`.

- virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const

Default implementation of `ex::normal()`.

- virtual `ex to_rational` (`exmap` &repl) const

Default implementation of `ex::to_rational()`.

- virtual `ex to_polynomial` (`exmap` &repl) const

- virtual `numeric integer_content` () const

- virtual `ex smod` (const `numeric` &xi) const

Apply symmetric modular homomorphism to an expanded multivariate polynomial.

- virtual `numeric max_coefficient` () const

Implementation `ex::max_coefficient()`.

- virtual `exvector get_free_indices` () const

Return a vector containing the free indices of an expression.

- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const

Add two indexed expressions.

- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const

Multiply an indexed expression with a scalar.

- virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const

Try to contract two indexed expressions that appear in the same product.

- virtual `ex conjugate` () const

- virtual `ex real_part` () const

- virtual `ex imag_part` () const

- template<class T >

void `print_dispatch` (const `print_context` &c, unsigned level) const

Like `print()`, but dispatch to the specified class.

- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const

- Like `print()`, but dispatch to the specified class.
- `ex subs_one_level` (const `exmap` &`m`, unsigned `options`) const
Helper function for `subs()`.
- `ex diff` (const `symbol` &`s`, unsigned `nth=1`) const
Default interface of `nth` derivative `ex::diff(s, n)`.
- `int compare` (const `basic` &`other`) const
Compare objects syntactically to establish canonical ordering.
- `bool is_equal` (const `basic` &`other`) const
Test for syntactic equality.
- `const basic & hold` () const
Stop further evaluation.
- `unsigned gethash` () const
- `const basic & setflag` (unsigned `f`) const
Set some `status_flags`.
- `const basic & clearflag` (unsigned `f`) const
Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- `unsigned int add_reference` () noexcept
- `unsigned int remove_reference` () noexcept
- `unsigned int get_refcount` () const noexcept
- `void set_refcount` (unsigned int `r`) noexcept

Protected Member Functions

- `ex eval_ncmul` (const `exvector` &`v`) const override
- `bool match_same_type` (const `basic` &`other`) const override
Returns true if the attributes of two objects are similar enough for a match.
- `unsigned return_type` () const override
- `return_type_t return_type_tinfo` () const override
- `unsigned calchash` () const override
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- `void do_print` (const `print_context` &`c`, unsigned `level`) const
- `void do_print_python_repr` (const `print_python_repr` &`c`, unsigned `level`) const

Protected Member Functions inherited from `GiNaC::basic`

- `basic` ()
- `virtual ex derivative` (const `symbol` &`s`) const
Default implementation of `ex::diff()`.
- `virtual int compare_same_type` (const `basic` &`other`) const
Returns order relation between two objects of same type.
- `virtual bool is_equal_same_type` (const `basic` &`other`) const
Returns true if two objects of same type are equal.
- `void ensure_if_modifiable` () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- `void do_print` (const `print_context` &`c`, unsigned `level`) const
Default output to stream.
- `void do_print_tree` (const `print_tree` &`c`, unsigned `level`) const
Tree output to stream.
- `void do_print_python_repr` (const `print_python_repr` &`c`, unsigned `level`) const
Python parsable output to stream.

Protected Attributes

- [ex lh](#)
- [ex rh](#)
- [operators o](#)

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

Private Types

- typedef void(safe_bool_helper::* [safe_bool](#)) ()

Private Member Functions

- [safe_bool make_safe_bool](#) (bool) const

6.149.1 Detailed Description

This class holds a relation consisting of two expressions and a logical relation between them.

6.149.2 Member Typedef Documentation**6.149.2.1 [safe_bool](#)**

```
typedef void(safe_bool_helper::* GiNaC::relational::safe_bool) () [private]
```

6.149.3 Member Enumeration Documentation**6.149.3.1 [operators](#)**

```
enum GiNaC::relational::operators
```

Enumerator

equal	
not_equal	
less	
less_or_equal	
greater	
greater_or_equal	

6.149.4 Constructor & Destructor Documentation

6.149.4.1 relational()

```
GiNaC::relational::relational (
    const ex & lhs,
    const ex & rhs,
    operators oper = equal )
```

6.149.5 Member Function Documentation

6.149.5.1 precedence()

```
unsigned GiNaC::relational::precedence ( ) const [inline], [override], [virtual]
```

Return relative operator precedence (for parenthezing output).

Reimplemented from [GiNaC::basic](#).

Referenced by [do_print\(\)](#).

6.149.5.2 info()

```
bool GiNaC::relational::info (
    unsigned inf ) const [override], [virtual]
```

Information about the object.

See also

class [info_flags](#)

Reimplemented from [GiNaC::basic](#).

References [equal](#), [greater](#), [greater_or_equal](#), [less](#), [less_or_equal](#), [not_equal](#), [o](#), [GiNaC::info_flags::relation](#), [GiNaC::info_flags::relation_equal](#), [GiNaC::info_flags::relation_greater](#), [GiNaC::info_flags::relation_greater_or_equal](#), [GiNaC::info_flags::relation_less](#), [GiNaC::info_flags::relation_less_or_equal](#), and [GiNaC::info_flags::relation_not_equal](#).

Referenced by [operator safe_bool\(\)](#).

6.149.5.3 nops()

```
size_t GiNaC::relational::nops ( ) const [override], [virtual]
```

Number of operands/members.

Reimplemented from [GiNaC::basic](#).

6.149.5.4 op()

```
ex GiNaC::relational::op (
    size_t i ) const [override], [virtual]
```

Return operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [GINAC_ASSERT](#), [lh](#), and [rh](#).

6.149.5.5 map()

```
ex GiNaC::relational::map (
    map_function & f ) const [override], [virtual]
```

Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).

Reimplemented from [GiNaC::basic](#).

References [GiNaC::are_ex_trivially_equal\(\)](#), [lh](#), [o](#), and [rh](#).

6.149.5.6 subs()

```
ex GiNaC::relational::subs (
    const exmap & m,
    unsigned options = 0 ) const [override], [virtual]
```

Substitute a set of objects by arbitrary expressions.

The ex returned will already be evaluated.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::are_ex_trivially_equal\(\)](#), [lh](#), [m](#), [o](#), [options](#), [rh](#), [GiNaC::ex::subs\(\)](#), and [GiNaC::basic::subs_one_level\(\)](#).

6.149.5.7 archive()

```
void GiNaC::relational::archive (
    archive_node & n ) const [override], [virtual]
```

Save (a.k.a.

serialize) object into archive.

Reimplemented from [GiNaC::basic](#).

References [lh](#), [n](#), [o](#), and [rh](#).

6.149.5.8 read_archive()

```
void GiNaC::relational::read_archive (
    const archive_node & n,
    lst & syms ) [override], [virtual]
```

Read (a.k.a.

deserialize) object from archive.

Reimplemented from [GiNaC::basic](#).

References [lh](#), [n](#), [o](#), and [rh](#).

6.149.5.9 canonical()

```
ex GiNaC::relational::canonical ( ) const
```

Returns an equivalent relational with zero right-hand side.

References [GiNaC::_ex0](#), [lh](#), [o](#), and [rh](#).

6.149.5.10 eval_ncmul()

```
ex GiNaC::relational::eval_ncmul (
    const exvector & v ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::ex::eval_ncmul\(\)](#), and [lh](#).

6.149.5.11 match_same_type()

```
bool GiNaC::relational::match_same_type (
    const basic & other ) const [override], [protected], [virtual]
```

Returns true if the attributes of two objects are similar enough for a match.

This function must not match subexpressions (this is already done by [basic::match\(\)](#)). Only attributes not accessible by [op\(\)](#) should be compared. This is also the reason why this function doesn't take the wildcard replacement list from [match\(\)](#) as an argument: only subexpressions are subject to wildcard matches. Also, this function only needs to be implemented for container classes because [is_equal_same_type\(\)](#) is automatically used instead of [match_same_type\(\)](#) if [nops\(\)](#) == 0.

See also

[basic::match](#)

Reimplemented from [GiNaC::basic](#).

References [GINAC_ASSERT](#), and [o](#).

6.149.5.12 return_type()

```
unsigned GiNaC::relational::return_type ( ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GINAC_ASSERT](#), [lh](#), [GiNaC::ex::return_type\(\)](#), and [rh](#).

6.149.5.13 return_type_tinfo()

```
return_type_t GiNaC::relational::return_type_tinfo ( ) const [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GINAC_ASSERT](#), [lh](#), [GiNaC::ex::return_type_tinfo\(\)](#), and [rh](#).

6.149.5.14 calchash()

```
unsigned GiNaC::relational::calchash ( ) const [override], [protected], [virtual]
```

Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.

The method inherited from class basic computes a hash value based on the type and hash values of possible members. For this reason it is well suited for container classes but atomic classes should override this implementation because otherwise they would all end up with the same hashvalue.

Reimplemented from [GiNaC::basic](#).

References [equal](#), [GiNaC::status_flags::evaluated](#), [GiNaC::basic::flags](#), [GiNaC::ex::gethash\(\)](#), [greater](#), [greater_or_equal](#), [GiNaC::status_flags::hash_calculated](#), [GiNaC::basic::hashvalue](#), [less](#), [less_or_equal](#), [lh](#), [GiNaC::make_hash_seed\(\)](#), [not_equal](#), [o](#), [rh](#), [GiNaC::rotate_left\(\)](#), and [GiNaC::basic::setflag\(\)](#).

6.149.5.15 do_print()

```
void GiNaC::relational::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

References [c](#), [lh](#), [o](#), [precedence\(\)](#), [GiNaC::ex::print\(\)](#), [GiNaC::print_operator\(\)](#), and [rh](#).

6.149.5.16 do_print_python_repr()

```
void GiNaC::relational::do_print_python_repr (
    const print_python_repr & c,
    unsigned level ) const [protected]
```

References [c](#), [lh](#), [o](#), [GiNaC::ex::print\(\)](#), [GiNaC::print_operator\(\)](#), and [rh](#).

6.149.5.17 lhs()

```
ex GiNaC::relational::lhs ( ) const [inline]
```

References [lh](#).

Referenced by [GiNaC::atan_series\(\)](#), [GiNaC::atanh_series\(\)](#), [GiNaC::beta_series\(\)](#), [GiNaC::Li2_series\(\)](#), [GiNaC::log_series\(\)](#), [GiNaC::tan_series\(\)](#), and [GiNaC::tanh_series\(\)](#).

6.149.5.18 rhs()

```
ex GiNaC::relational::rhs ( ) const [inline]
```

References [rh](#).

Referenced by [GiNaC::atan_series\(\)](#), [GiNaC::atanh_series\(\)](#), [GiNaC::Li2_series\(\)](#), and [GiNaC::log_series\(\)](#).

6.149.5.19 make_safe_bool()

```
relational::safe_bool GiNaC::relational::make_safe_bool (
    bool cond ) const [private]
```

References [GiNaC::relational::safe_bool_helper::nonnull\(\)](#).

Referenced by [operator safe_bool\(\)](#), and [operator!\(\)](#).

6.149.5.20 operator safe_bool()

```
GiNaC::relational::operator relational::safe_bool ( ) const
```

Cast the relational into a Boolean, mainly for evaluation within an if-statement.

Note that $(a < b) == \text{false}$ does not imply $(a \geq b) == \text{true}$ in the general symbolic case. A false result means the comparison is either false or undecidable (except of course for $!=$, where true means either unequal or undecidable).

References [GiNaC::num0_p](#), [equal](#), [greater](#), [greater_or_equal](#), [GiNaC::ex::info\(\)](#), [info\(\)](#), [GiNaC::ex::is_zero\(\)](#), [GiNaC::is_zero\(\)](#), [less](#), [less_or_equal](#), [lh](#), [make_safe_bool\(\)](#), [GiNaC::info_flags::negative](#), [GiNaC::info_flags::nonnegative](#), [not_equal](#), [o](#), [GiNaC::info_flags::positive](#), and [rh](#).

6.149.5.21 operator"!()

```
relational::safe_bool GiNaC::relational::operator! ( ) const [inline]
```

References [make_safe_bool\(\)](#).

6.149.6 Member Data Documentation

6.149.6.1 lh

`ex GiNaC::relational::lh` [protected]

Referenced by [archive\(\)](#), [calchash\(\)](#), [canonical\(\)](#), [do_print\(\)](#), [do_print_python_repr\(\)](#), [eval_ncmul\(\)](#), [lhs\(\)](#), [map\(\)](#), [op\(\)](#), [operator safe_bool\(\)](#), [read_archive\(\)](#), [return_type\(\)](#), [return_type_tinfo\(\)](#), and [subs\(\)](#).

6.149.6.2 rh

`ex GiNaC::relational::rh` [protected]

Referenced by [archive\(\)](#), [calchash\(\)](#), [canonical\(\)](#), [do_print\(\)](#), [do_print_python_repr\(\)](#), [map\(\)](#), [op\(\)](#), [operator safe_bool\(\)](#), [read_archive\(\)](#), [return_type\(\)](#), [return_type_tinfo\(\)](#), [rhs\(\)](#), and [subs\(\)](#).

6.149.6.3 o

`operators GiNaC::relational::o` [protected]

Referenced by [archive\(\)](#), [calchash\(\)](#), [canonical\(\)](#), [do_print\(\)](#), [do_print_python_repr\(\)](#), [info\(\)](#), [map\(\)](#), [match_same_type\(\)](#), [operator safe_bool\(\)](#), [read_archive\(\)](#), and [subs\(\)](#).

The documentation for this class was generated from the following files:

- [relational.h](#)
- [relational.cpp](#)

6.150 GiNaC::remember_strategies Class Reference

Strategies how to clean up the function remember cache.

```
#include <flags.h>
```

Public Types

- enum { [delete_never](#) , [delete_lru](#) , [delete_lfu](#) , [delete_cyclic](#) }

6.150.1 Detailed Description

Strategies how to clean up the function remember cache.

See also

[remember_table](#)

6.150.2 Member Enumeration Documentation

6.150.2.1 anonymous enum

anonymous enum

Enumerator

delete_never	Let table grow indefinitely.
delete_lru	Least recently used.
delete_lfu	Least frequently used.
delete_cyclic	First (oldest) one in list.

The documentation for this class was generated from the following file:

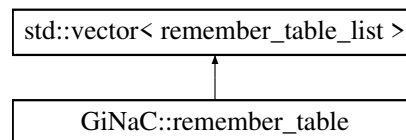
- [flags.h](#)

6.151 GiNaC::remember_table Class Reference

The remember table is organized like an n-fold associative cache in a microprocessor.

```
#include <remember.h>
```

Inheritance diagram for GiNaC::remember_table:



Public Member Functions

- [remember_table](#) ()
- [remember_table](#) (unsigned s, unsigned as, unsigned strat)
- bool [lookup_entry](#) (function const &f, [ex](#) &result) const
- void [add_entry](#) (function const &f, [ex](#) const &result)
- void [clear_all_entries](#) ()
- void [show_statistics](#) (std::ostream &os, unsigned level) const

Static Public Member Functions

- static std::vector< [remember_table](#) > & [remember_tables](#) ()

Protected Member Functions

- void [init_table](#) ()

Protected Attributes

- unsigned [table_size](#)
- unsigned [max_assoc_size](#)
- unsigned [remember_strategy](#)

6.151.1 Detailed Description

The remember table is organized like an n-fold associative cache in a microprocessor.

The table has a width of 's' (which is rounded to table_size, some power of 2 near 's', internally) and a depth of 'as' (unless you choose that entries are never discarded). The place where an entry is stored depends on the hashvalue of the parameters of the function (this corresponds to the address of byte to be cached). The 'log_2(table_size)' least significant bits of this hashvalue give the slot in which the entry will be stored or looked up. Each slot can take up to 'as' entries. If a slot is full, an older entry is removed by one of the following strategies:

- oldest entry (the first one in the list)
- least recently used (the one with the lowest 'last_access')
- least frequently used (the one with the lowest 'successful_hits') or all entries are kept which means that the table grows indefinitely.

6.151.2 Constructor & Destructor Documentation

6.151.2.1 remember_table() [1/2]

```
GiNaC::remember_table::remember_table ( )
```

References [GiNaC::remember_strategies::delete_never](#), [max_assoc_size](#), [remember_strategy](#), and [table_size](#).

6.151.2.2 remember_table() [2/2]

```
GiNaC::remember_table::remember_table (
    unsigned s,
    unsigned as,
    unsigned strat )
```

References [init_table\(\)](#), [GiNaC::log2\(\)](#), and [table_size](#).

6.151.3 Member Function Documentation

6.151.3.1 lookup_entry()

```
bool GiNaC::remember_table::lookup_entry (
    function const & f,
    ex & result ) const
```

References [GiNaC::basic::gethash\(\)](#), [GINAC_ASSERT](#), and [table_size](#).

6.151.3.2 add_entry()

```
void GiNaC::remember_table::add_entry (
    function const & f,
    ex const & result )
```

References [GiNaC::basic::gethash\(\)](#), [GINAC_ASSERT](#), and [table_size](#).

6.151.3.3 `clear_all_entries()`

```
void GiNaC::remember_table::clear_all_entries ( )
```

References [init_table\(\)](#).

6.151.3.4 `show_statistics()`

```
void GiNaC::remember_table::show_statistics (
    std::ostream & os,
    unsigned level ) const
```

6.151.3.5 `remember_tables()`

```
std::vector< remember_table > & GiNaC::remember_table::remember_tables ( ) [static]
```

Referenced by [GiNaC::function::lookup_remember_table\(\)](#), [GiNaC::function::register_new\(\)](#), and [GiNaC::function::store_remember_t](#)

6.151.3.6 `init_table()`

```
void GiNaC::remember_table::init_table ( ) [protected]
```

References [max_assoc_size](#), [remember_strategy](#), and [table_size](#).

Referenced by [clear_all_entries\(\)](#), and [remember_table\(\)](#).

6.151.4 Member Data Documentation

6.151.4.1 `table_size`

```
unsigned GiNaC::remember_table::table_size [protected]
```

Referenced by [add_entry\(\)](#), [init_table\(\)](#), [lookup_entry\(\)](#), [remember_table\(\)](#), and [remember_table\(\)](#).

6.151.4.2 `max_assoc_size`

```
unsigned GiNaC::remember_table::max_assoc_size [protected]
```

Referenced by [init_table\(\)](#), and [remember_table\(\)](#).

6.151.4.3 `remember_strategy`

```
unsigned GiNaC::remember_table::remember_strategy [protected]
```

Referenced by [init_table\(\)](#), and [remember_table\(\)](#).

The documentation for this class was generated from the following files:

- [remember.h](#)
- [remember.cpp](#)

6.152 GiNaC::remember_table_entry Class Reference

A single entry in the remember table of a function.

```
#include <remember.h>
```

Public Member Functions

- [remember_table_entry](#) ([function](#) const &f, [ex](#) const &r)
- bool [is_equal](#) ([function](#) const &f) const
- [ex](#) [get_result](#) () const
- unsigned long [get_last_access](#) () const
- unsigned long [get_successful_hits](#) () const

Protected Attributes

- unsigned [hashvalue](#)
- [exvector](#) [seq](#)
- [ex](#) [result](#)
- unsigned long [last_access](#)
- unsigned [successful_hits](#)

Static Protected Attributes

- static unsigned long [access_counter](#) = 0

6.152.1 Detailed Description

A single entry in the remember table of a function.

Needs to be a friend of class [function](#) to access 'seq'. 'last_access' and 'successful_hits' are updated at each successful 'is_equal'.

6.152.2 Constructor & Destructor Documentation

6.152.2.1 remember_table_entry()

```
GiNaC::remember_table_entry::remember_table_entry (  
    function const & f,  
    ex const & r )
```

References [access_counter](#), [last_access](#), and [successful_hits](#).

6.152.3 Member Function Documentation

6.152.3.1 `is_equal()`

```
bool GiNaC::remember_table_entry::is_equal (
    function const & f ) const
```

References [access_counter](#), [GiNaC::basic::gethash\(\)](#), [GINAC_ASSERT](#), [hashvalue](#), [is_equal\(\)](#), [last_access](#), [GiNaC::container_storage< C >::seq](#), [seq](#), and [successful_hits](#).

Referenced by [is_equal\(\)](#).

6.152.3.2 `get_result()`

```
ex GiNaC::remember_table_entry::get_result ( ) const [inline]
```

References [result](#).

6.152.3.3 `get_last_access()`

```
unsigned long GiNaC::remember_table_entry::get_last_access ( ) const [inline]
```

References [last_access](#).

6.152.3.4 `get_successful_hits()`

```
unsigned long GiNaC::remember_table_entry::get_successful_hits ( ) const [inline]
```

References [successful_hits](#).

6.152.4 Member Data Documentation

6.152.4.1 `hashvalue`

```
unsigned GiNaC::remember_table_entry::hashvalue [protected]
```

Referenced by [is_equal\(\)](#).

6.152.4.2 `seq`

```
exvector GiNaC::remember_table_entry::seq [protected]
```

Referenced by [is_equal\(\)](#).

6.152.4.3 result

`ex` GiNaC::remember_table_entry::result [protected]

Referenced by [get_result\(\)](#).

6.152.4.4 last_access

unsigned long GiNaC::remember_table_entry::last_access [mutable], [protected]

Referenced by [get_last_access\(\)](#), [is_equal\(\)](#), and [remember_table_entry\(\)](#).

6.152.4.5 successful_hits

unsigned GiNaC::remember_table_entry::successful_hits [mutable], [protected]

Referenced by [get_successful_hits\(\)](#), [is_equal\(\)](#), and [remember_table_entry\(\)](#).

6.152.4.6 access_counter

unsigned long GiNaC::remember_table_entry::access_counter = 0 [static], [protected]

Referenced by [is_equal\(\)](#), and [remember_table_entry\(\)](#).

The documentation for this class was generated from the following files:

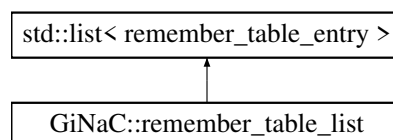
- [remember.h](#)
- [remember.cpp](#)

6.153 GiNaC::remember_table_list Class Reference

A list of entries in the remember table having some least significant bits of the hashvalue in common.

```
#include <remember.h>
```

Inheritance diagram for GiNaC::remember_table_list:



Public Member Functions

- [remember_table_list](#) (unsigned as, unsigned strat)
- void [add_entry](#) (function const &f, `ex` const &result)
- bool [lookup_entry](#) (function const &f, `ex` &result) const

Protected Attributes

- unsigned [max_assoc_size](#)
- unsigned [remember_strategy](#)

6.153.1 Detailed Description

A list of entries in the remember table having some least significant bits of the hashvalue in common.

6.153.2 Constructor & Destructor Documentation

6.153.2.1 [remember_table_list\(\)](#)

```
GiNaC::remember_table_list::remember_table_list (
    unsigned as,
    unsigned strat )
```

References [max_assoc_size](#), and [remember_strategy](#).

6.153.3 Member Function Documentation

6.153.3.1 [add_entry\(\)](#)

```
void GiNaC::remember_table_list::add_entry (
    function const & f,
    ex const & result )
```

References [GiNaC::remember_strategies::delete_cyclic](#), [GiNaC::remember_strategies::delete_lfu](#), [GiNaC::remember_strategies::delete_never](#), [GINAC_ASSERT](#), [max_assoc_size](#), and [remember_strategy](#).

6.153.3.2 [lookup_entry\(\)](#)

```
bool GiNaC::remember_table_list::lookup_entry (
    function const & f,
    ex & result ) const
```

6.153.4 Member Data Documentation

6.153.4.1 [max_assoc_size](#)

```
unsigned GiNaC::remember_table_list::max_assoc_size [protected]
```

Referenced by [add_entry\(\)](#), and [remember_table_list\(\)](#).

6.153.4.2 remember_strategy

```
unsigned GiNaC::remember_table_list::remember_strategy [protected]
```

Referenced by [add_entry\(\)](#), and [remember_table_list\(\)](#).

The documentation for this class was generated from the following files:

- [remember.h](#)
- [remember.cpp](#)

6.154 GiNaC::return_type_t Struct Reference

To distinguish between different kinds of non-commutative objects.

```
#include <registrar.h>
```

Public Member Functions

- bool [operator<](#) (const [return_type_t](#) &other) const
Strict weak ordering (so one can put [return_type_t](#)'s into a STL container).
- bool [operator==](#) (const [return_type_t](#) &other) const
- bool [operator!=](#) (const [return_type_t](#) &other) const

Public Attributes

- std::type_info const * [tinfo](#)
to distinguish between non-commutative objects of different type.
- unsigned [rl](#)
to distinguish between non-commutative objects of the same type.

6.154.1 Detailed Description

To distinguish between different kinds of non-commutative objects.

6.154.2 Member Function Documentation

6.154.2.1 operator<()

```
bool GiNaC::return_type_t::operator< (
    const return\_type\_t & other ) const [inline]
```

Strict weak ordering (so one can put [return_type_t](#)'s into a STL container).

References [rl](#), and [tinfo](#).

6.154.2.2 `operator==()`

```
bool GiNaC::return_type_t::operator== (
    const return\_type\_t & other ) const [inline]
```

References [rl](#), and [tinfo](#).

Referenced by [operator!=\(\)](#).

6.154.2.3 `operator"!=()`

```
bool GiNaC::return_type_t::operator!= (
    const return\_type\_t & other ) const [inline]
```

References [operator==\(\)](#).

6.154.3 Member Data Documentation

6.154.3.1 `tinfo`

```
std::type_info const* GiNaC::return_type_t::tinfo
```

to distinguish between non-commutative objects of different type.

Referenced by [GiNaC::is_clifford_tinfo\(\)](#), [GiNaC::is_color_tinfo\(\)](#), [GiNaC::make_return_type_t\(\)](#), [operator<\(\)](#), [operator==\(\)](#), and [GiNaC::basic::return_type_tinfo\(\)](#).

6.154.3.2 `rl`

```
unsigned GiNaC::return_type_t::rl
```

to distinguish between non-commutative objects of the same type.

Think of gamma matrices with different representation labels.

Referenced by [GiNaC::get_representation_label\(\)](#), [GiNaC::make_return_type_t\(\)](#), [operator<\(\)](#), [operator==\(\)](#), [GiNaC::basic::return_type_tinfo\(\)](#), and [GiNaC::structure< T, ComparisonPolicy >::return_type_tinfo\(\)](#).

The documentation for this struct was generated from the following file:

- [registrar.h](#)

6.155 `GiNaC::return_types` Class Reference

```
#include <flags.h>
```

Public Types

- enum { [commutative](#) , [noncommutative](#) , [noncommutative_composite](#) }

6.155.1 Member Enumeration Documentation

6.155.1.1 anonymous enum

```
anonymous enum
```


Enumerator

commutative	
noncommutative	
noncommutative_composite	

The documentation for this class was generated from the following file:

- [flags.h](#)

6.156 GiNaC::relational::safe_bool_helper Struct Reference

Public Member Functions

- void [nonnull](#) ()

6.156.1 Member Function Documentation

6.156.1.1 nonnull()

```
void GiNaC::relational::safe_bool_helper::nonnull ( ) [inline]
```

Referenced by [GiNaC::relational::make_safe_bool\(\)](#).

The documentation for this struct was generated from the following file:

- [relational.h](#)

6.157 GiNaC::scalar_products Class Reference

Helper class for storing information about known scalar products which are to be automatically replaced by [simplify_indexed\(\)](#).

```
#include <indexed.h>
```

Public Member Functions

- void [add](#) (const [ex](#) &v1, const [ex](#) &v2, const [ex](#) &sp)
Register scalar product pair and its value.
- void [add](#) (const [ex](#) &v1, const [ex](#) &v2, const [ex](#) &dim, const [ex](#) &sp)
Register scalar product pair and its value for a specific space dimension.
- void [add_vectors](#) (const [lst](#) &l, const [ex](#) &dim=[wild](#)())
Register list of vectors.
- void [clear](#) ()
Clear all registered scalar products.
- bool [is_defined](#) (const [ex](#) &v1, const [ex](#) &v2, const [ex](#) &dim) const
Check whether scalar product pair is defined.
- [ex](#) [evaluate](#) (const [ex](#) &v1, const [ex](#) &v2, const [ex](#) &dim) const
Return value of defined scalar product pair.
- void [debugprint](#) () const

Protected Attributes

- [spm](#) `spm`

6.157.1 Detailed Description

Helper class for storing information about known scalar products which are to be automatically replaced by [simplify_indexed\(\)](#).

See also

[simplify_indexed](#)

6.157.2 Member Function Documentation

6.157.2.1 `add()` [1/2]

```
void GiNaC::scalar_products::add (
    const ex & v1,
    const ex & v2,
    const ex & sp )
```

Register scalar product pair and its value.

References [spm](#).

6.157.2.2 `add()` [2/2]

```
void GiNaC::scalar_products::add (
    const ex & v1,
    const ex & v2,
    const ex & dim,
    const ex & sp )
```

Register scalar product pair and its value for a specific space dimension.

References [spm](#).

6.157.2.3 `add_vectors()`

```
void GiNaC::scalar_products::add_vectors (
    const lst & l,
    const ex & dim = wild\(\) )
```

Register list of vectors.

This adds all possible pairs of products $a_i * b_i$ with the value $a*b$ (note that this is not a scalar vector product but an ordinary product of scalars).

6.157.2.4 clear()

```
void GiNaC::scalar_products::clear ( )
```

Clear all registered scalar products.

References [spm](#).

6.157.2.5 is_defined()

```
bool GiNaC::scalar_products::is_defined (
    const ex & v1,
    const ex & v2,
    const ex & dim ) const
```

Check whether scalar product pair is defined.

References [spm](#).

6.157.2.6 evaluate()

```
ex GiNaC::scalar_products::evaluate (
    const ex & v1,
    const ex & v2,
    const ex & dim ) const
```

Return value of defined scalar product pair.

References [GiNaC::ex::find\(\)](#), and [spm](#).

6.157.2.7 debugprint()

```
void GiNaC::scalar_products::debugprint ( ) const
```

References [GiNaC::spmapkey::debugprint\(\)](#), [k](#), and [spm](#).

6.157.3 Member Data Documentation

6.157.3.1 spm

```
spmap GiNaC::scalar_products::spm [protected]
```

Referenced by [add\(\)](#), [add\(\)](#), [clear\(\)](#), [debugprint\(\)](#), [evaluate\(\)](#), and [is_defined\(\)](#).

The documentation for this class was generated from the following files:

- [indexed.h](#)
- [indexed.cpp](#)

6.158 GiNaC::series_options Class Reference

Flags to control series expansion.

```
#include <flags.h>
```

Public Types

- enum { [suppress_branchcut](#) = 0x0001 }

6.158.1 Detailed Description

Flags to control series expansion.

6.158.2 Member Enumeration Documentation

6.158.2.1 anonymous enum

anonymous enum

Enumerator

suppress_branchcut	Suppress branch cuts in series expansion. Branch cuts manifest themselves as step functions, if this option is not passed. If it is passed and expansion at a point on a cut is performed, then the analytic continuation of the function is expanded.

The documentation for this class was generated from the following file:

- [flags.h](#)

6.159 GiNaC::solve_algo Class Reference

Switch to control algorithm for linear system solving.

```
#include <flags.h>
```

Public Types

- enum {
[automatic](#) , [gauss](#) , [divfree](#) , [bareiss](#) ,
[markowitz](#) }

6.159.1 Detailed Description

Switch to control algorithm for linear system solving.

6.159.2 Member Enumeration Documentation

6.159.2.1 anonymous enum

anonymous enum

Enumerator

automatic	Let the system choose. A heuristics is applied for automatic determination of a suitable algorithm.
gauss	Gauss elimination. If $m_{i,j}^{(0)}$ are the entries of the original matrix, then the matrix is transformed into triangular form by applying the rules $m_{i,j}^{(k+1)} = m_{i,j}^{(k)} - m_{i,k}^{(k)} m_{k,j}^{(k)} / m_{k,k}^{(k)}$ This algorithm is well-suited for numerical matrices but generally suffers from the expensive division (and computation of GCDs) at each step.
divfree	Division-free elimination. This is a modification of Gauss elimination where the division by the pivot element is not carried out. If $m_{i,j}^{(0)}$ are the entries of the original matrix, then the matrix is transformed into triangular form by applying the rules $m_{i,j}^{(k+1)} = m_{i,j}^{(k)} m_{k,k}^{(k)} - m_{i,k}^{(k)} m_{k,j}^{(k)}$ This algorithm is only there for the purpose of cross-checks. It suffers from exponential intermediate expression swell. Use it only for small systems.
bareiss	Bareiss fraction-free elimination. This is a modification of Gauss elimination where the division by the pivot element is <i>delayed</i> until it can be carried out without computing GCDs. If $m_{i,j}^{(0)}$ are the entries of the original matrix, then the matrix is transformed into triangular form by applying the rules $m_{i,j}^{(k+1)} = (m_{i,j}^{(k)} m_{k,k}^{(k)} - m_{i,k}^{(k)} m_{k,j}^{(k)}) / m_{k-1,k-1}^{(k-1)}$ (We have set $m_{-1,-1}^{(-1)} = 1$ in order to avoid a case distinction in above formula.) It can be shown that nothing more than polynomial long division is needed for carrying out the division. This is generally the fastest algorithm for solving linear systems. In contrast to division-free elimination it only has a linear expression swell. For two-dimensional systems, the two algorithms are equivalent, however.
markowitz	Markowitz-ordered Gaussian elimination. Same as the usual Gaussian elimination, but with additional effort spent on selecting pivots that minimize fill-in. Faster than the methods above for large sparse matrices (particularly with symbolic coefficients), otherwise slightly slower than Gaussian elimination.

The documentation for this class was generated from the following file:

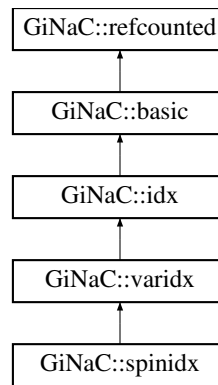
- [flags.h](#)

6.160 GiNaC::spinidx Class Reference

This class holds a spinor index that can be dotted or undotted and that also has a variance.

```
#include <idx.h>
```

Inheritance diagram for GiNaC::spinidx:



Public Member Functions

- `spinidx` (const `ex` &`v`, const `ex` &`dim`=2, bool `covariant`=false, bool `dotted`=false)
Construct index with given value, dimension, variance and dot.
- bool `is_dummy_pair_same_type` (const `basic` &`other`) const override
Check whether the index forms a dummy index pair with another index of the same type.
- `ex conjugate` () const override
- void `archive` (`archive_node` &`n`) const override
Save (serialize) the object into archive node.
- void `read_archive` (const `archive_node` &`n`, `lst` &`syms`) override
Load (deserialize) the object from an archive node.
- bool `is_dotted` () const
Check whether the index is dotted.
- bool `is_undotted` () const
Check whether the index is not dotted.
- `ex toggle_dot` () const
Make a new index with the same value and variance but the opposite dottedness.
- `ex toggle_variance_dot` () const
Make a new index with the same value but opposite variance and dottedness.

Public Member Functions inherited from `GiNaC::varidx`

- `varidx` (const `ex` &`v`, const `ex` &`dim`, bool `covariant`=false)
Construct index with given value, dimension and variance.
- bool `is_dummy_pair_same_type` (const `basic` &`other`) const override
Check whether the index forms a dummy index pair with another index of the same type.
- void `archive` (`archive_node` &`n`) const override
Save (serialize) the object into archive node.
- void `read_archive` (const `archive_node` &`n`, `lst` &`syms`) override
Load (deserialize) the object from an archive node.
- bool `is_covariant` () const
Check whether the index is covariant.
- bool `is_contravariant` () const
Check whether the index is contravariant (not covariant).
- `ex toggle_variance` () const
Make a new index with the same value but the opposite variance.

Public Member Functions inherited from GiNaC::idx

- `idx` (const `ex` &`v`, const `ex` &`dim`)
Construct index with given value and dimension.
- `bool info` (unsigned `inf`) const override
Information about the object.
- `size_t nops` () const override
Number of operands/members.
- `ex op` (size_t `i`) const override
Return operand/member at position `i`.
- `ex map` (`map_function` &`f`) const override
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- `ex evalf` () const override
By default, `basic::evalf` would evaluate the index value but we don't want `a.1` to become `a`.
- `ex subs` (const `exmap` &`m`, unsigned `options=0`) const override
Substitute a set of objects by arbitrary expressions.
- `ex get_value` () const
Get value of index.
- `bool is_numeric` () const
Check whether the index is numeric.
- `bool is_symbolic` () const
Check whether the index is symbolic.
- `ex get_dim` () const
Get dimension of index space.
- `bool is_dim_numeric` () const
Check whether the dimension is numeric.
- `bool is_dim_symbolic` () const
Check whether the dimension is symbolic.
- `ex replace_dim` (const `ex` &`new_dim`) const
Make a new index with the same value but a different dimension.
- `ex minimal_dim` (const `idx` &`other`) const
Return the minimum of the dimensions of this and another index.

Public Member Functions inherited from GiNaC::basic

- `virtual ~basic` ()
basic destructor, virtual because class `ex` will delete objects of derived classes via a `basic*`.
- `basic` (const `basic` &`other`)
- `const basic & operator=` (const `basic` &`other`)
basic assignment operator: the other object might be of a derived class.
- `virtual basic * duplicate` () const
Create a clone of this object on the heap.
- `virtual ex eval` () const
Perform automatic non-interruptive term rewriting rules.
- `virtual ex evalm` () const
Evaluate sums, products and integer powers of matrices.
- `virtual ex eval_integ` () const
Evaluate integrals, if result is known.
- `virtual ex eval_indexed` (const `basic` &`i`) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.

- virtual void `print` (const `print_context` &c, unsigned level=0) const
Output to stream.
- virtual void `dbgprint` () const
Little wrapper around print to be called within a debugger.
- virtual void `dbgprnttree` () const
Little wrapper around prnttree to be called within a debugger.
- virtual unsigned `precedence` () const
Return relative operator precedence (for parenthezing output).
- virtual `ex operator[]` (const `ex` &index) const
- virtual `ex operator[]` (size_t i) const
- virtual `ex & let_op` (size_t i)
Return modifiable operand/member at position i.
- virtual `ex & operator[]` (const `ex` &index)
- virtual `ex & operator[]` (size_t i)
- virtual bool `has` (const `ex` &other, unsigned `options`=0) const
Test for occurrence of a pattern.
- virtual bool `match` (const `ex` &pattern, `exmap` &repls) const
Check whether the expression matches a given pattern.
- virtual void `accept` (`GiNaC::visitor` &v) const
- virtual bool `is_polynomial` (const `ex` &var) const
Check whether this is a polynomial in the given variables.
- virtual int `degree` (const `ex` &s) const
Return degree of highest power in object s.
- virtual int `ldegree` (const `ex` &s) const
Return degree of lowest power in object s.
- virtual `ex coeff` (const `ex` &s, int n=1) const
Return coefficient of degree n in object s.
- virtual `ex expand` (unsigned `options`=0) const
Expand expression, i.e.
- virtual `ex collect` (const `ex` &s, bool distributed=false) const
Sort expanded expression in terms of powers of some object(s).
- virtual `ex series` (const `relational` &r, int `order`, unsigned `options`=0) const
Default implementation of `ex::series()`.
- virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const
Default implementation of `ex::normal()`.
- virtual `ex to_rational` (`exmap` &repl) const
Default implementation of `ex::to_rational()`.
- virtual `ex to_polynomial` (`exmap` &repl) const
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &xi) const
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient` () const
Implementation `ex::max_coefficient()`.
- virtual `exvector get_free_indices` () const
Return a vector containing the free indices of an expression.
- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
Add two indexed expressions.
- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
Multiply an indexed expression with a scalar.
- virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const
Try to contract two indexed expressions that appear in the same product.

- virtual unsigned `return_type` () const
- virtual `return_type_t return_type_tinfo` () const
- virtual `ex real_part` () const
- virtual `ex imag_part` () const
- template<class T >
void `print_dispatch` (const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- `ex subs_one_level` (const `exmap` &m, unsigned `options`) const
Helper function for `subs()`.
- `ex diff` (const `symbol` &s, unsigned nth=1) const
Default interface of nth derivative `ex::diff(s, n)`.
- int `compare` (const `basic` &other) const
Compare objects syntactically to establish canonical ordering.
- bool `is_equal` (const `basic` &other) const
Test for syntactic equality.
- const `basic` & `hold` () const
Stop further evaluation.
- unsigned `gethash` () const
- const `basic` & `setflag` (unsigned f) const
Set some `status_flags`.
- const `basic` & `clearflag` (unsigned f) const
Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Protected Member Functions

- bool `match_same_type` (const `basic` &other) const override
Returns true if the attributes of two objects are similar enough for a match.
- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_latex` (const `print_latex` &c, unsigned level) const
- void `do_print_tree` (const `print_tree` &c, unsigned level) const

Protected Member Functions inherited from `GiNaC::varidx`

- bool `match_same_type` (const `basic` &other) const override
Returns true if the attributes of two objects are similar enough for a match.
- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_tree` (const `print_tree` &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::idx](#)

- [ex derivative](#) (const [symbol](#) &s) const override
Implementation of [ex::diff\(\)](#) for an index always returns 0.
- unsigned [calchash](#) () const override
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [print_index](#) (const [print_context](#) &c, unsigned level) const
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
- void [do_print_csrc](#) (const [print_csrc](#) &c, unsigned level) const
- void [do_print_latex](#) (const [print_latex](#) &c, unsigned level) const
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Protected Attributes

- bool [dotted](#)

Protected Attributes inherited from [GiNaC::varidx](#)

- bool [covariant](#)
 $x.mu$, default is contravariant: $x \sim \mu$

Protected Attributes inherited from [GiNaC::idx](#)

- [ex value](#)
Expression that constitutes the index (numeric or symbolic name)
- [ex dim](#)
Dimension of space (can be symbolic or numeric)

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.160.1 Detailed Description

This class holds a spinor index that can be dotted or undotted and that also has a variance.

This is used in the Weyl-van-der-Waerden formalism where the dot indicates complex conjugation. There is an associated (asymmetric) metric tensor that can be used to raise/lower spinor indices.

6.160.2 Constructor & Destructor Documentation

6.160.2.1 spinidx()

```
GiNaC::spinidx::spinidx (
    const ex & v,
    const ex & dim = 2,
    bool covariant = false,
    bool dotted = false )
```

Construct index with given value, dimension, variance and dot.

Parameters

<i>v</i>	Value of index (numeric or symbolic)
<i>dim</i>	Dimension of index space (numeric or symbolic)
<i>covariant</i>	Make covariant index (default is contravariant)
<i>dotted</i>	Make covariant dotted (default is undotted)

6.160.3 Member Function Documentation

6.160.3.1 is_dummy_pair_same_type()

```
bool GiNaC::spinidx::is_dummy_pair_same_type (
    const basic & other ) const [override], [virtual]
```

Check whether the index forms a dummy index pair with another index of the same type.

Reimplemented from [GiNaC::idx](#).

References [dotted](#).

6.160.3.2 conjugate()

```
ex GiNaC::spinidx::conjugate ( ) const [inline], [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [toggle_dot\(\)](#).

6.160.3.3 archive()

```
void GiNaC::spinidx::archive (
    archive_node & n ) const [override], [virtual]
```

Save (serialize) the object into archive node.

Archive the object.

Losely speaking, this method turns an expression into a byte stream (which can be saved and restored later on, or sent via network, etc.)

Reimplemented from [GiNaC::idx](#).

References [dotted](#), and [n](#).

6.160.3.4 read_archive()

```
void GiNaC::spinidx::read_archive (
    const archive_node & n,
    lst & syms ) [override], [virtual]
```

Load (deserialize) the object from an archive node.

Construct object from [archive_node](#).

Note

This method is essentially a constructor. However, constructors can't be virtual. So, if unarchiving routines are implemented as constructors one would need to define such a constructor in every class, even if all it does is simply calling constructor of a superclass.

Reimplemented from [GiNaC::idx](#).

References [dotted](#), and [n](#).

6.160.3.5 match_same_type()

```
bool GiNaC::spinidx::match_same_type (
    const basic & other ) const [override], [protected], [virtual]
```

Returns true if the attributes of two objects are similar enough for a match.

This function must not match subexpressions (this is already done by [basic::match\(\)](#)). Only attributes not accessible by [op\(\)](#) should be compared. This is also the reason why this function doesn't take the wildcard replacement list from [match\(\)](#) as an argument: only subexpressions are subject to wildcard matches. Also, this function only needs to be implemented for container classes because [is_equal_same_type\(\)](#) is automatically used instead of [match_same_type\(\)](#) if [nops\(\)](#) == 0.

See also

[basic::match](#)

Reimplemented from [GiNaC::idx](#).

References [dotted](#), and [GINAC_ASSERT](#).

6.160.3.6 is_dotted()

```
bool GiNaC::spinidx::is_dotted ( ) const [inline]
```

Check whether the index is dotted.

References [dotted](#).

6.160.3.7 is_undotted()

```
bool GiNaC::spinidx::is_undotted ( ) const [inline]
```

Check whether the index is not dotted.

References [dotted](#).

6.160.3.8 toggle_dot()

```
ex GiNaC::spinidx::toggle_dot ( ) const
```

Make a new index with the same value and variance but the opposite dottedness.

References [GiNaC::basic::clearflag\(\)](#), [dotted](#), [GiNaC::basic::duplicate\(\)](#), and [GiNaC::status_flags::hash_calculated](#).

Referenced by [conjugate\(\)](#).

6.160.3.9 toggle_variance_dot()

```
ex GiNaC::spinidx::toggle_variance_dot ( ) const
```

Make a new index with the same value but opposite variance and dottedness.

References [GiNaC::basic::clearflag\(\)](#), [GiNaC::varidx::covariant](#), [dotted](#), [GiNaC::basic::duplicate\(\)](#), and [GiNaC::status_flags::hash_cal](#).

6.160.3.10 do_print()

```
void GiNaC::spinidx::do_print (
    const print\_context & c,
    unsigned level ) const [protected]
```

References [c](#), [GiNaC::varidx::covariant](#), [dotted](#), and [GiNaC::idx::print_index\(\)](#).

6.160.3.11 do_print_latex()

```
void GiNaC::spinidx::do_print_latex (
    const print\_latex & c,
    unsigned level ) const [protected]
```

References [c](#), [dotted](#), and [GiNaC::idx::print_index\(\)](#).

6.160.3.12 do_print_tree()

```
void GiNaC::spinidx::do_print_tree (
    const print\_tree & c,
    unsigned level ) const [protected]
```

References [c](#), [GiNaC::varidx::covariant](#), [GiNaC::idx::dim](#), [dotted](#), [GiNaC::basic::flags](#), [GiNaC::basic::hashvalue](#), [GiNaC::ex::print\(\)](#), and [GiNaC::idx::value](#).

6.160.4 Member Data Documentation

6.160.4.1 dotted

```
bool GiNaC::spinidx::dotted [protected]
```

Referenced by [archive\(\)](#), [do_print\(\)](#), [do_print_latex\(\)](#), [do_print_tree\(\)](#), [is_dotted\(\)](#), [is_dummy_pair_same_type\(\)](#), [is_undotted\(\)](#), [match_same_type\(\)](#), [read_archive\(\)](#), [toggle_dot\(\)](#), and [toggle_variance_dot\(\)](#).

The documentation for this class was generated from the following files:

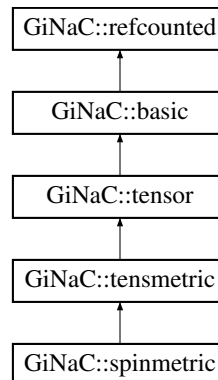
- [idx.h](#)
- [idx.cpp](#)

6.161 GiNaC::spinmetric Class Reference

This class represents an antisymmetric spinor metric tensor which can be used to raise/lower indices of 2-component Weyl spinors.

```
#include <tensor.h>
```

Inheritance diagram for GiNaC::spinmetric:



Public Member Functions

- bool [info](#) (unsigned inf) const override
Information about the object.
- [ex eval_indexed](#) (const [basic](#) &i) const override
Automatic symbolic evaluation of an indexed metric tensor.
- bool [contract_with](#) (exvector::iterator self, exvector::iterator other, [exvector](#) &v) const override
Contraction of an indexed spinor metric with something else.

Public Member Functions inherited from [GiNaC::tensmetric](#)

- bool [info](#) (unsigned inf) const override
Information about the object.
- [ex eval_indexed](#) (const [basic](#) &i) const override
Automatic symbolic evaluation of an indexed metric tensor.
- bool [contract_with](#) (exvector::iterator self, exvector::iterator other, [exvector](#) &v) const override
Contraction of an indexed metric tensor with something else.

Public Member Functions inherited from [GiNaC::tensor](#)

- bool [replace_contr_index](#) (exvector::iterator self, exvector::iterator other) const
Replace dummy index in contracted-with object by the contracting object's second index (used internally for delta and metric tensor contractions).

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic ()`
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic (const basic &other)`
- `const basic & operator= (const basic &other)`
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate () const`
Create a clone of this object on the heap.
- virtual `ex eval () const`
Perform automatic non-interruptive term rewriting rules.
- virtual `ex evalf () const`
Evaluate object numerically.
- virtual `ex evalm () const`
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ () const`
Evaluate integrals, if result is known.
- virtual `void print (const print_context &c, unsigned level=0) const`
Output to stream.
- virtual `void dbgprint () const`
Little wrapper around print to be called within a debugger.
- virtual `void dbgprintree () const`
Little wrapper around printree to be called within a debugger.
- virtual `unsigned precedence () const`
Return relative operator precedence (for parenthezing output).
- virtual `size_t nops () const`
Number of operands/members.
- virtual `ex op (size_t i) const`
Return operand/member at position i.
- virtual `ex operator[] (const ex &index) const`
- virtual `ex operator[] (size_t i) const`
- virtual `ex & let_op (size_t i)`
Return modifiable operand/member at position i.
- virtual `ex & operator[] (const ex &index)`
- virtual `ex & operator[] (size_t i)`
- virtual `bool has (const ex &other, unsigned options=0) const`
Test for occurrence of a pattern.
- virtual `bool match (const ex &pattern, exmap &repls) const`
Check whether the expression matches a given pattern.
- virtual `ex subs (const exmap &m, unsigned options=0) const`
Substitute a set of objects by arbitrary expressions.
- virtual `ex map (map_function &f) const`
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual `void accept (GiNaC::visitor &v) const`
- virtual `bool is_polynomial (const ex &var) const`
Check whether this is a polynomial in the given variables.
- virtual `int degree (const ex &s) const`
Return degree of highest power in object s.
- virtual `int ldegree (const ex &s) const`
Return degree of lowest power in object s.
- virtual `ex coeff (const ex &s, int n=1) const`

- Return coefficient of degree n in object s .*

 - virtual `ex expand` (unsigned `options=0`) const

Expand expression, i.e.
 - virtual `ex collect` (const `ex` & s , bool `distributed=false`) const

Sort expanded expression in terms of powers of some object(s).
 - virtual `ex series` (const `relational` & r , int `order`, unsigned `options=0`) const

Default implementation of `ex::series()`.
 - virtual `ex normal` (`exmap` & $repl$, `exmap` & rev_lookup , `lst` & $modifier$) const

Default implementation of `ex::normal()`.
 - virtual `ex to_rational` (`exmap` & $repl$) const

Default implementation of `ex::to_rational()`.
 - virtual `ex to_polynomial` (`exmap` & $repl$) const
 - virtual `numeric integer_content` () const
 - virtual `ex smod` (const `numeric` & xi) const
- Apply symmetric modular homomorphism to an expanded multivariate polynomial.*
- virtual `numeric max_coefficient` () const
- Implementation `ex::max_coefficient()`.*
- virtual `exvector get_free_indices` () const
- Return a vector containing the free indices of an expression.*
- virtual `ex add_indexed` (const `ex` & $self$, const `ex` & $other$) const
- Add two indexed expressions.*
- virtual `ex scalar_mul_indexed` (const `ex` & $self$, const `numeric` & $other$) const
- Multiply an indexed expression with a scalar.*
- virtual `return_type_t return_type_info` () const
 - virtual `ex conjugate` () const
 - virtual `ex real_part` () const
 - virtual `ex imag_part` () const
- template<class T >
 - void `print_dispatch` (const `print_context` & c , unsigned `level`) const
- Like `print()`, but dispatch to the specified class.*
- void `print_dispatch` (const `registered_class_info` & ri , const `print_context` & c , unsigned `level`) const
- Like `print()`, but dispatch to the specified class.*
- virtual void `archive` (`archive_node` & n) const
- Save (serialize) the object into archive node.*
- virtual void `read_archive` (const `archive_node` & n , `lst` & $syms$)
- Load (deserialize) the object from an archive node.*
- `ex subs_one_level` (const `exmap` & m , unsigned `options`) const
- Helper function for `subs()`.*
- `ex diff` (const `symbol` & s , unsigned `nth=1`) const
- Default interface of n th derivative `ex::diff(s, n)`.*
- int `compare` (const `basic` & $other$) const
- Compare objects syntactically to establish canonical ordering.*
- bool `is_equal` (const `basic` & $other$) const
- Test for syntactic equality.*
- const `basic` & `hold` () const
- Stop further evaluation.*
- unsigned `gethash` () const
 - const `basic` & `setflag` (unsigned `f`) const
- Set some `status_flags`.*
- const `basic` & `clearflag` (unsigned `f`) const
- Clear some `status_flags`.*

Public Member Functions inherited from [GiNaC::refcounted](#)

- [refcounted](#) () noexcept
- unsigned int [add_reference](#) () noexcept
- unsigned int [remove_reference](#) () noexcept
- unsigned int [get_refcount](#) () const noexcept
- void [set_refcount](#) (unsigned int r) noexcept

Protected Member Functions

- void [do_print](#) (const [print_context](#) &c, unsigned level) const
- void [do_print_latex](#) (const [print_latex](#) &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::tensmetric](#)

- unsigned [return_type](#) () const override
- void [do_print](#) (const [print_context](#) &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::tensor](#)

- unsigned [return_type](#) () const override

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual [ex derivative](#) (const [symbol](#) &s) const
Default implementation of [ex::diff\(\)](#).
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Additional Inherited Members

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.161.1 Detailed Description

This class represents an antisymmetric spinor metric tensor which can be used to raise/lower indices of 2-component Weyl spinors.

If indexed, it must have exactly two indices of the same type which must be of class `spinidx` or a subclass and have dimension 2.

6.161.2 Member Function Documentation

6.161.2.1 `info()`

```
bool GiNaC::spinmetric::info (
    unsigned inf ) const [override], [virtual]
```

Information about the object.

See also

class [info_flags](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::info_flags::real](#).

6.161.2.2 `eval_indexed()`

```
ex GiNaC::spinmetric::eval_indexed (
    const basic & i ) const [override], [virtual]
```

Automatic symbolic evaluation of an indexed metric tensor.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex0](#), [GiNaC::_ex1](#), [GiNaC::_ex_1](#), [GiNaC::idx::get_value\(\)](#), [GINAC_ASSERT](#), [GiNaC::basic::hold\(\)](#), [GiNaC::info_flags::nonnegint](#), [GiNaC::basic::nops\(\)](#), and [GiNaC::basic::op\(\)](#).

6.161.2.3 contract_with()

```
bool GiNaC::spinmetric::contract_with (
    exvector::iterator self,
    exvector::iterator other,
    exvector & v ) const [override], [virtual]
```

Contraction of an indexed spinor metric with something else.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex1](#), [GiNaC::_ex2](#), [GiNaC::_ex_2](#), [GiNaC::delta_tensor\(\)](#), [GINAC_ASSERT](#), [GiNaC::is_dummy_pair\(\)](#), and [GiNaC::idx::is_symbolic\(\)](#).

6.161.2.4 do_print()

```
void GiNaC::spinmetric::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

6.161.2.5 do_print_latex()

```
void GiNaC::spinmetric::do_print_latex (
    const print_latex & c,
    unsigned level ) const [protected]
```

The documentation for this class was generated from the following files:

- [tensor.h](#)
- [tensor.cpp](#)

6.162 GiNaC::spmapkey Class Reference

```
#include <indexed.h>
```

Public Member Functions

- [spmapkey](#) ()
- [spmapkey](#) (const [ex](#) &v1, const [ex](#) &v2, const [ex](#) &dim=[wild](#)())
- bool [operator==](#) (const [spmapkey](#) &other) const
- bool [operator<](#) (const [spmapkey](#) &other) const
- void [debugprint](#) () const

Protected Attributes

- [ex](#) v1
- [ex](#) v2
- [ex](#) dim

6.162.1 Constructor & Destructor Documentation

6.162.1.1 spmapkey() [1/2]

```
GiNaC::spmapkey::spmapkey ( ) [inline]
```

6.162.1.2 spmapkey() [2/2]

```
GiNaC::spmapkey::spmapkey (
    const ex & v1,
    const ex & v2,
    const ex & dim = wild() )
```

References [GiNaC::ex::compare\(\)](#), [GiNaC::ex::op\(\)](#), [v1](#), and [v2](#).

6.162.2 Member Function Documentation

6.162.2.1 operator==()

```
bool GiNaC::spmapkey::operator== (
    const spmapkey & other ) const
```

References [dim](#), [GiNaC::ex::is_equal\(\)](#), [v1](#), and [v2](#).

6.162.2.2 operator<()

```
bool GiNaC::spmapkey::operator< (
    const spmapkey & other ) const
```

References [GiNaC::ex::compare\(\)](#), [dim](#), [v1](#), and [v2](#).

6.162.2.3 debugprint()

```
void GiNaC::spmapkey::debugprint ( ) const
```

References [dim](#), [v1](#), and [v2](#).

Referenced by [GiNaC::scalar_products::debugprint\(\)](#).

6.162.3 Member Data Documentation

6.162.3.1 v1

```
ex GiNaC::spmapkey::v1 [protected]
```

Referenced by [debugprint\(\)](#), [operator<\(\)](#), [operator==\(\)](#), and [spmapkey\(\)](#).

6.162.3.2 v2

`ex GiNaC::spmapkey::v2 [protected]`

Referenced by [debugprint\(\)](#), [operator<\(\)](#), [operator==\(\)](#), and [spmapkey\(\)](#).

6.162.3.3 dim

`ex GiNaC::spmapkey::dim [protected]`

Referenced by [debugprint\(\)](#), [operator<\(\)](#), and [operator==\(\)](#).

The documentation for this class was generated from the following files:

- [indexed.h](#)
- [indexed.cpp](#)

6.163 GiNaC::status_flags Class Reference

Flags to store information about the state of an object.

```
#include <flags.h>
```

Public Types

- enum {
[dynallocated](#) = 0x0001 , [evaluated](#) = 0x0002 , [expanded](#) = 0x0004 , [hash_calculated](#) = 0x0008 ,
[not_shareable](#) = 0x0010 , [has_indices](#) = 0x0020 , [has_no_indices](#) = 0x0040 , [is_positive](#) = 0x0080 ,
[is_negative](#) = 0x0100 , [purely_indefinite](#) = 0x0200 }

6.163.1 Detailed Description

Flags to store information about the state of an object.

See also

[basic::flags](#)

6.163.2 Member Enumeration Documentation

6.163.2.1 anonymous enum

anonymous enum

Enumerator

dynallocated	heap-allocated (i.e. created by new if we want to be clever and bypass the stack, See also ex::construct_from_basic())
evaluated	.eval() has already done its job
expanded	.expand(0) has already done its job (other expand() options ignore this flag)
hash_calculated	.calchash() has already done its job
not_shareable	don't share instances of this object between different expressions unless explicitly asked to (used by ex::compare())
has_indices	
has_no_indices	
is_positive	
is_negative	
purely_indefinite	

The documentation for this class was generated from the following file:

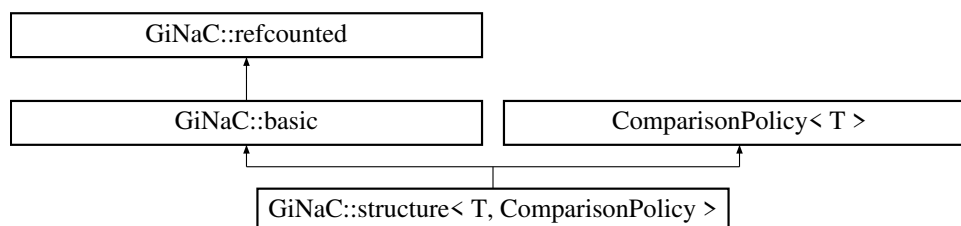
- [flags.h](#)

6.164 GiNaC::structure< T, ComparisonPolicy > Class Template Reference

Wrapper template for making [GiNaC](#) classes out of C++ structures.

```
#include <structure.h>
```

Inheritance diagram for GiNaC::structure< T, ComparisonPolicy >:



Public Member Functions

- [structure](#) (const T &t)
Construct structure as a copy of a given C++ structure.
- [ex eval](#) () const override
Perform automatic non-interruptive term rewriting rules.
- [ex evalm](#) () const override
Evaluate sums, products and integer powers of matrices.
- [ex eval_indexed](#) (const [basic](#) &i) const override

- Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.*

 - void `print` (const `print_context` &c, unsigned level=0) const override

Output to stream.
- unsigned `precedence` () const override

Return relative operator precedence (for parenthezing output).
- bool `info` (unsigned inf) const override

Information about the object.
- size_t `nops` () const override

Number of operands/members.
- `ex op` (size_t i) const override

Return operand/member at position i.
- `ex operator[]` (const `ex` &index) const override
- `ex operator[]` (size_t i) const override
- `ex & let_op` (size_t i) override

Return modifiable operand/member at position i.
- `ex & operator[]` (const `ex` &index) override
- `ex & operator[]` (size_t i) override
- bool `has` (const `ex` &other, unsigned `options`=0) const override

Test for occurrence of a pattern.
- bool `match` (const `ex` &pattern, `exmap` &repl_lst) const override

Check whether the expression matches a given pattern.
- `ex subs` (const `exmap` &m, unsigned `options`=0) const override

Substitute a set of objects by arbitrary expressions.
- `ex map` (`map_function` &f) const override

Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- int `degree` (const `ex` &s) const override

Return degree of highest power in object s.
- int `ldegree` (const `ex` &s) const override

Return degree of lowest power in object s.
- `ex coeff` (const `ex` &s, int n=1) const override

Return coefficient of degree n in object s.
- `ex expand` (unsigned `options`=0) const override

Expand expression, i.e.
- `ex collect` (const `ex` &s, bool distributed=false) const override

Sort expanded expression in terms of powers of some object(s).
- `ex series` (const `relational` &r, int `order`, unsigned `options`=0) const override

Default implementation of `ex::series()`.
- `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const override

Default implementation of `ex::normal()`.
- `ex to_rational` (`exmap` &repl) const override

Default implementation of `ex::to_rational()`.
- `ex to_polynomial` (`exmap` &repl) const override
- `numeric integer_content` () const override
- `ex smod` (const `numeric` &xi) const override

Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- `numeric max_coefficient` () const override

Implementation `ex::max_coefficient()`.
- `exvector get_free_indices` () const override

Return a vector containing the free indices of an expression.
- `ex add_indexed` (const `ex` &self, const `ex` &other) const override

Add two indexed expressions.

- `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const override
Multiply an indexed expression with a scalar.
- `bool contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const override
Try to contract two indexed expressions that appear in the same product.
- `unsigned return_type` () const override
- `return_type_t return_type_tinfo` () const override
- `const T * operator->` () const
- `T & get_struct` ()
- `const T & get_struct` () const

Public Member Functions inherited from `GiNaC::basic`

- `virtual ~basic` ()
basic destructor, virtual because class `ex` will delete objects of derived classes via a `basic*`.
- `basic` (const `basic` &other)
- `const basic & operator=` (const `basic` &other)
basic assignment operator: the other object might be of a derived class.
- `virtual basic * duplicate` () const
Create a clone of this object on the heap.
- `virtual ex evalf` () const
Evaluate object numerically.
- `virtual ex eval_integ` () const
Evaluate integrals, if result is known.
- `virtual void dbgprint` () const
Little wrapper around `print` to be called within a debugger.
- `virtual void dbgprinttree` () const
Little wrapper around `printtree` to be called within a debugger.
- `virtual void accept` (`GiNaC::visitor` &v) const
- `virtual bool is_polynomial` (const `ex` &var) const
Check whether this is a polynomial in the given variables.
- `virtual ex conjugate` () const
- `virtual ex real_part` () const
- `virtual ex imag_part` () const
- `template<class T>`
`void print_dispatch` (const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- `void print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- `virtual void archive` (`archive_node` &n) const
Save (serialize) the object into archive node.
- `virtual void read_archive` (const `archive_node` &n, `lst` &syms)
Load (deserialize) the object from an archive node.
- `ex subs_one_level` (const `exmap` &m, unsigned `options`) const
Helper function for `subs()`.
- `ex diff` (const `symbol` &s, unsigned `nth`=1) const
Default interface of `nth` derivative `ex::diff(s, n)`.
- `int compare` (const `basic` &other) const
Compare objects syntactically to establish canonical ordering.
- `bool is_equal` (const `basic` &other) const
Test for syntactic equality.
- `const basic & hold` () const

Stop further evaluation.

- unsigned [gethash](#) () const
- const [basic](#) & [setflag](#) (unsigned f) const

Set some [status_flags](#).

- const [basic](#) & [clearflag](#) (unsigned f) const

Clear some [status_flags](#).

Public Member Functions inherited from [GiNaC::refcounted](#)

- [refcounted](#) () noexcept
- unsigned int [add_reference](#) () noexcept
- unsigned int [remove_reference](#) () noexcept
- unsigned int [get_refcount](#) () const noexcept
- void [set_refcount](#) (unsigned int r) noexcept

Protected Member Functions

- [ex eval_ncmul](#) (const [exvector](#) &v) const override
- bool [match_same_type](#) (const [basic](#) &other) const override

Returns true if the attributes of two objects are similar enough for a match.
- [ex derivative](#) (const [symbol](#) &s) const override

Default implementation of [ex::diff\(\)](#).
- bool [is_equal_same_type](#) (const [basic](#) &other) const override

Returns true if two objects of same type are equal.
- unsigned [calchash](#) () const override

Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual int [compare_same_type](#) (const [basic](#) &other) const

Returns order relation between two objects of same type.
- void [ensure_if_modifiable](#) () const

Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const

Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const

Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const

Python parsable output to stream.

Static Private Member Functions

- static const char * [get_class_name](#) ()

Private Attributes

- T [obj](#)

Additional Inherited Members

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.164.1 Detailed Description

template<class T, template< class > class ComparisonPolicy>
class GiNaC::structure< T, ComparisonPolicy >

Wrapper template for making [GiNaC](#) classes out of C++ structures.

6.164.2 Constructor & Destructor Documentation

6.164.2.1 structure()

```
template<class T , template< class > class ComparisonPolicy>
GiNaC::structure< T, CP >::structure (
    const T & t ) [inline]
```

Construct structure as a copy of a given C++ structure.

Default constructor.

6.164.3 Member Function Documentation

6.164.3.1 get_class_name()

```
template<class T , template< class > class ComparisonPolicy>
static const char * GiNaC::structure< T, ComparisonPolicy >::get_class_name ( ) [inline],
[static], [private]
```

6.164.3.2 eval()

```
template<class T , template< class > class ComparisonPolicy>
ex GiNaC::structure< T, ComparisonPolicy >::eval ( ) const [inline], [override], [virtual]
```

Perform automatic non-interruptive term rewriting rules.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::basic::hold\(\)](#).

6.164.3.3 evalm()

```
template<class T , template< class > class ComparisonPolicy>
ex GiNaC::structure< T, ComparisonPolicy >::evalm ( ) const [inline], [override], [virtual]
```

Evaluate sums, products and integer powers of matrices.

Reimplemented from [GiNaC::basic](#).

6.164.3.4 eval_ncmul()

```
template<class T , template< class > class ComparisonPolicy>
ex GiNaC::structure< T, ComparisonPolicy >::eval_ncmul (
    const exvector & v ) const [inline], [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::hold_ncmul\(\)](#).

6.164.3.5 eval_indexed()

```
template<class T , template< class > class ComparisonPolicy>
ex GiNaC::structure< T, ComparisonPolicy >::eval_indexed (
    const basic & i ) const [inline], [override], [virtual]
```

Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::basic::hold\(\)](#).

6.164.3.6 print()

```
template<class T , template< class > class ComparisonPolicy>
void GiNaC::structure< T, ComparisonPolicy >::print (
    const print_context & c,
    unsigned level = 0 ) const [inline], [override], [virtual]
```

Output to stream.

This performs double dispatch on the dynamic type of *this and the dynamic type of the supplied print context.

Parameters

<i>c</i>	print context object that describes the output formatting
<i>level</i>	value that is used to identify the precedence or indentation level for placing parentheses and formatting

Reimplemented from [GiNaC::basic](#).

References [c](#).

6.164.3.7 precedence()

```
template<class T , template< class > class ComparisonPolicy>
unsigned GiNaC::structure< T, ComparisonPolicy >::precedence ( ) const [inline], [override],
[virtual]
```

Return relative operator precedence (for parenthezing output).

Reimplemented from [GiNaC::basic](#).

6.164.3.8 info()

```
template<class T , template< class > class ComparisonPolicy>
bool GiNaC::structure< T, ComparisonPolicy >::info (
    unsigned inf ) const [inline], [override], [virtual]
```

Information about the object.

See also

class [info_flags](#)

Reimplemented from [GiNaC::basic](#).

6.164.3.9 nops()

```
template<class T , template< class > class ComparisonPolicy>
size_t GiNaC::structure< T, ComparisonPolicy >::nops ( ) const [inline], [override], [virtual]
```

Number of operands/members.

Reimplemented from [GiNaC::basic](#).

6.164.3.10 op()

```
template<class T , template< class > class ComparisonPolicy>
ex GiNaC::structure< T, ComparisonPolicy >::op (
    size_t i ) const [inline], [override], [virtual]
```

Return operand/member at position i.

Reimplemented from [GiNaC::basic](#).

6.164.3.11 operator[]() [1/4]

```
template<class T , template< class > class ComparisonPolicy>
ex GiNaC::structure< T, ComparisonPolicy >::operator[] (
    const ex & index ) const [inline], [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

6.164.3.12 operator[]() [2/4]

```
template<class T , template< class > class ComparisonPolicy>
ex GiNaC::structure< T, ComparisonPolicy >::operator[] (
    size_t i ) const [inline], [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

6.164.3.13 let_op()

```
template<class T , template< class > class ComparisonPolicy>
ex & GiNaC::structure< T, ComparisonPolicy >::let_op (
    size_t i ) [inline], [override], [virtual]
```

Return modifiable operand/member at position i.

Reimplemented from [GiNaC::basic](#).

6.164.3.14 operator[]() [3/4]

```
template<class T , template< class > class ComparisonPolicy>
ex & GiNaC::structure< T, ComparisonPolicy >::operator[] (
    const ex & index ) [inline], [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

6.164.3.15 operator[]() [4/4]

```
template<class T , template< class > class ComparisonPolicy>
ex & GiNaC::structure< T, ComparisonPolicy >::operator[] (
    size_t i ) [inline], [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

6.164.3.16 has()

```
template<class T , template< class > class ComparisonPolicy>
bool GiNaC::structure< T, ComparisonPolicy >::has (
    const ex & pattern,
    unsigned options = 0 ) const [inline], [override], [virtual]
```

Test for occurrence of a pattern.

An object 'has' a pattern if it matches the pattern itself or one of the children 'has' it. As a consequence (according to the definition of children) given $e=x+y+z$, $e.has(x)$ is true but $e.has(x+y)$ is false.

Reimplemented from [GiNaC::basic](#).

References [options](#).

6.164.3.17 match()

```
template<class T , template< class > class ComparisonPolicy>
bool GiNaC::structure< T, ComparisonPolicy >::match (
    const ex & pattern,
    exmap & repl_lst ) const [inline], [override], [virtual]
```

Check whether the expression matches a given pattern.

For every wildcard object in the pattern, a pair with the wildcard as a key and matching expression as a value is added to *repl_lst*.

Reimplemented from [GiNaC::basic](#).

6.164.3.18 match_same_type()

```
template<class T , template< class > class ComparisonPolicy>
bool GiNaC::structure< T, ComparisonPolicy >::match_same_type (
    const basic & other ) const [inline], [override], [protected], [virtual]
```

Returns true if the attributes of two objects are similar enough for a match.

This function must not match subexpressions (this is already done by [basic::match\(\)](#)). Only attributes not accessible by [op\(\)](#) should be compared. This is also the reason why this function doesn't take the wildcard replacement list from [match\(\)](#) as an argument: only subexpressions are subject to wildcard matches. Also, this function only needs to be implemented for container classes because [is_equal_same_type\(\)](#) is automatically used instead of [match_same_type\(\)](#) if [nops\(\)](#) == 0.

See also

[basic::match](#)

Reimplemented from [GiNaC::basic](#).

6.164.3.19 subs()

```
template<class T , template< class > class ComparisonPolicy>
ex GiNaC::structure< T, ComparisonPolicy >::subs (
    const exmap & m,
    unsigned options = 0 ) const [inline], [override], [virtual]
```

Substitute a set of objects by arbitrary expressions.

The *ex* returned will already be evaluated.

Reimplemented from [GiNaC::basic](#).

References [m](#), and [options](#).

6.164.3.20 map()

```
template<class T , template< class > class ComparisonPolicy>
ex GiNaC::structure< T, ComparisonPolicy >::map (
    map_function & f ) const [inline], [override], [virtual]
```

Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).

Reimplemented from [GiNaC::basic](#).

6.164.3.21 degree()

```
template<class T , template< class > class ComparisonPolicy>
int GiNaC::structure< T, ComparisonPolicy >::degree (
    const ex & s ) const [inline], [override], [virtual]
```

Return degree of highest power in object s.

Reimplemented from [GiNaC::basic](#).

6.164.3.22 ldegree()

```
template<class T , template< class > class ComparisonPolicy>
int GiNaC::structure< T, ComparisonPolicy >::ldegree (
    const ex & s ) const [inline], [override], [virtual]
```

Return degree of lowest power in object s.

Reimplemented from [GiNaC::basic](#).

6.164.3.23 coeff()

```
template<class T , template< class > class ComparisonPolicy>
ex GiNaC::structure< T, ComparisonPolicy >::coeff (
    const ex & s,
    int n = 1 ) const [inline], [override], [virtual]
```

Return coefficient of degree n in object s.

Reimplemented from [GiNaC::basic](#).

References [n](#).

6.164.3.24 expand()

```
template<class T , template< class > class ComparisonPolicy>
ex GiNaC::structure< T, ComparisonPolicy >::expand (
    unsigned options = 0 ) const [inline], [override], [virtual]
```

Expand expression, i.e.

multiply it out and return the result as a new expression.

Reimplemented from [GiNaC::basic](#).

References [options](#).

6.164.3.25 collect()

```
template<class T , template< class > class ComparisonPolicy>
ex GiNaC::structure< T, ComparisonPolicy >::collect (
    const ex & s,
    bool distributed = false ) const [inline], [override], [virtual]
```

Sort expanded expression in terms of powers of some object(s).

Parameters

<i>s</i>	object(s) to sort in
<i>distributed</i>	recursive or distributed form (only used when s is a list)

Reimplemented from [GiNaC::basic](#).

6.164.3.26 derivative()

```
template<class T , template< class > class ComparisonPolicy>
ex GiNaC::structure< T, ComparisonPolicy >::derivative (
    const symbol & s ) const [inline], [override], [protected], [virtual]
```

Default implementation of [ex::diff\(\)](#).

It maps the operation on the operands (or returns 0 when the object has no operands).

See also

[ex::diff](#)

Reimplemented from [GiNaC::basic](#).

6.164.3.27 series()

```
template<class T , template< class > class ComparisonPolicy>
ex GiNaC::structure< T, ComparisonPolicy >::series (
    const relational & r,
    int order,
    unsigned options = 0 ) const [inline], [override], [virtual]
```

Default implementation of [ex::series\(\)](#).

This performs Taylor expansion.

See also

[ex::series](#)

Reimplemented from [GiNaC::basic](#).

References [options](#), [order](#), and [r](#).

6.164.3.28 normal()

```
template<class T , template< class > class ComparisonPolicy>
ex GiNaC::structure< T, ComparisonPolicy >::normal (
    exmap & repl,
    exmap & rev_lookup,
    lst & modifier ) const [inline], [override], [virtual]
```

Default implementation of [ex::normal\(\)](#).

It normalizes the children and replaces the object with a temporary symbol.

See also

[ex::normal](#)

Reimplemented from [GiNaC::basic](#).

6.164.3.29 to_rational()

```
template<class T , template< class > class ComparisonPolicy>
ex GiNaC::structure< T, ComparisonPolicy >::to_rational (
    exmap & repl ) const [inline], [override], [virtual]
```

Default implementation of [ex::to_rational\(\)](#).

This replaces the object with a temporary symbol.

Reimplemented from [GiNaC::basic](#).

6.164.3.30 to_polynomial()

```
template<class T , template< class > class ComparisonPolicy>
ex GiNaC::structure< T, ComparisonPolicy >::to_polynomial (
    exmap & repl ) const [inline], [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

6.164.3.31 integer_content()

```
template<class T , template< class > class ComparisonPolicy>
numeric GiNaC::structure< T, ComparisonPolicy >::integer_content ( ) const [inline], [override],
[virtual]
```

Reimplemented from [GiNaC::basic](#).

6.164.3.32 smod()

```
template<class T , template< class > class ComparisonPolicy>
ex GiNaC::structure< T, ComparisonPolicy >::smod (
    const numeric & xi ) const [inline], [override], [virtual]
```

Apply symmetric modular homomorphism to an expanded multivariate polynomial.

This function is usually used internally by [heur_gcd\(\)](#).

Parameters

xi	modulus

Returns

mapped polynomial

See also

[heur_gcd](#)

Reimplemented from [GiNaC::basic](#).

6.164.3.33 max_coefficient()

```
template<class T , template< class > class ComparisonPolicy>
numeric GiNaC::structure< T, ComparisonPolicy >::max_coefficient ( ) const [inline], [override],
[virtual]
```

Implementation [ex::max_coefficient\(\)](#).

See also

[heur_gcd](#)

Reimplemented from [GiNaC::basic](#).

6.164.3.34 get_free_indices()

```
template<class T , template< class > class ComparisonPolicy>
exvector GiNaC::structure< T, ComparisonPolicy >::get_free_indices ( ) const [inline], [override],
[virtual]
```

Return a vector containing the free indices of an expression.

Reimplemented from [GiNaC::basic](#).

6.164.3.35 add_indexed()

```
template<class T , template< class > class ComparisonPolicy>
ex GiNaC::structure< T, ComparisonPolicy >::add_indexed (
    const ex & self,
    const ex & other ) const [inline], [override], [virtual]
```

Add two indexed expressions.

They are guaranteed to be of class indexed (or a subclass) and their indices are compatible. This function is used internally by [simplify_indexed\(\)](#).

Parameters

<i>self</i>	First indexed expression; its base object is *this
<i>other</i>	Second indexed expression

Returns

sum of self and other

See also

[ex::simplify_indexed\(\)](#)

Reimplemented from [GiNaC::basic](#).

6.164.3.36 scalar_mul_indexed()

```
template<class T , template< class > class ComparisonPolicy>
ex GiNaC::structure< T, ComparisonPolicy >::scalar_mul_indexed (
    const ex & self,
    const numeric & other ) const [inline], [override], [virtual]
```

Multiply an indexed expression with a scalar.

This function is used internally by [simplify_indexed\(\)](#).

Parameters

<i>self</i>	Indexed expression; its base object is *this
<i>other</i>	Numeric value

Returns

product of self and other

See also

[ex::simplify_indexed\(\)](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::basic::ex](#).

6.164.3.37 contract_with()

```
template<class T , template< class > class ComparisonPolicy>
bool GiNaC::structure< T, ComparisonPolicy >::contract_with (
    exvector::iterator self,
```

```
exvector::iterator other,
exvector & v ) const [inline], [override], [virtual]
```

Try to contract two indexed expressions that appear in the same product.

If a contraction exists, the function overwrites one or both of the expressions and returns true. Otherwise it returns false. It is guaranteed that both expressions are of class indexed (or a subclass) and that at least one dummy index has been found. This functions is used internally by [simplify_indexed\(\)](#).

Parameters

<i>self</i>	Pointer to first indexed expression; its base object is *this
<i>other</i>	Pointer to second indexed expression
<i>v</i>	The complete vector of factors

Returns

true if the contraction was successful, false otherwise

See also

[ex::simplify_indexed\(\)](#)

Reimplemented from [GiNaC::basic](#).

6.164.3.38 return_type()

```
template<class T , template< class > class ComparisonPolicy>
unsigned GiNaC::structure< T, ComparisonPolicy >::return_type ( ) const [inline], [override],
[virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::return_types::commutative](#).

6.164.3.39 return_type_tinfo()

```
template<class T , template< class > class ComparisonPolicy>
return_type_t GiNaC::structure< T, ComparisonPolicy >::return_type_tinfo ( ) const [inline],
[override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [r](#), and [GiNaC::return_type_t::rl](#).

6.164.3.40 is_equal_same_type()

```
template<class T , template< class > class ComparisonPolicy>
bool GiNaC::structure< T, ComparisonPolicy >::is_equal_same_type (
    const basic & other ) const [inline], [override], [protected], [virtual]
```

Returns true if two objects of same type are equal.

Normally needs not be reimplemented as long as it wasn't overwritten by some parent class, since it just calls [compare_same_type\(\)](#). The reason why this function exists is that sometimes it is easier to determine equality than an order relation and then it can be overridden.

Reimplemented from [GiNaC::basic](#).

References [GINAC_ASSERT](#), and [GiNaC::structure< T, ComparisonPolicy >::obj](#).

6.164.3.41 calchash()

```
template<class T , template< class > class ComparisonPolicy>
unsigned GiNaC::structure< T, ComparisonPolicy >::calchash ( ) const [inline], [override],
[protected], [virtual]
```

Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.

The method inherited from class basic computes a hash value based on the type and hash values of possible members. For this reason it is well suited for container classes but atomic classes should override this implementation because otherwise they would all end up with the same hashvalue.

Reimplemented from [GiNaC::basic](#).

6.164.3.42 operator->()

```
template<class T , template< class > class ComparisonPolicy>
const T * GiNaC::structure< T, ComparisonPolicy >::operator-> ( ) const [inline]
```

References [GiNaC::structure< T, ComparisonPolicy >::obj](#).

6.164.3.43 get_struct() [1/2]

```
template<class T , template< class > class ComparisonPolicy>
T & GiNaC::structure< T, ComparisonPolicy >::get_struct ( ) [inline]
```

References [GiNaC::structure< T, ComparisonPolicy >::obj](#).

6.164.3.44 get_struct() [2/2]

```
template<class T , template< class > class ComparisonPolicy>
const T & GiNaC::structure< T, ComparisonPolicy >::get_struct ( ) const [inline]
```

References [GiNaC::structure< T, ComparisonPolicy >::obj](#).

6.164.4 Member Data Documentation

6.164.4.1 obj

```
template<class T , template< class > class ComparisonPolicy>
T GiNaC::structure< T, ComparisonPolicy >::obj [private]
```

Referenced by [GiNaC::structure< T, ComparisonPolicy >::get_struct\(\)](#), [GiNaC::structure< T, ComparisonPolicy >::get_struct\(\)](#), [GiNaC::structure< T, ComparisonPolicy >::is_equal_same_type\(\)](#), and [GiNaC::structure< T, ComparisonPolicy >::operator->\(\)](#).

The documentation for this class was generated from the following file:

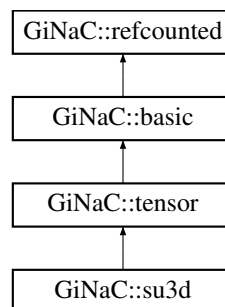
- [structure.h](#)

6.165 GiNaC::su3d Class Reference

This class represents the tensor of symmetric su(3) structure constants.

```
#include <color.h>
```

Inheritance diagram for GiNaC::su3d:



Public Member Functions

- [ex eval_indexed](#) (const [basic](#) &i) const override
Automatic symbolic evaluation of indexed symmetric structure constant.
- bool [contract_with](#) (exvector::iterator self, exvector::iterator other, [exvector](#) &v) const override
Contraction of an indexed symmetric structure constant with something else.

Public Member Functions inherited from [GiNaC::tensor](#)

- bool [replace_contr_index](#) (exvector::iterator self, exvector::iterator other) const
Replace dummy index in contracted-with object by the contracting object's second index (used internally for delta and metric tensor contractions).

Public Member Functions inherited from GiNaC::basic

- virtual `~basic ()`
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic (const basic &other)`
- `const basic & operator= (const basic &other)`
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate () const`
Create a clone of this object on the heap.
- virtual `ex eval () const`
Perform automatic non-interruptive term rewriting rules.
- virtual `ex evalf () const`
Evaluate object numerically.
- virtual `ex evalm () const`
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ () const`
Evaluate integrals, if result is known.
- virtual void `print (const print_context &c, unsigned level=0) const`
Output to stream.
- virtual void `dbgprint () const`
Little wrapper around print to be called within a debugger.
- virtual void `dbgprnttree () const`
Little wrapper around printtree to be called within a debugger.
- virtual unsigned `precedence () const`
Return relative operator precedence (for parenthezing output).
- virtual bool `info (unsigned inf) const`
Information about the object.
- virtual size_t `nops () const`
Number of operands/members.
- virtual `ex op (size_t i) const`
Return operand/member at position i.
- virtual `ex operator[] (const ex &index) const`
- virtual `ex operator[] (size_t i) const`
- virtual `ex & let_op (size_t i)`
Return modifiable operand/member at position i.
- virtual `ex & operator[] (const ex &index)`
- virtual `ex & operator[] (size_t i)`
- virtual bool `has (const ex &other, unsigned options=0) const`
Test for occurrence of a pattern.
- virtual bool `match (const ex &pattern, exmap &repls) const`
Check whether the expression matches a given pattern.
- virtual `ex subs (const exmap &m, unsigned options=0) const`
Substitute a set of objects by arbitrary expressions.
- virtual `ex map (map_function &f) const`
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void `accept (GiNaC::visitor &v) const`
- virtual bool `is_polynomial (const ex &var) const`
Check whether this is a polynomial in the given variables.
- virtual int `degree (const ex &s) const`
Return degree of highest power in object s.
- virtual int `ldegree (const ex &s) const`

- Return degree of lowest power in object s.*

 - virtual `ex coeff` (const `ex` &s, int n=1) const

Return coefficient of degree n in object s.
- virtual `ex expand` (unsigned options=0) const

Expand expression, i.e.
- virtual `ex collect` (const `ex` &s, bool distributed=false) const

Sort expanded expression in terms of powers of some object(s).
- virtual `ex series` (const `relational` &r, int order, unsigned options=0) const

Default implementation of `ex::series()`.
- virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const

Default implementation of `ex::normal()`.
- virtual `ex to_rational` (`exmap` &repl) const

Default implementation of `ex::to_rational()`.
- virtual `ex to_polynomial` (`exmap` &repl) const
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &xi) const

Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient` () const

Implementation `ex::max_coefficient()`.
- virtual `exvector get_free_indices` () const

Return a vector containing the free indices of an expression.
- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const

Add two indexed expressions.
- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const

Multiply an indexed expression with a scalar.
- virtual `return_type_t return_type_tinfo` () const
- virtual `ex conjugate` () const
- virtual `ex real_part` () const
- virtual `ex imag_part` () const
- template<class T >
 - void `print_dispatch` (const `print_context` &c, unsigned level) const

Like `print()`, but dispatch to the specified class.
- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const

Like `print()`, but dispatch to the specified class.
- virtual void `archive` (`archive_node` &n) const

Save (serialize) the object into archive node.
- virtual void `read_archive` (const `archive_node` &n, `lst` &syms)

Load (deserialize) the object from an archive node.
- `ex subs_one_level` (const `exmap` &m, unsigned options) const

Helper function for `subs()`.
- `ex diff` (const `symbol` &s, unsigned nth=1) const

Default interface of nth derivative `ex::diff(s, n)`.
- int `compare` (const `basic` &other) const

Compare objects syntactically to establish canonical ordering.
- bool `is_equal` (const `basic` &other) const

Test for syntactic equality.
- const `basic` & `hold` () const

Stop further evaluation.
- unsigned `gethash` () const
- const `basic` & `setflag` (unsigned f) const

Set some `status_flags`.
- const `basic` & `clearflag` (unsigned f) const

Clear some `status_flags`.

Public Member Functions inherited from GiNaC::refcounted

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Protected Member Functions

- unsigned `return_type` () const override
- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_latex` (const `print_latex` &c, unsigned level) const

Protected Member Functions inherited from GiNaC::tensor

- unsigned `return_type` () const override

Protected Member Functions inherited from GiNaC::basic

- `basic` ()
- virtual `ex eval_ncmul` (const `exvector` &v) const
- virtual bool `match_same_type` (const `basic` &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual `ex derivative` (const `symbol` &s) const
Default implementation of `ex::diff()`.
- virtual int `compare_same_type` (const `basic` &other) const
Returns order relation between two objects of same type.
- virtual bool `is_equal_same_type` (const `basic` &other) const
Returns true if two objects of same type are equal.
- virtual unsigned `calchash` () const
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- void `ensure_if_modifiable` () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void `do_print` (const `print_context` &c, unsigned level) const
Default output to stream.
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
Tree output to stream.
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
Python parsable output to stream.

Additional Inherited Members

Protected Attributes inherited from GiNaC::basic

- unsigned `flags`
of type `status_flags`
- unsigned `hashvalue`
hash value

6.165.1 Detailed Description

This class represents the tensor of symmetric $\text{su}(3)$ structure constants.

6.165.2 Member Function Documentation

6.165.2.1 `eval_indexed()`

```
ex GiNaC::su3d::eval_indexed (
    const basic & i ) const [override], [virtual]
```

Automatic symbolic evaluation of indexed symmetric structure constant.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex0](#), [GiNaC::_ex1_2](#), [GiNaC::_ex1_3](#), [GiNaC::_ex3](#), [GiNaC::_ex_1_2](#), [GiNaC::_ex_1_3](#), [GiNaC::_ex_6](#), [CMPINDICES](#), [GINAC_ASSERT](#), [GiNaC::basic::hold\(\)](#), [GiNaC::info_flags::nonnegint](#), [GiNaC::basic::nops\(\)](#), [GiNaC::basic::op\(\)](#), [GiNaC::sqrt\(\)](#), and [std::swap\(\)](#).

6.165.2.2 `contract_with()`

```
bool GiNaC::su3d::contract_with (
    exvector::iterator self,
    exvector::iterator other,
    exvector & v ) const [override], [virtual]
```

Contraction of an indexed symmetric structure constant with something else.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex1](#), [GiNaC::color_T\(\)](#), [GiNaC::delta_tensor\(\)](#), [GiNaC::find_free_and_dummy\(\)](#), [GiNaC::get_representation_label\(\)](#), [GINAC_ASSERT](#), [GiNaC::ex::op\(\)](#), and [GiNaC::permute_free_index_to_front\(\)](#).

6.165.2.3 `return_type()`

```
unsigned GiNaC::su3d::return_type ( ) const [inline], [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::return_types::commutative](#).

6.165.2.4 `do_print()`

```
void GiNaC::su3d::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

6.165.2.5 do_print_latex()

```
void GiNaC::su3d::do_print_latex (
    const print_latex & c,
    unsigned level ) const [protected]
```

The documentation for this class was generated from the following files:

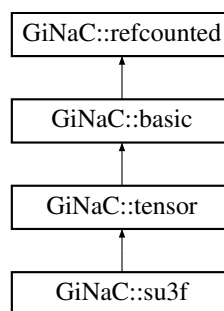
- [color.h](#)
- [color.cpp](#)

6.166 GiNaC::su3f Class Reference

This class represents the tensor of antisymmetric su(3) structure constants.

```
#include <color.h>
```

Inheritance diagram for GiNaC::su3f:



Public Member Functions

- [ex eval_indexed](#) (const [basic](#) &i) const override
Automatic symbolic evaluation of indexed antisymmetric structure constant.
- bool [contract_with](#) (exvector::iterator self, exvector::iterator other, [exvector](#) &v) const override
Contraction of an indexed antisymmetric structure constant with something else.

Public Member Functions inherited from [GiNaC::tensor](#)

- bool [replace_contr_index](#) (exvector::iterator self, exvector::iterator other) const
Replace dummy index in contracted-with object by the contracting object's second index (used internally for delta and metric tensor contractions).

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic ()`
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic (const basic &other)`
- `const basic & operator= (const basic &other)`
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate () const`
Create a clone of this object on the heap.
- virtual `ex eval () const`
Perform automatic non-interruptive term rewriting rules.
- virtual `ex evalf () const`
Evaluate object numerically.
- virtual `ex evalm () const`
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ () const`
Evaluate integrals, if result is known.
- virtual void `print (const print_context &c, unsigned level=0) const`
Output to stream.
- virtual void `dbgprint () const`
Little wrapper around print to be called within a debugger.
- virtual void `dbgprnttree () const`
Little wrapper around printtree to be called within a debugger.
- virtual unsigned `precedence () const`
Return relative operator precedence (for parenthezing output).
- virtual bool `info (unsigned inf) const`
Information about the object.
- virtual size_t `nops () const`
Number of operands/members.
- virtual `ex op (size_t i) const`
Return operand/member at position i.
- virtual `ex operator[] (const ex &index) const`
- virtual `ex operator[] (size_t i) const`
- virtual `ex & let_op (size_t i)`
Return modifiable operand/member at position i.
- virtual `ex & operator[] (const ex &index)`
- virtual `ex & operator[] (size_t i)`
- virtual bool `has (const ex &other, unsigned options=0) const`
Test for occurrence of a pattern.
- virtual bool `match (const ex &pattern, exmap &repls) const`
Check whether the expression matches a given pattern.
- virtual `ex subs (const exmap &m, unsigned options=0) const`
Substitute a set of objects by arbitrary expressions.
- virtual `ex map (map_function &f) const`
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void `accept (GiNaC::visitor &v) const`
- virtual bool `is_polynomial (const ex &var) const`
Check whether this is a polynomial in the given variables.
- virtual int `degree (const ex &s) const`
Return degree of highest power in object s.
- virtual int `ldegree (const ex &s) const`

- Return degree of lowest power in object s.*

 - virtual `ex coeff` (const `ex` &s, int n=1) const

Return coefficient of degree n in object s.

 - virtual `ex expand` (unsigned options=0) const

Expand expression, i.e.

 - virtual `ex collect` (const `ex` &s, bool distributed=false) const

Sort expanded expression in terms of powers of some object(s).

 - virtual `ex series` (const `relational` &r, int order, unsigned options=0) const

Default implementation of `ex::series()`.

 - virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const

Default implementation of `ex::normal()`.

 - virtual `ex to_rational` (`exmap` &repl) const

Default implementation of `ex::to_rational()`.

 - virtual `ex to_polynomial` (`exmap` &repl) const
 - virtual `numeric integer_content` () const
 - virtual `ex smod` (const `numeric` &xi) const

Apply symmetric modular homomorphism to an expanded multivariate polynomial.

 - virtual `numeric max_coefficient` () const

Implementation `ex::max_coefficient()`.

 - virtual `exvector get_free_indices` () const

Return a vector containing the free indices of an expression.

 - virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const

Add two indexed expressions.

 - virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const

Multiply an indexed expression with a scalar.

 - virtual `return_type_t return_type_tinfo` () const
 - virtual `ex conjugate` () const
 - virtual `ex real_part` () const
 - virtual `ex imag_part` () const- template<class T >
 - void `print_dispatch` (const `print_context` &c, unsigned level) const

Like `print()`, but dispatch to the specified class.

 - void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const

Like `print()`, but dispatch to the specified class.

 - virtual void `archive` (`archive_node` &n) const

Save (serialize) the object into archive node.

 - virtual void `read_archive` (const `archive_node` &n, `lst` &syms)

Load (deserialize) the object from an archive node.

 - `ex subs_one_level` (const `exmap` &m, unsigned options) const

Helper function for `subs()`.

 - `ex diff` (const `symbol` &s, unsigned nth=1) const

Default interface of nth derivative `ex::diff(s, n)`.

 - int `compare` (const `basic` &other) const

Compare objects syntactically to establish canonical ordering.

 - bool `is_equal` (const `basic` &other) const

Test for syntactic equality.

 - const `basic` & `hold` () const

Stop further evaluation.

 - unsigned `gethash` () const
 - const `basic` & `setflag` (unsigned f) const

Set some `status_flags`.

 - const `basic` & `clearflag` (unsigned f) const

Clear some `status_flags`.

Public Member Functions inherited from [GiNaC::refcounted](#)

- [refcounted](#) () noexcept
- unsigned int [add_reference](#) () noexcept
- unsigned int [remove_reference](#) () noexcept
- unsigned int [get_refcount](#) () const noexcept
- void [set_refcount](#) (unsigned int r) noexcept

Protected Member Functions

- unsigned [return_type](#) () const override
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
- void [do_print_latex](#) (const [print_latex](#) &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::tensor](#)

- unsigned [return_type](#) () const override

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual [ex derivative](#) (const [symbol](#) &s) const
Default implementation of [ex::diff\(\)](#).
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Additional Inherited Members

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.166.1 Detailed Description

This class represents the tensor of antisymmetric su(3) structure constants.

6.166.2 Member Function Documentation

6.166.2.1 eval_indexed()

```
ex GiNaC::su3f::eval_indexed (
    const basic & i ) const [override], [virtual]
```

Automatic symbolic evaluation of indexed antisymmetric structure constant.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex0](#), [GiNaC::_ex1_2](#), [GiNaC::_ex3](#), [GiNaC::_ex_1_2](#), [CMPINDICES](#), [GINAC_ASSERT](#), [GiNaC::basic::hold\(\)](#), [GiNaC::info_flags::nonnegint](#), [GiNaC::basic::nops\(\)](#), [GiNaC::basic::op\(\)](#), [GiNaC::sqrt\(\)](#), and [std::swap\(\)](#).

6.166.2.2 contract_with()

```
bool GiNaC::su3f::contract_with (
    exvector::iterator self,
    exvector::iterator other,
    exvector & v ) const [override], [virtual]
```

Contraction of an indexed antisymmetric structure constant with something else.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex1](#), [GiNaC::color_T\(\)](#), [GiNaC::delta_tensor\(\)](#), [GiNaC::get_representation_label\(\)](#), [GINAC_ASSERT](#), [GiNaC::l](#), [GiNaC::ex::op\(\)](#), and [GiNaC::permute_free_index_to_front\(\)](#).

6.166.2.3 return_type()

```
unsigned GiNaC::su3f::return_type ( ) const [inline], [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::return_types::commutative](#).

6.166.2.4 do_print()

```
void GiNaC::su3f::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

6.166.2.5 do_print_latex()

```
void GiNaC::su3f::do_print_latex (
    const print\_latex & c,
    unsigned level ) const [protected]
```

The documentation for this class was generated from the following files:

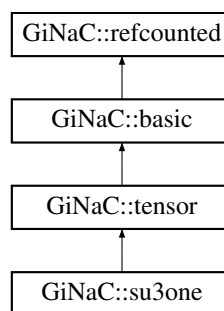
- [color.h](#)
- [color.cpp](#)

6.167 GiNaC::su3one Class Reference

This class represents the su(3) unity element.

```
#include <color.h>
```

Inheritance diagram for GiNaC::su3one:



Protected Member Functions

- void [do_print](#) (const [print_context](#) &c, unsigned level) const
- void [do_print_latex](#) (const [print_latex](#) &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::tensor](#)

- unsigned [return_type](#) () const override

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual [ex derivative](#) (const [symbol](#) &s) const
Default implementation of [ex::diff\(\)](#).
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Additional Inherited Members

Public Member Functions inherited from [GiNaC::tensor](#)

- bool [replace_contr_index](#) (exvector::iterator self, exvector::iterator other) const
Replace dummy index in contracted-with object by the contracting object's second index (used internally for delta and metric tensor contractions).

Public Member Functions inherited from [GiNaC::basic](#)

- virtual [~basic](#) ()
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- [basic](#) (const [basic](#) &other)
- const [basic](#) & [operator=](#) (const [basic](#) &other)
basic assignment operator: the other object might be of a derived class.
- virtual [basic](#) * [duplicate](#) () const
Create a clone of this object on the heap.
- virtual [ex eval](#) () const
Perform automatic non-interruptive term rewriting rules.
- virtual [ex evalf](#) () const
Evaluate object numerically.
- virtual [ex evalm](#) () const
Evaluate sums, products and integer powers of matrices.
- virtual [ex eval_integ](#) () const
Evaluate integrals, if result is known.
- virtual [ex eval_indexed](#) (const [basic](#) &i) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.

- virtual void `print` (const `print_context` &c, unsigned level=0) const
Output to stream.
- virtual void `dbgprint` () const
Little wrapper around print to be called within a debugger.
- virtual void `dbgprnttree` () const
Little wrapper around prnttree to be called within a debugger.
- virtual unsigned `precedence` () const
Return relative operator precedence (for parenthezing output).
- virtual bool `info` (unsigned inf) const
Information about the object.
- virtual size_t `nops` () const
Number of operands/members.
- virtual `ex op` (size_t i) const
Return operand/member at position i.
- virtual `ex operator[]` (const `ex` &index) const
- virtual `ex operator[]` (size_t i) const
- virtual `ex & let_op` (size_t i)
Return modifiable operand/member at position i.
- virtual `ex & operator[]` (const `ex` &index)
- virtual `ex & operator[]` (size_t i)
- virtual bool `has` (const `ex` &other, unsigned `options`=0) const
Test for occurrence of a pattern.
- virtual bool `match` (const `ex` &pattern, `exmap` &repls) const
Check whether the expression matches a given pattern.
- virtual `ex subs` (const `exmap` &m, unsigned `options`=0) const
Substitute a set of objects by arbitrary expressions.
- virtual `ex map` (`map_function` &f) const
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void `accept` (`GiNaC::visitor` &v) const
- virtual bool `is_polynomial` (const `ex` &var) const
Check whether this is a polynomial in the given variables.
- virtual int `degree` (const `ex` &s) const
Return degree of highest power in object s.
- virtual int `ldegree` (const `ex` &s) const
Return degree of lowest power in object s.
- virtual `ex coeff` (const `ex` &s, int n=1) const
Return coefficient of degree n in object s.
- virtual `ex expand` (unsigned `options`=0) const
Expand expression, i.e.
- virtual `ex collect` (const `ex` &s, bool distributed=false) const
Sort expanded expression in terms of powers of some object(s).
- virtual `ex series` (const `relational` &r, int `order`, unsigned `options`=0) const
Default implementation of `ex::series()`.
- virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const
Default implementation of `ex::normal()`.
- virtual `ex to_rational` (`exmap` &repl) const
Default implementation of `ex::to_rational()`.
- virtual `ex to_polynomial` (`exmap` &repl) const
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &xi) const
Apply symmetric modular homomorphism to an expanded multivariate polynomial.

- virtual [numeric max_coefficient](#) () const
Implementation [ex::max_coefficient\(\)](#).
- virtual [exvector get_free_indices](#) () const
Return a vector containing the free indices of an expression.
- virtual [ex add_indexed](#) (const [ex](#) &self, const [ex](#) &other) const
Add two indexed expressions.
- virtual [ex scalar_mul_indexed](#) (const [ex](#) &self, const [numeric](#) &other) const
Multiply an indexed expression with a scalar.
- virtual bool [contract_with](#) (exvector::iterator self, exvector::iterator other, [exvector](#) &v) const
Try to contract two indexed expressions that appear in the same product.
- virtual [return_type_t return_type_tinfo](#) () const
- virtual [ex conjugate](#) () const
- virtual [ex real_part](#) () const
- virtual [ex imag_part](#) () const
- template<class T >
void [print_dispatch](#) (const [print_context](#) &c, unsigned level) const
Like [print\(\)](#), but dispatch to the specified class.
- void [print_dispatch](#) (const [registered_class_info](#) &ri, const [print_context](#) &c, unsigned level) const
Like [print\(\)](#), but dispatch to the specified class.
- virtual void [archive](#) ([archive_node](#) &n) const
Save (serialize) the object into archive node.
- virtual void [read_archive](#) (const [archive_node](#) &n, [lst](#) &syms)
Load (deserialize) the object from an archive node.
- [ex subs_one_level](#) (const [exmap](#) &m, unsigned [options](#)) const
Helper function for [subs\(\)](#).
- [ex diff](#) (const [symbol](#) &s, unsigned nth=1) const
Default interface of nth derivative [ex::diff\(s, n\)](#).
- int [compare](#) (const [basic](#) &other) const
Compare objects syntactically to establish canonical ordering.
- bool [is_equal](#) (const [basic](#) &other) const
Test for syntactic equality.
- const [basic](#) & [hold](#) () const
Stop further evaluation.
- unsigned [gethash](#) () const
- const [basic](#) & [setflag](#) (unsigned f) const
Set some [status_flags](#).
- const [basic](#) & [clearflag](#) (unsigned f) const
Clear some [status_flags](#).

Public Member Functions inherited from [GiNaC::refcounted](#)

- [refcounted](#) () noexcept
- unsigned int [add_reference](#) () noexcept
- unsigned int [remove_reference](#) () noexcept
- unsigned int [get_refcount](#) () const noexcept
- void [set_refcount](#) (unsigned int r) noexcept

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.167.1 Detailed Description

This class represents the $\text{su}(3)$ unity element.

6.167.2 Member Function Documentation

6.167.2.1 `do_print()`

```
void GiNaC::su3one::do_print (
    const print\_context & c,
    unsigned level ) const [protected]
```

6.167.2.2 `do_print_latex()`

```
void GiNaC::su3one::do_print_latex (
    const print\_latex & c,
    unsigned level ) const [protected]
```

The documentation for this class was generated from the following file:

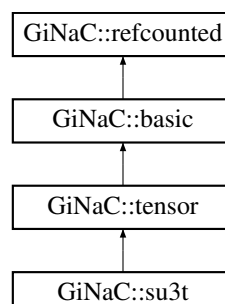
- [color.h](#)

6.168 GiNaC::su3t Class Reference

This class represents an $\text{su}(3)$ generator.

```
#include <color.h>
```

Inheritance diagram for `GiNaC::su3t`:



Public Member Functions

- bool [contract_with](#) (exvector::iterator self, exvector::iterator other, [exvector](#) &v) const override
Contraction of generator with something else.

Public Member Functions inherited from [GiNaC::tensor](#)

- bool [replace_contr_index](#) (exvector::iterator self, exvector::iterator other) const
Replace dummy index in contracted-with object by the contracting object's second index (used internally for delta and metric tensor contractions).

Public Member Functions inherited from [GiNaC::basic](#)

- virtual [~basic](#) ()
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- [basic](#) (const [basic](#) &other)
- const [basic](#) & [operator=](#) (const [basic](#) &other)
basic assignment operator: the other object might be of a derived class.
- virtual [basic](#) * [duplicate](#) () const
Create a clone of this object on the heap.
- virtual [ex eval](#) () const
Perform automatic non-interruptive term rewriting rules.
- virtual [ex evalf](#) () const
Evaluate object numerically.
- virtual [ex evalm](#) () const
Evaluate sums, products and integer powers of matrices.
- virtual [ex eval_integ](#) () const
Evaluate integrals, if result is known.
- virtual [ex eval_indexed](#) (const [basic](#) &i) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void [print](#) (const [print_context](#) &c, unsigned level=0) const
Output to stream.
- virtual void [dbgprint](#) () const
Little wrapper around print to be called within a debugger.
- virtual void [dbgprinttree](#) () const
Little wrapper around printtree to be called within a debugger.
- virtual unsigned [precedence](#) () const
Return relative operator precedence (for parenthesizing output).
- virtual bool [info](#) (unsigned inf) const
Information about the object.
- virtual size_t [nops](#) () const
Number of operands/members.
- virtual [ex op](#) (size_t i) const
Return operand/member at position i.
- virtual [ex operator\[\]](#) (const [ex](#) &index) const
- virtual [ex operator\[\]](#) (size_t i) const
- virtual [ex](#) & [let_op](#) (size_t i)
Return modifiable operand/member at position i.
- virtual [ex](#) & [operator\[\]](#) (const [ex](#) &index)
- virtual [ex](#) & [operator\[\]](#) (size_t i)

- virtual bool [has](#) (const [ex](#) &other, unsigned [options](#)=0) const
Test for occurrence of a pattern.
- virtual bool [match](#) (const [ex](#) &pattern, [exmap](#) &repls) const
Check whether the expression matches a given pattern.
- virtual [ex subs](#) (const [exmap](#) &m, unsigned [options](#)=0) const
Substitute a set of objects by arbitrary expressions.
- virtual [ex map](#) ([map_function](#) &f) const
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void [accept](#) ([GiNaC::visitor](#) &v) const
- virtual bool [is_polynomial](#) (const [ex](#) &var) const
Check whether this is a polynomial in the given variables.
- virtual int [degree](#) (const [ex](#) &s) const
Return degree of highest power in object s.
- virtual int [ldegree](#) (const [ex](#) &s) const
Return degree of lowest power in object s.
- virtual [ex coeff](#) (const [ex](#) &s, int [n](#)=1) const
Return coefficient of degree n in object s.
- virtual [ex expand](#) (unsigned [options](#)=0) const
Expand expression, i.e.
- virtual [ex collect](#) (const [ex](#) &s, bool distributed=false) const
Sort expanded expression in terms of powers of some object(s).
- virtual [ex series](#) (const [relational](#) &r, int [order](#), unsigned [options](#)=0) const
Default implementation of [ex::series\(\)](#).
- virtual [ex normal](#) ([exmap](#) &repl, [exmap](#) &rev_lookup, [lst](#) &modifier) const
Default implementation of [ex::normal\(\)](#).
- virtual [ex to_rational](#) ([exmap](#) &repl) const
Default implementation of [ex::to_rational\(\)](#).
- virtual [ex to_polynomial](#) ([exmap](#) &repl) const
- virtual [numeric integer_content](#) () const
- virtual [ex smod](#) (const [numeric](#) &xi) const
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual [numeric max_coefficient](#) () const
Implementation [ex::max_coefficient\(\)](#).
- virtual [exvector get_free_indices](#) () const
Return a vector containing the free indices of an expression.
- virtual [ex add_indexed](#) (const [ex](#) &self, const [ex](#) &other) const
Add two indexed expressions.
- virtual [ex scalar_mul_indexed](#) (const [ex](#) &self, const [numeric](#) &other) const
Multiply an indexed expression with a scalar.
- virtual [return_type_t return_type_tinfo](#) () const
- virtual [ex conjugate](#) () const
- virtual [ex real_part](#) () const
- virtual [ex imag_part](#) () const
- template<class T >
void [print_dispatch](#) (const [print_context](#) &c, unsigned level) const
Like [print\(\)](#), but dispatch to the specified class.
- void [print_dispatch](#) (const [registered_class_info](#) &ri, const [print_context](#) &c, unsigned level) const
Like [print\(\)](#), but dispatch to the specified class.
- virtual void [archive](#) ([archive_node](#) &n) const
Save (serialize) the object into archive node.
- virtual void [read_archive](#) (const [archive_node](#) &n, [lst](#) &syms)

- *Load (deserialize) the object from an archive node.*
- `ex subs_one_level` (const `exmap` &`m`, unsigned `options`) const
Helper function for `subs()`.
- `ex diff` (const `symbol` &`s`, unsigned `nth`=1) const
Default interface of `nth` derivative `ex::diff(s, n)`.
- `int compare` (const `basic` &`other`) const
Compare objects syntactically to establish canonical ordering.
- `bool is_equal` (const `basic` &`other`) const
Test for syntactic equality.
- `const basic & hold` () const
Stop further evaluation.
- `unsigned gethash` () const
- `const basic & setflag` (unsigned `f`) const
Set some `status_flags`.
- `const basic & clearflag` (unsigned `f`) const
Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- `unsigned int add_reference` () noexcept
- `unsigned int remove_reference` () noexcept
- `unsigned int get_refcount` () const noexcept
- `void set_refcount` (unsigned int `r`) noexcept

Protected Member Functions

- `void do_print` (const `print_context` &`c`, unsigned `level`) const
- `void do_print_latex` (const `print_latex` &`c`, unsigned `level`) const

Protected Member Functions inherited from `GiNaC::tensor`

- `unsigned return_type` () const override

Protected Member Functions inherited from `GiNaC::basic`

- `basic` ()
- `virtual ex eval_ncmul` (const `exvector` &`v`) const
- `virtual bool match_same_type` (const `basic` &`other`) const
Returns true if the attributes of two objects are similar enough for a match.
- `virtual ex derivative` (const `symbol` &`s`) const
Default implementation of `ex::diff()`.
- `virtual int compare_same_type` (const `basic` &`other`) const
Returns order relation between two objects of same type.
- `virtual bool is_equal_same_type` (const `basic` &`other`) const
Returns true if two objects of same type are equal.
- `virtual unsigned calchash` () const
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- `void ensure_if_modifiable` () const

Ensure the object may be modified without hurting others, throws if this is not the case.

- void `do_print` (const `print_context` &`c`, unsigned level) const

Default output to stream.

- void `do_print_tree` (const `print_tree` &`c`, unsigned level) const

Tree output to stream.

- void `do_print_python_repr` (const `print_python_repr` &`c`, unsigned level) const

Python parsable output to stream.

Additional Inherited Members

Protected Attributes inherited from `GiNaC::basic`

- unsigned `flags`
of type `status_flags`
- unsigned `hashvalue`
hash value

6.168.1 Detailed Description

This class represents an $su(3)$ generator.

6.168.2 Member Function Documentation

6.168.2.1 `contract_with()`

```
bool GiNaC::su3t::contract_with (
    exvector::iterator self,
    exvector::iterator other,
    exvector & v ) const [override], [virtual]
```

Contraction of generator with something else.

Reimplemented from `GiNaC::basic`.

References `GiNaC::_ex1`, `GiNaC::color_ONE()`, `GiNaC::color_trace()`, and `GINAC_ASSERT`.

6.168.2.2 `do_print()`

```
void GiNaC::su3t::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

6.168.2.3 do_print_latex()

```
void GiNaC::su3t::do_print_latex (
    const print\_latex & c,
    unsigned level ) const [protected]
```

The documentation for this class was generated from the following files:

- [color.h](#)
- [color.cpp](#)

6.169 GiNaC::subs_options Class Reference

Flags to control the behavior of [subs\(\)](#).

```
#include <flags.h>
```

Public Types

- enum {
[no_pattern](#) = 0x0001 , [subs_no_pattern](#) = 0x0001 , [algebraic](#) = 0x0002 , [subs_algebraic](#) = 0x0002 ,
[pattern_is_product](#) = 0x0004 , [pattern_is_not_product](#) = 0x0008 , [no_index_renaming](#) = 0x0010 ,
[really_subs_idx](#) = 0x0020 }

6.169.1 Detailed Description

Flags to control the behavior of [subs\(\)](#).

6.169.2 Member Enumeration Documentation

6.169.2.1 anonymous enum

```
anonymous enum
```

Enumerator

no_pattern	disable pattern matching
subs_no_pattern	
algebraic	enable algebraic substitutions
subs_algebraic	
pattern_is_product	used internally by expairseq::subschildren()
pattern_is_not_product	used internally by expairseq::subschildren()
no_index_renaming	
really_subs_idx	

The documentation for this class was generated from the following file:

- [flags.h](#)

6.170 GiNaC::sy_is_less Class Reference

Public Member Functions

- [sy_is_less](#) (exvector::iterator v_)
- bool [operator\(\)](#) (const [ex](#) &lh, const [ex](#) &rh) const

Private Attributes

- exvector::iterator [v](#)

6.170.1 Constructor & Destructor Documentation

6.170.1.1 sy_is_less()

```
GiNaC::sy_is_less::sy_is_less (
    exvector::iterator v_ ) [inline]
```

6.170.2 Member Function Documentation

6.170.2.1 operator>()

```
bool GiNaC::sy_is_less::operator() (
    const ex & lh,
    const ex & rh ) const [inline]
```

References [GINAC_ASSERT](#), and [v](#).

6.170.3 Member Data Documentation

6.170.3.1 v

```
exvector::iterator GiNaC::sy_is_less::v [private]
```

Referenced by [operator>\(\)](#).

The documentation for this class was generated from the following file:

- [symmetry.cpp](#)

6.171 GiNaC::sy_swap Class Reference

Public Member Functions

- [sy_swap](#) (exvector::iterator v_, bool &s)
- void [operator\(\)](#) (const [ex](#) &lh, const [ex](#) &rh)

Public Attributes

- bool & [swapped](#)

Private Attributes

- exvector::iterator [v](#)

6.171.1 Constructor & Destructor Documentation

6.171.1.1 sy_swap()

```
GiNaC::sy_swap::sy_swap (  
    exvector::iterator v_,  
    bool & s ) [inline]
```

6.171.2 Member Function Documentation

6.171.2.1 operator>()

```
void GiNaC::sy_swap::operator() (  
    const ex & lh,  
    const ex & rh ) [inline]
```

References [GINAC_ASSERT](#), [swapped](#), and [v](#).

6.171.3 Member Data Documentation

6.171.3.1 v

```
exvector::iterator GiNaC::sy_swap::v [private]
```

Referenced by [operator>\(\)](#).

6.171.3.2 swapped

```
bool& GiNaC::sy_swap::swapped
```

Referenced by [operator>\(\)](#).

The documentation for this class was generated from the following file:

- [symmetry.cpp](#)

6.172 GiNaC::sym_desc Struct Reference

This structure holds information about the highest and lowest degrees in which a symbol appears in two multivariate polynomials "a" and "b".

Public Member Functions

- [sym_desc](#) (const [ex](#) &s)
Initialize symbol, leave other variables uninitialized.
- bool [operator<](#) (const [sym_desc](#) &x) const
Comparison operator for sorting.

Public Attributes

- [ex](#) [sym](#)
Reference to symbol.
- int [deg_a](#)
Highest degree of symbol in polynomial "a".
- int [deg_b](#)
Highest degree of symbol in polynomial "b".
- int [ldeg_a](#)
Lowest degree of symbol in polynomial "a".
- int [ldeg_b](#)
Lowest degree of symbol in polynomial "b".
- int [max_deg](#)
Maximum of [deg_a](#) and [deg_b](#) (Used for sorting)
- size_t [max_lcnops](#)
Maximum number of terms of leading coefficient of symbol in both polynomials.

6.172.1 Detailed Description

This structure holds information about the highest and lowest degrees in which a symbol appears in two multivariate polynomials "a" and "b".

A vector of these structures with information about all symbols in two polynomials can be created with the function [get_symbol_stats\(\)](#).

See also

[get_symbol_stats](#)

6.172.2 Constructor & Destructor Documentation

6.172.2.1 sym_desc()

```
GiNaC::sym_desc::sym_desc (
    const ex & s ) [inline]
```

Initialize symbol, leave other variables uninitialized.

6.172.3 Member Function Documentation

6.172.3.1 operator<()

```
bool GiNaC::sym_desc::operator< (
    const sym\_desc & x ) const [inline]
```

Comparison operator for sorting.

References [max_deg](#), [max_lcnops](#), and [x](#).

6.172.4 Member Data Documentation

6.172.4.1 sym

```
ex GiNaC::sym_desc::sym
```

Reference to symbol.

6.172.4.2 deg_a

```
int GiNaC::sym_desc::deg_a
```

Highest degree of symbol in polynomial "a".

6.172.4.3 deg_b

```
int GiNaC::sym_desc::deg_b
```

Highest degree of symbol in polynomial "b".

6.172.4.4 ldeg_a

```
int GiNaC::sym_desc::ldeg_a
```

Lowest degree of symbol in polynomial "a".

6.172.4.5 ldeg_b

```
int GiNaC::sym_desc::ldeg_b
```

Lowest degree of symbol in polynomial "b".

6.172.4.6 max_deg

```
int GiNaC::sym_desc::max_deg
```

Maximum of deg_a and deg_b (Used for sorting)

Referenced by [operator<\(\)](#).

6.172.4.7 max_lcnops

```
size_t GiNaC::sym_desc::max_lcnops
```

Maximum number of terms of leading coefficient of symbol in both polynomials.

Referenced by [operator<\(\)](#).

The documentation for this struct was generated from the following file:

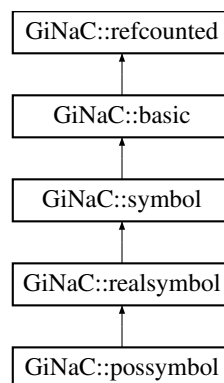
- [normal.cpp](#)

6.173 GiNaC::symbol Class Reference

Basic CAS symbol.

```
#include <symbol.h>
```

Inheritance diagram for GiNaC::symbol:



Public Member Functions

- [symbol](#) (const std::string &initname)
- [symbol](#) (const std::string &initname, const std::string &texname)
- bool [info](#) (unsigned inf) const override
Information about the object.
- [ex eval](#) () const override
Perform automatic non-interruptive term rewriting rules.
- [ex evalf](#) () const override
Evaluate object numerically.
- [ex series](#) (const [relational](#) &s, int [order](#), unsigned [options](#)=0) const override
Implementation of [ex::series\(\)](#) for symbols.
- [ex subs](#) (const [exmap](#) &m, unsigned [options](#)=0) const override
Substitute a set of objects by arbitrary expressions.
- [ex normal](#) ([exmap](#) &repl, [exmap](#) &rev_lookup, [lst](#) &modifier) const override
Implementation of [ex::normal\(\)](#) for symbols.
- [ex to_rational](#) ([exmap](#) &repl) const override
Implementation of [ex::to_rational\(\)](#) for symbols.
- [ex to_polynomial](#) ([exmap](#) &repl) const override
Implementation of [ex::to_polynomial\(\)](#) for symbols.
- [ex conjugate](#) () const override
- [ex real_part](#) () const override
- [ex imag_part](#) () const override
- bool [is_polynomial](#) (const [ex](#) &var) const override
Check whether this is a polynomial in the given variables.
- void [archive](#) ([archive_node](#) &n) const override
Save (a.k.a.
- void [read_archive](#) (const [archive_node](#) &n, [lst](#) &syms) override
Read (a.k.a.
- virtual unsigned [get_domain](#) () const
- void [set_name](#) (const std::string &n)
- void [set_TeX_name](#) (const std::string &n)
- std::string [get_name](#) () const
- std::string [get_TeX_name](#) () const

Public Member Functions inherited from [GiNaC::basic](#)

- virtual [~basic](#) ()
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- [basic](#) (const [basic](#) &other)
- const [basic](#) & [operator=](#) (const [basic](#) &other)
basic assignment operator: the other object might be of a derived class.
- virtual [basic](#) * [duplicate](#) () const
Create a clone of this object on the heap.
- virtual [ex evalm](#) () const
Evaluate sums, products and integer powers of matrices.
- virtual [ex eval_integ](#) () const
Evaluate integrals, if result is known.
- virtual [ex eval_indexed](#) (const [basic](#) &i) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void [print](#) (const [print_context](#) &c, unsigned level=0) const

- *Output to stream.*
virtual void [dbgprint](#) () const
- *Little wrapper around print to be called within a debugger.*
virtual void [dbgprinttree](#) () const
- *Little wrapper around printtree to be called within a debugger.*
virtual unsigned [precedence](#) () const
- *Return relative operator precedence (for parenthezing output).*
virtual size_t [nops](#) () const
- *Number of operands/members.*
virtual [ex op](#) (size_t i) const
- *Return operand/member at position i.*
virtual [ex operator\[\]](#) (const [ex](#) &index) const
- virtual [ex operator\[\]](#) (size_t i) const
- virtual [ex & let_op](#) (size_t i)
Return modifiable operand/member at position i.
- virtual [ex & operator\[\]](#) (const [ex](#) &index)
- virtual [ex & operator\[\]](#) (size_t i)
- virtual bool [has](#) (const [ex](#) &other, unsigned [options](#)=0) const
Test for occurrence of a pattern.
- virtual bool [match](#) (const [ex](#) &pattern, [exmap](#) &repls) const
Check whether the expression matches a given pattern.
- virtual [ex map](#) ([map_function](#) &f) const
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void [accept](#) ([GiNaC::visitor](#) &v) const
- virtual int [degree](#) (const [ex](#) &s) const
Return degree of highest power in object s.
- virtual int [ldegree](#) (const [ex](#) &s) const
Return degree of lowest power in object s.
- virtual [ex coeff](#) (const [ex](#) &s, int [n](#)=1) const
Return coefficient of degree n in object s.
- virtual [ex expand](#) (unsigned [options](#)=0) const
Expand expression, i.e.
- virtual [ex collect](#) (const [ex](#) &s, bool [distributed](#)=false) const
Sort expanded expression in terms of powers of some object(s).
- virtual [numeric integer_content](#) () const
- virtual [ex smod](#) (const [numeric](#) &xi) const
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual [numeric max_coefficient](#) () const
Implementation [ex::max_coefficient\(\)](#).
- virtual [exvector get_free_indices](#) () const
Return a vector containing the free indices of an expression.
- virtual [ex add_indexed](#) (const [ex](#) &self, const [ex](#) &other) const
Add two indexed expressions.
- virtual [ex scalar_mul_indexed](#) (const [ex](#) &self, const [numeric](#) &other) const
Multiply an indexed expression with a scalar.
- virtual bool [contract_with](#) (exvector::iterator self, exvector::iterator other, [exvector](#) &v) const
Try to contract two indexed expressions that appear in the same product.
- virtual unsigned [return_type](#) () const
- virtual [return_type_t return_type_tinfo](#) () const
- template<class T >
void [print_dispatch](#) (const [print_context](#) &c, unsigned level) const

- Like `print()`, but dispatch to the specified class.
- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const
- Like `print()`, but dispatch to the specified class.
- `ex subs_one_level` (const `exmap` &m, unsigned `options`) const
- Helper function for `subs()`.
- `ex diff` (const `symbol` &s, unsigned nth=1) const
- Default interface of nth derivative `ex::diff(s, n)`.
- int `compare` (const `basic` &other) const
- Compare objects syntactically to establish canonical ordering.
- bool `is_equal` (const `basic` &other) const
- Test for syntactic equality.
- const `basic` & `hold` () const
- Stop further evaluation.
- unsigned `gethash` () const
- const `basic` & `setflag` (unsigned f) const
- Set some `status_flags`.
- const `basic` & `clearflag` (unsigned f) const
- Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Protected Member Functions

- `ex derivative` (const `symbol` &s) const override
- Implementation of `ex::diff()` for single differentiation of a symbol.
- bool `is_equal_same_type` (const `basic` &other) const override
- Returns true if two objects of same type are equal.
- unsigned `calchash` () const override
- Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_latex` (const `print_latex` &c, unsigned level) const
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const

Protected Member Functions inherited from `GiNaC::basic`

- `basic` ()
- virtual `ex eval_ncmul` (const `exvector` &v) const
- virtual bool `match_same_type` (const `basic` &other) const
- Returns true if the attributes of two objects are similar enough for a match.
- virtual int `compare_same_type` (const `basic` &other) const
- Returns order relation between two objects of same type.
- void `ensure_if_modifiable` () const
- Ensure the object may be modified without hurting others, throws if this is not the case.

- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Protected Attributes

- unsigned [serial](#)
unique serial number for comparison
- std::string [name](#)
printname of this symbol
- std::string [TeX_name](#)
LaTeX name of this symbol.

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

Static Private Attributes

- static unsigned [next_serial](#) = 0

6.173.1 Detailed Description

Basic CAS symbol.

It has a name because it must know how to output itself.

6.173.2 Constructor & Destructor Documentation

6.173.2.1 [symbol\(\)](#) [1/2]

```
GiNaC::symbol::symbol (
    const std::string & initname ) [explicit]
```

References [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::expanded](#), and [GiNaC::basic::setflag\(\)](#).

6.173.2.2 [symbol\(\)](#) [2/2]

```
GiNaC::symbol::symbol (
    const std::string & initname,
    const std::string & texname )
```

References [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::expanded](#), and [GiNaC::basic::setflag\(\)](#).

6.173.3 Member Function Documentation

6.173.3.1 info()

```
bool GiNaC::symbol::info (
    unsigned inf ) const [override], [virtual]
```

Information about the object.

See also

class [info_flags](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::info_flags::cinteger_polynomial](#), [GiNaC::info_flags::crational_polynomial](#), [GiNaC::info_flags::expanded](#), [get_domain\(\)](#), [GiNaC::info_flags::has_indices](#), [GiNaC::info_flags::integer_polynomial](#), [GiNaC::info_flags::nonnegative](#), [GiNaC::info_flags::polynomial](#), [GiNaC::domain::positive](#), [GiNaC::info_flags::positive](#), [GiNaC::info_flags::rational_function](#), [GiNaC::info_flags::rational_polynomial](#), [GiNaC::domain::real](#), [GiNaC::info_flags::real](#), and [GiNaC::info_flags::symbol](#).

Referenced by [GiNaC::conjugate_expl_derivative\(\)](#), [GiNaC::imag_part_expl_derivative\(\)](#), and [GiNaC::real_part_expl_derivative\(\)](#).

6.173.3.2 eval()

```
ex GiNaC::symbol::eval ( ) const [inline], [override], [virtual]
```

Perform automatic non-interruptive term rewriting rules.

Reimplemented from [GiNaC::basic](#).

6.173.3.3 evalf()

```
ex GiNaC::symbol::evalf ( ) const [inline], [override], [virtual]
```

Evaluate object numerically.

Reimplemented from [GiNaC::basic](#).

6.173.3.4 series()

```
ex GiNaC::symbol::series (
    const relational & r,
    int order,
    unsigned options = 0 ) const [override], [virtual]
```

Implementation of [ex::series\(\)](#) for symbols.

See also

[ex::series](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex0](#), [GiNaC::_ex1](#), [GINAC_ASSERT](#), [is_equal_same_type\(\)](#), [GiNaC::ex::is_zero\(\)](#), [order](#), [r](#), and [GiNaC::ex::rhs\(\)](#).

Referenced by [GiNaC::EllipticE_series\(\)](#), and [GiNaC::EllipticK_series\(\)](#).

6.173.3.5 subs()

```
ex GiNaC::symbol::subs (
    const exmap & m,
    unsigned options = 0 ) const [inline], [override], [virtual]
```

Substitute a set of objects by arbitrary expressions.

The ex returned will already be evaluated.

Reimplemented from [GiNaC::basic](#).

References [m](#), [options](#), and [GiNaC::basic::subs_one_level\(\)](#).

6.173.3.6 normal()

```
ex GiNaC::symbol::normal (
    exmap & repl,
    exmap & rev_lookup,
    lst & modifier ) const [override], [virtual]
```

Implementation of [ex::normal\(\)](#) for symbols.

This returns the unmodified symbol.

See also

[ex::normal](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex1](#).

6.173.3.7 to_rational()

```
ex GiNaC::symbol::to_rational (
    exmap & repl ) const [override], [virtual]
```

Implementation of [ex::to_rational\(\)](#) for symbols.

This returns the unmodified symbol.

Reimplemented from [GiNaC::basic](#).

6.173.3.8 to_polynomial()

```
ex GiNaC::symbol::to_polynomial (
    exmap & repl ) const [override], [virtual]
```

Implementation of [ex::to_polynomial\(\)](#) for symbols.

This returns the unmodified symbol.

Reimplemented from [GiNaC::basic](#).

6.173.3.9 conjugate()

```
ex GiNaC::symbol::conjugate ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

6.173.3.10 real_part()

```
ex GiNaC::symbol::real_part ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

6.173.3.11 imag_part()

```
ex GiNaC::symbol::imag_part ( ) const [override], [virtual]
```

Reimplemented from [GiNaC::basic](#).

6.173.3.12 is_polynomial()

```
bool GiNaC::symbol::is_polynomial (
    const ex & var ) const [override], [virtual]
```

Check whether this is a polynomial in the given variables.

Reimplemented from [GiNaC::basic](#).

6.173.3.13 archive()

```
void GiNaC::symbol::archive (
    archive_node & n ) const [override], [virtual]
```

Save (a.k.a.

Archive the object.

serialize) object into archive.

Reimplemented from [GiNaC::basic](#).

References [n](#), [name](#), and [TeX_name](#).

6.173.3.14 read_archive()

```
void GiNaC::symbol::read_archive (
    const archive_node & n,
    lst & syms ) [override], [virtual]
```

Read (a.k.a.

Read object from [archive_node](#).

deserialize) object from archive.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::container< C >::append\(\)](#), [GiNaC::status_flags::dynallocated](#), [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::expanded](#), [n](#), [name](#), [next_serial](#), [serial](#), [GiNaC::basic::setflag\(\)](#), and [TeX_name](#).

6.173.3.15 derivative()

```
ex GiNaC::symbol::derivative (
    const symbol & s ) const [override], [protected], [virtual]
```

Implementation of [ex::diff\(\)](#) for single differentiation of a symbol.

It returns 1 or 0.

See also

[ex::diff](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex0](#), [GiNaC::_ex1](#), and [GiNaC::basic::compare_same_type\(\)](#).

6.173.3.16 is_equal_same_type()

```
bool GiNaC::symbol::is_equal_same_type (
    const basic & other ) const [override], [protected], [virtual]
```

Returns true if two objects of same type are equal.

Normally needs not be reimplemented as long as it wasn't overwritten by some parent class, since it just calls [compare_same_type\(\)](#). The reason why this function exists is that sometimes it is easier to determine equality than an order relation and then it can be overridden.

Reimplemented from [GiNaC::basic](#).

References [GINAC_ASSERT](#), and [serial](#).

Referenced by [series\(\)](#).

6.173.3.17 calchash()

```
unsigned GiNaC::symbol::calchash ( ) const [override], [protected], [virtual]
```

Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.

The method inherited from class `basic` computes a hash value based on the type and hash values of possible members. For this reason it is well suited for container classes but atomic classes should override this implementation because otherwise they would all end up with the same hashvalue.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::golden_ratio_hash\(\)](#), [GiNaC::status_flags::hash_calculated](#), [GiNaC::basic::hashvalue](#), [GiNaC::make_hash_seed\(\)](#), [serial](#), and [GiNaC::basic::setflag\(\)](#).

6.173.3.18 get_domain()

```
virtual unsigned GiNaC::symbol::get_domain ( ) const [inline], [virtual]
```

Reimplemented in [GiNaC::realsymbol](#), and [GiNaC::possymbol](#).

References [GiNaC::domain::complex](#).

Referenced by [do_print_tree\(\)](#), and [info\(\)](#).

6.173.3.19 set_name()

```
void GiNaC::symbol::set_name (
    const std::string & n ) [inline]
```

References [n](#), and [name](#).

6.173.3.20 set_TeX_name()

```
void GiNaC::symbol::set_TeX_name (
    const std::string & n ) [inline]
```

References [n](#), and [TeX_name](#).

6.173.3.21 get_name()

```
std::string GiNaC::symbol::get_name ( ) const
```

References [name](#), and [serial](#).

Referenced by [do_print\(\)](#), and [get_TeX_name\(\)](#).

6.173.3.22 `get_TeX_name()`

```
std::string GiNaC::symbol::get_TeX_name ( ) const
```

References [GiNaC::get_default_TeX_name\(\)](#), [get_name\(\)](#), and [TeX_name](#).

6.173.3.23 `do_print()`

```
void GiNaC::symbol::do_print (
    const print\_context & c,
    unsigned level ) const [protected]
```

References [c](#), and [get_name\(\)](#).

6.173.3.24 `do_print_latex()`

```
void GiNaC::symbol::do_print_latex (
    const print\_latex & c,
    unsigned level ) const [protected]
```

References [c](#), [GiNaC::get_default_TeX_name\(\)](#), [name](#), [serial](#), and [TeX_name](#).

6.173.3.25 `do_print_tree()`

```
void GiNaC::symbol::do_print_tree (
    const print\_tree & c,
    unsigned level ) const [protected]
```

References [c](#), [GiNaC::basic::flags](#), [get_domain\(\)](#), [GiNaC::basic::hashvalue](#), [name](#), and [serial](#).

6.173.3.26 `do_print_python_repr()`

```
void GiNaC::symbol::do_print_python_repr (
    const print\_python\_repr & c,
    unsigned level ) const [protected]
```

References [c](#), [name](#), [serial](#), and [TeX_name](#).

6.173.4 Member Data Documentation

6.173.4.1 `serial`

```
unsigned GiNaC::symbol::serial [protected]
```

unique serial number for comparison

Referenced by [calchash\(\)](#), [do_print_latex\(\)](#), [do_print_python_repr\(\)](#), [do_print_tree\(\)](#), [get_name\(\)](#), [is_equal_same_type\(\)](#), and [read_archive\(\)](#).

6.173.4.2 name

```
std::string GiNaC::symbol::name [mutable], [protected]
```

printname of this symbol

Referenced by [archive\(\)](#), [do_print_latex\(\)](#), [do_print_python_repr\(\)](#), [do_print_tree\(\)](#), [get_name\(\)](#), [read_archive\(\)](#), and [set_name\(\)](#).

6.173.4.3 TeX_name

```
std::string GiNaC::symbol::TeX_name [protected]
```

LaTeX name of this symbol.

Referenced by [archive\(\)](#), [do_print_latex\(\)](#), [do_print_python_repr\(\)](#), [get_TeX_name\(\)](#), [read_archive\(\)](#), and [set_TeX_name\(\)](#).

6.173.4.4 next_serial

```
unsigned GiNaC::symbol::next_serial = 0 [static], [private]
```

Referenced by [read_archive\(\)](#).

The documentation for this class was generated from the following files:

- [symbol.h](#)
- [normal.cpp](#)
- [pseries.cpp](#)
- [symbol.cpp](#)

6.174 GiNaC::symbolset Class Reference

Public Member Functions

- [symbolset](#) (const [ex](#) &e)
- bool [has](#) (const [ex](#) &e) const

Private Member Functions

- void [insert_symbols](#) (const [ex](#) &e)

Private Attributes

- [exset](#) s

6.174.1 Constructor & Destructor Documentation

6.174.1.1 `symbolset()`

```
GiNaC::symbolset::symbolset (
    const ex & e ) [inline], [explicit]
```

References [insert_symbols\(\)](#).

6.174.2 Member Function Documentation

6.174.2.1 `insert_symbols()`

```
void GiNaC::symbolset::insert_symbols (
    const ex & e ) [inline], [private]
```

References [insert_symbols\(\)](#), and [s](#).

Referenced by [insert_symbols\(\)](#), and [symbolset\(\)](#).

6.174.2.2 `has()`

```
bool GiNaC::symbolset::has (
    const ex & e ) const [inline]
```

References [s](#).

Referenced by [GiNaC::!solve\(\)](#).

6.174.3 Member Data Documentation

6.174.3.1 `s`

```
exset GiNaC::symbolset::s [private]
```

Referenced by [has\(\)](#), and [insert_symbols\(\)](#).

The documentation for this class was generated from the following file:

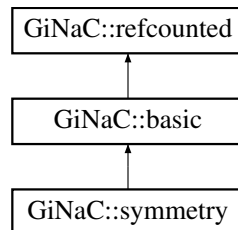
- [inifcns.cpp](#)

6.175 GiNaC::symmetry Class Reference

This class describes the symmetry of a group of indices.

```
#include <symmetry.h>
```

Inheritance diagram for GiNaC::symmetry:



Public Types

- enum [symmetry_type](#) { [none](#) , [symmetric](#) , [antisymmetric](#) , [cyclic](#) }
- Type of symmetry.*

Public Member Functions

- [symmetry](#) (unsigned i)
Create leaf node that represents one index.
- [symmetry](#) ([symmetry_type](#) t, const [symmetry](#) &c1, const [symmetry](#) &c2)
Create node with two children.
- void [archive](#) ([archive_node](#) &n) const override
Save (a.k.a.
- void [read_archive](#) (const [archive_node](#) &n, [lst](#) &syms) override
Read (a.k.a.
- [symmetry_type](#) [get_type](#) () const
Get symmetry type.
- void [set_type](#) ([symmetry_type](#) t)
Set symmetry type.
- [symmetry](#) & [add](#) (const [symmetry](#) &c)
Add child node, check index sets for consistency.
- void [validate](#) (unsigned n)
Verify that all indices of this node are in the range [0..n-1].
- bool [has_symmetry](#) () const
Check whether this node actually represents any kind of symmetry.
- bool [has_nonsymmetric](#) () const
Check whether this node involves anything non symmetric.
- bool [has_cyclic](#) () const
Check whether this node involves a cyclic symmetry.

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic ()`
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic (const basic &other)`
- `const basic & operator= (const basic &other)`
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate () const`
Create a clone of this object on the heap.
- virtual `ex eval () const`
Perform automatic non-interruptive term rewriting rules.
- virtual `ex evalf () const`
Evaluate object numerically.
- virtual `ex evalm () const`
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ () const`
Evaluate integrals, if result is known.
- virtual `ex eval_indexed (const basic &i) const`
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual `void print (const print_context &c, unsigned level=0) const`
Output to stream.
- virtual `void dbgprint () const`
Little wrapper around print to be called within a debugger.
- virtual `void dbgprinttree () const`
Little wrapper around printtree to be called within a debugger.
- virtual `unsigned precedence () const`
Return relative operator precedence (for parenthezing output).
- virtual `bool info (unsigned inf) const`
Information about the object.
- virtual `size_t nops () const`
Number of operands/members.
- virtual `ex op (size_t i) const`
Return operand/member at position i.
- virtual `ex operator[] (const ex &index) const`
- virtual `ex operator[] (size_t i) const`
- virtual `ex & let_op (size_t i)`
Return modifiable operand/member at position i.
- virtual `ex & operator[] (const ex &index)`
- virtual `ex & operator[] (size_t i)`
- virtual `bool has (const ex &other, unsigned options=0) const`
Test for occurrence of a pattern.
- virtual `bool match (const ex &pattern, exmap &repls) const`
Check whether the expression matches a given pattern.
- virtual `ex subs (const exmap &m, unsigned options=0) const`
Substitute a set of objects by arbitrary expressions.
- virtual `ex map (map_function &f) const`
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual `void accept (GiNaC::visitor &v) const`
- virtual `bool is_polynomial (const ex &var) const`
Check whether this is a polynomial in the given variables.
- virtual `int degree (const ex &s) const`

- Return degree of highest power in object s.*

 - virtual int `ldegree` (const `ex` &s) const
- Return degree of lowest power in object s.*

 - virtual `ex coeff` (const `ex` &s, int n=1) const
- Return coefficient of degree n in object s.*

 - virtual `ex expand` (unsigned `options`=0) const
- Expand expression, i.e.*

 - virtual `ex collect` (const `ex` &s, bool distributed=false) const
- Sort expanded expression in terms of powers of some object(s).*

 - virtual `ex series` (const `relational` &r, int `order`, unsigned `options`=0) const
- Default implementation of `ex::series()`.*

 - virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const
- Default implementation of `ex::normal()`.*

 - virtual `ex to_rational` (`exmap` &repl) const
- Default implementation of `ex::to_rational()`.*

 - virtual `ex to_polynomial` (`exmap` &repl) const
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &xi) const
- Apply symmetric modular homomorphism to an expanded multivariate polynomial.*

 - virtual `numeric max_coefficient` () const
- Implementation `ex::max_coefficient()`.*

 - virtual `exvector get_free_indices` () const
- Return a vector containing the free indices of an expression.*

 - virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
- Add two indexed expressions.*

 - virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
- Multiply an indexed expression with a scalar.*

 - virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const
- Try to contract two indexed expressions that appear in the same product.*

 - virtual unsigned `return_type` () const
- virtual `return_type_t return_type_info` () const
- virtual `ex conjugate` () const
- virtual `ex real_part` () const
- virtual `ex imag_part` () const
- template<class T >
 - void `print_dispatch` (const `print_context` &c, unsigned level) const
- Like `print()`, but dispatch to the specified class.*

 - void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const
- Like `print()`, but dispatch to the specified class.*

 - `ex subs_one_level` (const `exmap` &m, unsigned `options`) const
- Helper function for `subs()`.*

 - `ex diff` (const `symbol` &s, unsigned nth=1) const
- Default interface of nth derivative `ex::diff(s, n)`.*

 - int `compare` (const `basic` &other) const
- Compare objects syntactically to establish canonical ordering.*

 - bool `is_equal` (const `basic` &other) const
- Test for syntactic equality.*

 - const `basic` & `hold` () const
- Stop further evaluation.*

 - unsigned `gethash` () const
- const `basic` & `setflag` (unsigned f) const
- Set some `status_flags`.*

 - const `basic` & `clearflag` (unsigned f) const
- Clear some `status_flags`.*

Public Member Functions inherited from [GiNaC::refcounted](#)

- [refcounted](#) () noexcept
- unsigned int [add_reference](#) () noexcept
- unsigned int [remove_reference](#) () noexcept
- unsigned int [get_refcount](#) () const noexcept
- void [set_refcount](#) (unsigned int r) noexcept

Protected Member Functions

- unsigned [calchash](#) () const override
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual [ex derivative](#) (const [symbol](#) &s) const
Default implementation of [ex::diff\(\)](#).
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Private Attributes

- [symmetry_type](#) type
Type of symmetry described by this node.
- std::set< unsigned > [indices](#)
Sorted union set of all indices handled by this node.
- [exvector](#) children
Vector of child nodes.

Friends

- class [sy_is_less](#)
- class [sy_swap](#)
- int [canonicalize](#) (exvector::iterator v, const [symmetry](#) &symm)
Canonicalize the order of elements of an expression vector, according to the symmetry properties defined in a symmetry tree.

Additional Inherited Members

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.175.1 Detailed Description

This class describes the symmetry of a group of indices.

These objects can be grouped into a tree to form complex mixed symmetries.

6.175.2 Member Enumeration Documentation

6.175.2.1 symmetry_type

```
enum GiNaC::symmetry::symmetry\_type
```

Type of symmetry.

Enumerator

none	no symmetry properties
symmetric	totally symmetric
antisymmetric	totally antisymmetric
cyclic	cyclic symmetry

6.175.3 Constructor & Destructor Documentation

6.175.3.1 symmetry() [1/2]

```
GiNaC::symmetry::symmetry (  
    unsigned i )
```

Create leaf node that represents one index.

References [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::expanded](#), [indices](#), and [GiNaC::basic::setflag\(\)](#).

6.175.3.2 symmetry() [2/2]

```
GiNaC::symmetry::symmetry (  
    symmetry\_type t,  
    const symmetry & c1,  
    const symmetry & c2 )
```

Create node with two children.

References [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::expanded](#), and [GiNaC::basic::setflag\(\)](#).

6.175.4 Member Function Documentation

6.175.4.1 archive()

```
void GiNaC::symmetry::archive (
    archive_node & n ) const [override], [virtual]
```

Save (a.k.a.

Archive the object.

serialize) object into archive.

Reimplemented from [GiNaC::basic](#).

References [children](#), [indices](#), [n](#), and [type](#).

6.175.4.2 read_archive()

```
void GiNaC::symmetry::read_archive (
    const archive_node & n,
    lst & syms ) [override], [virtual]
```

Read (a.k.a.

Construct object from [archive_node](#).

deserialize) object from archive.

Reimplemented from [GiNaC::basic](#).

References [indices](#), [n](#), and [type](#).

6.175.4.3 calchash()

```
unsigned GiNaC::symmetry::calchash ( ) const [override], [protected], [virtual]
```

Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.

The method inherited from class basic computes a hash value based on the type and hash values of possible members. For this reason it is well suited for container classes but atomic classes should override this implementation because otherwise they would all end up with the same hashvalue.

Reimplemented from [GiNaC::basic](#).

References [children](#), [GiNaC::status_flags::evaluated](#), [GiNaC::basic::flags](#), [GiNaC::status_flags::hash_calculated](#), [GiNaC::basic::hashvalue](#), [indices](#), [GiNaC::make_hash_seed\(\)](#), [none](#), [GiNaC::rotate_left\(\)](#), [GiNaC::basic::setflag\(\)](#), and [type](#).

6.175.4.4 get_type()

```
symmetry_type GiNaC::symmetry::get_type ( ) const [inline]
```

Get symmetry type.

References [type](#).

6.175.4.5 set_type()

```
void GiNaC::symmetry::set_type (
    symmetry_type t ) [inline]
```

Set symmetry type.

References [type](#).

Referenced by [GiNaC::sy_anti\(\)](#), [GiNaC::sy_cycl\(\)](#), and [GiNaC::sy_symm\(\)](#).

6.175.4.6 add()

```
symmetry & GiNaC::symmetry::add (
    const symmetry & c )
```

Add child node, check index sets for consistency.

References [c](#), [children](#), [GINAC_ASSERT](#), [indices](#), [none](#), and [type](#).

Referenced by [GiNaC::sy_anti\(\)](#), [GiNaC::sy_anti\(\)](#), [GiNaC::sy_cycl\(\)](#), [GiNaC::sy_cycl\(\)](#), [GiNaC::sy_none\(\)](#), [GiNaC::sy_none\(\)](#), [GiNaC::sy_symm\(\)](#), and [GiNaC::sy_symm\(\)](#).

6.175.4.7 validate()

```
void GiNaC::symmetry::validate (
    unsigned n )
```

Verify that all indices of this node are in the range [0..n-1].

This function throws an exception if the verification fails. If the top node has a type != none and no children, add all indices in the range [0..n-1] as children.

References [indices](#), [n](#), [none](#), and [type](#).

6.175.4.8 has_symmetry()

```
bool GiNaC::symmetry::has_symmetry ( ) const [inline]
```

Check whether this node actually represents any kind of symmetry.

References [children](#), [none](#), and [type](#).

6.175.4.9 has_nonsymmetric()

```
bool GiNaC::symmetry::has_nonsymmetric ( ) const
```

Check whether this node involves anything non symmetric.

References [antisymmetric](#), [children](#), [cyclic](#), and [type](#).

6.175.4.10 has_cyclic()

```
bool GiNaC::symmetry::has_cyclic ( ) const
```

Check whether this node involves a cyclic symmetry.

References [children](#), [cyclic](#), and [type](#).

6.175.4.11 do_print()

```
void GiNaC::symmetry::do_print (
    const print\_context & c,
    unsigned level ) const [protected]
```

References [antisymmetric](#), [c](#), [children](#), [cyclic](#), [indices](#), [none](#), [symmetric](#), and [type](#).

6.175.4.12 do_print_tree()

```
void GiNaC::symmetry::do_print_tree (
    const print\_tree & c,
    unsigned level ) const [protected]
```

References [antisymmetric](#), [c](#), [children](#), [cyclic](#), [GiNaC::basic::flags](#), [GiNaC::basic::hashvalue](#), [indices](#), [none](#), [symmetric](#), and [type](#).

6.175.5 Friends And Related Symbol Documentation**6.175.5.1 sy_is_less**

```
friend class sy\_is\_less [friend]
```

6.175.5.2 sy_swap

```
friend class sy\_swap [friend]
```

6.175.5.3 canonicalize

```
int canonicalize (
    exvector::iterator v,
    const symmetry & symm ) [friend]
```

Canonicalize the order of elements of an expression vector, according to the symmetry properties defined in a symmetry tree.

Parameters

<i>v</i>	Start of expression vector
<i>symm</i>	Root node of symmetry tree

Returns

the overall sign introduced by the reordering (+1, -1 or 0) or `numeric_limits<int>::max()` if nothing changed

6.175.6 Member Data Documentation

6.175.6.1 type

```
symmetry_type GiNaC::symmetry::type [private]
```

Type of symmetry described by this node.

Referenced by [add\(\)](#), [archive\(\)](#), [calchash\(\)](#), [do_print\(\)](#), [do_print_tree\(\)](#), [get_type\(\)](#), [has_cyclic\(\)](#), [has_nonsymmetric\(\)](#), [has_symmetry\(\)](#), [read_archive\(\)](#), [set_type\(\)](#), and [validate\(\)](#).

6.175.6.2 indices

```
std::set<unsigned> GiNaC::symmetry::indices [private]
```

Sorted union set of all indices handled by this node.

Referenced by [add\(\)](#), [archive\(\)](#), [calchash\(\)](#), [do_print\(\)](#), [do_print_tree\(\)](#), [read_archive\(\)](#), [symmetry\(\)](#), and [validate\(\)](#).

6.175.6.3 children

```
exvector GiNaC::symmetry::children [private]
```

Vector of child nodes.

Referenced by [add\(\)](#), [archive\(\)](#), [calchash\(\)](#), [do_print\(\)](#), [do_print_tree\(\)](#), [has_cyclic\(\)](#), [has_nonsymmetric\(\)](#), and [has_symmetry\(\)](#).

The documentation for this class was generated from the following files:

- [symmetry.h](#)
- [symmetry.cpp](#)

6.176 GiNaC::symminfo Class Reference

This structure stores the individual symmetrized terms obtained during the simplification of sums.

Public Member Functions

- [symminfo](#) ()
- [symminfo](#) (const [ex](#) &symmterm_, const [ex](#) &orig_, size_t num_)

Public Attributes

- [ex symmterm](#)
symmetrized term
- [ex coeff](#)
coefficient of symmetrized term
- [ex orig](#)
original term
- [size_t num](#)
how many symmetrized terms resulted from the original term

6.176.1 Detailed Description

This structure stores the individual symmetrized terms obtained during the simplification of sums.

6.176.2 Constructor & Destructor Documentation

6.176.2.1 [symminfo\(\)](#) [1/2]

```
GiNaC::symminfo::symminfo ( ) [inline]
```

6.176.2.2 [symminfo\(\)](#) [2/2]

```
GiNaC::symminfo::symminfo (
    const ex & symmterm_,
    const ex & orig_,
    size_t num_ ) [inline]
```

References [coeff](#), [GiNaC::ex::nops\(\)](#), [GiNaC::ex::op\(\)](#), and [symmterm](#).

6.176.3 Member Data Documentation

6.176.3.1 [symmterm](#)

```
ex GiNaC::symminfo::symmterm
```

symmetrized term

Referenced by [GiNaC::symminfo_is_less_by_symmterm::operator\(\)\(\)](#), and [symminfo\(\)](#).

6.176.3.2 coeff

`ex GiNaC::symminfo::coeff`

coefficient of symmetrized term

Referenced by [symminfo\(\)](#).

6.176.3.3 orig

`ex GiNaC::symminfo::orig`

original term

Referenced by [GiNaC::symminfo_is_less_by_orig::operator\(\)](#).

6.176.3.4 num

`size_t GiNaC::symminfo::num`

how many symmetrized terms resulted from the original term

The documentation for this class was generated from the following file:

- [indexed.cpp](#)

6.177 GiNaC::symminfo_is_less_by_orig Class Reference

Public Member Functions

- `bool operator()` (const [symminfo](#) &si1, const [symminfo](#) &si2) const

6.177.1 Member Function Documentation

6.177.1.1 operator()

```
bool GiNaC::symminfo_is_less_by_orig::operator() (
    const symminfo & si1,
    const symminfo & si2 ) const [inline]
```

References [GiNaC::ex::compare\(\)](#), and [GiNaC::symminfo::orig](#).

The documentation for this class was generated from the following file:

- [indexed.cpp](#)

6.178 GiNaC::symminfo_is_less_by_symmterm Class Reference

Public Member Functions

- bool [operator\(\)](#) (const [symminfo](#) &si1, const [symminfo](#) &si2) const

6.178.1 Member Function Documentation

6.178.1.1 operator>()

```
bool GiNaC::symminfo_is_less_by_symmterm::operator() (
    const symminfo & si1,
    const symminfo & si2 ) const [inline]
```

References [GiNaC::ex::compare\(\)](#), and [GiNaC::symminfo::symmterm](#).

The documentation for this class was generated from the following file:

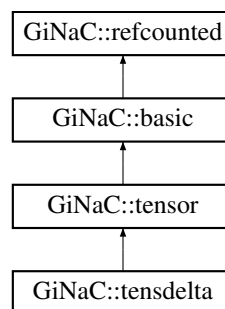
- [indexed.cpp](#)

6.179 GiNaC::tensdelta Class Reference

This class represents the delta tensor.

```
#include <tensor.h>
```

Inheritance diagram for GiNaC::tensdelta:



Public Member Functions

- bool [info](#) (unsigned inf) const override
Information about the object.
- [ex eval_indexed](#) (const [basic](#) &i) const override
Automatic symbolic evaluation of an indexed delta tensor.
- bool [contract_with](#) (exvector::iterator self, exvector::iterator other, [exvector](#) &v) const override
Contraction of an indexed delta tensor with something else.

Public Member Functions inherited from GiNaC::tensor

- bool [replace_contr_index](#) (exvector::iterator self, exvector::iterator other) const
Replace dummy index in contracted-with object by the contracting object's second index (used internally for delta and metric tensor contractions).

Public Member Functions inherited from GiNaC::basic

- virtual [~basic](#) ()
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- [basic](#) (const [basic](#) &other)
- const [basic](#) & [operator=](#) (const [basic](#) &other)
basic assignment operator: the other object might be of a derived class.
- virtual [basic](#) * [duplicate](#) () const
Create a clone of this object on the heap.
- virtual [ex eval](#) () const
Perform automatic non-interruptive term rewriting rules.
- virtual [ex evalf](#) () const
Evaluate object numerically.
- virtual [ex evalm](#) () const
Evaluate sums, products and integer powers of matrices.
- virtual [ex eval_integ](#) () const
Evaluate integrals, if result is known.
- virtual void [print](#) (const [print_context](#) &c, unsigned level=0) const
Output to stream.
- virtual void [dbgprint](#) () const
Little wrapper around print to be called within a debugger.
- virtual void [dbgprinttree](#) () const
Little wrapper around printtree to be called within a debugger.
- virtual unsigned [precedence](#) () const
Return relative operator precedence (for parenthezing output).
- virtual size_t [nops](#) () const
Number of operands/members.
- virtual [ex op](#) (size_t i) const
Return operand/member at position i.
- virtual [ex operator\[\]](#) (const [ex](#) &index) const
- virtual [ex operator\[\]](#) (size_t i) const
- virtual [ex](#) & [let_op](#) (size_t i)
Return modifiable operand/member at position i.
- virtual [ex](#) & [operator\[\]](#) (const [ex](#) &index)
- virtual [ex](#) & [operator\[\]](#) (size_t i)
- virtual bool [has](#) (const [ex](#) &other, unsigned [options](#)=0) const
Test for occurrence of a pattern.
- virtual bool [match](#) (const [ex](#) &pattern, [exmap](#) &repls) const
Check whether the expression matches a given pattern.
- virtual [ex subs](#) (const [exmap](#) &m, unsigned [options](#)=0) const
Substitute a set of objects by arbitrary expressions.
- virtual [ex map](#) ([map_function](#) &f) const
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void [accept](#) ([GiNaC::visitor](#) &v) const
- virtual bool [is_polynomial](#) (const [ex](#) &var) const

- Check whether this is a polynomial in the given variables.*

 - virtual int [degree](#) (const [ex](#) &s) const

Return degree of highest power in object s.
- virtual int [ldegree](#) (const [ex](#) &s) const

Return degree of lowest power in object s.
- virtual [ex](#) [coeff](#) (const [ex](#) &s, int n=1) const

Return coefficient of degree n in object s.
- virtual [ex](#) [expand](#) (unsigned [options](#)=0) const

Expand expression, i.e.
- virtual [ex](#) [collect](#) (const [ex](#) &s, bool distributed=false) const

Sort expanded expression in terms of powers of some object(s).
- virtual [ex](#) [series](#) (const [relational](#) &r, int [order](#), unsigned [options](#)=0) const

Default implementation of [ex::series\(\)](#).
- virtual [ex](#) [normal](#) ([exmap](#) &repl, [exmap](#) &rev_lookup, [lst](#) &modifier) const

Default implementation of [ex::normal\(\)](#).
- virtual [ex](#) [to_rational](#) ([exmap](#) &repl) const

Default implementation of [ex::to_rational\(\)](#).
- virtual [ex](#) [to_polynomial](#) ([exmap](#) &repl) const
- virtual [numeric](#) [integer_content](#) () const
- virtual [ex](#) [smod](#) (const [numeric](#) &xi) const

Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual [numeric](#) [max_coefficient](#) () const

Implementation [ex::max_coefficient\(\)](#).
- virtual [exvector](#) [get_free_indices](#) () const

Return a vector containing the free indices of an expression.
- virtual [ex](#) [add_indexed](#) (const [ex](#) &self, const [ex](#) &other) const

Add two indexed expressions.
- virtual [ex](#) [scalar_mul_indexed](#) (const [ex](#) &self, const [numeric](#) &other) const

Multiply an indexed expression with a scalar.
- virtual [return_type_t](#) [return_type_tinfo](#) () const
- virtual [ex](#) [conjugate](#) () const
- virtual [ex](#) [real_part](#) () const
- virtual [ex](#) [imag_part](#) () const
- template<class T >
 - void [print_dispatch](#) (const [print_context](#) &c, unsigned level) const

Like [print\(\)](#), but dispatch to the specified class.
- void [print_dispatch](#) (const [registered_class_info](#) &ri, const [print_context](#) &c, unsigned level) const

Like [print\(\)](#), but dispatch to the specified class.
- virtual void [archive](#) ([archive_node](#) &n) const

Save (serialize) the object into archive node.
- virtual void [read_archive](#) (const [archive_node](#) &n, [lst](#) &syms)

Load (deserialize) the object from an archive node.
- [ex](#) [subs_one_level](#) (const [exmap](#) &m, unsigned [options](#)) const

Helper function for [subs\(\)](#).
- [ex](#) [diff](#) (const [symbol](#) &s, unsigned nth=1) const

Default interface of nth derivative [ex::diff\(s, n\)](#).
- int [compare](#) (const [basic](#) &other) const

Compare objects syntactically to establish canonical ordering.
- bool [is_equal](#) (const [basic](#) &other) const

Test for syntactic equality.
- const [basic](#) & [hold](#) () const

Stop further evaluation.

- unsigned [gethash](#) () const
- const [basic](#) & [setflag](#) (unsigned f) const

Set some [status_flags](#).

- const [basic](#) & [clearflag](#) (unsigned f) const

Clear some [status_flags](#).

Public Member Functions inherited from [GiNaC::refcounted](#)

- [refcounted](#) () noexcept
- unsigned int [add_reference](#) () noexcept
- unsigned int [remove_reference](#) () noexcept
- unsigned int [get_refcount](#) () const noexcept
- void [set_refcount](#) (unsigned int r) noexcept

Protected Member Functions

- unsigned [return_type](#) () const override
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
- void [do_print_latex](#) (const [print_latex](#) &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::tensor](#)

- unsigned [return_type](#) () const override

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual [ex derivative](#) (const [symbol](#) &s) const
Default implementation of [ex::diff\(\)](#).
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Additional Inherited Members

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.179.1 Detailed Description

This class represents the delta tensor.

If indexed, it must have exactly two indices of the same type.

6.179.2 Member Function Documentation

6.179.2.1 [info\(\)](#)

```
bool GiNaC::tensdelta::info (
    unsigned inf ) const [override], [virtual]
```

Information about the object.

See also

class [info_flags](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::info_flags::real](#).

6.179.2.2 [eval_indexed\(\)](#)

```
ex GiNaC::tensdelta::eval_indexed (
    const basic & i ) const [override], [virtual]
```

Automatic symbolic evaluation of an indexed delta tensor.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex0](#), [GiNaC::_ex1](#), [GiNaC::idx::get_dim\(\)](#), [GiNaC::idx::get_value\(\)](#), [GINAC_ASSERT](#), [GiNaC::info_flags::integer](#), [GiNaC::is_dummy_pair\(\)](#), [GiNaC::ex::is_equal\(\)](#), [m](#), [GiNaC::idx::minimal_dim\(\)](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::idx::replace_dim\(\)](#), and [GiNaC::ex::subs\(\)](#).

6.179.2.3 contract_with()

```
bool GiNaC::tensdelta::contract_with (
    exvector::iterator self,
    exvector::iterator other,
    exvector & v ) const [override], [virtual]
```

Contraction of an indexed delta tensor with something else.

Reimplemented from [GiNaC::basic](#).

References [GINAC_ASSERT](#), and [GiNaC::tensor::replace_contr_index\(\)](#).

6.179.2.4 return_type()

```
unsigned GiNaC::tensdelta::return_type ( ) const [inline], [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::return_types::commutative](#).

6.179.2.5 do_print()

```
void GiNaC::tensdelta::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

6.179.2.6 do_print_latex()

```
void GiNaC::tensdelta::do_print_latex (
    const print_latex & c,
    unsigned level ) const [protected]
```

The documentation for this class was generated from the following files:

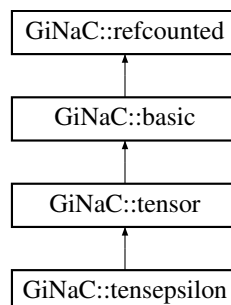
- [tensor.h](#)
- [tensor.cpp](#)

6.180 GiNaC::tensepsilon Class Reference

This class represents the totally antisymmetric epsilon tensor.

```
#include <tensor.h>
```

Inheritance diagram for GiNaC::tensepsilon:



Public Member Functions

- `tensepsilon` (bool `minkowski`, bool `pos_sig`)
- bool `info` (unsigned int) const override
Information about the object.
- `ex eval_indexed` (const `basic` &i) const override
Automatic symbolic evaluation of an indexed epsilon tensor.
- bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const override
Contraction of epsilon tensor with something else.
- void `archive` (`archive_node` &n) const override
Save (a.k.a.
- void `read_archive` (const `archive_node` &n, `lst` &syms) override
Read (a.k.a.

Public Member Functions inherited from `GiNaC::tensor`

- bool `replace_contr_index` (exvector::iterator self, exvector::iterator other) const
Replace dummy index in contracted-with object by the contracting object's second index (used internally for delta and metric tensor contractions.

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic` ()
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic` (const `basic` &other)
- const `basic` & `operator=` (const `basic` &other)
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate` () const
Create a clone of this object on the heap.
- virtual `ex eval` () const
Perform automatic non-interruptive term rewriting rules.
- virtual `ex evalf` () const
Evaluate object numerically.
- virtual `ex evalm` () const
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ` () const
Evaluate integrals, if result is known.
- virtual void `print` (const `print_context` &c, unsigned level=0) const
Output to stream.
- virtual void `dbgprint` () const
Little wrapper around print to be called within a debugger.
- virtual void `dbgprinttree` () const
Little wrapper around printtree to be called within a debugger.
- virtual unsigned `precedence` () const
Return relative operator precedence (for parenthezing output).
- virtual size_t `nops` () const
Number of operands/members.
- virtual `ex op` (size_t i) const
Return operand/member at position i.
- virtual `ex operator[]` (const `ex` &index) const

- virtual `ex operator[]` (size_t i) const
- virtual `ex & let_op` (size_t i)
Return modifiable operand/member at position i.
- virtual `ex & operator[]` (const `ex` &index)
- virtual `ex & operator[]` (size_t i)
- virtual bool `has` (const `ex` &other, unsigned `options`=0) const
Test for occurrence of a pattern.
- virtual bool `match` (const `ex` &pattern, `exmap` &repls) const
Check whether the expression matches a given pattern.
- virtual `ex subs` (const `exmap` &m, unsigned `options`=0) const
Substitute a set of objects by arbitrary expressions.
- virtual `ex map` (`map_function` &f) const
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void `accept` (GiNaC::visitor &v) const
- virtual bool `is_polynomial` (const `ex` &var) const
Check whether this is a polynomial in the given variables.
- virtual int `degree` (const `ex` &s) const
Return degree of highest power in object s.
- virtual int `ldegree` (const `ex` &s) const
Return degree of lowest power in object s.
- virtual `ex coeff` (const `ex` &s, int n=1) const
Return coefficient of degree n in object s.
- virtual `ex expand` (unsigned `options`=0) const
Expand expression, i.e.
- virtual `ex collect` (const `ex` &s, bool distributed=false) const
Sort expanded expression in terms of powers of some object(s).
- virtual `ex series` (const `relational` &r, int `order`, unsigned `options`=0) const
Default implementation of `ex::series()`.
- virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const
Default implementation of `ex::normal()`.
- virtual `ex to_rational` (`exmap` &repl) const
Default implementation of `ex::to_rational()`.
- virtual `ex to_polynomial` (`exmap` &repl) const
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &xi) const
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient` () const
Implementation `ex::max_coefficient()`.
- virtual `exvector get_free_indices` () const
Return a vector containing the free indices of an expression.
- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
Add two indexed expressions.
- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
Multiply an indexed expression with a scalar.
- virtual `return_type_t return_type_info` () const
- virtual `ex conjugate` () const
- virtual `ex real_part` () const
- virtual `ex imag_part` () const
- template<class T >
void `print_dispatch` (const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.

- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- `ex subs_one_level` (const `exmap` &m, unsigned `options`) const
Helper function for `subs()`.
- `ex diff` (const `symbol` &s, unsigned nth=1) const
Default interface of nth derivative `ex::diff(s, n)`.
- int `compare` (const `basic` &other) const
Compare objects syntactically to establish canonical ordering.
- bool `is_equal` (const `basic` &other) const
Test for syntactic equality.
- const `basic` & `hold` () const
Stop further evaluation.
- unsigned `gethash` () const
- const `basic` & `setflag` (unsigned f) const
Set some `status_flags`.
- const `basic` & `clearflag` (unsigned f) const
Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Protected Member Functions

- unsigned `return_type` () const override
- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_latex` (const `print_latex` &c, unsigned level) const

Protected Member Functions inherited from `GiNaC::tensor`

- unsigned `return_type` () const override

Protected Member Functions inherited from `GiNaC::basic`

- `basic` ()
- virtual `ex eval_ncmul` (const `exvector` &v) const
- virtual bool `match_same_type` (const `basic` &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual `ex derivative` (const `symbol` &s) const
Default implementation of `ex::diff()`.
- virtual int `compare_same_type` (const `basic` &other) const
Returns order relation between two objects of same type.
- virtual bool `is_equal_same_type` (const `basic` &other) const
Returns true if two objects of same type are equal.
- virtual unsigned `calchash` () const

Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.

- void `ensure_if_modifiable` () const

Ensure the object may be modified without hurting others, throws if this is not the case.

- void `do_print` (const `print_context` &c, unsigned level) const

Default output to stream.

- void `do_print_tree` (const `print_tree` &c, unsigned level) const

Tree output to stream.

- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const

Python parsable output to stream.

Private Attributes

- bool `minkowski`

If true, tensor is in Minkowski-type space.

- bool `pos_sig`

If true, the metric is assumed to be $\text{diag}(-1, 1, 1 \dots)$.

Additional Inherited Members

Protected Attributes inherited from `GiNaC::basic`

- unsigned `flags`

of type `status_flags`

- unsigned `hashvalue`

hash value

6.180.1 Detailed Description

This class represents the totally antisymmetric epsilon tensor.

If indexed, all indices must be of the same type and their number must be equal to the dimension of the index space.

6.180.2 Constructor & Destructor Documentation

6.180.2.1 `tensepsilon()`

```
GiNaC::tensepsilon::tensepsilon (
    bool minkowski,
    bool pos_sig )
```

6.180.3 Member Function Documentation

6.180.3.1 info()

```
bool GiNaC::tensepsilon::info (
    unsigned inf ) const [override], [virtual]
```

Information about the object.

See also

class [info_flags](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::info_flags::real](#).

6.180.3.2 eval_indexed()

```
ex GiNaC::tensepsilon::eval_indexed (
    const basic & i ) const [override], [virtual]
```

Automatic symbolic evaluation of an indexed epsilon tensor.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex0](#), [GINAC_ASSERT](#), [GiNaC::basic::hold\(\)](#), [GiNaC::is_zero\(\)](#), [minkowski](#), [GiNaC::info_flags::nonnegint](#), [GiNaC::basic::nops\(\)](#), [GiNaC::basic::op\(\)](#), [GiNaC::permutation_sign\(\)](#), [pos_sig](#), and [x](#).

6.180.3.3 contract_with()

```
bool GiNaC::tensepsilon::contract_with (
    exvector::iterator self,
    exvector::iterator other,
    exvector & v ) const [override], [virtual]
```

Contraction of epsilon tensor with something else.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::_ex1](#), [GiNaC::delta_tensor\(\)](#), [GiNaC::matrix::determinant\(\)](#), [GINAC_ASSERT](#), [GiNaC::lorentz_g\(\)](#), [GiNaC::metric_tensor\(\)](#), [minkowski](#), [pos_sig](#), and [GiNaC::ex::simplify_indexed\(\)](#).

6.180.3.4 archive()

```
void GiNaC::tensepsilon::archive (
    archive\_node & n ) const [override], [virtual]
```

Save (a.k.a.

serialize) object into archive.

Reimplemented from [GiNaC::basic](#).

References [minkowski](#), [n](#), and [pos_sig](#).

6.180.3.5 read_archive()

```
void GiNaC::tensepsilon::read_archive (
    const archive\_node & n,
    lst & syms ) [override], [virtual]
```

Read (a.k.a.

deserialize) object from archive.

Reimplemented from [GiNaC::basic](#).

References [minkowski](#), [n](#), and [pos_sig](#).

6.180.3.6 return_type()

```
unsigned GiNaC::tensepsilon::return_type ( ) const [inline], [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::return_types::commutative](#).

6.180.3.7 do_print()

```
void GiNaC::tensepsilon::do_print (
    const print\_context & c,
    unsigned level ) const [protected]
```

6.180.3.8 do_print_latex()

```
void GiNaC::tensepsilon::do_print_latex (
    const print\_latex & c,
    unsigned level ) const [protected]
```

6.180.4 Member Data Documentation

6.180.4.1 minkowski

```
bool GiNaC::tensepsilon::minkowski [private]
```

If true, tensor is in Minkowski-type space.

Otherwise it is in a Euclidean space.

Referenced by [archive\(\)](#), [contract_with\(\)](#), [eval_indexed\(\)](#), and [read_archive\(\)](#).

6.180.4.2 pos_sig

```
bool GiNaC::tensepsilon::pos_sig [private]
```

If true, the metric is assumed to be $\text{diag}(-1, 1, 1, \dots)$.

Otherwise it is $\text{diag}(1, -1, -1, \dots)$. This is only relevant if `minkowski = true`.

Referenced by [archive\(\)](#), [contract_with\(\)](#), [eval_indexed\(\)](#), and [read_archive\(\)](#).

The documentation for this class was generated from the following files:

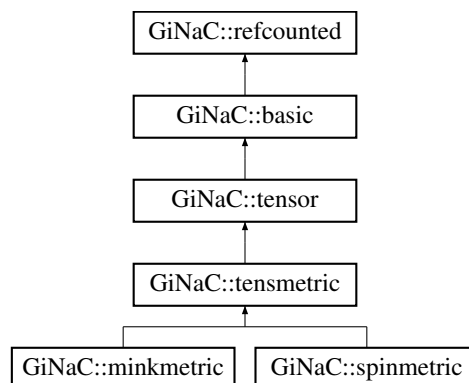
- [tensor.h](#)
- [tensor.cpp](#)

6.181 GiNaC::tensmetric Class Reference

This class represents a general metric tensor which can be used to raise/lower indices.

```
#include <tensor.h>
```

Inheritance diagram for `GiNaC::tensmetric`:



Public Member Functions

- bool [info](#) (unsigned inf) const override
Information about the object.
- `ex` [eval_indexed](#) (const [basic](#) &i) const override
Automatic symbolic evaluation of an indexed metric tensor.
- bool [contract_with](#) (exvector::iterator self, exvector::iterator other, [exvector](#) &v) const override
Contraction of an indexed metric tensor with something else.

Public Member Functions inherited from [GiNaC::tensor](#)

- bool [replace_contr_index](#) (exvector::iterator self, exvector::iterator other) const
Replace dummy index in contracted-with object by the contracting object's second index (used internally for delta and metric tensor contractions).

Public Member Functions inherited from **GiNaC::basic**

- virtual `~basic ()`
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic (const basic &other)`
- `const basic & operator= (const basic &other)`
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate () const`
Create a clone of this object on the heap.
- virtual `ex eval () const`
Perform automatic non-interruptive term rewriting rules.
- virtual `ex evalf () const`
Evaluate object numerically.
- virtual `ex evalm () const`
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ () const`
Evaluate integrals, if result is known.
- virtual void `print (const print_context &c, unsigned level=0) const`
Output to stream.
- virtual void `dbgprint () const`
Little wrapper around print to be called within a debugger.
- virtual void `dbgprnttree () const`
Little wrapper around printtree to be called within a debugger.
- virtual unsigned `precedence () const`
Return relative operator precedence (for parenthezing output).
- virtual size_t `nops () const`
Number of operands/members.
- virtual `ex op (size_t i) const`
Return operand/member at position i.
- virtual `ex operator[] (const ex &index) const`
- virtual `ex operator[] (size_t i) const`
- virtual `ex & let_op (size_t i)`
Return modifiable operand/member at position i.
- virtual `ex & operator[] (const ex &index)`
- virtual `ex & operator[] (size_t i)`
- virtual bool `has (const ex &other, unsigned options=0) const`
Test for occurrence of a pattern.
- virtual bool `match (const ex &pattern, exmap &repls) const`
Check whether the expression matches a given pattern.
- virtual `ex subs (const exmap &m, unsigned options=0) const`
Substitute a set of objects by arbitrary expressions.
- virtual `ex map (map_function &f) const`
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void `accept (GiNaC::visitor &v) const`
- virtual bool `is_polynomial (const ex &var) const`
Check whether this is a polynomial in the given variables.
- virtual int `degree (const ex &s) const`
Return degree of highest power in object s.
- virtual int `ldegree (const ex &s) const`
Return degree of lowest power in object s.
- virtual `ex coeff (const ex &s, int n=1) const`

- Return coefficient of degree n in object s .*

 - virtual `ex expand` (unsigned `options=0`) const

Expand expression, i.e.
 - virtual `ex collect` (const `ex` & s , bool `distributed=false`) const

Sort expanded expression in terms of powers of some object(s).
 - virtual `ex series` (const `relational` & r , int `order`, unsigned `options=0`) const

Default implementation of `ex::series()`.
 - virtual `ex normal` (`exmap` & $repl$, `exmap` & rev_lookup , `lst` & $modifier$) const

Default implementation of `ex::normal()`.
 - virtual `ex to_rational` (`exmap` & $repl$) const

Default implementation of `ex::to_rational()`.
 - virtual `ex to_polynomial` (`exmap` & $repl$) const
 - virtual `numeric integer_content` () const
 - virtual `ex smod` (const `numeric` & xi) const
- Apply symmetric modular homomorphism to an expanded multivariate polynomial.*
- virtual `numeric max_coefficient` () const
- Implementation `ex::max_coefficient()`.*
- virtual `exvector get_free_indices` () const
- Return a vector containing the free indices of an expression.*
- virtual `ex add_indexed` (const `ex` & $self$, const `ex` & $other$) const
- Add two indexed expressions.*
- virtual `ex scalar_mul_indexed` (const `ex` & $self$, const `numeric` & $other$) const
- Multiply an indexed expression with a scalar.*
- virtual `return_type_t return_type_info` () const
 - virtual `ex conjugate` () const
 - virtual `ex real_part` () const
 - virtual `ex imag_part` () const
- template<class T >
 - void `print_dispatch` (const `print_context` & c , unsigned `level`) const
- Like `print()`, but dispatch to the specified class.*
- void `print_dispatch` (const `registered_class_info` & ri , const `print_context` & c , unsigned `level`) const
- Like `print()`, but dispatch to the specified class.*
- virtual void `archive` (`archive_node` & n) const
- Save (serialize) the object into archive node.*
- virtual void `read_archive` (const `archive_node` & n , `lst` & $syms$)
- Load (deserialize) the object from an archive node.*
- `ex subs_one_level` (const `exmap` & m , unsigned `options`) const
- Helper function for `subs()`.*
- `ex diff` (const `symbol` & s , unsigned `nth=1`) const
- Default interface of n th derivative `ex::diff(s, n)`.*
- int `compare` (const `basic` & $other$) const
- Compare objects syntactically to establish canonical ordering.*
- bool `is_equal` (const `basic` & $other$) const
- Test for syntactic equality.*
- const `basic` & `hold` () const
- Stop further evaluation.*
- unsigned `gethash` () const
 - const `basic` & `setflag` (unsigned `f`) const
- Set some `status_flags`.*
- const `basic` & `clearflag` (unsigned `f`) const
- Clear some `status_flags`.*

Public Member Functions inherited from GiNaC::refcounted

- [refcounted](#) () noexcept
- unsigned int [add_reference](#) () noexcept
- unsigned int [remove_reference](#) () noexcept
- unsigned int [get_refcount](#) () const noexcept
- void [set_refcount](#) (unsigned int r) noexcept

Protected Member Functions

- unsigned [return_type](#) () const override
- void [do_print](#) (const [print_context](#) &c, unsigned level) const

Protected Member Functions inherited from GiNaC::tensor

- unsigned [return_type](#) () const override

Protected Member Functions inherited from GiNaC::basic

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual [ex derivative](#) (const [symbol](#) &s) const
Default implementation of [ex::diff\(\)](#).
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Additional Inherited Members

Protected Attributes inherited from GiNaC::basic

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.181.1 Detailed Description

This class represents a general metric tensor which can be used to raise/lower indices.

If indexed, it must have exactly two indices of the same type which must be of class `varidx` or a subclass.

6.181.2 Member Function Documentation

6.181.2.1 `info()`

```
bool GiNaC::tensmetric::info (
    unsigned inf ) const [override], [virtual]
```

Information about the object.

See also

class [info_flags](#)

Reimplemented from [GiNaC::basic](#).

References [GiNaC::info_flags::real](#).

6.181.2.2 `eval_indexed()`

```
ex GiNaC::tensmetric::eval_indexed (
    const basic & i ) const [override], [virtual]
```

Automatic symbolic evaluation of an indexed metric tensor.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::delta_tensor\(\)](#), [GiNaC::idx::get_dim\(\)](#), [GINAC_ASSERT](#), [GiNaC::basic::hold\(\)](#), [GiNaC::varidx::is_covariant\(\)](#), [GiNaC::ex::is_equal\(\)](#), [m](#), [GiNaC::idx::minimal_dim\(\)](#), [GiNaC::subs_options::no_pattern](#), [GiNaC::basic::nops\(\)](#), [GiNaC::basic::op\(\)](#), [GiNaC::idx::replace_dim\(\)](#), and [GiNaC::basic::subs\(\)](#).

6.181.2.3 `contract_with()`

```
bool GiNaC::tensmetric::contract_with (
    exvector::iterator self,
    exvector::iterator other,
    exvector & v ) const [override], [virtual]
```

Contraction of an indexed metric tensor with something else.

Reimplemented from [GiNaC::basic](#).

References [GINAC_ASSERT](#), and [GiNaC::tensor::replace_contr_index\(\)](#).

6.181.2.4 return_type()

```
unsigned GiNaC::tensmetric::return_type ( ) const [inline], [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::return_types::commutative](#).

6.181.2.5 do_print()

```
void GiNaC::tensmetric::do_print (
    const print\_context & c,
    unsigned level ) const [protected]
```

The documentation for this class was generated from the following files:

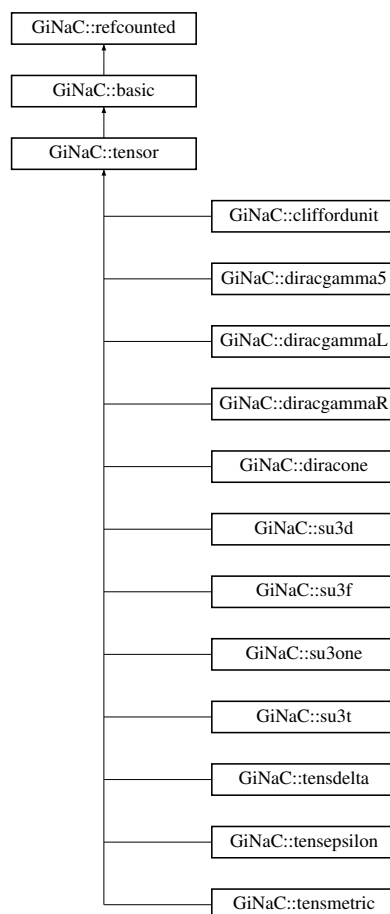
- [tensor.h](#)
- [tensor.cpp](#)

6.182 GiNaC::tensor Class Reference

This class holds one of [GiNaC](#)'s predefined special tensors such as the delta and the metric tensors.

```
#include <tensor.h>
```

Inheritance diagram for GiNaC::tensor:



Public Member Functions

- bool `replace_contr_index` (exvector::iterator self, exvector::iterator other) const
Replace dummy index in contracted-with object by the contracting object's second index (used internally for delta and metric tensor contractions).

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic` ()
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic` (const `basic` &other)
- const `basic` & `operator=` (const `basic` &other)
basic assignment operator: the other object might be of a derived class.
- virtual `basic` * `duplicate` () const
Create a clone of this object on the heap.
- virtual `ex eval` () const
Perform automatic non-interruptive term rewriting rules.
- virtual `ex evalf` () const
Evaluate object numerically.
- virtual `ex evalm` () const
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ` () const
Evaluate integrals, if result is known.
- virtual `ex eval_indexed` (const `basic` &i) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void `print` (const `print_context` &c, unsigned level=0) const
Output to stream.
- virtual void `dbgprint` () const
Little wrapper around print to be called within a debugger.
- virtual void `dbgprinttree` () const
Little wrapper around printtree to be called within a debugger.
- virtual unsigned `precedence` () const
Return relative operator precedence (for parenthezing output).
- virtual bool `info` (unsigned inf) const
Information about the object.
- virtual size_t `nops` () const
Number of operands/members.
- virtual `ex op` (size_t i) const
Return operand/member at position i.
- virtual `ex operator[]` (const `ex` &index) const
- virtual `ex operator[]` (size_t i) const
- virtual `ex` & `let_op` (size_t i)
Return modifiable operand/member at position i.
- virtual `ex` & `operator[]` (const `ex` &index)
- virtual `ex` & `operator[]` (size_t i)
- virtual bool `has` (const `ex` &other, unsigned `options`=0) const
Test for occurrence of a pattern.
- virtual bool `match` (const `ex` &pattern, `exmap` &repls) const
Check whether the expression matches a given pattern.
- virtual `ex subs` (const `exmap` &m, unsigned `options`=0) const
Substitute a set of objects by arbitrary expressions.

- virtual `ex map` (`map_function` &f) const
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual void `accept` (`GiNaC::visitor` &v) const
- virtual bool `is_polynomial` (const `ex` &var) const
Check whether this is a polynomial in the given variables.
- virtual int `degree` (const `ex` &s) const
Return degree of highest power in object s.
- virtual int `ldegree` (const `ex` &s) const
Return degree of lowest power in object s.
- virtual `ex coeff` (const `ex` &s, int n=1) const
Return coefficient of degree n in object s.
- virtual `ex expand` (unsigned `options`=0) const
Expand expression, i.e.
- virtual `ex collect` (const `ex` &s, bool distributed=false) const
Sort expanded expression in terms of powers of some object(s).
- virtual `ex series` (const `relational` &r, int `order`, unsigned `options`=0) const
Default implementation of `ex::series()`.
- virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const
Default implementation of `ex::normal()`.
- virtual `ex to_rational` (`exmap` &repl) const
Default implementation of `ex::to_rational()`.
- virtual `ex to_polynomial` (`exmap` &repl) const
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &xi) const
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual `numeric max_coefficient` () const
Implementation `ex::max_coefficient()`.
- virtual `exvector get_free_indices` () const
Return a vector containing the free indices of an expression.
- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
Add two indexed expressions.
- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
Multiply an indexed expression with a scalar.
- virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const
Try to contract two indexed expressions that appear in the same product.
- virtual `return_type_t return_type_tinfo` () const
- virtual `ex conjugate` () const
- virtual `ex real_part` () const
- virtual `ex imag_part` () const
- template<class T >
void `print_dispatch` (const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const
Like `print()`, but dispatch to the specified class.
- virtual void `archive` (`archive_node` &n) const
Save (serialize) the object into archive node.
- virtual void `read_archive` (const `archive_node` &n, `lst` &syms)
Load (deserialize) the object from an archive node.
- `ex subs_one_level` (const `exmap` &m, unsigned `options`) const
Helper function for `subs()`.
- `ex diff` (const `symbol` &s, unsigned nth=1) const

Default interface of nth derivative `ex::diff(s, n)`.

- int `compare` (const `basic` &other) const
Compare objects syntactically to establish canonical ordering.
- bool `is_equal` (const `basic` &other) const
Test for syntactic equality.
- const `basic` & `hold` () const
Stop further evaluation.
- unsigned `gethash` () const
- const `basic` & `setflag` (unsigned f) const
Set some `status_flags`.
- const `basic` & `clearflag` (unsigned f) const
Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Protected Member Functions

- unsigned `return_type` () const override

Protected Member Functions inherited from `GiNaC::basic`

- `basic` ()
- virtual `ex eval_ncmul` (const `exvector` &v) const
- virtual bool `match_same_type` (const `basic` &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual `ex derivative` (const `symbol` &s) const
Default implementation of `ex::diff()`.
- virtual int `compare_same_type` (const `basic` &other) const
Returns order relation between two objects of same type.
- virtual bool `is_equal_same_type` (const `basic` &other) const
Returns true if two objects of same type are equal.
- virtual unsigned `calchash` () const
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- void `ensure_if_modifiable` () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void `do_print` (const `print_context` &c, unsigned level) const
Default output to stream.
- void `do_print_tree` (const `print_tree` &c, unsigned level) const
Tree output to stream.
- void `do_print_python_repr` (const `print_python_repr` &c, unsigned level) const
Python parsable output to stream.

Additional Inherited Members

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.182.1 Detailed Description

This class holds one of [GiNaC](#)'s predefined special tensors such as the delta and the metric tensors.

They are represented without indices. To attach indices to them, wrap them in an object of class [indexed](#).

6.182.2 Member Function Documentation

6.182.2.1 [return_type\(\)](#)

```
unsigned GiNaC::tensor::return_type ( ) const [inline], [override], [protected], [virtual]
```

Reimplemented from [GiNaC::basic](#).

References [GiNaC::return_types::noncommutative_composite](#).

6.182.2.2 [replace_contr_index\(\)](#)

```
bool GiNaC::tensor::replace_contr_index (
    exvector::iterator self,
    exvector::iterator other ) const
```

Replace dummy index in contracted-with object by the contracting object's second index (used internally for delta and metric tensor contractions).

References [GiNaC::ex1](#), [GiNaC::is_dummy_pair\(\)](#), [GiNaC::idx::is_symbolic\(\)](#), [GiNaC::idx::minimal_dim\(\)](#), [GiNaC::idx::replace_dim\(\)](#), and [GiNaC::ex::subs\(\)](#).

Referenced by [GiNaC::tensdelta::contract_with\(\)](#), and [GiNaC::tensmetric::contract_with\(\)](#).

The documentation for this class was generated from the following files:

- [tensor.h](#)
- [tensor.cpp](#)

6.183 GiNaC::terminfo Class Reference

This structure stores the original and symmetrized versions of terms obtained during the simplification of sums.

Public Member Functions

- [terminfo](#) (const [ex](#) &orig_, const [ex](#) &symm_)

Public Attributes

- [ex orig](#)
original term
- [ex symm](#)
symmetrized term

6.183.1 Detailed Description

This structure stores the original and symmetrized versions of terms obtained during the simplification of sums.

6.183.2 Constructor & Destructor Documentation

6.183.2.1 [terminfo\(\)](#)

```
GiNaC::terminfo::terminfo (  
    const ex & orig_,  
    const ex & symm_ ) [inline]
```

6.183.3 Member Data Documentation

6.183.3.1 [orig](#)

[ex](#) [GiNaC::terminfo::orig](#)

original term

6.183.3.2 [symm](#)

[ex](#) [GiNaC::terminfo::symm](#)

symmetrized term

Referenced by [GiNaC::terminfo_is_less::operator>\(\)\(\)](#).

The documentation for this class was generated from the following file:

- [indexed.cpp](#)

6.184 GiNaC::terminfo_is_less Class Reference

Public Member Functions

- bool [operator\(\)](#) (const [terminfo](#) &ti1, const [terminfo](#) &ti2) const

6.184.1 Member Function Documentation

6.184.1.1 [operator\(\)](#)()

```
bool GiNaC::terminfo_is_less::operator() (
    const terminfo & ti1,
    const terminfo & ti2 ) const [inline]
```

References [GiNaC::ex::compare\(\)](#), and [GiNaC::terminfo::symm](#).

The documentation for this class was generated from the following file:

- [indexed.cpp](#)

6.185 GiNaC::Gt_detail::TransformExpressionWithCache< Type > Class Template Reference

```
#include <Gt_helpers.h>
```

Public Member Functions

- [TransformExpressionWithCache](#) (const std::function< [ex](#)(const [ex](#) &)> &func_obj, const std::function< [ex](#)(const [ex](#) &)> &func_default=[](const [ex](#) &obj) -> [ex](#) { return obj;})
- [ex operator\(\)](#) (const [ex](#) &input)
- void [print_cache](#) () const
- const [exhashmap](#)< [ex](#) > & [get_cache](#) () const
- void [clear_cache](#) ()

Private Member Functions

- [ex impl](#) (const [ex](#) &expr)

Static Private Member Functions

- template<typename T = Type>
static std::enable_if< std::is_base_of< [basic](#), T >::value, bool >::type [check_type](#) (const [ex](#) &expr)
- template<typename T = Type>
static std::enable_if< !std::is_base_of< [basic](#), T >::value, bool >::type [check_type](#) (const [ex](#) &expr)

Private Attributes

- `const std::function< ex(const ex &> > func\_obj`
- `const std::function< ex(const ex &> > func\_default`
- `exhashmap< ex > cache`

6.185.1 Constructor & Destructor Documentation

6.185.1.1 TransformExpressionWithCache()

```
template<typename Type >
GiNaC::Gt_detail::TransformExpressionWithCache< Type >::TransformExpressionWithCache (
    const std::function< ex(const ex &> > & func\_obj,
    const std::function< ex(const ex &> > & func\_default = [] (const ex& obj) -> ex { return obj; }
) [inline]
```

6.185.2 Member Function Documentation

6.185.2.1 operator>()()

```
template<typename Type >
ex GiNaC::Gt_detail::TransformExpressionWithCache< Type >::operator() (
    const ex & input ) [inline]
```

References [GiNaC::Gt_detail::TransformExpressionWithCache< Type >::impl\(\)](#).

6.185.2.2 print_cache()

```
template<typename Type >
void GiNaC::Gt_detail::TransformExpressionWithCache< Type >::print_cache ( ) const
```

References [cache](#), and [n](#).

6.185.2.3 get_cache()

```
template<typename Type >
const exhashmap< ex > & GiNaC::Gt_detail::TransformExpressionWithCache< Type >::get_cache ( )
const [inline]
```

References [GiNaC::Gt_detail::TransformExpressionWithCache< Type >::cache](#).

6.185.2.4 clear_cache()

```
template<typename Type >
void GiNaC::Gt_detail::TransformExpressionWithCache< Type >::clear_cache ( ) [inline]
```

References [GiNaC::Gt_detail::TransformExpressionWithCache< Type >::cache](#).

6.185.2.5 check_type() [1/2]

```
template<typename Type >
template<typename T >
std::enable_if<!std::is_base_of< basic, T >::value, bool >::type GiNaC::Gt_detail::TransformExpressionWithCa
Type >::check_type (
    const ex & expr ) [static], [private]
```

6.185.2.6 check_type() [2/2]

```
template<typename Type >
template<typename T = Type>
static std::enable_if<!std::is_base_of< basic, T >::value, bool >::type GiNaC::Gt_detail::TransformExpressio
Type >::check_type (
    const ex & expr ) [static], [private]
```

6.185.2.7 impl()

```
template<typename Type >
ex GiNaC::Gt_detail::TransformExpressionWithCache< Type >::impl (
    const ex & expr ) [private]
```

References [GiNaC::container< C >::append\(\)](#), [cache](#), [GiNaC::ex::find\(\)](#), [GiNaC::ex::nops\(\)](#), and [GiNaC::ex::op\(\)](#).

Referenced by [GiNaC::Gt_detail::TransformExpressionWithCache< Type >::operator\(\)\(\)](#).

6.185.3 Member Data Documentation**6.185.3.1 func_obj**

```
template<typename Type >
const std::function<ex(const ex&)> GiNaC::Gt_detail::TransformExpressionWithCache< Type >↔
::func_obj [private]
```

6.185.3.2 func_default

```
template<typename Type >
const std::function<ex(const ex&)> GiNaC::Gt_detail::TransformExpressionWithCache< Type >↔
::func_default [private]
```

6.185.3.3 cache

```
template<typename Type >
exhashmap<ex> GiNaC::Gt_detail::TransformExpressionWithCache< Type >::cache [private]
```

Referenced by [GiNaC::Gt_detail::TransformExpressionWithCache< Type >::clear_cache\(\)](#), and [GiNaC::Gt_detail::TransformExpressio](#)

The documentation for this class was generated from the following files:

- [Gt.h](#)
- [Gt_helpers.h](#)
- [Gt_helpers.cpp](#)

6.186 GiNaC::class_info< OPT >::tree_node Struct Reference

Public Member Functions

- [tree_node](#) ([class_info](#) *)
- void [add_child](#) ([tree_node](#) *n)

Public Attributes

- `std::vector< tree\_node * >` [children](#)
- [class_info](#) * [info](#)

6.186.1 Constructor & Destructor Documentation

6.186.1.1 tree_node()

```
template<class OPT >
GiNaC::class_info< OPT >::tree_node::tree_node (
    class_info * i ) [inline]
```

6.186.2 Member Function Documentation

6.186.2.1 add_child()

```
template<class OPT >
void GiNaC::class_info< OPT >::tree_node::add_child (
    tree_node * n ) [inline]
```

References [GiNaC::class_info< OPT >::tree_node::children](#), and [n](#).

6.186.3 Member Data Documentation

6.186.3.1 children

```
template<class OPT >
std::vector<tree_node *> GiNaC::class_info< OPT >::tree_node::children
```

Referenced by [GiNaC::class_info< OPT >::tree_node::add_child\(\)](#).

6.186.3.2 info

```
template<class OPT >
class_info* GiNaC::class_info< OPT >::tree_node::info
```

The documentation for this struct was generated from the following file:

- [class_info.h](#)

6.187 GiNaC::unarchive_table_t Class Reference

```
#include <archive.h>
```

Public Member Functions

- [unarchive_table_t\(\)](#)
- [~unarchive_table_t\(\)](#)
- [synthesize_func find](#) (const std::string &classname) const
- void [insert](#) (const std::string &classname, [synthesize_func](#) f)

Static Private Attributes

- static int [usecount](#) = 0
- static [unarchive_map_t](#) * [unarch_map](#) = nullptr

6.187.1 Constructor & Destructor Documentation

6.187.1.1 unarchive_table_t()

```
GiNaC::unarchive_table_t::unarchive_table_t ( )
```

References [unarch_map](#), and [usecount](#).

6.187.1.2 ~unarchive_table_t()

```
GiNaC::unarchive_table_t::~~unarchive_table_t ( )
```

References [unarch_map](#), and [usecount](#).

6.187.2 Member Function Documentation

6.187.2.1 find()

```
synthesize\_func GiNaC::unarchive_table_t::find (  
    const std::string & classname ) const
```

References [unarch_map](#).

Referenced by [GiNaC::find_factory_fcn\(\)](#).

6.187.2.2 insert()

```
void GiNaC::unarchive_table_t::insert (  
    const std::string & classname,  
    synthesize\_func f )
```

References [unarch_map](#).

6.187.3 Member Data Documentation

6.187.3.1 usecount

```
int GiNaC::unarchive_table_t::usecount = 0 [static], [private]
```

Referenced by [unarchive_table_t\(\)](#), and [~unarchive_table_t\(\)](#).

6.187.3.2 unarch_map

```
unarchive_map_t * GiNaC::unarchive_table_t::unarch_map = nullptr [static], [private]
```

Referenced by [find\(\)](#), [insert\(\)](#), [unarchive_table_t\(\)](#), and [~unarchive_table_t\(\)](#).

The documentation for this class was generated from the following files:

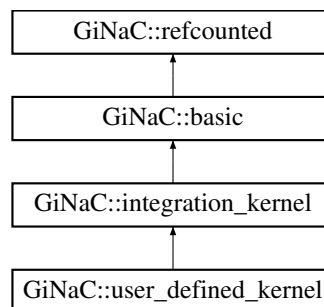
- [archive.h](#)
- [archive.cpp](#)

6.188 GiNaC::user_defined_kernel Class Reference

A user-defined integration kernel.

```
#include <integration_kernel.h>
```

Inheritance diagram for GiNaC::user_defined_kernel:



Public Member Functions

- [user_defined_kernel](#) (const [ex](#) &*f*, const [ex](#) &*x*)
- [size_t nops](#) () const override
Number of operands/members.
- [ex op](#) (size_t *i*) const override
Return operand/member at position i.
- [ex & let_op](#) (size_t *i*) override
Return modifiable operand/member at position i.
- bool [is_numeric](#) (void) const override
This routine returns true, if the integration kernel can be evaluated numerically.
- [ex Laurent_series](#) (const [ex](#) &*x*, int *order*) const override
Returns the Laurent series, starting possibly with the pole term.

Public Member Functions inherited from GiNaC::integration_kernel

- **ex series** (const relational &r, int order, unsigned options=0) const override
Default implementation of ex::series().
- virtual bool **has_trailing_zero** (void) const
This routine returns true, if the integration kernel has a trailing zero.
- virtual **ex get_numerical_value** (const ex &lambda, int N_trunc=0) const
Evaluates the integrand at lambda.
- size_t **get_cache_size** (void) const
Returns the current size of the cache.
- void **set_cache_step** (int cache_steps) const
Sets the step size by which the cache is increased.
- **ex get_series_coeff** (int i) const
Wrapper around series_coeff(i), converts cl_N to numeric.
- cln::cl_N **series_coeff** (int i) const
Subclasses have either to implement series_coeff_impl or the two methods Laurent_series and uses_Laurent_series.

Public Member Functions inherited from GiNaC::basic

- virtual **~basic** ()
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- **basic** (const basic &other)
- const **basic & operator=** (const basic &other)
basic assignment operator: the other object might be of a derived class.
- virtual **basic * duplicate** () const
Create a clone of this object on the heap.
- virtual **ex eval** () const
Perform automatic non-interruptive term rewriting rules.
- virtual **ex evalf** () const
Evaluate object numerically.
- virtual **ex evalm** () const
Evaluate sums, products and integer powers of matrices.
- virtual **ex eval_integ** () const
Evaluate integrals, if result is known.
- virtual **ex eval_indexed** (const basic &i) const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void **print** (const print_context &c, unsigned level=0) const
Output to stream.
- virtual void **dbgprint** () const
Little wrapper around print to be called within a debugger.
- virtual void **dbgprinttree** () const
Little wrapper around printtree to be called within a debugger.
- virtual unsigned **precedence** () const
Return relative operator precedence (for parenthezing output).
- virtual bool **info** (unsigned inf) const
Information about the object.
- virtual **ex operator[]** (const ex &index) const
- virtual **ex operator[]** (size_t i) const
- virtual **ex & operator[]** (const ex &index)
- virtual **ex & operator[]** (size_t i)
- virtual bool **has** (const ex &other, unsigned options=0) const

- Test for occurrence of a pattern.*

 - virtual bool `match` (const `ex` &pattern, `exmap` &repls) const
- Check whether the expression matches a given pattern.*

 - virtual `ex subs` (const `exmap` &m, unsigned `options`=0) const
- Substitute a set of objects by arbitrary expressions.*

 - virtual `ex map` (`map_function` &f) const
- Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).*

 - virtual void `accept` (`GiNaC::visitor` &v) const
 - virtual bool `is_polynomial` (const `ex` &var) const
- Check whether this is a polynomial in the given variables.*

 - virtual int `degree` (const `ex` &s) const
- Return degree of highest power in object s.*

 - virtual int `ldegree` (const `ex` &s) const
- Return degree of lowest power in object s.*

 - virtual `ex coeff` (const `ex` &s, int `n`=1) const
- Return coefficient of degree n in object s.*

 - virtual `ex expand` (unsigned `options`=0) const
- Expand expression, i.e.*

 - virtual `ex collect` (const `ex` &s, bool `distributed`=false) const
- Sort expanded expression in terms of powers of some object(s).*

 - virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const
- Default implementation of `ex::normal()`.*

 - virtual `ex to_rational` (`exmap` &repl) const
- Default implementation of `ex::to_rational()`.*

 - virtual `ex to_polynomial` (`exmap` &repl) const
 - virtual `numeric integer_content` () const
 - virtual `ex smod` (const `numeric` &xi) const
- Apply symmetric modular homomorphism to an expanded multivariate polynomial.*

 - virtual `numeric max_coefficient` () const
- Implementation `ex::max_coefficient()`.*

 - virtual `exvector get_free_indices` () const
- Return a vector containing the free indices of an expression.*

 - virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
- Add two indexed expressions.*

 - virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
- Multiply an indexed expression with a scalar.*

 - virtual bool `contract_with` (`exvector::iterator` self, `exvector::iterator` other, `exvector` &v) const
- Try to contract two indexed expressions that appear in the same product.*

 - virtual unsigned `return_type` () const
 - virtual `return_type_t return_type_tinfo` () const
 - virtual `ex conjugate` () const
 - virtual `ex real_part` () const
 - virtual `ex imag_part` () const
- template<class T >

 - void `print_dispatch` (const `print_context` &c, unsigned level) const
- Like `print()`, but dispatch to the specified class.*

 - void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned level) const
- Like `print()`, but dispatch to the specified class.*

 - virtual void `archive` (`archive_node` &n) const
- Save (serialize) the object into archive node.*

 - virtual void `read_archive` (const `archive_node` &n, `lst` &syms)

- *Load (deserialize) the object from an archive node.*
- `ex subs_one_level` (const `exmap` &m, unsigned `options`) const
Helper function for `subs()`.
- `ex diff` (const `symbol` &s, unsigned `nth`=1) const
Default interface of `nth` derivative `ex::diff(s, n)`.
- `int compare` (const `basic` &other) const
Compare objects syntactically to establish canonical ordering.
- `bool is_equal` (const `basic` &other) const
Test for syntactic equality.
- `const basic & hold` () const
Stop further evaluation.
- `unsigned gethash` () const
- `const basic & setflag` (unsigned `f`) const
Set some `status_flags`.
- `const basic & clearflag` (unsigned `f`) const
Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- `unsigned int add_reference` () noexcept
- `unsigned int remove_reference` () noexcept
- `unsigned int get_refcount` () const noexcept
- `void set_refcount` (unsigned int `r`) noexcept

Protected Member Functions

- `bool uses_Laurent_series` () const override
Returns true, if the coefficients are computed from the Laurent series (in which case the method `Laurent_series` needs to be implemented).
- `void do_print` (const `print_context` &c, unsigned `level`) const

Protected Member Functions inherited from `GiNaC::integration_kernel`

- `virtual cln::cl_N series_coeff_impl` (int `i`) const
For $\omega = d\lambda$ only the coefficient of λ^0 is non-zero.
- `ex get_numerical_value_impl` (const `ex` &lambda, const `ex` &pre, int `shift`, int `N_trunc`) const
The actual implementation for computing a numerical value for the integrand.
- `void do_print` (const `print_context` &c, unsigned `level`) const

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual [ex derivative](#) (const [symbol](#) &s) const
Default implementation of [ex::diff\(\)](#).
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- virtual unsigned [calchash](#) () const
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Protected Attributes

- [ex f](#)
- [ex x](#)

Protected Attributes inherited from [GiNaC::integration_kernel](#)

- int [cache_step_size](#)
- `std::vector< cln::cl\_N >` [series_vec](#)

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.188.1 Detailed Description

A user-defined integration kernel.

The input is an expression f , depending on a variable x . It is assumed that f has a Laurent expansion around $x = 0$ and maximally a simple pole at $x = 0$.

6.188.2 Constructor & Destructor Documentation

6.188.2.1 user_defined_kernel()

```
GiNaC::user_defined_kernel::user_defined_kernel (
    const ex & f,
    const ex & x )
```

6.188.3 Member Function Documentation

6.188.3.1 nops()

```
size_t GiNaC::user_defined_kernel::nops ( ) const [override], [virtual]
```

Number of operands/members.

Reimplemented from [GiNaC::basic](#).

6.188.3.2 op()

```
ex GiNaC::user_defined_kernel::op (
    size_t i ) const [override], [virtual]
```

Return operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [f](#), and [x](#).

6.188.3.3 let_op()

```
ex & GiNaC::user_defined_kernel::let_op (
    size_t i ) [override], [virtual]
```

Return modifiable operand/member at position i.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::basic::ensure_if_modifiable\(\)](#), [f](#), and [x](#).

6.188.3.4 is_numeric()

```
bool GiNaC::user_defined_kernel::is_numeric (
    void ) const [override], [virtual]
```

This routine returns true, if the integration kernel can be evaluated numerically.

Reimplemented from [GiNaC::integration_kernel](#).

References [GiNaC::ex::evalf\(\)](#), [f](#), [GiNaC::ex::info\(\)](#), [GiNaC::info_flags::numeric](#), [GiNaC::ex::subs\(\)](#), and [x](#).

6.188.3.5 Laurent_series()

```
ex GiNaC::user_defined_kernel::Laurent_series (
    const ex & x,
    int order ) const [override], [virtual]
```

Returns the Laurent series, starting possibly with the pole term.

Neglected terms are of order $O(x^{order})$.

Reimplemented from [GiNaC::integration_kernel](#).

References [f](#), [order](#), [GiNaC::ex::series\(\)](#), [GiNaC::ex::subs\(\)](#), and [x](#).

6.188.3.6 uses_Laurent_series()

```
bool GiNaC::user_defined_kernel::uses_Laurent_series ( ) const [override], [protected], [virtual]
```

Returns true, if the coefficients are computed from the Laurent series (in which case the method `Laurent_series` needs to be implemented).

Returns false if this is not the case (and the class has an implementation of `series_coeff_impl`).

Reimplemented from [GiNaC::integration_kernel](#).

6.188.3.7 do_print()

```
void GiNaC::user_defined_kernel::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

References [c](#), [f](#), [GiNaC::ex::print\(\)](#), and [x](#).

6.188.4 Member Data Documentation

6.188.4.1 f

```
ex GiNaC::user_defined_kernel::f [protected]
```

Referenced by [do_print\(\)](#), [is_numeric\(\)](#), [Laurent_series\(\)](#), [let_op\(\)](#), and [op\(\)](#).

6.188.4.2 x

```
ex GiNaC::user_defined_kernel::x [protected]
```

Referenced by [do_print\(\)](#), [is_numeric\(\)](#), [Laurent_series\(\)](#), [let_op\(\)](#), and [op\(\)](#).

The documentation for this class was generated from the following files:

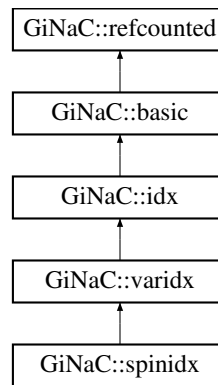
- [integration_kernel.h](#)
- [integration_kernel.cpp](#)

6.189 GiNaC::varidx Class Reference

This class holds an index with a variance (co- or contravariant).

```
#include <idx.h>
```

Inheritance diagram for GiNaC::varidx:



Public Member Functions

- `varidx` (const `ex` &`v`, const `ex` &`dim`, bool `covariant`=false)
Construct index with given value, dimension and variance.
- bool `is_dummy_pair_same_type` (const `basic` &`other`) const override
Check whether the index forms a dummy index pair with another index of the same type.
- void `archive` (`archive_node` &`n`) const override
Save (serialize) the object into archive node.
- void `read_archive` (const `archive_node` &`n`, `lst` &`syms`) override
Load (deserialize) the object from an archive node.
- bool `is_covariant` () const
Check whether the index is covariant.
- bool `is_contravariant` () const
Check whether the index is contravariant (not covariant).
- `ex toggle_variance` () const
Make a new index with the same value but the opposite variance.

Public Member Functions inherited from `GiNaC::idx`

- `idx` (const `ex` &`v`, const `ex` &`dim`)
Construct index with given value and dimension.
- bool `info` (unsigned `inf`) const override
Information about the object.
- `size_t nops` () const override
Number of operands/members.
- `ex op` (`size_t` `i`) const override
Return operand/member at position `i`.
- `ex map` (`map_function` &`f`) const override
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).

- `ex evalf ()` const override
By default, `basic::evalf` would evaluate the index value but we don't want a.1 to become a.
- `ex subs (const exmap &m, unsigned options=0)` const override
Substitute a set of objects by arbitrary expressions.
- `ex get_value ()` const
Get value of index.
- `bool is_numeric ()` const
Check whether the index is numeric.
- `bool is_symbolic ()` const
Check whether the index is symbolic.
- `ex get_dim ()` const
Get dimension of index space.
- `bool is_dim_numeric ()` const
Check whether the dimension is numeric.
- `bool is_dim_symbolic ()` const
Check whether the dimension is symbolic.
- `ex replace_dim (const ex &new_dim)` const
Make a new index with the same value but a different dimension.
- `ex minimal_dim (const idx &other)` const
Return the minimum of the dimensions of this and another index.

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic ()`
basic destructor, virtual because class `ex` will delete objects of derived classes via a `basic*`.
- `basic (const basic &other)`
- `const basic & operator= (const basic &other)`
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate ()` const
Create a clone of this object on the heap.
- virtual `ex eval ()` const
Perform automatic non-interruptive term rewriting rules.
- virtual `ex evalm ()` const
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ ()` const
Evaluate integrals, if result is known.
- virtual `ex eval_indexed (const basic &i)` const
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual void `print (const print_context &c, unsigned level=0)` const
Output to stream.
- virtual void `dbgprint ()` const
Little wrapper around `print` to be called within a debugger.
- virtual void `dbgprinttree ()` const
Little wrapper around `printtree` to be called within a debugger.
- virtual unsigned `precedence ()` const
Return relative operator precedence (for parenthesizing output).
- virtual `ex operator[] (const ex &index)` const
- virtual `ex operator[] (size_t i)` const
- virtual `ex & let_op (size_t i)`
Return modifiable operand/member at position `i`.

- virtual [ex](#) & [operator\[\]](#) (const [ex](#) &index)
- virtual [ex](#) & [operator\[\]](#) (size_t i)
- virtual bool [has](#) (const [ex](#) &other, unsigned [options](#)=0) const
Test for occurrence of a pattern.
- virtual bool [match](#) (const [ex](#) &pattern, [exmap](#) &repls) const
Check whether the expression matches a given pattern.
- virtual void [accept](#) ([GiNaC::visitor](#) &v) const
- virtual bool [is_polynomial](#) (const [ex](#) &var) const
Check whether this is a polynomial in the given variables.
- virtual int [degree](#) (const [ex](#) &s) const
Return degree of highest power in object s.
- virtual int [ldegree](#) (const [ex](#) &s) const
Return degree of lowest power in object s.
- virtual [ex](#) [coeff](#) (const [ex](#) &s, int [n](#)=1) const
Return coefficient of degree n in object s.
- virtual [ex](#) [expand](#) (unsigned [options](#)=0) const
Expand expression, i.e.
- virtual [ex](#) [collect](#) (const [ex](#) &s, bool [distributed](#)=false) const
Sort expanded expression in terms of powers of some object(s).
- virtual [ex](#) [series](#) (const [relational](#) &r, int [order](#), unsigned [options](#)=0) const
Default implementation of [ex::series\(\)](#).
- virtual [ex](#) [normal](#) ([exmap](#) &repl, [exmap](#) &rev_lookup, [lst](#) &modifier) const
Default implementation of [ex::normal\(\)](#).
- virtual [ex](#) [to_rational](#) ([exmap](#) &repl) const
Default implementation of [ex::to_rational\(\)](#).
- virtual [ex](#) [to_polynomial](#) ([exmap](#) &repl) const
- virtual [numeric](#) [integer_content](#) () const
- virtual [ex](#) [smod](#) (const [numeric](#) &xi) const
Apply symmetric modular homomorphism to an expanded multivariate polynomial.
- virtual [numeric](#) [max_coefficient](#) () const
Implementation [ex::max_coefficient\(\)](#).
- virtual [exvector](#) [get_free_indices](#) () const
Return a vector containing the free indices of an expression.
- virtual [ex](#) [add_indexed](#) (const [ex](#) &self, const [ex](#) &other) const
Add two indexed expressions.
- virtual [ex](#) [scalar_mul_indexed](#) (const [ex](#) &self, const [numeric](#) &other) const
Multiply an indexed expression with a scalar.
- virtual bool [contract_with](#) ([exvector::iterator](#) self, [exvector::iterator](#) other, [exvector](#) &v) const
Try to contract two indexed expressions that appear in the same product.
- virtual unsigned [return_type](#) () const
- virtual [return_type_t](#) [return_type_tinfo](#) () const
- virtual [ex](#) [conjugate](#) () const
- virtual [ex](#) [real_part](#) () const
- virtual [ex](#) [imag_part](#) () const
- template<class T >
void [print_dispatch](#) (const [print_context](#) &c, unsigned level) const
Like [print\(\)](#), but dispatch to the specified class.
- void [print_dispatch](#) (const [registered_class_info](#) &ri, const [print_context](#) &c, unsigned level) const
Like [print\(\)](#), but dispatch to the specified class.
- [ex](#) [subs_one_level](#) (const [exmap](#) &m, unsigned [options](#)) const
Helper function for [subs\(\)](#).

- `ex diff` (const `symbol` &s, unsigned nth=1) const
Default interface of nth derivative `ex::diff(s, n)`.
- int `compare` (const `basic` &other) const
Compare objects syntactically to establish canonical ordering.
- bool `is_equal` (const `basic` &other) const
Test for syntactic equality.
- const `basic` & `hold` () const
Stop further evaluation.
- unsigned `gethash` () const
- const `basic` & `setflag` (unsigned f) const
Set some `status_flags`.
- const `basic` & `clearflag` (unsigned f) const
Clear some `status_flags`.

Public Member Functions inherited from `GiNaC::refcounted`

- `refcounted` () noexcept
- unsigned int `add_reference` () noexcept
- unsigned int `remove_reference` () noexcept
- unsigned int `get_refcount` () const noexcept
- void `set_refcount` (unsigned int r) noexcept

Protected Member Functions

- bool `match_same_type` (const `basic` &other) const override
Returns true if the attributes of two objects are similar enough for a match.
- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_tree` (const `print_tree` &c, unsigned level) const

Protected Member Functions inherited from `GiNaC::idx`

- `ex derivative` (const `symbol` &s) const override
Implementation of `ex::diff()` for an index always returns 0.
- unsigned `calchash` () const override
Compute the hash value of an object and if it makes sense to store it in the objects `status_flags`, do so.
- void `print_index` (const `print_context` &c, unsigned level) const
- void `do_print` (const `print_context` &c, unsigned level) const
- void `do_print_csrc` (const `print_csrc` &c, unsigned level) const
- void `do_print_latex` (const `print_latex` &c, unsigned level) const
- void `do_print_tree` (const `print_tree` &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Protected Attributes

- bool [covariant](#)
 $x.\mu$, default is contravariant: $x \sim \mu$

Protected Attributes inherited from [GiNaC::idx](#)

- [ex value](#)
Expression that constitutes the index (numeric or symbolic name)
- [ex dim](#)
Dimension of space (can be symbolic or numeric)

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.189.1 Detailed Description

This class holds an index with a variance (co- or contravariant).

There is an associated metric tensor that can be used to raise/lower indices.

6.189.2 Constructor & Destructor Documentation

6.189.2.1 [varidx\(\)](#)

```
GiNaC::varidx::varidx (
    const ex & v,
    const ex & dim,
    bool covariant = false )
```

Construct index with given value, dimension and variance.

Parameters

<i>v</i>	Value of index (numeric or symbolic)
<i>dim</i>	Dimension of index space (numeric or symbolic)
<i>covariant</i>	Make covariant index (default is contravariant)

6.189.3 Member Function Documentation

6.189.3.1 `is_dummy_pair_same_type()`

```
bool GiNaC::varidx::is_dummy_pair_same_type (
    const basic & other ) const [override], [virtual]
```

Check whether the index forms a dummy index pair with another index of the same type.

Reimplemented from [GiNaC::idx](#).

References [covariant](#).

6.189.3.2 `archive()`

```
void GiNaC::varidx::archive (
    archive_node & n ) const [override], [virtual]
```

Save (serialize) the object into archive node.

Archive the object.

Losely speaking, this method turns an expression into a byte stream (which can be saved and restored later on, or sent via network, etc.)

Reimplemented from [GiNaC::idx](#).

References [covariant](#), and [n](#).

6.189.3.3 `read_archive()`

```
void GiNaC::varidx::read_archive (
    const archive_node & n,
    lst & syms ) [override], [virtual]
```

Load (deserialize) the object from an archive node.

Construct object from [archive_node](#).

Note

This method is essentially a constructor. However, constructors can't be virtual. So, if unarchiving routines are implemented as constructors one would need to define such a constructor in every class, even if all it does is simply calling constructor of a superclass.

Reimplemented from [GiNaC::idx](#).

References [covariant](#), and [n](#).

6.189.3.4 match_same_type()

```
bool GiNaC::varidx::match_same_type (
    const basic & other ) const [override], [protected], [virtual]
```

Returns true if the attributes of two objects are similar enough for a match.

This function must not match subexpressions (this is already done by [basic::match\(\)](#)). Only attributes not accessible by [op\(\)](#) should be compared. This is also the reason why this function doesn't take the wildcard replacement list from [match\(\)](#) as an argument: only subexpressions are subject to wildcard matches. Also, this function only needs to be implemented for container classes because [is_equal_same_type\(\)](#) is automatically used instead of [match_same_type\(\)](#) if [nops\(\)](#) == 0.

See also

[basic::match](#)

Reimplemented from [GiNaC::idx](#).

References [covariant](#), and [GINAC_ASSERT](#).

6.189.3.5 is_covariant()

```
bool GiNaC::varidx::is_covariant ( ) const [inline]
```

Check whether the index is covariant.

References [covariant](#).

Referenced by [GiNaC::tensmetric::eval_indexed\(\)](#).

6.189.3.6 is_contravariant()

```
bool GiNaC::varidx::is_contravariant ( ) const [inline]
```

Check whether the index is contravariant (not covariant).

References [covariant](#).

6.189.3.7 toggle_variance()

```
ex GiNaC::varidx::toggle_variance ( ) const
```

Make a new index with the same value but the opposite variance.

References [GiNaC::basic::clearflag\(\)](#), [covariant](#), [GiNaC::basic::duplicate\(\)](#), and [GiNaC::status_flags::hash_calculated](#).

6.189.3.8 do_print()

```
void GiNaC::varidx::do_print (
    const print\_context & c,
    unsigned level ) const [protected]
```

References [c](#), [covariant](#), and [GiNaC::idx::print_index\(\)](#).

6.189.3.9 do_print_tree()

```
void GiNaC::varidx::do_print_tree (
    const print\_tree & c,
    unsigned level ) const [protected]
```

References [c](#), [covariant](#), [GiNaC::idx::dim](#), [GiNaC::basic::flags](#), [GiNaC::basic::hashvalue](#), [GiNaC::ex::print\(\)](#), and [GiNaC::idx::value](#).

6.189.4 Member Data Documentation

6.189.4.1 covariant

```
bool GiNaC::varidx::covariant [protected]
```

x.mu, default is contravariant: $x \sim \mu$

Referenced by [archive\(\)](#), [do_print\(\)](#), [GiNaC::spinidx::do_print\(\)](#), [do_print_tree\(\)](#), [GiNaC::spinidx::do_print_tree\(\)](#), [is_contravariant\(\)](#), [is_covariant\(\)](#), [is_dummy_pair_same_type\(\)](#), [match_same_type\(\)](#), [read_archive\(\)](#), [toggle_variance\(\)](#), and [GiNaC::spinidx::toggle_variance_dot\(\)](#).

The documentation for this class was generated from the following files:

- [idx.h](#)
- [idx.cpp](#)

6.190 GiNaC::visitor Class Reference

Degenerate base class for visitors.

```
#include <basic.h>
```

Protected Member Functions

- virtual [~visitor\(\)](#)

6.190.1 Detailed Description

Degenerate base class for visitors.

basic and derivative classes support Robert C. Martin's Acyclic Visitor pattern (cf. <http://condor.depaul.edu/dmumaugh/OOT/Design-Principles/acv.pdf> or chapter 10 of Andrei Alexandrescu's "Modern C++ Design").

6.190.2 Constructor & Destructor Documentation

6.190.2.1 ~visitor()

```
virtual GiNaC::visitor::~~visitor ( ) [inline], [protected], [virtual]
```

The documentation for this class was generated from the following file:

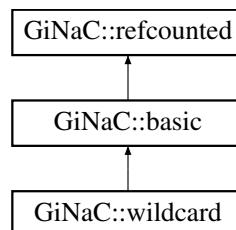
- [basic.h](#)

6.191 GiNaC::wildcard Class Reference

This class acts as a wildcard for [subs\(\)](#), [match\(\)](#), [has\(\)](#) and [find\(\)](#).

```
#include <wildcard.h>
```

Inheritance diagram for GiNaC::wildcard:



Public Member Functions

- [wildcard](#) (unsigned [label](#))
Construct wildcard with specified label.
- bool [match](#) (const [ex](#) &pattern, [exmap](#) &repl_lst) const override
Check whether the expression matches a given pattern.
- void [archive](#) ([archive_node](#) &n) const override
Save (a.k.a.
- void [read_archive](#) (const [archive_node](#) &n, [lst](#) &syms) override
Read (a.k.a.
- unsigned [get_label](#) () const

Public Member Functions inherited from `GiNaC::basic`

- virtual `~basic ()`
basic destructor, virtual because class ex will delete objects of derived classes via a basic.*
- `basic (const basic &other)`
- `const basic & operator= (const basic &other)`
basic assignment operator: the other object might be of a derived class.
- virtual `basic * duplicate () const`
Create a clone of this object on the heap.
- virtual `ex eval () const`
Perform automatic non-interruptive term rewriting rules.
- virtual `ex evalf () const`
Evaluate object numerically.
- virtual `ex evalm () const`
Evaluate sums, products and integer powers of matrices.
- virtual `ex eval_integ () const`
Evaluate integrals, if result is known.
- virtual `ex eval_indexed (const basic &i) const`
Perform automatic symbolic evaluations on indexed expression that contains this object as the base expression.
- virtual `void print (const print_context &c, unsigned level=0) const`
Output to stream.
- virtual `void dbgprint () const`
Little wrapper around print to be called within a debugger.
- virtual `void dbgprinttree () const`
Little wrapper around printtree to be called within a debugger.
- virtual `unsigned precedence () const`
Return relative operator precedence (for parenthezing output).
- virtual `bool info (unsigned inf) const`
Information about the object.
- virtual `size_t nops () const`
Number of operands/members.
- virtual `ex op (size_t i) const`
Return operand/member at position i.
- virtual `ex operator[] (const ex &index) const`
- virtual `ex operator[] (size_t i) const`
- virtual `ex & let_op (size_t i)`
Return modifiable operand/member at position i.
- virtual `ex & operator[] (const ex &index)`
- virtual `ex & operator[] (size_t i)`
- virtual `bool has (const ex &other, unsigned options=0) const`
Test for occurrence of a pattern.
- virtual `ex subs (const exmap &m, unsigned options=0) const`
Substitute a set of objects by arbitrary expressions.
- virtual `ex map (map_function &f) const`
Construct new expression by applying the specified function to all sub-expressions (one level only, not recursively).
- virtual `void accept (GiNaC::visitor &v) const`
- virtual `bool is_polynomial (const ex &var) const`
Check whether this is a polynomial in the given variables.
- virtual `int degree (const ex &s) const`
Return degree of highest power in object s.
- virtual `int ldegree (const ex &s) const`

- Return degree of lowest power in object s.*
- virtual `ex coeff` (const `ex` &s, int `n`=1) const
- Return coefficient of degree n in object s.*
- virtual `ex expand` (unsigned `options`=0) const
- Expand expression, i.e.*
- virtual `ex collect` (const `ex` &s, bool `distributed`=false) const
- Sort expanded expression in terms of powers of some object(s).*
- virtual `ex series` (const `relational` &r, int `order`, unsigned `options`=0) const
- Default implementation of `ex::series()`.*
- virtual `ex normal` (`exmap` &repl, `exmap` &rev_lookup, `lst` &modifier) const
- Default implementation of `ex::normal()`.*
- virtual `ex to_rational` (`exmap` &repl) const
- Default implementation of `ex::to_rational()`.*
- virtual `ex to_polynomial` (`exmap` &repl) const
- virtual `numeric integer_content` () const
- virtual `ex smod` (const `numeric` &xi) const
- Apply symmetric modular homomorphism to an expanded multivariate polynomial.*
- virtual `numeric max_coefficient` () const
- Implementation `ex::max_coefficient()`.*
- virtual `exvector get_free_indices` () const
- Return a vector containing the free indices of an expression.*
- virtual `ex add_indexed` (const `ex` &self, const `ex` &other) const
- Add two indexed expressions.*
- virtual `ex scalar_mul_indexed` (const `ex` &self, const `numeric` &other) const
- Multiply an indexed expression with a scalar.*
- virtual bool `contract_with` (exvector::iterator self, exvector::iterator other, `exvector` &v) const
- Try to contract two indexed expressions that appear in the same product.*
- virtual unsigned `return_type` () const
- virtual `return_type_t return_type_tinfo` () const
- virtual `ex conjugate` () const
- virtual `ex real_part` () const
- virtual `ex imag_part` () const
- template<class T >
- void `print_dispatch` (const `print_context` &c, unsigned `level`) const
 - Like `print()`, but dispatch to the specified class.*
- void `print_dispatch` (const `registered_class_info` &ri, const `print_context` &c, unsigned `level`) const
- Like `print()`, but dispatch to the specified class.*
- `ex subs_one_level` (const `exmap` &m, unsigned `options`) const
- Helper function for `subs()`.*
- `ex diff` (const `symbol` &s, unsigned `nth`=1) const
- Default interface of nth derivative `ex::diff(s, n)`.*
- int `compare` (const `basic` &other) const
- Compare objects syntactically to establish canonical ordering.*
- bool `is_equal` (const `basic` &other) const
- Test for syntactic equality.*
- const `basic` & `hold` () const
- Stop further evaluation.*
- unsigned `gethash` () const
- const `basic` & `setflag` (unsigned `f`) const
- Set some `status_flags`.*
- const `basic` & `clearflag` (unsigned `f`) const
- Clear some `status_flags`.*

Public Member Functions inherited from [GiNaC::refcounted](#)

- [refcounted](#) () noexcept
- unsigned int [add_reference](#) () noexcept
- unsigned int [remove_reference](#) () noexcept
- unsigned int [get_refcount](#) () const noexcept
- void [set_refcount](#) (unsigned int r) noexcept

Protected Member Functions

- unsigned [calchash](#) () const override
Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const

Protected Member Functions inherited from [GiNaC::basic](#)

- [basic](#) ()
- virtual [ex eval_ncmul](#) (const [exvector](#) &v) const
- virtual bool [match_same_type](#) (const [basic](#) &other) const
Returns true if the attributes of two objects are similar enough for a match.
- virtual [ex derivative](#) (const [symbol](#) &s) const
Default implementation of [ex::diff\(\)](#).
- virtual int [compare_same_type](#) (const [basic](#) &other) const
Returns order relation between two objects of same type.
- virtual bool [is_equal_same_type](#) (const [basic](#) &other) const
Returns true if two objects of same type are equal.
- void [ensure_if_modifiable](#) () const
Ensure the object may be modified without hurting others, throws if this is not the case.
- void [do_print](#) (const [print_context](#) &c, unsigned level) const
Default output to stream.
- void [do_print_tree](#) (const [print_tree](#) &c, unsigned level) const
Tree output to stream.
- void [do_print_python_repr](#) (const [print_python_repr](#) &c, unsigned level) const
Python parsable output to stream.

Private Attributes

- unsigned [label](#)
Label used to distinguish different wildcards.

Additional Inherited Members

Protected Attributes inherited from [GiNaC::basic](#)

- unsigned [flags](#)
of type [status_flags](#)
- unsigned [hashvalue](#)
hash value

6.191.1 Detailed Description

This class acts as a wildcard for [subs\(\)](#), [match\(\)](#), [has\(\)](#) and [find\(\)](#).

An integer label is used to identify different wildcards.

6.191.2 Constructor & Destructor Documentation

6.191.2.1 wildcard()

```
GiNaC::wildcard::wildcard (
    unsigned label )
```

Construct wildcard with specified label.

References [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::expanded](#), and [GiNaC::basic::setflag\(\)](#).

6.191.3 Member Function Documentation

6.191.3.1 match()

```
bool GiNaC::wildcard::match (
    const ex & pattern,
    exmap & repl_lst ) const [override], [virtual]
```

Check whether the expression matches a given pattern.

For every wildcard object in the pattern, a pair with the wildcard as a key and matching expression as a value is added to repl_lst.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::basic::is_equal\(\)](#).

6.191.3.2 archive()

```
void GiNaC::wildcard::archive (
    archive_node & n ) const [override], [virtual]
```

Save (a.k.a.

serialize) object into archive.

Reimplemented from [GiNaC::basic](#).

References [label](#), and [n](#).

6.191.3.3 read_archive()

```
void GiNaC::wildcard::read_archive (
    const archive_node & n,
    lst & syms ) [override], [virtual]
```

Read (a.k.a.

deserialize) object from archive.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::status_flags::evaluated](#), [GiNaC::status_flags::expanded](#), [label](#), [n](#), and [GiNaC::basic::setflag\(\)](#).

6.191.3.4 calchash()

```
unsigned GiNaC::wildcard::calchash ( ) const [override], [protected], [virtual]
```

Compute the hash value of an object and if it makes sense to store it in the objects [status_flags](#), do so.

The method inherited from class basic computes a hash value based on the type and hash values of possible members. For this reason it is well suited for container classes but atomic classes should override this implementation because otherwise they would all end up with the same hashvalue.

Reimplemented from [GiNaC::basic](#).

References [GiNaC::golden_ratio_hash\(\)](#), [GiNaC::status_flags::hash_calculated](#), [GiNaC::basic::hashvalue](#), [label](#), [GiNaC::make_hash_seed\(\)](#), and [GiNaC::basic::setflag\(\)](#).

6.191.3.5 get_label()

```
unsigned GiNaC::wildcard::get_label ( ) const [inline]
```

References [label](#).

6.191.3.6 do_print()

```
void GiNaC::wildcard::do_print (
    const print_context & c,
    unsigned level ) const [protected]
```

References [c](#), and [label](#).

6.191.3.7 do_print_tree()

```
void GiNaC::wildcard::do_print_tree (
    const print_tree & c,
    unsigned level ) const [protected]
```

References [c](#), [GiNaC::basic::flags](#), [GiNaC::basic::hashvalue](#), and [label](#).

6.191.3.8 do_print_python_repr()

```
void GiNaC::wildcard::do_print_python_repr (
    const print\_python\_repr & c,
    unsigned level ) const [protected]
```

References [c](#), and [label](#).

6.191.4 Member Data Documentation

6.191.4.1 label

```
unsigned GiNaC::wildcard::label [private]
```

Label used to distinguish different wildcards.

Referenced by [archive\(\)](#), [calchash\(\)](#), [do_print\(\)](#), [do_print_python_repr\(\)](#), [do_print_tree\(\)](#), [get_label\(\)](#), and [read_archive\(\)](#).

The documentation for this class was generated from the following files:

- [wildcard.h](#)
- [wildcard.cpp](#)

6.192 GiNaC::zeta1_SERIAL Class Reference

Complex conjugate.

```
#include <inifcns.h>
```

Static Public Attributes

- static unsigned [serial](#)

6.192.1 Detailed Description

Complex conjugate.

Real part. Imaginary part. Absolute value. Step function. Complex sign. Eta function: $\log(a*b) == \log(a) + \log(b) + \eta(a, b)$. Sine. Cosine. Tangent. Exponential function. Natural logarithm. Inverse sine (arc sine). Inverse cosine (arc cosine). Inverse tangent (arc tangent). Inverse tangent with two arguments. Hyperbolic Sine. Hyperbolic Cosine. Hyperbolic Tangent. Inverse hyperbolic Sine (area hyperbolic sine). Inverse hyperbolic Cosine (area hyperbolic cosine). Inverse hyperbolic Tangent (area hyperbolic tangent). Dilogarithm. Trilogarithm. Derivatives of Riemann's Zeta-function. Multiple zeta value including Riemann's zeta-function.

6.192.2 Member Data Documentation

6.192.2.1 serial

```
unsigned GiNaC::zeta1_SERIAL::serial [static]
```

Initial value:

```
= function::register_new(function_options("zeta", 1).
    evalf_func(zeta1_evalf).
    eval_func(zeta1_eval).
    derivative_func(zeta1_deriv).
    print_func<print_latex>(zeta1_print_latex).
    do_not_evalf_params().
    overloaded(2))
```

Referenced by [GiNaC::zeta\(\)](#).

The documentation for this class was generated from the following files:

- [inifcns.h](#)
- [inifcns_nstdsums.cpp](#)

6.193 GiNaC::zeta2_SERIAL Class Reference

Alternating Euler sum or colored MZV.

```
#include <inifcns.h>
```

Static Public Attributes

- static unsigned [serial](#)

6.193.1 Detailed Description

Alternating Euler sum or colored MZV.

6.193.2 Member Data Documentation

6.193.2.1 serial

```
unsigned GiNaC::zeta2_SERIAL::serial [static]
```

Initial value:

```
= function::register_new(function_options("zeta", 2).
    evalf_func(zeta2_evalf).
    eval_func(zeta2_eval).
    derivative_func(zeta2_deriv).
    print_func<print_latex>(zeta2_print_latex).
    do_not_evalf_params().
    overloaded(2))
```

Referenced by [GiNaC::zeta\(\)](#).

The documentation for this class was generated from the following files:

- [inifcns.h](#)
- [inifcns_nstdsums.cpp](#)

Chapter 7

File Documentation

7.1 add.cpp File Reference

Implementation of [GiNaC](#)'s sums of expressions.

```
#include "add.h"
#include "mul.h"
#include "archive.h"
#include "operators.h"
#include "matrix.h"
#include "utils.h"
#include "clifford.h"
#include "ncmul.h"
#include "compiler.h"
#include <limits>
#include <stdexcept>
```

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) (add, expairseq, print_func< [print_context](#) >(&add::do_print). print_func< [print_latex](#) >(&add::do_print_latex). print_func< [print_csrc](#) >(&add::do_print_csrc). print_func< [print_tree](#) >(&add::do_print_tree). print_func< [print_python_repr](#) >(&add::do_print_python_repr))
add
- [GiNaC::GINAC_BIND_UNARCHIVER](#) (add)

7.1.1 Detailed Description

Implementation of [GiNaC](#)'s sums of expressions.

7.2 add.h File Reference

Interface to [GiNaC](#)'s sums of expressions.

```
#include "expairseq.h"
```

Classes

- class [GiNaC::add](#)
Sum of expressions.

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([add](#))

7.2.1 Detailed Description

Interface to [GiNaC](#)'s sums of expressions.

7.3 archive.cpp File Reference

Archiving of [GiNaC](#) expressions.

```
#include "archive.h"  
#include "registrar.h"  
#include "ex.h"  
#include "lst.h"  
#include "version.h"  
#include <stdexcept>
```

Namespaces

- namespace [GiNaC](#)

Functions

- static void [GiNaC::write_unsigned](#) (std::ostream &os, unsigned val)
Write unsigned integer quantity to stream.
- static unsigned [GiNaC::read_unsigned](#) (std::istream &is)
Read unsigned integer quantity from stream.
- std::ostream & [GiNaC::operator<<](#) (std::ostream &os, const [archive_node](#) &n)
Write [archive_node](#) to binary data stream.
- std::ostream & [GiNaC::operator<<](#) (std::ostream &os, const [archive](#) &ar)
Write archive to binary data stream.
- std::istream & [GiNaC::operator>>](#) (std::istream &is, [archive_node](#) &n)
Read [archive_node](#) from binary data stream.
- std::istream & [GiNaC::operator>>](#) (std::istream &is, [archive](#) &ar)
Read archive from binary data stream.
- static [synthesize_func](#) [GiNaC::find_factory_fcn](#) (const std::string &name)

7.3.1 Detailed Description

Archiving of [GiNaC](#) expressions.

7.4 archive.h File Reference

Archiving of [GiNaC](#) expressions.

```
#include "ex.h"
#include <iosfwd>
#include <map>
#include <string>
#include <vector>
```

Classes

- class [GiNaC::archive_node](#)
This class stores all properties needed to record/retrieve the state of one object of class basic (or a derived class).
- struct [GiNaC::archive_node::property_info](#)
Information about a stored property.
- struct [GiNaC::archive_node::property](#)
Archived property (data type, name and associated data)
- struct [GiNaC::archive_node::archive_node_cit_range](#)
- class [GiNaC::unarchive_table_t](#)
- class [GiNaC::archive](#)
This class holds archived versions of [GiNaC](#) expressions (class [ex](#)).
- struct [GiNaC::archive::archived_ex](#)
Archived expression descriptor.

Namespaces

- namespace [GiNaC](#)

Macros

- `#define GINAC_DECLARE_UNARCHIVER(classname)`
Helper macros to register a class with (un)archiving (a.k.a.
- `#define GINAC_BIND_UNARCHIVER(classname)`

Typedefs

- `typedef unsigned GiNaC::archive_node_id`
Numerical ID value to refer to an [archive_node](#).
- `typedef unsigned GiNaC::archive_atom`
Numerical ID value to refer to a string.
- `typedef basic *(* GiNaC::synthesize_func) ()`
- `typedef std::map< std::string, synthesize_func > GiNaC::unarchive_map_t`

Functions

- `std::ostream & GiNaC::operator<< (std::ostream &os, const archive &ar)`
Write archive to binary data stream.
- `std::istream & GiNaC::operator>> (std::istream &is, archive &ar)`
Read archive from binary data stream.

Variables

- static `unarchive_table_t GiNaC::unarch_table_instance`

7.4.1 Detailed Description

Archiving of [GiNaC](#) expressions.

7.4.2 Macro Definition Documentation

7.4.2.1 GINAC_DECLARE_UNARCHIVER

```
#define GINAC_DECLARE_UNARCHIVER(  
    classname )
```

Value:

```
class classname ## _unarchiver \
{ \
    static int usecount; \
public: \
    static GiNaC::basic* create(); \
    classname ## _unarchiver(); \
    ~ classname ## _unarchiver(); \
}; \
static classname ## _unarchiver classname ## _unarchiver_instance
```

Helper macros to register a class with (un)archiving (a.k.a.

(de)serialization).

Usage: put

`GINAC_DECLARE_UNARCHIVER(myclass);`

into the header file (in the global or namespace scope), and

`GINAC_BIND_UNARCHIVER(myclass);`

into the source file.

Effect: the 'myclass' (being a class derived directly or indirectly from `GiNaC::basic`) can be archived and unarchived.

Note: you need to use `GINAC_{DECLARE,BIND}_UNARCHIVER` incantations in order to make your class (un)archivable *even if your class does not overload 'read_archive' method*. Sorry for inconvenience.

How it works:

The 'basic' class has a 'read_archive' virtual method which reads an expression from archive. Derived classes can overload that method. There's a small problem, though. On unarchiving all we have is a set of named byte streams. In C++ the class name (as written in the source code) has nothing to do with its actual type. Thus, we need establish a correspondence ourselves. To do so we maintain a 'class_name' => 'function_pointer' table (see the `unarchive_table_t` class above). Every function in this table is supposed to create a new object of the 'class_name' type. The 'archive_node' class uses that table to construct an object of correct type. Next it invokes `read_archive` virtual method of newly created object, which does the actual job.

Note: this approach is very simple-minded (it does not handle classes with same names from different namespaces, multiple inheritance, etc), but it happens to work surprisingly well.

7.4.2.2 GINAC_BIND_UNARCHIVER

```
#define GINAC_BIND_UNARCHIVER(  
    classname )
```

Value:

```
classname ## _unarchiver::classname ## _unarchiver() \
{ \
    static GiNaC::unarchive_table_t table; \
    if (usecount++ == 0) { \
        table.insert(std::string(#classname), \
            &(classname ## _unarchiver::create)); \
    } \
} \
GiNaC::basic* classname ## _unarchiver::create() \
{ \
    return new classname(); \
} \
classname ## _unarchiver::~~ classname ## _unarchiver() { } \
int classname ## _unarchiver::usecount = 0
```

7.5 assertion.h File Reference

Assertion macro definition.

Macros

- `#define GINAC_ASSERT(X) ((void)0)`
Assertion macro for checking invariances.

7.5.1 Detailed Description

Assertion macro definition.

7.5.2 Macro Definition Documentation

7.5.2.1 GINAC_ASSERT

```
#define GINAC_ASSERT(  
    X ) ((void)0)
```

Assertion macro for checking invariances.

7.6 basic.cpp File Reference

Implementation of [GiNaC's ABC](#).

```
#include "basic.h"  
#include "ex.h"  
#include "numeric.h"  
#include "power.h"  
#include "add.h"  
#include "symbol.h"  
#include "lst.h"  
#include "ncmul.h"  
#include "relational.h"  
#include "operators.h"  
#include "wildcard.h"  
#include "archive.h"  
#include "utils.h"  
#include "hash_seed.h"  
#include "inifcns.h"  
#include <iostream>  
#include <stdexcept>  
#include <typeinfo>
```

Classes

- struct [GiNaC::evalf_map_function](#)
Function object to be applied by *basic::evalf()*.
- struct [GiNaC::evalm_map_function](#)
Function object to be applied by *basic::evalm()*.
- struct [GiNaC::eval_integ_map_function](#)
Function object to be applied by *basic::eval_integ()*.
- struct [GiNaC::derivative_map_function](#)
Function object to be applied by *basic::derivative()*.
- struct [GiNaC::expand_map_function](#)
Function object to be applied by *basic::expand()*.

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([basic](#), void, [print_func](#)< [print_context](#)>(&[basic::do_print](#)), [print_func](#)< [print_tree](#)>(&[basic::do_print_tree](#)), [print_func](#)< [print_python_repr](#)>(&[basic::do_print_python_repr](#))) [basic](#)

basic copy constructor: implicitly assumes that the other class is of the exact same type (as it's used by duplicate()), so it can copy the tinfo_key and the hash value.

Variables

- [GiNaC::evalm_map_function](#) [GiNaC::map_evalm](#)
- [GiNaC::eval_integ_map_function](#) [GiNaC::map_eval_integ](#)

7.6.1 Detailed Description

Implementation of [GiNaC](#)'s ABC.

7.7 basic.h File Reference

Interface to [GiNaC](#)'s ABC.

```
#include "flags.h"
#include "ptr.h"
#include "assertion.h"
#include "registrar.h"
#include <cstddef>
#include <map>
#include <set>
#include <typeinfo>
#include <vector>
#include <utility>
```

Classes

- struct [GiNaC::map_function](#)
Function object for map().
- class [GiNaC::visitor](#)
Degenerate base class for visitors.
- class [GiNaC::basic](#)
This class is the ABC (abstract base class) of [GiNaC](#)'s class hierarchy.

Namespaces

- namespace [GiNaC](#)

Typedefs

- typedef std::vector< [ex](#) > [GiNaC::exvector](#)
- typedef std::set< [ex](#), [ex_is_less](#) > [GiNaC::exset](#)
- typedef std::map< [ex](#), [ex](#), [ex_is_less](#) > [GiNaC::exmap](#)

Functions

- template<class T >
bool [GiNaC::is_a](#) (const [basic](#) &obj)
Check if obj is a T, including base classes.
- template<class T >
bool [GiNaC::is_exactly_a](#) (const [basic](#) &obj)
Check if obj is a T, not including base classes.
- template<class B , typename... Args>
B & [GiNaC::dynallocate](#) (Args &&... args)
Constructs a new (class basic or derived) B object on the heap.
- template<class B >
B & [GiNaC::dynallocate](#) (std::initializer_list< [ex](#) > il)
Constructs a new (class basic or derived) B object on the heap.

7.7.1 Detailed Description

Interface to [GiNaC](#)'s ABC.

7.8 class_info.h File Reference

Helper templates to provide per-class information for class hierarchies.

```
#include <cstdint>
#include <cstring>
#include <iomanip>
#include <iostream>
#include <map>
#include <stdexcept>
#include <string>
#include <vector>
```

Classes

- class [GiNaC::class_info](#)< OPT >
- struct [GiNaC::class_info](#)< OPT >::tree_node

Namespaces

- namespace [GiNaC](#)

7.8.1 Detailed Description

Helper templates to provide per-class information for class hierarchies.

7.9 clifford.cpp File Reference

Implementation of [GiNaC](#)'s clifford algebra (Dirac gamma) objects.

```
#include "clifford.h"
#include "ex.h"
#include "idx.h"
#include "ncmul.h"
#include "symbol.h"
#include "numeric.h"
#include "symmetry.h"
#include "lst.h"
#include "relational.h"
#include "operators.h"
#include "add.h"
#include "mul.h"
#include "power.h"
#include "matrix.h"
#include "archive.h"
#include "utils.h"
#include <algorithm>
#include <stdexcept>
```

Classes

- struct [GiNaC::is_not_a_clifford](#)
Predicate for finding non-clifford objects.

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([clifford](#), [indexed](#), [print_func< print_dflt >\(&clifford::do_print_dflt\)](#), [print_func< print_latex >\(&clifford::do_print_latex\)](#), [print_func< print_tree >\(&clifford::do_print_tree\)](#)) [GINAC_IMPLEMENT_REGISTERED_CLASS_OPT\(diracone](#)
- [GiNaC::print_func< print_dflt >\(&diracone::do_print\)](#), [print_func< print_latex >\(&diracone](#)
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([clifford](#))
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([cliffordunit](#))
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([diracone](#))
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([diracgamma](#))
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([diracgamma5](#))
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([diracgammaL](#))
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([diracgammaR](#))
- static bool [GiNaC::is_dirac_slash](#) (const [ex](#) &seq0)

- static void `GiNaC::base_and_index` (const `ex` &c, `ex` &b, `ex` &i)
This function decomposes $\gamma_{\sim\mu} \rightarrow (1, \mu)$ and $a_I \rightarrow (a.Ix, ix)$
- `ex` `GiNaC::dirac_ONE` (unsigned char rl=0)
Create a Clifford unity object.
- static unsigned `GiNaC::get_dim_uint` (const `ex` &e)
- `ex` `GiNaC::clifford_unit` (const `ex` &mu, const `ex` &metr, unsigned char rl=0)
Create a Clifford unit object.
- `ex` `GiNaC::dirac_gamma` (const `ex` &mu, unsigned char rl=0)
Create a Dirac gamma object.
- `ex` `GiNaC::dirac_gamma5` (unsigned char rl=0)
Create a Dirac gamma5 object.
- `ex` `GiNaC::dirac_gammaL` (unsigned char rl=0)
Create a Dirac gammaL object.
- `ex` `GiNaC::dirac_gammaR` (unsigned char rl=0)
Create a Dirac gammaR object.
- `ex` `GiNaC::dirac_slash` (const `ex` &e, const `ex` &dim, unsigned char rl=0)
Create a term of the form $e_{\mu} * \gamma_{\sim\mu}$ with a unique index μ .
- static unsigned char `GiNaC::get_representation_label` (const `return_type_t` &ti)
Extract representation label from tinfo key (as returned by `return_type_tinfo()`).
- static `ex` `GiNaC::trace_string` (exvector::const_iterator ix, size_t num)
Take trace of a string of an even number of Dirac gammas given a vector of indices.
- `ex` `GiNaC::dirac_trace` (const `ex` &e, const std::set< unsigned char > &rls, const `ex` &trONE=4)
Calculate dirac traces over the specified set of representation labels.
- `ex` `GiNaC::dirac_trace` (const `ex` &e, const `lst` &rl, const `ex` &trONE=4)
Calculate dirac traces over the specified list of representation labels.
- `ex` `GiNaC::dirac_trace` (const `ex` &e, unsigned char rl=0, const `ex` &trONE=4)
Calculate the trace of an expression containing gamma objects with a specified representation label.
- `ex` `GiNaC::canonicalize_clifford` (const `ex` &e)
Bring all products of clifford objects in an expression into a canonical order.
- `ex` `GiNaC::clifford_star_bar` (const `ex` &e, bool do_bar, unsigned `options`)
An auxillary function performing `clifford_star()` and `clifford_bar()`.
- `ex` `GiNaC::clifford_prime` (const `ex` &e)
Automorphism of the Clifford algebra, simply changes signs of all clifford units.
- `ex` `GiNaC::remove_dirac_ONE` (const `ex` &e, unsigned char rl=0, unsigned `options`=0)
Replaces `dirac_ONE`'s (with a `representation_label` no less than `rl`) in `e` with 1.
- int `GiNaC::clifford_max_label` (const `ex` &e, bool ignore_ONE=false)
Returns the maximal representation label of a clifford object if `e` contains at least one, otherwise returns -1.
- `ex` `GiNaC::clifford_norm` (const `ex` &e)
Calculation of the norm in the Clifford algebra.
- `ex` `GiNaC::clifford_inverse` (const `ex` &e)
Calculation of the inverse in the Clifford algebra.
- `ex` `GiNaC::lst_to_clifford` (const `ex` &v, const `ex` &mu, const `ex` &metr, unsigned char rl=0)
List or vector conversion into the Clifford vector.
- `ex` `GiNaC::lst_to_clifford` (const `ex` &v, const `ex` &e)
List or vector conversion into the Clifford vector.
- static `ex` `GiNaC::get_clifford_comp` (const `ex` &e, const `ex` &c, bool root=true)
Auxiliary structure to define a function for stripping one Clifford unit from vectors.
- `lst` `GiNaC::clifford_to_lst` (const `ex` &e, const `ex` &c, bool algebraic=true)
An inverse function to `lst_to_clifford()`.
- `ex` `GiNaC::clifford_moebius_map` (const `ex` &a, const `ex` &b, const `ex` &c, const `ex` &d, const `ex` &v, const `ex` &G, unsigned char rl=0)

Calculations of Moebius transformations (conformal map) defined by a 2x2 Clifford matrix (a b|c d) in linear spaces with arbitrary signature.

- [ex GiNaC::clifford_moebius_map](#) (const [ex](#) &M, const [ex](#) &v, const [ex](#) &G, unsigned char rl=0)

The second form of Moebius transformations defined by a 2x2 Clifford matrix M This function takes the transformation matrix M as a single entity.

Variables

- [GiNaC::tensor](#)

7.9.1 Detailed Description

Implementation of [GiNaC](#)'s clifford algebra (Dirac gamma) objects.

7.10 clifford.h File Reference

Interface to [GiNaC](#)'s clifford algebra (Dirac gamma) objects.

```
#include "indexed.h"
#include "tensor.h"
#include "symbol.h"
#include "idx.h"
#include <set>
```

Classes

- class [GiNaC::clifford](#)
This class holds an object representing an element of the Clifford algebra (the Dirac gamma matrices).
- class [GiNaC::diracone](#)
This class represents the Clifford algebra unity element.
- class [GiNaC::cliffordunit](#)
This class represents the Clifford algebra generators (units).
- class [GiNaC::diracgamma](#)
This class represents the Dirac gamma Lorentz vector.
- class [GiNaC::diracgamma5](#)
This class represents the Dirac gamma5 object which anticommutes with all other gammas.
- class [GiNaC::diracgammaL](#)
This class represents the Dirac gammaL object which behaves like 1/2 (1-gamma5).
- class [GiNaC::diracgammaR](#)
This class represents the Dirac gammaL object which behaves like 1/2 (1+gamma5).

Namespaces

- namespace [GiNaC](#)

Functions

- `GiNaC::GINAC_DECLARE_UNARCHIVER` (`clifford`)
- `GiNaC::GINAC_DECLARE_UNARCHIVER` (`diracone`)
- `GiNaC::GINAC_DECLARE_UNARCHIVER` (`cliffordunit`)
- `GiNaC::GINAC_DECLARE_UNARCHIVER` (`diracgamma`)
- `GiNaC::GINAC_DECLARE_UNARCHIVER` (`diracgamma5`)
- `GiNaC::GINAC_DECLARE_UNARCHIVER` (`diracgammaL`)
- `GiNaC::GINAC_DECLARE_UNARCHIVER` (`diracgammaR`)
- `bool GiNaC::is_clifford_tinfo` (`const return_type_t &ti`)
Check whether a given `return_type_t` object (as returned by `return_type_tinfo()`) is that of a clifford object (with an arbitrary representation label).
- `ex GiNaC::dirac_ONE` (`unsigned char rl=0`)
Create a Clifford unity object.
- `ex GiNaC::clifford_unit` (`const ex &mu`, `const ex &metr`, `unsigned char rl=0`)
Create a Clifford unit object.
- `ex GiNaC::dirac_gamma` (`const ex &mu`, `unsigned char rl=0`)
Create a Dirac gamma object.
- `ex GiNaC::dirac_gamma5` (`unsigned char rl=0`)
Create a Dirac gamma5 object.
- `ex GiNaC::dirac_gammaL` (`unsigned char rl=0`)
Create a Dirac gammaL object.
- `ex GiNaC::dirac_gammaR` (`unsigned char rl=0`)
Create a Dirac gammaR object.
- `ex GiNaC::dirac_slash` (`const ex &e`, `const ex &dim`, `unsigned char rl=0`)
Create a term of the form $e_{\mu} * \gamma_{\sim\mu}$ with a unique index μ .
- `ex GiNaC::dirac_trace` (`const ex &e`, `const std::set< unsigned char > &rls`, `const ex &trONE=4`)
Calculate dirac traces over the specified set of representation labels.
- `ex GiNaC::dirac_trace` (`const ex &e`, `const lst &rls`, `const ex &trONE=4`)
Calculate dirac traces over the specified list of representation labels.
- `ex GiNaC::dirac_trace` (`const ex &e`, `unsigned char rl=0`, `const ex &trONE=4`)
Calculate the trace of an expression containing gamma objects with a specified representation label.
- `ex GiNaC::canonicalize_clifford` (`const ex &e`)
Bring all products of clifford objects in an expression into a canonical order.
- `ex GiNaC::clifford_prime` (`const ex &e`)
Automorphism of the Clifford algebra, simply changes signs of all clifford units.
- `ex GiNaC::clifford_star_bar` (`const ex &e`, `bool do_bar`, `unsigned options`)
An auxillary function performing `clifford_star()` and `clifford_bar()`.
- `ex GiNaC::clifford_bar` (`const ex &e`)
Main anti-automorphism of the Clifford algebra: makes reversion and changes signs of all clifford units.
- `ex GiNaC::clifford_star` (`const ex &e`)
Reversion of the Clifford algebra, reverse the order of all clifford units in `ncmul`.
- `ex GiNaC::remove_dirac_ONE` (`const ex &e`, `unsigned char rl=0`, `unsigned options=0`)
Replaces `dirac_ONE`'s (with a representation_label no less than `rl`) in `e` with 1.
- `int GiNaC::clifford_max_label` (`const ex &e`, `bool ignore_ONE=false`)
Returns the maximal representation label of a clifford object if `e` contains at least one, otherwise returns -1.
- `ex GiNaC::clifford_norm` (`const ex &e`)
Calculation of the norm in the Clifford algebra.
- `ex GiNaC::clifford_inverse` (`const ex &e`)
Calculation of the inverse in the Clifford algebra.
- `ex GiNaC::lst_to_clifford` (`const ex &v`, `const ex &mu`, `const ex &metr`, `unsigned char rl=0`)
List or vector conversion into the Clifford vector.

- `ex GiNaC::lst_to_clifford` (const `ex` &v, const `ex` &e)
List or vector conversion into the Clifford vector.
- `lst GiNaC::clifford_to_lst` (const `ex` &e, const `ex` &c, bool algebraic=true)
An inverse function to `lst_to_clifford()`.
- `ex GiNaC::clifford_moebius_map` (const `ex` &a, const `ex` &b, const `ex` &c, const `ex` &d, const `ex` &v, const `ex` &G, unsigned char rl=0)
Calculations of Moebius transformations (conformal map) defined by a 2x2 Clifford matrix (a b|c d) in linear spaces with arbitrary signature.
- `ex GiNaC::clifford_moebius_map` (const `ex` &M, const `ex` &v, const `ex` &G, unsigned char rl=0)
The second form of Moebius transformations defined by a 2x2 Clifford matrix M This function takes the transformation matrix M as a single entity.

7.10.1 Detailed Description

Interface to [GiNaC](#)'s clifford algebra (Dirac gamma) objects.

7.11 color.cpp File Reference

Implementation of [GiNaC](#)'s color (SU(3) Lie algebra) objects.

```
#include "color.h"
#include "idx.h"
#include "ncmul.h"
#include "symmetry.h"
#include "operators.h"
#include "numeric.h"
#include "mul.h"
#include "power.h"
#include "symbol.h"
#include "archive.h"
#include "utils.h"
#include <stdexcept>
```

Namespaces

- namespace [GiNaC](#)

Macros

- `#define TEST_PERMUTATION(A, B, C, P)`
- `#define CMPINDICES(A, B, C) ((v[0] == (A)) && (v[1] == (B)) && (v[2] == (C)))`

Functions

- `GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`su3one`, `tensor`, `print_func< print_dflt >(&su3one::do_print)`). `print_func< print_latex >(&su3one::do_print_latex)`) `GINAC_IMPLEMENT_↵`
`REGISTERED_CLASS_OPT(su3t`
- `GiNaC::print_func< print_dflt > (&su3t::do_print)`. `print_func< print_latex >(&su3t`
- `GiNaC::GINAC_BIND_UNARCHIVER` (`color`)
- `GiNaC::GINAC_BIND_UNARCHIVER` (`su3one`)
- `GiNaC::GINAC_BIND_UNARCHIVER` (`su3t`)
- `GiNaC::GINAC_BIND_UNARCHIVER` (`su3f`)
- `GiNaC::GINAC_BIND_UNARCHIVER` (`su3d`)
- static `ex` `GiNaC::permute_free_index_to_front` (const `exvector` &iv3, const `exvector` &iv2, int &sig)
Given a vector iv3 of three indices and a vector iv2 of two indices that is a subset of iv3, return the (free) index that is in iv3 but not in iv2 and the sign introduced by permuting that index to the front.
- `ex` `GiNaC::color_ONE` (unsigned char rl=0)
Create the su(3) unity element.
- `ex` `GiNaC::color_T` (const `ex` &a, unsigned char rl=0)
Create an su(3) generator.
- `ex` `GiNaC::color_f` (const `ex` &a, const `ex` &b, const `ex` &c)
Create an su(3) antisymmetric structure constant.
- `ex` `GiNaC::color_d` (const `ex` &a, const `ex` &b, const `ex` &c)
Create an su(3) symmetric structure constant.
- `ex` `GiNaC::color_h` (const `ex` &a, const `ex` &b, const `ex` &c)
*This returns the linear combination d.a.b.c+l*f.a.b.c.*
- static bool `GiNaC::is_color_tinfo` (const `return_type_t` &ti)
Check whether a given tinfo key (as returned by `return_type_tinfo()`) is that of a color object (with an arbitrary representation label).
- static unsigned char `GiNaC::get_representation_label` (const `return_type_t` &ti)
Extract representation label from tinfo key (as returned by `return_type_tinfo()`).
- `ex` `GiNaC::color_trace` (const `ex` &e, const `std::set< unsigned char >` &rls)
Calculate color traces over the specified set of representation labels.
- `ex` `GiNaC::color_trace` (const `ex` &e, const `lst` &rl)
Calculate color traces over the specified list of representation labels.
- `ex` `GiNaC::color_trace` (const `ex` &e, unsigned char rl=0)
Calculate the trace of an expression containing color objects with a specified representation label.

7.11.1 Detailed Description

Implementation of `GiNaC`'s color (SU(3) Lie algebra) objects.

7.11.2 Macro Definition Documentation

7.11.2.1 TEST_PERMUTATION

```
#define TEST_PERMUTATION(
    A,
    B,
    C,
    P )
```

Value:

```
if (iv3[B].is_equal(iv2[0]) && iv3[C].is_equal(iv2[1])) { \
    sig = P; \
    return iv3[A]; \
}
```


7.11.2.2 CMPINDICES

```
#define CMPINDICES(
    A,
    B,
    C ) ((v[0] == (A)) && (v[1] == (B)) && (v[2] == (C)))
```

7.12 color.h File Reference

Interface to GiNaC's color (SU(3) Lie algebra) objects.

```
#include "indexed.h"
#include "tensor.h"
#include <set>
```

Classes

- class [GiNaC::color](#)
This class holds a generator T_a or the unity element of the Lie algebra of SU(3), as used for calculations in quantum chromodynamics.
- class [GiNaC::su3one](#)
This class represents the su(3) unity element.
- class [GiNaC::su3t](#)
This class represents an su(3) generator.
- class [GiNaC::su3f](#)
This class represents the tensor of antisymmetric su(3) structure constants.
- class [GiNaC::su3d](#)
This class represents the tensor of symmetric su(3) structure constants.

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([color](#))
- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([su3one](#))
- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([su3t](#))
- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([su3f](#))
- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([su3d](#))
- [ex GiNaC::color_ONE](#) (unsigned char rl=0)
Create the su(3) unity element.
- [ex GiNaC::color_T](#) (const [ex](#) &a, unsigned char rl=0)
Create an su(3) generator.
- [ex GiNaC::color_f](#) (const [ex](#) &a, const [ex](#) &b, const [ex](#) &c)
Create an su(3) antisymmetric structure constant.
- [ex GiNaC::color_d](#) (const [ex](#) &a, const [ex](#) &b, const [ex](#) &c)
Create an su(3) symmetric structure constant.

- `ex GiNaC::color_h` (const `ex` &a, const `ex` &b, const `ex` &c)
This returns the linear combination $d.a.b.c + l*f.a.b.c$.
- `ex GiNaC::color_trace` (const `ex` &e, const `std::set< unsigned char >` &rls)
Calculate color traces over the specified set of representation labels.
- `ex GiNaC::color_trace` (const `ex` &e, const `lst` &rl)
Calculate color traces over the specified list of representation labels.
- `ex GiNaC::color_trace` (const `ex` &e, unsigned char rl=0)
Calculate the trace of an expression containing color objects with a specified representation label.

7.12.1 Detailed Description

Interface to `GiNaC`'s color (SU(3) Lie algebra) objects.

7.13 compiler.h File Reference

Definition of optimizing macros.

Macros

- `#define unlikely`(cond) (cond)
- `#define likely`(cond) (cond)
- `#define attribute_pure`

7.13.1 Detailed Description

Definition of optimizing macros.

7.13.2 Macro Definition Documentation

7.13.2.1 unlikely

```
#define unlikely(  
    cond ) (cond)
```

7.13.2.2 likely

```
#define likely(  
    cond ) (cond)
```

7.13.2.3 attribute_pure

```
#define attribute_pure
```

7.14 constant.cpp File Reference

Implementation of [GiNaC](#)'s constant types and some special constants.

```
#include "constant.h"
#include "numeric.h"
#include "ex.h"
#include "archive.h"
#include "utils.h"
#include "inifcns.h"
#include <stdexcept>
#include <string>
```

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([constant](#), [basic](#), [print_func< print_context >\(&constant::do_print\)](#), [print_func< print_latex >\(&constant::do_print_latex\)](#), [print_func< print_tree >\(&constant::do_print_tree\)](#), [print_func< print_python_repr >\(&constant::do_print_python_repr\)](#)) [constant](#)
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([constant](#))

Variables

- [const constant GiNaC::Pi](#) ("Pi", [PiEvalf](#), "\\pi", [domain::positive](#))
Pi.
- [const constant GiNaC::Euler](#) ("Euler", [EulerEvalf](#), "\\gamma_E", [domain::positive](#))
Euler's constant.
- [const constant GiNaC::Catalan](#) ("Catalan", [CatalanEvalf](#), "G", [domain::positive](#))
Catalan's constant.

7.14.1 Detailed Description

Implementation of [GiNaC](#)'s constant types and some special constants.

7.15 constant.h File Reference

Interface to [GiNaC](#)'s constant types and some special constants.

```
#include "basic.h"
#include "ex.h"
#include "archive.h"
#include <string>
```

Classes

- class [GiNaC::constant](#)
This class holds constants, symbols with specific numerical value.

Namespaces

- namespace [GiNaC](#)

Typedefs

- typedef [ex](#)(* [GiNaC::evalffunctype](#)) ()

Functions

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) (constant)

7.15.1 Detailed Description

Interface to [GiNaC](#)'s constant types and some special constants.

7.16 container.h File Reference

Wrapper template for making [GiNaC](#) classes out of STL containers.

```
#include "ex.h"
#include "print.h"
#include "archive.h"
#include "assertion.h"
#include "compiler.h"
#include <algorithm>
#include <iterator>
#include <list>
#include <memory>
#include <stdexcept>
#include <vector>
```

Classes

- class [GiNaC::container_storage< C >](#)
Helper template for encapsulating the [reserve\(\)](#) mechanics of STL containers.
- class [GiNaC::container< C >](#)
Wrapper template for making [GiNaC](#) classes out of STL containers.

Namespaces

- namespace [GiNaC](#)

7.16.1 Detailed Description

Wrapper template for making [GiNaC](#) classes out of STL containers.

7.17 crc32.h File Reference

CRC32 hash function.

Namespaces

- namespace [GiNaC](#)

Functions

- static unsigned [GiNaC::crc32](#) (const char *[c](#), const unsigned [len](#), const unsigned [crcinit](#))

Variables

- static unsigned const [GiNaC::crctab](#) [256]

7.17.1 Detailed Description

CRC32 hash function.

Shamelessly stolen from GNU coreutils (cksum.c).

7.18 ex.cpp File Reference

Implementation of [GiNaC](#)'s light-weight expression handles.

```
#include "ex.h"
#include "add.h"
#include "mul.h"
#include "ncmul.h"
#include "numeric.h"
#include "matrix.h"
#include "power.h"
#include "lst.h"
#include "relational.h"
#include "utils.h"
#include <stdexcept>
```

Namespaces

- namespace [GiNaC](#)

7.18.1 Detailed Description

Implementation of [GiNaC](#)'s light-weight expression handles.

7.19 ex.h File Reference

Interface to [GiNaC](#)'s light-weight expression handles.

```
#include "basic.h"
#include "ptr.h"
#include "compiler.h"
#include <functional>
#include <iosfwd>
#include <iterator>
#include <memory>
#include <stack>
```

Classes

- class [GiNaC::library_init](#)
Helper class to initialize the library.
- class [GiNaC::ex](#)
Lightweight wrapper for [GiNaC](#)'s symbolic objects.
- class [GiNaC::const_iterator](#)
- struct [GiNaC::internal::_iter_rep](#)
- class [GiNaC::const_preorder_iterator](#)
- class [GiNaC::const_postorder_iterator](#)
- struct [GiNaC::ex_is_less](#)
- struct [GiNaC::ex_is_equal](#)
- struct [GiNaC::op0_is_equal](#)
- struct [GiNaC::ex_swap](#)
- class [GiNaC::pointer_to_map_function](#)
- class [GiNaC::pointer_to_map_function_1arg< T1 >](#)
- class [GiNaC::pointer_to_map_function_2args< T1, T2 >](#)
- class [GiNaC::pointer_to_map_function_3args< T1, T2, T3 >](#)
- class [GiNaC::pointer_to_member_to_map_function< C >](#)
- class [GiNaC::pointer_to_member_to_map_function_1arg< C, T1 >](#)
- class [GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 >](#)
- class [GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >](#)
- struct [std::hash< GiNaC::ex >](#)
Specialization of `std::hash()` for `ex` objects.
- struct [std::equal_to< GiNaC::ex >](#)
Specialization of `std::equal_to()` for `ex` objects.

Namespaces

- namespace [GiNaC](#)
- namespace [GiNaC::internal](#)
- namespace [std](#)

Functions

- `bool GiNaC::are_ex_trivially_equal` (const `ex` &e1, const `ex` &e2)
Compare two objects of class quickly without doing a deep tree traversal.
- `std::ostream & GiNaC::operator<<` (std::ostream &os, const `exvector` &e)
- `std::ostream & GiNaC::operator<<` (std::ostream &os, const `exset` &e)
- `std::ostream & GiNaC::operator<<` (std::ostream &os, const `exmap` &e)
- `size_t GiNaC::nops` (const `ex` &thisex)
- `ex GiNaC::expand` (const `ex` &thisex, unsigned `options`=0)
- `ex GiNaC::conjugate` (const `ex` &thisex)
- `ex GiNaC::real_part` (const `ex` &thisex)
- `ex GiNaC::imag_part` (const `ex` &thisex)
- `bool GiNaC::has` (const `ex` &thisex, const `ex` &pattern, unsigned `options`=0)
- `bool GiNaC::find` (const `ex` &thisex, const `ex` &pattern, `exset` &found)
- `bool GiNaC::is_polynomial` (const `ex` &thisex, const `ex` &vars)
- `int GiNaC::degree` (const `ex` &thisex, const `ex` &s)
- `int GiNaC::ldegree` (const `ex` &thisex, const `ex` &s)
- `ex GiNaC::coeff` (const `ex` &thisex, const `ex` &s, int `n`=1)
- `ex GiNaC::numer` (const `ex` &thisex)
- `ex GiNaC::denom` (const `ex` &thisex)
- `ex GiNaC::numer_denom` (const `ex` &thisex)
- `ex GiNaC::normal` (const `ex` &thisex)
- `ex GiNaC::to_rational` (const `ex` &thisex, `exmap` &repl)
- `ex GiNaC::to_polynomial` (const `ex` &thisex, `exmap` &repl)
- `ex GiNaC::collect` (const `ex` &thisex, const `ex` &s, bool `distributed`=false)
- `ex GiNaC::eval` (const `ex` &thisex)
- `ex GiNaC::evalf` (const `ex` &thisex)
- `ex GiNaC::evalm` (const `ex` &thisex)
- `ex GiNaC::eval_integ` (const `ex` &thisex)
- `ex GiNaC::diff` (const `ex` &thisex, const `symbol` &s, unsigned `nth`=1)
- `ex GiNaC::series` (const `ex` &thisex, const `ex` &r, int `order`, unsigned `options`=0)
- `bool GiNaC::match` (const `ex` &thisex, const `ex` &pattern, `exmap` &repl_lst)
- `ex GiNaC::simplify_indexed` (const `ex` &thisex, unsigned `options`=0)
- `ex GiNaC::simplify_indexed` (const `ex` &thisex, const `scalar_products` &sp, unsigned `options`=0)
- `ex GiNaC::symmetrize` (const `ex` &thisex)
- `ex GiNaC::symmetrize` (const `ex` &thisex, const `lst` &l)
- `ex GiNaC::antisymmetrize` (const `ex` &thisex)
- `ex GiNaC::antisymmetrize` (const `ex` &thisex, const `lst` &l)
- `ex GiNaC::symmetrize_cyclic` (const `ex` &thisex)
- `ex GiNaC::symmetrize_cyclic` (const `ex` &thisex, const `lst` &l)
- `ex GiNaC::op` (const `ex` &thisex, `size_t` i)
- `ex GiNaC::lhs` (const `ex` &thisex)
- `ex GiNaC::rhs` (const `ex` &thisex)
- `bool GiNaC::is_zero` (const `ex` &thisex)
- `void GiNaC::swap` (`ex` &e1, `ex` &e2)
- `ex GiNaC::subs` (const `ex` &thisex, const `exmap` &m, unsigned `options`=0)
- `ex GiNaC::subs` (const `ex` &thisex, const `lst` &ls, const `lst` &lr, unsigned `options`=0)
- `ex GiNaC::subs` (const `ex` &thisex, const `ex` &e, unsigned `options`=0)
- `template<class T >`
`bool GiNaC::is_a` (const `ex` &obj)
Check if ex is a handle to a T, including base classes.
- `template<class T >`
`bool GiNaC::is_exactly_a` (const `ex` &obj)
Check if ex is a handle to a T, not including base classes.

- `template<class T >`
`attribute_pure` `const T & GiNaC::ex_to (const ex &e)`
Return a reference to the basic-derived class T object embedded in an expression.
- `template<> void std::swap (GiNaC::ex &a, GiNaC::ex &b)`
Specialization of `std::swap()` for ex objects.

Variables

- static `library_init` `GiNaC::library_initializer`
For construction of flyweights, etc.
- const `basic * GiNaC::_num0_bp`

7.19.1 Detailed Description

Interface to [GiNaC](#)'s light-weight expression handles.

7.20 excompiler.cpp File Reference

Functions to facilitate the conversion of a ex to a function pointer suited for fast numerical integration.

```
#include "excompiler.h"
#include "ex.h"
#include "lst.h"
#include "operators.h"
#include "relational.h"
#include "symbol.h"
#include <cstdlib>
#include <fstream>
#include <ios>
#include <sstream>
#include <stdexcept>
#include <string>
#include <vector>
```

Namespaces

- namespace [GiNaC](#)

Functions

- void `GiNaC::compile_ex` (const `ex` &expr, const `symbol` &sym, `FUNC_P_1P` &fp, const std::string filename="")
Takes an expression and produces a function pointer to the compiled and linked C code equivalent in double precision.
- void `GiNaC::compile_ex` (const `ex` &expr, const `symbol` &sym1, const `symbol` &sym2, `FUNC_P_2P` &fp, const std::string filename="")
Takes an expression and produces a function pointer to the compiled and linked C code equivalent in double precision.
- void `GiNaC::compile_ex` (const `lst` &exprs, const `lst` &syms, `FUNC_P_CUBA` &fp, const std::string filename="")
Takes an expression and produces a function pointer to the compiled and linked C code equivalent in double precision.
- void `GiNaC::link_ex` (const std::string filename, `FUNC_P_1P` &fp)
Opens an existing so-file and returns a function pointer of type `FUNC_P_1P` to the contained function.
- void `GiNaC::link_ex` (const std::string filename, `FUNC_P_2P` &fp)
Opens an existing so-file and returns a function pointer of type `FUNC_P_2P` to the contained function.
- void `GiNaC::link_ex` (const std::string filename, `FUNC_P_CUBA` &fp)
Opens an existing so-file and returns a function pointer of type `FUNC_P_CUBA` to the contained function.
- void `GiNaC::unlink_ex` (const std::string filename)
Closes all linked .so files that have the supplied filename.

7.20.1 Detailed Description

Functions to facilitate the conversion of a `ex` to a function pointer suited for fast numerical integration.

7.21 excompiler.h File Reference

Functions to facilitate the conversion of a `ex` to a function pointer suited for fast numerical integration.

```
#include "lst.h"
#include <string>
```

Namespaces

- namespace `GiNaC`

Typedefs

- typedef double(* `GiNaC::FUNC_P_1P`) (double)
Function pointer with one function parameter.
- typedef double(* `GiNaC::FUNC_P_2P`) (double, double)
Function pointer with two function parameters.
- typedef void(* `GiNaC::FUNC_P_CUBA`) (const int *, const double[], const int *, double[])
Function pointer for use with the CUBA library (<http://www.feynarts.de/cuba>).

Functions

- void `GiNaC::compile_ex` (const `ex` &expr, const `symbol` &sym, `FUNC_P_1P` &fp, const std::string filename="")
Takes an expression and produces a function pointer to the compiled and linked C code equivalent in double precision.
- void `GiNaC::compile_ex` (const `ex` &expr, const `symbol` &sym1, const `symbol` &sym2, `FUNC_P_2P` &fp, const std::string filename="")
Takes an expression and produces a function pointer to the compiled and linked C code equivalent in double precision.
- void `GiNaC::compile_ex` (const `lst` &exprs, const `lst` &syms, `FUNC_P_CUBA` &fp, const std::string filename="")
Takes an expression and produces a function pointer to the compiled and linked C code equivalent in double precision.
- void `GiNaC::link_ex` (const std::string filename, `FUNC_P_1P` &fp)
Opens an existing so-file and returns a function pointer of type `FUNC_P_1P` to the contained function.
- void `GiNaC::link_ex` (const std::string filename, `FUNC_P_2P` &fp)
Opens an existing so-file and returns a function pointer of type `FUNC_P_2P` to the contained function.
- void `GiNaC::link_ex` (const std::string filename, `FUNC_P_CUBA` &fp)
Opens an existing so-file and returns a function pointer of type `FUNC_P_CUBA` to the contained function.
- void `GiNaC::unlink_ex` (const std::string filename)
Closes all linked .so files that have the supplied filename.

7.21.1 Detailed Description

Functions to facilitate the conversion of a `ex` to a function pointer suited for fast numerical integration.

7.22 `expair.cpp` File Reference

Implementation of expression pairs (building blocks of `expairseq`).

```
#include "expair.h"
#include "operators.h"
```

Namespaces

- namespace `GiNaC`

7.22.1 Detailed Description

Implementation of expression pairs (building blocks of `expairseq`).

7.23 `expair.h` File Reference

Definition of expression pairs (building blocks of `expairseq`).

```
#include "ex.h"
#include "numeric.h"
#include "print.h"
```

Classes

- class [GiNaC::expair](#)
A pair of expressions.
- struct [GiNaC::expair_is_less](#)
Function object for insertion into third argument of STL's sort() etc.
- struct [GiNaC::expair_rest_is_less](#)
Function object not caring about the numerical coefficients for insertion into third argument of STL's sort().
- struct [GiNaC::expair_swap](#)

Namespaces

- namespace [GiNaC](#)

Functions

- void [GiNaC::swap](#) ([expair](#) &e1, [expair](#) &e2)

7.23.1 Detailed Description

Definition of expression pairs (building blocks of expairseq).

7.24 expairseq.cpp File Reference

Implementation of sequences of expression pairs.

```
#include "expairseq.h"
#include "lst.h"
#include "add.h"
#include "mul.h"
#include "power.h"
#include "relational.h"
#include "wildcard.h"
#include "archive.h"
#include "operators.h"
#include "utils.h"
#include "hash_seed.h"
#include "indexed.h"
#include "compiler.h"
#include <algorithm>
#include <iostream>
#include <iterator>
#include <memory>
#include <stdexcept>
#include <string>
```

Namespaces

- namespace [GiNaC](#)

Functions

- `GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`expairseq`, `basic`, `print_func` < `print_context` >(&`expairseq::do_print`). `print_func` < `print_tree` >(&`expairseq::do_print_tree`)) `class` `epp_is_less`
- `epvector * GiNaC::conjugateepvector` (`const epvector &`)
Complex conjugate every element of an epvector.

7.24.1 Detailed Description

Implementation of sequences of expression pairs.

7.25 expairseq.h File Reference

Interface to sequences of expression pairs.

```
#include "expair.h"
#include "indexed.h"
#include <vector>
```

Classes

- `class GiNaC::expairseq`
A sequence of class expair.
- `class GiNaC::make_flat_inserter`
Class to handle the renaming of dummy indices.

Namespaces

- namespace `GiNaC`

Typedefs

- `typedef std::vector< expair > GiNaC::epvector`
expair-vector
- `typedef epvector::iterator GiNaC::epp`
expair-vector pointer

Functions

- `epvector * GiNaC::conjugateepvector` (`const epvector &`)
Complex conjugate every element of an epvector.

7.25.1 Detailed Description

Interface to sequences of expression pairs.

7.26 exprseq.cpp File Reference

Implementation of [GiNaC](#)'s exprseq.

```
#include "exprseq.h"
```

Namespaces

- namespace [GiNaC](#)

7.26.1 Detailed Description

Implementation of [GiNaC](#)'s exprseq.

7.27 exprseq.h File Reference

Definition of [GiNaC](#)'s exprseq.

```
#include "container.h"  
#include <vector>
```

Namespaces

- namespace [GiNaC](#)

Typedefs

- typedef [container](#)< std::vector > [GiNaC::exprseq](#)

7.27.1 Detailed Description

Definition of [GiNaC](#)'s exprseq.

7.28 factor.cpp File Reference

Polynomial factorization (implementation).

```
#include "factor.h"
#include "ex.h"
#include "numeric.h"
#include "operators.h"
#include "inifcns.h"
#include "symbol.h"
#include "relational.h"
#include "power.h"
#include "mul.h"
#include "normal.h"
#include "add.h"
#include <type_traits>
#include <algorithm>
#include <limits>
#include <list>
#include <vector>
#include <stack>
#include <cln/cln.h>
```

Namespaces

- namespace [GiNaC](#)

Macros

- `#define` [DCOUT](#)(str)
- `#define` [DCOUTVAR](#)(var)
- `#define` [DCOUT2](#)(str, var)
- `#define` [USE_SAME_DEGREE_FACTOR](#)

Functions

- `ex` [GiNaC::factor](#) (const `ex` &`poly`, unsigned `options`)

Interface function to the outside world.

7.28.1 Detailed Description

Polynomial factorization (implementation).

The interface function `factor()` at the end of this file is defined in the [GiNaC](#) namespace. All other utility functions and classes are defined in an additional anonymous namespace.

Factorization starts by doing a square free factorization and making the coefficients integer. Then, depending on the number of free variables it proceeds either in dedicated univariate or multivariate factorization code.

Univariate factorization does a modular factorization via Berlekamp's algorithm and distinct degree factorization. Hensel lifting is used at the end.

Multivariate factorization uses the univariate factorization (applying a evaluation homomorphism first) and Hensel lifting raises the answer to the multivariate domain. The Hensel lifting code is completely distinct from the code used by the univariate factorization.

Algorithms used can be found in [Wan] An Improved Multivariate Polynomial Factoring Algorithm, P.S.Wang, Mathematics of Computation, Vol. 32, No. 144 (1978) 1215–1231. [GCL] Algorithms for Computer Algebra, K.O.Geddes, S.R.Czapor, G.Labahn, Springer Verlag, 1992. [Mig] Some Useful Bounds, M.Mignotte, In "Computer Algebra, Symbolic and Algebraic Computation" (B.Buchberger et al., eds.), pp. 259-263, Springer-Verlag, New York, 1982.

7.28.2 Macro Definition Documentation

7.28.2.1 DCOUT

```
#define DCOUT(  
    str )
```

7.28.2.2 DCOUTVAR

```
#define DCOUTVAR(  
    var )
```

7.28.2.3 DCOUT2

```
#define DCOUT2(  
    str,  
    var )
```

7.28.2.4 USE_SAME_DEGREE_FACTOR

```
#define USE_SAME_DEGREE_FACTOR
```

7.28.3 Variable Documentation

7.28.3.1 value

```
const bool value = false [static]
```

Referenced by [GiNaC::archive_node::add_bool\(\)](#), [GiNaC::archive_node::add_ex\(\)](#), [GiNaC::archive_node::add_string\(\)](#), [GiNaC::archive_node::add_unsigned\(\)](#), [GiNaC::Li2_\(\)](#), [GiNaC::matrix::set\(\)](#), and [GiNaC::subsvalue\(\)](#).

7.28.3.2 r

```
size_t r [private]
```

Referenced by [GiNaC::matrix::charpoly\(\)](#), [GiNaC::collect_common_factors\(\)](#), [GiNaC::ex::content\(\)](#), [GiNaC::numeric::csgn\(\)](#), [GiNaC::numeric::denom\(\)](#), [GiNaC::matrix::determinant\(\)](#), [GiNaC::matrix::determinant_minor\(\)](#), [GiNaC::divide\(\)](#), [GiNaC::divide_in_z\(\)](#), [GiNaC::matrix::division_free_elimination\(\)](#), [GiNaC::matrix::echelon_form\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::power::expand\(\)](#), [GiNaC::power::expand_add\(\)](#), [GiNaC::power::expand_add_2\(\)](#), [GiNaC::matrix::fraction_free_elimination\(\)](#), [GiNaC::matrix::gauss_elimination\(\)](#), [GiNaC::idx_symmetrization\(\)](#), [GiNaC::matrix::inverse\(\)](#), [GiNaC::iquo\(\)](#), [GiNaC::lsolve\(\)](#), [GiNaC::matrix::markowitz_elimination\(\)](#), [GiNaC::matrix::matrix\(\)](#), [GiNaC::matrix::matrix\(\)](#), [GiNaC::matrix::mul\(\)](#), [GiNaC::matrix::mul_scalar\(\)](#), [GiNaC::numeric::numer\(\)](#), [GiNaC::Order_series\(\)](#), [GiNaC::matrix::pow\(\)](#), [GiNaC::prem\(\)](#), [GiNaC::numeric::print_numeric\(\)](#), [GiNaC::quo\(\)](#), [GiNaC::matrix::rank\(\)](#), [GiNaC::reduced_matrix\(\)](#), [GiNaC::rem\(\)](#), [GiNaC::structure< T, ComparisonPolicy >::return_type_tinfo\(\)](#), [GiNaC::ex::series\(\)](#), [GiNaC::series\(\)](#), [GiNaC::basic::series\(\)](#), [GiNaC::add::series\(\)](#), [GiNaC::fderivative::series\(\)](#), [GiNaC::function::series\(\)](#), [GiNaC::integral::series\(\)](#), [GiNaC::integration_kernel::series\(\)](#), [GiNaC::Eisenstein_kernel::series\(\)](#), [GiNaC::modular_form_kernel::series\(\)](#), [GiNaC::pseries::series\(\)](#), [GiNaC::structure< T, ComparisonPolicy >::series\(\)](#), [GiNaC::mul::series\(\)](#), [GiNaC::power::series\(\)](#), [GiNaC::symbol::series\(\)](#), [GiNaC::refcounted::set_refcount\(\)](#), [GiNaC::matrix::solve\(\)](#), [GiNaC::sprem\(\)](#), [GiNaC::sqrfree_parfrac\(\)](#), [GiNaC::sr_gcd\(\)](#), [GiNaC::numeric::step\(\)](#), [GiNaC::sub_matrix\(\)](#), [GiNaC::ex::subs\(\)](#), [GiNaC::matrix::subs\(\)](#), [GiNaC::symbolic_matrix\(\)](#), [GiNaC::symbolic_matrix\(\)](#), [GiNaC::matrix::trace\(\)](#), [GiNaC::matrix::transpose\(\)](#), [GiNaC::unit_matrix\(\)](#), and [GiNaC::zeta1_evalf\(\)](#).

7.28.3.3 c

```
size_t c [private]
```

Referenced by [GiNaC::abs_print_csrc_float\(\)](#), [GiNaC::abs_print_latex\(\)](#), [GiNaC::symmetry::add\(\)](#), [GiNaC::bernoulli\(\)](#), [GiNaC::matrix::charpoly\(\)](#), [GiNaC::mul::coeff\(\)](#), [GiNaC::ncmul::coeff\(\)](#), [GiNaC::color_d\(\)](#), [GiNaC::color_f\(\)](#), [GiNaC::color_h\(\)](#), [GiNaC::expairseq::combine_ex_with_coeff_to_pair\(\)](#), [GiNaC::add::combine_ex_with_coeff_to_pair\(\)](#), [GiNaC::mul::combine_ex_with_coeff_to_pair\(\)](#), [GiNaC::expairseq::combine_overall_coeff\(\)](#), [GiNaC::mul::combine_overall_coeff\(\)](#), [GiNaC::expairseq::combine_pair_with_coeff_to_pair\(\)](#), [GiNaC::add::combine_pair_with_coeff_to_pair\(\)](#), [GiNaC::mul::combine_pair_v](#), [GiNaC::mul::conjugate\(\)](#), [GiNaC::conjugate_print_latex\(\)](#), [GiNaC::ex::content\(\)](#), [GiNaC::crc32\(\)](#), [GiNaC::pseries::derivative\(\)](#), [GiNaC::matrix::determinant\(\)](#), [GiNaC::matrix::determinant_minor\(\)](#), [GiNaC::divide_in_z\(\)](#), [GiNaC::matrix::division_free_elimination\(\)](#), [GiNaC::add::do_print\(\)](#), [GiNaC::basic::do_print\(\)](#), [GiNaC::constant::do_print\(\)](#), [GiNaC::container< C >::do_print\(\)](#), [GiNaC::expairseq::do_print\(\)](#), [GiNaC::fderivative::do_print\(\)](#), [GiNaC::Gt::do_print\(\)](#), [GiNaC::idx::do_print\(\)](#), [GiNaC::varidx::do_print\(\)](#), [GiNaC::spinidx::do_print\(\)](#), [GiNaC::indexed::do_print\(\)](#), [GiNaC::integral::do_print\(\)](#), [GiNaC::integration_kernel::do_print\(\)](#), [GiNaC::basic_log_kernel::do_print\(\)](#), [GiNaC::multiple_polylog_kernel::do_print\(\)](#), [GiNaC::ELi_kernel::do_print\(\)](#), [GiNaC::Ebar_kernel::do_print\(\)](#), [GiNaC::Kronecker_dtau_kernel::do_print\(\)](#), [GiNaC::Kronecker_dz_kernel::do_print\(\)](#), [GiNaC::Eisenstein_kernel::do_print\(\)](#), [GiNaC::Eisenstein_h_kernel::do_print\(\)](#), [GiNaC::modular_form_kernel::do_print\(\)](#), [GiNaC::user_defined_kernel::do_print\(\)](#), [GiNaC::matrix::do_print\(\)](#), [GiNaC::mul::do_print\(\)](#), [GiNaC::ncmul::do_print\(\)](#), [GiNaC::numeric::do_print\(\)](#), [GiNaC::pseries::do_print\(\)](#), [GiNaC::relational::do_print\(\)](#), [GiNaC::symbol::do_print\(\)](#), [GiNaC::symmetry::do_print\(\)](#), [GiNaC::wildcard::do_print\(\)](#), [GiNaC::ncmul::do_print_csrc\(\)](#), [GiNaC::add::do_print_csrc\(\)](#), [GiNaC::fderivative::do_print_csrc\(\)](#), [GiNaC::idx::do_print_csrc\(\)](#), [GiNaC::mul::do_print_csrc\(\)](#), [GiNaC::numeric::do_print_csrc\(\)](#), [GiNaC::power::do_print_csrc\(\)](#), [GiNaC::numeric::do_print_csrc_cl_N\(\)](#), [GiNaC::power::do_print_csrc_cl_N\(\)](#), [GiNaC::clifford::do_print_dflt\(\)](#), [GiNaC::power::do_print_dflt\(\)](#), [GiNaC::fderivative::do_print_latex\(\)](#), [GiNaC::add::do_print_latex\(\)](#), [GiNaC::clifford::do_print_latex\(\)](#), [GiNaC::constant::do_print_latex\(\)](#), [GiNaC::Gt::do_print_latex\(\)](#), [GiNaC::idx::do_print_latex\(\)](#), [GiNaC::spinidx::do_print_latex\(\)](#), [GiNaC::indexed::do_print_latex\(\)](#), [GiNaC::integral::do_print_latex\(\)](#), [GiNaC::matrix::do_print_latex\(\)](#), [GiNaC::mul::do_print_latex\(\)](#), [GiNaC::numeric::do_print_latex\(\)](#), [GiNaC::power::do_print_latex\(\)](#), [GiNaC::pseries::do_print_latex\(\)](#), [GiNaC::symbol::do_print_latex\(\)](#), [GiNaC::container< C >::do_print_python\(\)](#), [GiNaC::power::do_print_python\(\)](#), [GiNaC::pseries::do_print_python\(\)](#), [GiNaC::add::do_print_python_repr\(\)](#), [GiNaC::basic::do_print_python_repr\(\)](#), [GiNaC::constant::do_print_python_repr\(\)](#), [GiNaC::container< C >::do_print_python_repr\(\)](#), [GiNaC::matrix::do_print_python_repr\(\)](#), [GiNaC::mul::do_print_python_repr\(\)](#), [GiNaC::numeric::do_print_python_repr\(\)](#), [GiNaC::power::do_print_python_repr\(\)](#), [GiNaC::pseries::do_print_python_repr\(\)](#), [GiNaC::relational::do_print_python_repr\(\)](#), [GiNaC::symbol::do_print_python_repr\(\)](#), [GiNaC::wildcard::do_print_python_repr\(\)](#), [GiNaC::basic::do_print_tree\(\)](#), [GiNaC::clifford::do_print_tree\(\)](#), [GiNaC::constant::do_print_tree\(\)](#), [GiNaC::container< C >::do_print_tree\(\)](#), [GiNaC::expairseq::do_print_tree\(\)](#), [GiNaC::fderivative::do_print_tree\(\)](#), [GiNaC::idx::do_print_tree\(\)](#), [GiNaC::varidx::do_print_tree\(\)](#), [GiNaC::spinidx::do_print_tree\(\)](#), [GiNaC::indexed::do_print_tree\(\)](#), [GiNaC::numeric::do_print_tree\(\)](#), [GiNaC::pseries::do_print_tree\(\)](#), [GiNaC::symbol::do_print_tree\(\)](#), [GiNaC::symmetry::do_print_tree\(\)](#), [GiNaC::wildcard::do_print_tree\(\)](#), [GiNaC::matrix::echelon_form\(\)](#), [GiNaC::EllipticE_print_latex\(\)](#), [GiNaC::EllipticK_print_latex\(\)](#), [GiNaC::mul::eval\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::power::expand_add\(\)](#), [GiNaC::power::expand_add_2\(\)](#), [GiNaC::factorial_print_dflt_latex\(\)](#), [GiNaC::matrix::fraction_free_elimination\(\)](#), [GiNaC::matrix::gauss_elimination\(\)](#), [GiNaC::H_print_latex\(\)](#), [GiNaC::power::imag_part\(\)](#), [GiNaC::imag_part_print_latex\(\)](#), [GiNaC::add::integer_content\(\)](#), [GiNaC::matrix::inverse\(\)](#), [GiNaC::lcmcoeff\(\)](#), [GiNaC::Li_print_latex\(\)](#), [GiNaC::lsolve\(\)](#), [GiNaC::matrix::markowitz_elimination\(\)](#), [GiNaC::matrix::matrix\(\)](#), [GiNaC::matrix::matrix\(\)](#), [GiNaC::matrix::mul\(\)](#), [GiNaC::matrix::mul\(\)](#), [GiNaC::matrix::mul_scalar\(\)](#), [GiNaC::print_functor::operator\(\)](#), [GiNaC::print_ptrfun_handler< T, C >::operator\(\)](#), [GiNaC::print_memfun_handler< T, C >::operator\(\)](#), [GiNaC::error_and_integral_is_less::operator\(\)](#), [GiNaC::matrix::pivot\(\)](#), [GiNaC::pseries::power_const\(\)](#), [GiNaC::ex::primpart\(\)](#), [GiNaC::ex::primpart\(\)](#), [GiNaC::basic::print\(\)](#), [GiNaC::ex::print\(\)](#), [GiNaC::fderivative::print\(\)](#), [GiNaC::function::print\(\)](#), [GiNaC::structure< T, ComparisonPolicy >::print\(\)](#), [GiNaC::expair::print\(\)](#), [GiNaC::add::print_add\(\)](#), [GiNaC::basic::print_dispatch\(\)](#), [GiNaC::basic::print_dispatch\(\)](#), [GiNaC::matrix::print_elements\(\)](#), [GiNaC::idx::print_index\(\)](#), [GiNaC::indexed::print_indexed\(\)](#), [GiNaC::print_integer_csrc\(\)](#), [GiNaC::numeric::print_numeric\(\)](#), [GiNaC::print_operator\(\)](#), [GiNaC::mul::print_overall_coeff\(\)](#), [GiNaC::power::print_power\(\)](#), [GiNaC::print_real_cl_N\(\)](#), [GiNaC::print_real_csrc\(\)](#), [GiNaC::print_real_number\(\)](#), [GiNaC::pseries::print_series\(\)](#), [GiNaC::print_sym_pow\(\)](#), [GiNaC::indexed::printindices\(\)](#), [GiNaC::expairseq::printpair\(\)](#), [GiNaC::expairseq::printseq\(\)](#), [GiNaC::container< C >::printseq\(\)](#), [GiNaC::numeric::read_archive\(\)](#), [GiNaC::power::real_part\(\)](#), [GiNaC::real_part_print_latex\(\)](#), [GiNaC::reduced_matrix\(\)](#), [GiNaC::S_print_latex\(\)](#), [GiNaC::set_print_context\(\)](#), [GiNaC::matrix::solve\(\)](#), [GiNaC::sr_gcd\(\)](#), [GiNaC::sub_matrix\(\)](#), [GiNaC::clifford::subs\(\)](#), [GiNaC::matrix::subs\(\)](#), [GiNaC::symbolic_matrix\(\)](#), [GiNaC::symbolic_matrix\(\)](#), [GiNaC::Gt::tauToFundamental\(\)](#), [GiNaC::matrix::transpose\(\)](#), [GiNaC::ex::unit\(\)](#), [GiNaC::unit_matrix\(\)](#), [GiNaC::ex::unitcontprim\(\)](#), [GiNaC::zeta1_print_latex\(\)](#), and [GiNaC::zeta2_print_latex\(\)](#).

7.28.3.4 m

```
mvec m [private]
```

Referenced by [GiNaC::mul::algebraic_subs_mul\(\)](#), [GiNaC::charpoly\(\)](#), [GiNaC::Eisenstein_h_kernel::coefficient_an\(\)](#), [GiNaC::cols\(\)](#), [GiNaC::convert_H_to_Li\(\)](#), [GiNaC::determinant\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::tensdelta::eval_indexed\(\)](#), [GiNaC::tensmetric::eval_indexed\(\)](#), [GiNaC::evalf\(\)](#), [GiNaC::add::evalm\(\)](#), [GiNaC::mul::evalm\(\)](#), [GiNaC::expand\(\)](#), [GiNaC::power::expand\(\)](#), [GiNaC::power::expand_mul\(\)](#), [GiNaC::fibonacci\(\)](#), [GiNaC::H_deriv\(\)](#), [GiNaC::H_eval\(\)](#), [GiNaC::H_evalf\(\)](#), [GiNaC::H_print_latex\(\)](#), [GiNaC::H_series\(\)](#), [GiNaC::Gt_detail::integer_partition\(\)](#), [GiNaC::inverse\(\)](#), [GiNaC::inverse\(\)](#), [GiNaC::lgamma_series\(\)](#), [GiNaC::Li_deriv\(\)](#), [GiNaC::Li_eval\(\)](#), [GiNaC::Li_evalf\(\)](#), [GiNaC::Li_print_latex\(\)](#), [GiNaC::Li_series\(\)](#), [GiNaC::nops\(\)](#), [GiNaC::Order_eval\(\)](#), [GiNaC::psi1_series\(\)](#), [GiNaC::psi2_eval\(\)](#), [GiNaC::psi2_series\(\)](#), [GiNaC::rank\(\)](#), [GiNaC::rank\(\)](#), [GiNaC::reduced_matrix\(\)](#), [GiNaC::resultant\(\)](#), [GiNaC::rows\(\)](#), [GiNaC::S_eval\(\)](#), [GiNaC::smod\(\)](#), [GiNaC::sub_matrix\(\)](#), [GiNaC::ex::subs\(\)](#), [GiNaC::subs\(\)](#), [GiNaC::basic::subs\(\)](#), [GiNaC::ex::subs\(\)](#), [GiNaC::clifford::subs\(\)](#), [GiNaC::container< C >::subs\(\)](#), [GiNaC::expairseq::subs\(\)](#), [GiNaC::Gt::subs\(\)](#), [GiNaC::idx::subs\(\)](#), [GiNaC::numeric::subs\(\)](#), [GiNaC::power::subs\(\)](#), [GiNaC::pseries::subs\(\)](#), [GiNaC::relational::subs\(\)](#), [GiNaC::structure< T, ComparisonF >::subs\(\)](#), [GiNaC::symbol::subs\(\)](#), [GiNaC::ex::subs\(\)](#), [GiNaC::basic::subs_one_level\(\)](#), [GiNaC::container< C >::subchildren\(\)](#), [GiNaC::expairseq::subchildren\(\)](#), [GiNaC::tgamma_series\(\)](#), [GiNaC::trace\(\)](#), [GiNaC::transpose\(\)](#), [GiNaC::zeta1_deriv\(\)](#), [GiNaC::zeta1_eval\(\)](#), [GiNaC::zeta1_print_latex\(\)](#), [GiNaC::zeta2_deriv\(\)](#), [GiNaC::zeta2_eval\(\)](#), and [GiNaC::zeta2_print_latex\(\)](#).

7.28.3.5 lr

```
umodpoly lr[2] [private]
```

Referenced by [GiNaC::subs\(\)](#), and [GiNaC::ex::subs\(\)](#).

7.28.3.6 cache

```
vector<vector<umodpoly> > cache [private]
```

Referenced by [GiNaC::Gt_detail::TransformExpressionWithCache< Type >::impl\(\)](#), and [GiNaC::Gt_detail::TransformExpressionWithCache< Type >::impl\(\)](#).

7.28.3.7 factors

```
upvec factors [private]
```

Referenced by [GiNaC::ncmul::count_factors\(\)](#), [GiNaC::Gt_detail::deconcatenate_path\(\)](#), [GiNaC::ncmul::eval\(\)](#), [GiNaC::mul::expand\(\)](#), [GiNaC::mul::mul\(\)](#), [GiNaC::sqrfree\(\)](#), and [GiNaC::sqrfree_yun\(\)](#).

7.28.3.8 one

```
umodpoly one [private]
```

Referenced by [GiNaC::integration_kernel::get_numerical_value_impl\(\)](#), and [GiNaC::iterated_integral_evalf_impl\(\)](#).

7.28.3.9 n

size_t n [private]

Referenced by [GiNaC::class_info< OPT >::tree_node::add_child\(\)](#), [GiNaC::archive::add_node\(\)](#), [GiNaC::basic::archive\(\)](#), [GiNaC::clifford::archive\(\)](#), [GiNaC::color::archive\(\)](#), [GiNaC::constant::archive\(\)](#), [GiNaC::container< C >::archive\(\)](#), [GiNaC::expairseq::archive\(\)](#), [GiNaC::fderivative::archive\(\)](#), [GiNaC::function::archive\(\)](#), [GiNaC::Gt::archive\(\)](#), [GiNaC::idx::archive\(\)](#), [GiNaC::varidx::archive\(\)](#), [GiNaC::spinidx::archive\(\)](#), [GiNaC::indexed::archive\(\)](#), [GiNaC::integral::archive\(\)](#), [GiNaC::matrix::archive\(\)](#), [GiNaC::numeric::archive\(\)](#), [GiNaC::power::archive\(\)](#), [GiNaC::pseries::archive\(\)](#), [GiNaC::relational::archive\(\)](#), [GiNaC::symbol::archive\(\)](#), [GiNaC::symmetry::archive\(\)](#), [GiNaC::minkmetric::archive\(\)](#), [GiNaC::tensepsilon::archive\(\)](#), [GiNaC::wildcard::archive\(\)](#), [GiNaC::archive::archive\(\)](#), [GiNaC::bernoulli\(\)](#), [GiNaC::binomial\(\)](#), [GiNaC::basic::coeff\(\)](#), [GiNaC::ex::coeff\(\)](#), [GiNaC::add::coeff\(\)](#), [GiNaC::mul::coeff\(\)](#), [GiNaC::ncmul::coeff\(\)](#), [GiNaC::numeric::coeff\(\)](#), [GiNaC::power::coeff\(\)](#), [GiNaC::pseries::coeff\(\)](#), [GiNaC::structure< T, ComparisonPolicy >::coeff\(\)](#), [GiNaC::coeff\(\)](#), [GiNaC::Eisenstein_h_kernel::coefficient_an\(\)](#), [GiNaC::basic::collect\(\)](#), [GiNaC::const_postorder_iterator::const_postorder_iterator\(\)](#), [GiNaC::const_preorder_iterator::const_preorder_iterator\(\)](#), [GiNaC::composition_generator::coolmulti::coolmulti\(\)](#), [GiNaC::matrix::determinant_minor\(\)](#), [GiNaC::dirichlet_character\(\)](#), [GiNaC::matrix::division_free_elimination\(\)](#), [GiNaC::doublefactorial\(\)](#), [GiNaC::class_info< OPT >::dump_tree\(\)](#), [GiNaC::matrix::echelon_form\(\)](#), [GiNaC::power::eval\(\)](#), [GiNaC::mul::expand\(\)](#), [GiNaC::power::expand_add\(\)](#), [GiNaC::power::expand_mul\(\)](#), [GiNaC::factorial\(\)](#), [GiNaC::fibonacci\(\)](#), [GiNaC::frac_cancel\(\)](#), [GiNaC::matrix::fraction_free_elimination\(\)](#), [GiNaC::function_options::function_options\(\)](#), [GiNaC::function_options::function_options\(\)](#), [GiNaC::matrix::gauss_elimination\(\)](#), [GiNaC::gcd\(\)](#), [GiNaC::golden_ratio_hash\(\)](#), [GiNaC::H_eval\(\)](#), [GiNaC::ifactor\(\)](#), [GiNaC::power::imag_part\(\)](#), [GiNaC::Gt_detail::integer_partition\(\)](#), [GiNaC::is_discriminant_of_quad\(\)](#), [GiNaC::kronecker_symbol\(\)](#), [GiNaC::integration_kernel::Laurent_series\(\)](#), [GiNaC::log2\(\)](#), [GiNaC::log_series\(\)](#), [GiNaC::basic::map\(\)](#), [GiNaC::matrix::markowitz_elimination\(\)](#), [GiNaC::multinomial_coefficient\(\)](#), [GiNaC::add::normal\(\)](#), [GiNaC::mul::normal\(\)](#), [GiNaC::pseries::normal\(\)](#), [GiNaC::const_iterator::operator+\(\)](#), [GiNaC::const_iterator::operator+=\(\(\)\)](#), [GiNaC::const_iterator::operator-\(\)](#), [GiNaC::const_iterator::operator-=\(\)](#), [GiNaC::const_iterator::operator\[\]\(\)](#), [GiNaC::primitive_dirichlet_character\(\)](#), [GiNaC::Gt_detail::TransformExpressionWithCache< Type >::print_cache\(\)](#), [GiNaC::psi2_deriv\(\)](#), [GiNaC::psi2_eval\(\)](#), [GiNaC::psi2_evalf\(\)](#), [GiNaC::psi2_series\(\)](#), [GiNaC::clifford::read_archive\(\)](#), [GiNaC::color::read_archive\(\)](#), [GiNaC::container< C >::read_archive\(\)](#), [GiNaC::constant::read_archive\(\)](#), [GiNaC::expairseq::read_archive\(\)](#), [GiNaC::fderivative::read_archive\(\)](#), [GiNaC::function::read_archive\(\)](#), [GiNaC::Gt::read_archive\(\)](#), [GiNaC::idx::read_archive\(\)](#), [GiNaC::varidx::read_archive\(\)](#), [GiNaC::spinidx::read_archive\(\)](#), [GiNaC::indexed::read_archive\(\)](#), [GiNaC::integral::read_archive\(\)](#), [GiNaC::matrix::read_archive\(\)](#), [GiNaC::numeric::read_archive\(\)](#), [GiNaC::power::read_archive\(\)](#), [GiNaC::pseries::read_archive\(\)](#), [GiNaC::relational::read_archive\(\)](#), [GiNaC::symbol::read_archive\(\)](#), [GiNaC::symmetry::read_archive\(\)](#), [GiNaC::minkmetric::read_archive\(\)](#), [GiNaC::tensepsilon::read_archive\(\)](#), [GiNaC::wildcard::read_archive\(\)](#), [GiNaC::power::real_part\(\)](#), [GiNaC::Gt::regularise\(\)](#), [GiNaC::container_storage< C >::reserve\(\)](#), [GiNaC::container_storage< C >::reserve\(\)](#), [GiNaC::rotate_left\(\)](#), [GiNaC::S_deriv\(\)](#), [GiNaC::S_eval\(\)](#), [GiNaC::S_evalf\(\)](#), [GiNaC::S_print_latex\(\)](#), [GiNaC::S_series\(\)](#), [GiNaC::basic::series\(\)](#), [GiNaC::symbol::set_name\(\)](#), [GiNaC::function_options::set_name\(\)](#), [GiNaC::symbol::set_TeX_name\(\)](#), [GiNaC::matrix::solve\(\)](#), [GiNaC::sqfree_parfrac\(\)](#), [GiNaC::function_options::test_and_set_nparams\(\)](#), [GiNaC::tgamma_eval\(\)](#), [GiNaC::Gt::to_integer\(\)](#), [GiNaC::ex::traverse_postorder\(\)](#), [GiNaC::ex::traverse_preorder\(\)](#), [GiNaC::symmetry::validate\(\)](#), [GiNaC::write_real_float\(\)](#), [GiNaC::zetaderiv_deriv\(\)](#), and [GiNaC::zetaderiv_eval\(\)](#).

7.28.3.10 len

size_t len [private]

Referenced by [GiNaC::crc32\(\)](#).

7.28.3.11 last

size_t last [private]

Referenced by [GiNaC::antisymmetrize\(\)](#), [GiNaC::expairseq::combine_same_terms_sorted_seq\(\)](#), [GiNaC::expairseq::construct_from\(\)](#), [GiNaC::cyclic_permutation\(\)](#), [GiNaC::mul::eval\(\)](#), [GiNaC::expairseq::evalchildren\(\)](#), [GiNaC::power::expand_add_2\(\)](#), [GiNaC::expairseq::expandchildren\(\)](#), [GiNaC::mul::expandchildren\(\)](#), [GiNaC::find_free_and_dummy\(\)](#), [GiNaC::permutation_sign\(\)](#), [GiNaC::permutation_sign\(\)](#), [GiNaC::shaker_sort\(\)](#), [GiNaC::expairseq::subschildren\(\)](#), [GiNaC::symm\(\)](#), [GiNaC::symmetrize\(\)](#), and [GiNaC::symmetrize_cyclic\(\)](#).

7.28.3.12 k

```
vector<int> k [private]
```

Referenced by [GiNaC::Gt::applyIntegrationPath\(\)](#), [GiNaC::bernoulli\(\)](#), [GiNaC::Bernoulli_polynomial\(\)](#), [GiNaC::binomial\(\)](#), [GiNaC::Gt::calchash\(\)](#), [GiNaC::scalar_products::debugprint\(\)](#), [GiNaC::Gt::decomposeIntegrationPath\(\)](#), [GiNaC::divide_in_z\(\)](#), [GiNaC::EllipticE_deriv\(\)](#), [GiNaC::EllipticE_eval\(\)](#), [GiNaC::EllipticE_evalf\(\)](#), [GiNaC::EllipticE_print_latex\(\)](#), [GiNaC::EllipticE_series\(\)](#), [GiNaC::EllipticK_deriv\(\)](#), [GiNaC::EllipticK_eval\(\)](#), [GiNaC::EllipticK_evalf\(\)](#), [GiNaC::EllipticK_print_latex\(\)](#), [GiNaC::EllipticK_series\(\)](#), [GiNaC::Gt::evalf\(\)](#), [GiNaC::ncmul::expand\(\)](#), [GiNaC::power::expand_add\(\)](#), [GiNaC::find_common_factor\(\)](#), [GiNaC::generalised_Bernoulli_number\(\)](#), [GiNaC::multi_iterator_permutation< T >::get_sign\(\)](#), [GiNaC::Gt::Gt\(\)](#), [GiNaC::Gt::has\(\)](#), [GiNaC::Gt_detail::integer_partition\(\)](#), [GiNaC::log2\(\)](#), [GiNaC::matrix::markowitz_elimination\(\)](#), [GiNaC::basic_partition_generator::mpartition2::mpartition2\(\)](#), [GiNaC::basic_partition_generator::mpartition2::next_partition\(\)](#), [GiNaC::composition_generator::coolmulti::next_permutation\(\)](#), [GiNaC::multi_iterator_ordered< T >::operator++\(\)](#), [GiNaC::multi_iterator_ordered_eq< T >::operator++\(\)](#), [GiNaC::multi_iterator_ordered_eq_indv< T >::operator++\(\)](#), [GiNaC::multi_iterator_counter< T >::operator++\(\)](#), [GiNaC::multi_iterator_counter_indv< T >::operator++\(\)](#), [GiNaC::multi_iterator_permutation< T >::operator++\(\)](#), [GiNaC::multi_iterator_shuffle< T >::operator++\(\)](#), [GiNaC::matrix::pivot\(\)](#), [GiNaC::Gt::qExpand\(\)](#), [GiNaC::Gt::regularise\(\)](#), [GiNaC::resultant\(\)](#), [GiNaC::ELi_kernel::series_coeff_impl\(\)](#), [GiNaC::Ebar_kernel::series_coeff_impl\(\)](#), [GiNaC::sqrfree_parfrac\(\)](#), [GiNaC::Gt::subs\(\)](#), [GiNaC::Gt::tauToFundamental\(\)](#), and [GiNaC::Gt::zisToFundamental\(\)](#).

7.28.3.13 poly

```
const ex poly
```

Referenced by [GiNaC::matrix::charpoly\(\)](#), and [GiNaC::factor\(\)](#).

7.28.3.14 x

```
const ex x
```

Referenced by [GiNaC::abs\(\)](#), [GiNaC::acos\(\)](#), [GiNaC::acos_conjugate\(\)](#), [GiNaC::acos_deriv\(\)](#), [GiNaC::acos_eval\(\)](#), [GiNaC::acos_evalf\(\)](#), [GiNaC::acosh\(\)](#), [GiNaC::acosh_conjugate\(\)](#), [GiNaC::acosh_deriv\(\)](#), [GiNaC::acosh_eval\(\)](#), [GiNaC::acosh_evalf\(\)](#), [GiNaC::adaptivesimpson\(\)](#), [GiNaC::asin\(\)](#), [GiNaC::asin_conjugate\(\)](#), [GiNaC::asin_deriv\(\)](#), [GiNaC::asin_eval\(\)](#), [GiNaC::asin_evalf\(\)](#), [GiNaC::asin_info\(\)](#), [GiNaC::asinh\(\)](#), [GiNaC::asinh_conjugate\(\)](#), [GiNaC::asinh_deriv\(\)](#), [GiNaC::asinh_eval\(\)](#), [GiNaC::asinh_evalf\(\)](#), [GiNaC::atan\(\)](#), [GiNaC::atan2_deriv\(\)](#), [GiNaC::atan2_eval\(\)](#), [GiNaC::atan2_evalf\(\)](#), [GiNaC::atan2_info\(\)](#), [GiNaC::atan_conjugate\(\)](#), [GiNaC::atan_deriv\(\)](#), [GiNaC::atan_eval\(\)](#), [GiNaC::atan_evalf\(\)](#), [GiNaC::atan_info\(\)](#), [GiNaC::atanh\(\)](#), [GiNaC::atanh_conjugate\(\)](#), [GiNaC::atanh_deriv\(\)](#), [GiNaC::atanh_eval\(\)](#), [GiNaC::atanh_evalf\(\)](#), [GiNaC::Bernoulli_polynomial\(\)](#), [GiNaC::beta_deriv\(\)](#), [GiNaC::beta_eval\(\)](#), [GiNaC::beta_evalf\(\)](#), [GiNaC::binomial_conjugate\(\)](#), [GiNaC::binomial_eval\(\)](#), [GiNaC::binomial_evalf\(\)](#), [GiNaC::binomial_real_part\(\)](#), [GiNaC::binomial_sym\(\)](#), [GiNaC::lanczos_coeffs::calc_lanczos_A\(\)](#), [GiNaC::basic::collect\(\)](#), [GiNaC::container< C >::conjugate\(\)](#), [GiNaC::expairseq::conjugate\(\)](#), [GiNaC::matrix::conjugate\(\)](#), [GiNaC::mul::conjugate\(\)](#), [GiNaC::conjugateepvector\(\)](#), [GiNaC::ex::content\(\)](#), [GiNaC::convert_H_to_Li\(\)](#), [GiNaC::cos\(\)](#), [GiNaC::cos_conjugate\(\)](#), [GiNaC::cos_deriv\(\)](#), [GiNaC::cos_eval\(\)](#), [GiNaC::cos_evalf\(\)](#), [GiNaC::cos_imag_part\(\)](#), [GiNaC::cos_real_part\(\)](#), [GiNaC::cosh\(\)](#), [GiNaC::cosh_conjugate\(\)](#), [GiNaC::cosh_deriv\(\)](#), [GiNaC::cosh_eval\(\)](#), [GiNaC::cosh_evalf\(\)](#), [GiNaC::cosh_imag_part\(\)](#), [GiNaC::cosh_real_part\(\)](#), [GiNaC::csgn\(\)](#), [GiNaC::decomp_rational\(\)](#), [GiNaC::denom\(\)](#), [GiNaC::divide\(\)](#), [GiNaC::divide_in_z\(\)](#), [GiNaC::eta_conjugate\(\)](#), [GiNaC::eta_eval\(\)](#), [GiNaC::eta_evalf\(\)](#), [GiNaC::eta_imag_part\(\)](#), [GiNaC::eta_series\(\)](#), [GiNaC::tensepsilon::eval_indexed\(\)](#), [GiNaC::exp\(\)](#), [GiNaC::exp_conjugate\(\)](#), [GiNaC::exp_deriv\(\)](#), [GiNaC::exp_eval\(\)](#), [GiNaC::exp_evalf\(\)](#), [GiNaC::exp_imag_part\(\)](#), [GiNaC::exp_info\(\)](#), [GiNaC::exp_power\(\)](#), [GiNaC::exp_real_part\(\)](#), [GiNaC::factorial_conjugate\(\)](#), [GiNaC::factorial_eval\(\)](#), [GiNaC::factorial_evalf\(\)](#), [GiNaC::factorial_print_dflft_latex\(\)](#), [GiNaC::factorial_real_part\(\)](#), [GiNaC::find_common_factor\(\)](#), [GiNaC::frac_cancel\(\)](#), [GiNaC::fsolve\(\)](#), [GiNaC::G\(\)](#), [GiNaC::G2_eval\(\)](#), [GiNaC::G2_evalf\(\)](#), [GiNaC::G3_eval\(\)](#), [GiNaC::G3_evalf\(\)](#), [GiNaC::gcd\(\)](#), [GiNaC::generalised_Bernoulli_number\(\)](#), [GiNaC::get_first_symbol\(\)](#), [GiNaC::guess_precision\(\)](#), [GiNaC::H_deriv\(\)](#), [GiNaC::H_eval\(\)](#), [GiNaC::H_evalf\(\)](#), [GiNaC::H_print_latex\(\)](#), [GiNaC::H_series\(\)](#), [GiNaC::make_flat_inserter::handle_factor\(\)](#), [GiNaC::hasindex\(\)](#), [GiNaC::haswild\(\)](#), [GiNaC::heur_gcd_z\(\)](#), [GiNaC::imag\(\)](#), [GiNaC::Gt_detail::integer_partition\(\)](#),

[GiNaC::interpolate\(\)](#), [GiNaC::inverse\(\)](#), [GiNaC::is_cinteger\(\)](#), [GiNaC::is_crational\(\)](#), [GiNaC::is_even\(\)](#), [GiNaC::is_integer\(\)](#), [GiNaC::is_negative\(\)](#), [GiNaC::is_nonneg_integer\(\)](#), [GiNaC::is_odd\(\)](#), [GiNaC::is_pos_integer\(\)](#), [GiNaC::is_positive\(\)](#), [GiNaC::is_prime\(\)](#), [GiNaC::is_rational\(\)](#), [GiNaC::is_real\(\)](#), [GiNaC::is_the_function\(\)](#), [GiNaC::is_the_function< G_SERIAL >\(\)](#), [GiNaC::is_the_function< iterated_integral_SERIAL >\(\)](#), [GiNaC::is_the_function< psi_SERIAL >\(\)](#), [GiNaC::is_the_function< zeta_SERIAL >\(\)](#), [GiNaC::is_zero\(\)](#), [GiNaC::isqrt\(\)](#), [GiNaC::Eisenstein_kernel::Laurent_series\(\)](#), [GiNaC::Eisenstein_h_kernel::Laurent_series\(\)](#), [GiNaC::lgamma\(\)](#), [GiNaC::lgamma\(\)](#), [GiNaC::lgamma_conjugate\(\)](#), [GiNaC::lgamma_deriv\(\)](#), [GiNaC::lgamma_eval\(\)](#), [GiNaC::lgamma_evalf\(\)](#), [GiNaC::Li2\(\)](#), [GiNaC::Li2_conjugate\(\)](#), [GiNaC::Li2_deriv\(\)](#), [GiNaC::Li2_eval\(\)](#), [GiNaC::Li2_evalf\(\)](#), [GiNaC::Li2_projection\(\)](#), [GiNaC::Li2_series\(\)](#), [GiNaC::Li2_series\(\)](#), [GiNaC::Li3_eval\(\)](#), [GiNaC::Li_deriv\(\)](#), [GiNaC::Li_eval\(\)](#), [GiNaC::Li_evalf\(\)](#), [GiNaC::Li_print_latex\(\)](#), [GiNaC::Li_series\(\)](#), [GiNaC::log\(\)](#), [GiNaC::log_conjugate\(\)](#), [GiNaC::log_deriv\(\)](#), [GiNaC::log_eval\(\)](#), [GiNaC::log_evalf\(\)](#), [GiNaC::log_imag_part\(\)](#), [GiNaC::log_info\(\)](#), [GiNaC::log_real_part\(\)](#), [GiNaC::make_real_float\(\)](#), [GiNaC::matrix::matrix\(\)](#), [GiNaC::numer\(\)](#), [GiNaC::sym_desc::operator<\(\)](#), [GiNaC::Order_conjugate\(\)](#), [GiNaC::Order_eval\(\)](#), [GiNaC::Order_imag_part\(\)](#), [GiNaC::Order_power\(\)](#), [GiNaC::Order_real_part\(\)](#), [GiNaC::Order_series\(\)](#), [GiNaC::pow\(\)](#), [GiNaC::prem\(\)](#), [GiNaC::ex::primpart\(\)](#), [GiNaC::ex::primpart\(\)](#), [GiNaC::print_integer_csrc\(\)](#), [GiNaC::print_real_cl_N\(\)](#), [GiNaC::print_real_csrc\(\)](#), [GiNaC::print_real_number\(\)](#), [GiNaC::print_sym_pow\(\)](#), [GiNaC::psi1_deriv\(\)](#), [GiNaC::psi1_eval\(\)](#), [GiNaC::psi1_evalf\(\)](#), [GiNaC::psi2_deriv\(\)](#), [GiNaC::psi2_eval\(\)](#), [GiNaC::psi2_evalf\(\)](#), [GiNaC::quo\(\)](#), [GiNaC::read_real_float\(\)](#), [GiNaC::real\(\)](#), [GiNaC::rem\(\)](#), [GiNaC::S_deriv\(\)](#), [GiNaC::S_eval\(\)](#), [GiNaC::S_evalf\(\)](#), [GiNaC::S_print_latex\(\)](#), [GiNaC::S_series\(\)](#), [GiNaC::sin\(\)](#), [GiNaC::sin_conjugate\(\)](#), [GiNaC::sin_deriv\(\)](#), [GiNaC::sin_eval\(\)](#), [GiNaC::sin_evalf\(\)](#), [GiNaC::sin_imag_part\(\)](#), [GiNaC::sin_real_part\(\)](#), [GiNaC::sinh\(\)](#), [GiNaC::sinh_conjugate\(\)](#), [GiNaC::sinh_deriv\(\)](#), [GiNaC::sinh_eval\(\)](#), [GiNaC::sinh_evalf\(\)](#), [GiNaC::sinh_imag_part\(\)](#), [GiNaC::sinh_real_part\(\)](#), [GiNaC::sprem\(\)](#), [GiNaC::sqrfree\(\)](#), [GiNaC::sqrfree_pfrac\(\)](#), [GiNaC::sqrfree_yun\(\)](#), [GiNaC::sqrt\(\)](#), [GiNaC::sr_gcd\(\)](#), [GiNaC::step\(\)](#), [GiNaC::tan\(\)](#), [GiNaC::tan_conjugate\(\)](#), [GiNaC::tan_deriv\(\)](#), [GiNaC::tan_eval\(\)](#), [GiNaC::tan_evalf\(\)](#), [GiNaC::tan_imag_part\(\)](#), [GiNaC::tan_real_part\(\)](#), [GiNaC::tan_series\(\)](#), [GiNaC::tanh\(\)](#), [GiNaC::tanh_conjugate\(\)](#), [GiNaC::tanh_deriv\(\)](#), [GiNaC::tanh_eval\(\)](#), [GiNaC::tanh_evalf\(\)](#), [GiNaC::tanh_imag_part\(\)](#), [GiNaC::tanh_real_part\(\)](#), [GiNaC::tanh_series\(\)](#), [GiNaC::tgamma\(\)](#), [GiNaC::tgamma\(\)](#), [GiNaC::tgamma_conjugate\(\)](#), [GiNaC::tgamma_deriv\(\)](#), [GiNaC::tgamma_eval\(\)](#), [GiNaC::tgamma_evalf\(\)](#), [GiNaC::to_double\(\)](#), [GiNaC::to_int\(\)](#), [GiNaC::to_long\(\)](#), [GiNaC::trig_info\(\)](#), [GiNaC::ex::unit\(\)](#), [GiNaC::unit_matrix\(\)](#), [GiNaC::ex::unitcontprim\(\)](#), [GiNaC::zeta\(\)](#), [GiNaC::zeta1_evalf\(\)](#), [GiNaC::zeta2_evalf\(\)](#), [GiNaC::zetaderiv_deriv\(\)](#), and [GiNaC::zetaderiv_eval\(\)](#).

7.28.3.15 evalpoint

```
int evalpoint
```

7.28.3.16 R

```
cl_modint_ring R
```

7.28.3.17 syms_wox

```
const exset syms_wox
```

7.28.3.18 unit

```
ex unit
```

Referenced by [GiNaC::kronecker_symbol\(\)](#), [GiNaC::ex::primpart\(\)](#), and [GiNaC::ex::unitcontprim\(\)](#).

7.28.3.19 cont

```
ex cont
```

Referenced by [GiNaC::ex::content\(\)](#), [GiNaC::container< C >::imag_part\(\)](#), and [GiNaC::container< C >::real_part\(\)](#).

7.28.3.20 pp

ex pp

7.28.3.21 vn

ex vn

7.28.3.22 vnlst

exvector vnlst

7.28.3.23 modulus

numeric modulus

7.28.3.24 syms

exset syms

Referenced by [GiNaC::lsolve\(\)](#).

7.28.3.25 options

unsigned options

Referenced by [GiNaC::abs_expand\(\)](#), [GiNaC::mul::algebraic_subs_mul\(\)](#), [GiNaC::atan_series\(\)](#), [GiNaC::atanh_series\(\)](#), [GiNaC::beta_series\(\)](#), [GiNaC::csgn_series\(\)](#), [GiNaC::determinant\(\)](#), [GiNaC::exp_expand\(\)](#), [GiNaC::expand\(\)](#), [GiNaC::expand\(\)](#), [GiNaC::basic::expand\(\)](#), [GiNaC::ex::expand\(\)](#), [GiNaC::add::expand\(\)](#), [GiNaC::expairseq::expand\(\)](#), [GiNaC::function::expand\(\)](#), [GiNaC::indexed::expand\(\)](#), [GiNaC::integral::expand\(\)](#), [GiNaC::mul::expand\(\)](#), [GiNaC::ncmul::expand\(\)](#), [GiNaC::power::expand\(\)](#), [GiNaC::pseries::expand\(\)](#), [GiNaC::structure< T, ComparisonPolicy >::expand\(\)](#), [GiNaC::power::expand_add\(\)](#), [GiNaC::power::expand_add_2\(\)](#), [GiNaC::power::expand_mul\(\)](#), [GiNaC::expairseq::expandchildren\(\)](#), [GiNaC::mul::expandchildren\(\)](#), [GiNaC::ncmul::expandchildren\(\)](#), [GiNaC::factor\(\)](#), [GiNaC::gcd\(\)](#), [GiNaC::basic::has\(\)](#), [GiNaC::Gt::has\(\)](#), [GiNaC::mul::has\(\)](#), [GiNaC::power::has\(\)](#), [GiNaC::structure< T, ComparisonPolicy >::has\(\)](#), [GiNaC::ex::has\(\)](#), [GiNaC::has\(\)](#), [GiNaC::lgamma_series\(\)](#), [GiNaC::Li2_series\(\)](#), [GiNaC::log_expand\(\)](#), [GiNaC::log_series\(\)](#), [GiNaC::lsolve\(\)](#), [GiNaC::expand_map_function::operator\(\)](#), [GiNaC::registered_class_options::print_func\(\)](#), [GiNaC::function_options::print_func\(\)](#), [GiNaC::function_options::print_func\(\)](#), [GiNaC::function_options::print_func\(\)](#), [GiNaC::function_options::print_func\(\)](#), [GiNaC::function_options::print_func\(\)](#), [GiNaC::function_options::print_func\(\)](#), [GiNaC::function_options::print_func\(\)](#), [GiNaC::function_options::print_func\(\)](#), [GiNaC::function_options::print_func\(\)](#), [GiNaC::function_options::print_func\(\)](#), [GiNaC::function_options::print_func\(\)](#), [GiNaC::function_options::print_func\(\)](#), [GiNaC::function_options::print_func\(\)](#), [GiNaC::registered_class_options::print_func\(\)](#), [GiNaC::registered_class_options::print_func\(\)](#), [GiNaC::psi1_series\(\)](#), [GiNaC::psi2_series\(\)](#), [GiNaC::S_series\(\)](#), [GiNaC::ex::series\(\)](#), [GiNaC::series\(\)](#), [GiNaC::add::series\(\)](#), [GiNaC::fderivative::series\(\)](#), [GiNaC::function::series\(\)](#), [GiNaC::integral::series\(\)](#), [GiNaC::pseries::series\(\)](#), [GiNaC::structure< T, ComparisonPolicy >::series\(\)](#), [GiNaC::mul::series\(\)](#), [GiNaC::power::series\(\)](#), [GiNaC::set_print_context\(\)](#), [GiNaC::set_print_func\(\)](#), [GiNaC::set_print_func\(\)](#), [GiNaC::set_print_options\(\)](#), [GiNaC::simplify_indexed\(\)](#), [GiNaC::simplify_indexed\(\)](#), [GiNaC::step_series\(\)](#), [GiNaC::ex::subs\(\)](#), [GiNaC::subs\(\)](#), [GiNaC::subs\(\)](#), [GiNaC::subs\(\)](#), [GiNaC::basic::subs\(\)](#), [GiNaC::ex::subs\(\)](#), [GiNaC::clifford::subs\(\)](#), [GiNaC::container< C >::subs\(\)](#), [GiNaC::expairseq::subs\(\)](#), [GiNaC::Gt::subs\(\)](#), [GiNaC::idx::subs\(\)](#), [GiNaC::matrix::subs\(\)](#), [GiNaC::numeric::subs\(\)](#), [GiNaC::power::subs\(\)](#), [GiNaC::pseries::subs\(\)](#), [GiNaC::relational::subs\(\)](#), [GiNaC::structure< T, ComparisonPolicy >::subs\(\)](#), [GiNaC::symbol::subs\(\)](#), [GiNaC::ex::subs\(\)](#), [GiNaC::basic::subs_one_level\(\)](#), [GiNaC::container< C >::subschildren\(\)](#), [GiNaC::expairseq::subschildren\(\)](#), [GiNaC::tan_series\(\)](#), [GiNaC::tanh_series\(\)](#), and [GiNaC::tgamma_series\(\)](#).

7.29 factor.h File Reference

Polynomial factorization.

Namespaces

- namespace [GiNaC](#)

Functions

- [ex](#) [GiNaC::factor](#) (const [ex](#) &[poly](#), unsigned [options](#))
Interface function to the outside world.

7.29.1 Detailed Description

Polynomial factorization.

7.30 fail.cpp File Reference

Implementation of class signaling failure of operation.

```
#include "fail.h"  
#include "archive.h"  
#include "utils.h"
```

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([fail](#), [basic](#), [print_func](#)< [print_context](#)>(&[fail::do_print](#)). [print_func](#)< [print_tree](#)>(&[fail::do_print_tree](#))) [GINAC_BIND_UNARCHIVER](#)([fail](#))

7.30.1 Detailed Description

Implementation of class signaling failure of operation.

Considered somewhat obsolete (most of this can be replaced by exceptions).

7.31 fail.h File Reference

Interface to class signaling failure of operation.

```
#include "basic.h"
#include "archive.h"
```

Classes

- class [GiNaC::fail](#)

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([fail](#))

7.31.1 Detailed Description

Interface to class signaling failure of operation.

Considered obsolete somewhat obsolete (most of this can be replaced by exceptions).

7.32 fderivative.cpp File Reference

Implementation of abstract derivatives of functions.

```
#include "fderivative.h"
#include "operators.h"
#include "archive.h"
#include "utils.h"
```

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([fderivative](#), [function](#), [print_func< print_context >\(&fderivative::do_print\)](#), [print_func< print_latex >\(&fderivative::do_print_latex\)](#), [print_func< print_csrc >\(&fderivative::do_print_csrc\)](#), [print_func< print_tree >\(&fderivative::do_print_tree\)](#)) [fderivative](#)
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([fderivative](#))

7.32.1 Detailed Description

Implementation of abstract derivatives of functions.

7.33 fderivative.h File Reference

Interface to abstract derivatives of functions.

```
#include "function.h"  
#include <set>
```

Classes

- class [GiNaC::fderivative](#)

This class represents the (abstract) derivative of a symbolic function.

Namespaces

- namespace [GiNaC](#)

Typedefs

- typedef std::multiset< unsigned > [GiNaC::paramset](#)

Functions

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([fderivative](#))

7.33.1 Detailed Description

Interface to abstract derivatives of functions.

7.34 flags.h File Reference

Collection of all flags used through the [GiNaC](#) framework.

Classes

- class [GiNaC::expand_options](#)
Flags to control the behavior of `expand()`.
- class [GiNaC::has_options](#)
Flags to control the behavior of `has()`.
- class [GiNaC::subs_options](#)
Flags to control the behavior of `subs()`.
- class [GiNaC::domain](#)
Domain of an object.
- class [GiNaC::series_options](#)
Flags to control series expansion.
- class [GiNaC::determinant_algo](#)
Switch to control algorithm for determinant computation.
- class [GiNaC::solve_algo](#)
Switch to control algorithm for linear system solving.
- class [GiNaC::status_flags](#)
Flags to store information about the state of an object.
- class [GiNaC::info_flags](#)
Possible attributes an object can have.
- class [GiNaC::return_types](#)
- class [GiNaC::remember_strategies](#)
Strategies how to clean up the function remember cache.
- class [GiNaC::factor_options](#)
Flags to control the polynomial factorization.

Namespaces

- namespace [GiNaC](#)

7.34.1 Detailed Description

Collection of all flags used through the [GiNaC](#) framework.

7.35 function.cpp File Reference

Implementation of class of symbolic functions.

```
#include "function.h"
#include "operators.h"
#include "fderivative.h"
#include "ex.h"
#include "lst.h"
#include "symmetry.h"
#include "print.h"
#include "power.h"
#include "archive.h"
#include "inifcns.h"
#include "utils.h"
```

```
#include "hash_seed.h"
#include "remember.h"
#include <iostream>
#include <limits>
#include <list>
#include <stdexcept>
#include <string>
```

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_BIND_UNARCHIVER](#) (function)

7.35.1 Detailed Description

Implementation of class of symbolic functions.

7.36 function.h File Reference

Interface to class of symbolic functions.

```
#include "exprseq.h"
#include <string>
#include <vector>
```

Classes

- class [GiNaC::function_options](#)
- class [GiNaC::do_taylor](#)

Exception class thrown by classes which provide their own series expansion to signal that ordinary Taylor expansion is safe.

- class [GiNaC::function](#)

The class function is used to implement builtin functions like sin, cos... and user defined functions.

Namespaces

- namespace [GiNaC](#)

Macros

- `#define DECLARE_FUNCTION_1P(NAME)`
- `#define DECLARE_FUNCTION_2P(NAME)`
- `#define DECLARE_FUNCTION_3P(NAME)`
- `#define DECLARE_FUNCTION_4P(NAME)`
- `#define DECLARE_FUNCTION_5P(NAME)`
- `#define DECLARE_FUNCTION_6P(NAME)`
- `#define DECLARE_FUNCTION_7P(NAME)`
- `#define DECLARE_FUNCTION_8P(NAME)`
- `#define DECLARE_FUNCTION_9P(NAME)`
- `#define DECLARE_FUNCTION_10P(NAME)`
- `#define DECLARE_FUNCTION_11P(NAME)`
- `#define DECLARE_FUNCTION_12P(NAME)`
- `#define DECLARE_FUNCTION_13P(NAME)`
- `#define DECLARE_FUNCTION_14P(NAME)`
- `#define REGISTER_FUNCTION(NAME, OPT)`
- `#define is_ex_the_function(OBJ, FUNCNAME) (GiNaC::is_the_function<FUNCNAME##_SERIAL>(OBJ))`

Typedefs

- `typedef ex(* GiNaC::eval_funcp) ()`
- `typedef ex(* GiNaC::evalf_funcp) ()`
- `typedef ex(* GiNaC::conjugate_funcp) ()`
- `typedef ex(* GiNaC::real_part_funcp) ()`
- `typedef ex(* GiNaC::imag_part_funcp) ()`
- `typedef ex(* GiNaC::expand_funcp) ()`
- `typedef ex(* GiNaC::derivative_funcp) ()`
- `typedef ex(* GiNaC::expl_derivative_funcp) ()`
- `typedef ex(* GiNaC::power_funcp) ()`
- `typedef ex(* GiNaC::series_funcp) ()`
- `typedef void(* GiNaC::print_funcp) ()`
- `typedef bool(* GiNaC::info_funcp) ()`
- `typedef ex(* GiNaC::eval_funcp_1) (const ex &)`
- `typedef ex(* GiNaC::evalf_funcp_1) (const ex &)`
- `typedef ex(* GiNaC::conjugate_funcp_1) (const ex &)`
- `typedef ex(* GiNaC::real_part_funcp_1) (const ex &)`
- `typedef ex(* GiNaC::imag_part_funcp_1) (const ex &)`
- `typedef ex(* GiNaC::expand_funcp_1) (const ex &, unsigned)`
- `typedef ex(* GiNaC::derivative_funcp_1) (const ex &, unsigned)`
- `typedef ex(* GiNaC::expl_derivative_funcp_1) (const ex &, const symbol &)`
- `typedef ex(* GiNaC::power_funcp_1) (const ex &, const ex &)`
- `typedef ex(* GiNaC::series_funcp_1) (const ex &, const relational &, int, unsigned)`
- `typedef void(* GiNaC::print_funcp_1) (const ex &, const print_context &)`
- `typedef bool(* GiNaC::info_funcp_1) (const ex &, unsigned)`
- `typedef ex(* GiNaC::eval_funcp_2) (const ex &, const ex &)`
- `typedef ex(* GiNaC::evalf_funcp_2) (const ex &, const ex &)`
- `typedef ex(* GiNaC::conjugate_funcp_2) (const ex &, const ex &)`
- `typedef ex(* GiNaC::real_part_funcp_2) (const ex &, const ex &)`
- `typedef ex(* GiNaC::imag_part_funcp_2) (const ex &, const ex &)`
- `typedef ex(* GiNaC::expand_funcp_2) (const ex &, const ex &, unsigned)`
- `typedef ex(* GiNaC::derivative_funcp_2) (const ex &, const ex &, unsigned)`
- `typedef ex(* GiNaC::expl_derivative_funcp_2) (const ex &, const ex &, const symbol &)`
- `typedef ex(* GiNaC::power_funcp_2) (const ex &, const ex &, const ex &)`

- [illegible]

- [illegible]

Functions

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) (function)
- `template<typename T >`
`bool GiNaC::is\_the\_function (const ex &x)`

7.36.1 Detailed Description

Interface to class of symbolic functions.

7.36.2 Macro Definition Documentation

7.36.2.1 DECLARE_FUNCTION_1P

```
#define DECLARE_FUNCTION_1P(  
    NAME )
```

Value:

```
class NAME##_SERIAL { public: static unsigned serial; }; \
const unsigned NAME##_NPARAMS = 1; \
template< typename T1 > const GiNaC::function NAME( const T1 & p1 ) { \
    return GiNaC::function(NAME##_SERIAL::serial, GiNaC::ex(p1) ); \
}
```

7.36.2.2 DECLARE_FUNCTION_2P

```
#define DECLARE_FUNCTION_2P(  
    NAME )
```

Value:

```
class NAME##_SERIAL { public: static unsigned serial; }; \
const unsigned NAME##_NPARAMS = 2; \
template< typename T1, typename T2 > const GiNaC::function NAME( const T1 & p1, const T2 & p2 ) { \
    return GiNaC::function(NAME##_SERIAL::serial, GiNaC::ex(p1), GiNaC::ex(p2) ); \
}
```

7.36.2.3 DECLARE_FUNCTION_3P

```
#define DECLARE_FUNCTION_3P(  
    NAME )
```

Value:

```
class NAME##_SERIAL { public: static unsigned serial; }; \
const unsigned NAME##_NPARAMS = 3; \
template< typename T1, typename T2, typename T3 > const GiNaC::function NAME( const T1 & p1, const T2 & p2, \
    const T3 & p3 ) { \
    return GiNaC::function(NAME##_SERIAL::serial, GiNaC::ex(p1), GiNaC::ex(p2), GiNaC::ex(p3) ); \
}
```

7.36.2.4 DECLARE_FUNCTION_4P

```
#define DECLARE_FUNCTION_4P (
    NAME )
```

Value:

```
class NAME##_SERIAL { public: static unsigned serial; }; \
const unsigned NAME##_NPARAMS = 4; \
template< typename T1, typename T2, typename T3, typename T4 > const GiNaC::function NAME( const T1 & p1,
    const T2 & p2, const T3 & p3, const T4 & p4 ) { \
    return GiNaC::function( NAME##_SERIAL::serial, GiNaC::ex(p1), GiNaC::ex(p2), GiNaC::ex(p3), GiNaC::ex(p4)
    ); \
}
```

7.36.2.5 DECLARE_FUNCTION_5P

```
#define DECLARE_FUNCTION_5P (
    NAME )
```

Value:

```
class NAME##_SERIAL { public: static unsigned serial; }; \
const unsigned NAME##_NPARAMS = 5; \
template< typename T1, typename T2, typename T3, typename T4, typename T5 > const GiNaC::function NAME(
    const T1 & p1, const T2 & p2, const T3 & p3, const T4 & p4, const T5 & p5 ) { \
    return GiNaC::function( NAME##_SERIAL::serial, GiNaC::ex(p1), GiNaC::ex(p2), GiNaC::ex(p3),
    GiNaC::ex(p4), GiNaC::ex(p5) ); \
}
```

7.36.2.6 DECLARE_FUNCTION_6P

```
#define DECLARE_FUNCTION_6P (
    NAME )
```

Value:

```
class NAME##_SERIAL { public: static unsigned serial; }; \
const unsigned NAME##_NPARAMS = 6; \
template< typename T1, typename T2, typename T3, typename T4, typename T5, typename T6 > const
    GiNaC::function NAME( const T1 & p1, const T2 & p2, const T3 & p3, const T4 & p4, const T5 & p5, const
    T6 & p6 ) { \
    return GiNaC::function( NAME##_SERIAL::serial, GiNaC::ex(p1), GiNaC::ex(p2), GiNaC::ex(p3),
    GiNaC::ex(p4), GiNaC::ex(p5), GiNaC::ex(p6) ); \
}
```

7.36.2.7 DECLARE_FUNCTION_7P

```
#define DECLARE_FUNCTION_7P (
    NAME )
```

Value:

```
class NAME##_SERIAL { public: static unsigned serial; }; \
const unsigned NAME##_NPARAMS = 7; \
template< typename T1, typename T2, typename T3, typename T4, typename T5, typename T6, typename T7 > const
    GiNaC::function NAME( const T1 & p1, const T2 & p2, const T3 & p3, const T4 & p4, const T5 & p5, const
    T6 & p6, const T7 & p7 ) { \
    return GiNaC::function( NAME##_SERIAL::serial, GiNaC::ex(p1), GiNaC::ex(p2), GiNaC::ex(p3),
    GiNaC::ex(p4), GiNaC::ex(p5), GiNaC::ex(p6), GiNaC::ex(p7) ); \
}
```

7.36.2.8 DECLARE_FUNCTION_8P

```
#define DECLARE_FUNCTION_8P(
    NAME )
```

Value:

```
class NAME##_SERIAL { public: static unsigned serial; }; \
const unsigned NAME##_NPARAMS = 8; \
template< typename T1, typename T2, typename T3, typename T4, typename T5, typename T6, typename T7,
    typename T8 > const GiNaC::function NAME( const T1 & p1, const T2 & p2, const T3 & p3, const T4 & p4,
    const T5 & p5, const T6 & p6, const T7 & p7, const T8 & p8 ) { \
    return GiNaC::function(NAME##_SERIAL::serial, GiNaC::ex(p1), GiNaC::ex(p2), GiNaC::ex(p3),
    GiNaC::ex(p4), GiNaC::ex(p5), GiNaC::ex(p6), GiNaC::ex(p7), GiNaC::ex(p8) ); \
}
```

7.36.2.9 DECLARE_FUNCTION_9P

```
#define DECLARE_FUNCTION_9P(
    NAME )
```

Value:

```
class NAME##_SERIAL { public: static unsigned serial; }; \
const unsigned NAME##_NPARAMS = 9; \
template< typename T1, typename T2, typename T3, typename T4, typename T5, typename T6, typename T7,
    typename T8, typename T9 > const GiNaC::function NAME( const T1 & p1, const T2 & p2, const T3 & p3,
    const T4 & p4, const T5 & p5, const T6 & p6, const T7 & p7, const T8 & p8, const T9 & p9 ) { \
    return GiNaC::function(NAME##_SERIAL::serial, GiNaC::ex(p1), GiNaC::ex(p2), GiNaC::ex(p3),
    GiNaC::ex(p4), GiNaC::ex(p5), GiNaC::ex(p6), GiNaC::ex(p7), GiNaC::ex(p8), GiNaC::ex(p9) ); \
}
```

7.36.2.10 DECLARE_FUNCTION_10P

```
#define DECLARE_FUNCTION_10P(
    NAME )
```

Value:

```
class NAME##_SERIAL { public: static unsigned serial; }; \
const unsigned NAME##_NPARAMS = 10; \
template< typename T1, typename T2, typename T3, typename T4, typename T5, typename T6, typename T7,
    typename T8, typename T9, typename T10 > const GiNaC::function NAME( const T1 & p1, const T2 & p2,
    const T3 & p3, const T4 & p4, const T5 & p5, const T6 & p6, const T7 & p7, const T8 & p8, const T9 &
    p9, const T10 & p10 ) { \
    return GiNaC::function(NAME##_SERIAL::serial, GiNaC::ex(p1), GiNaC::ex(p2), GiNaC::ex(p3),
    GiNaC::ex(p4), GiNaC::ex(p5), GiNaC::ex(p6), GiNaC::ex(p7), GiNaC::ex(p8), GiNaC::ex(p9),
    GiNaC::ex(p10) ); \
}
```

7.36.2.11 DECLARE_FUNCTION_11P

```
#define DECLARE_FUNCTION_11P(
    NAME )
```

Value:

```
class NAME##_SERIAL { public: static unsigned serial; }; \
const unsigned NAME##_NPARAMS = 11; \
template< typename T1, typename T2, typename T3, typename T4, typename T5, typename T6, typename T7,
    typename T8, typename T9, typename T10, typename T11 > const GiNaC::function NAME( const T1 & p1,
    const T2 & p2, const T3 & p3, const T4 & p4, const T5 & p5, const T6 & p6, const T7 & p7, const T8 &
    p8, const T9 & p9, const T10 & p10, const T11 & p11 ) { \
    return GiNaC::function(NAME##_SERIAL::serial, GiNaC::ex(p1), GiNaC::ex(p2), GiNaC::ex(p3),
    GiNaC::ex(p4), GiNaC::ex(p5), GiNaC::ex(p6), GiNaC::ex(p7), GiNaC::ex(p8), GiNaC::ex(p9),
    GiNaC::ex(p10), GiNaC::ex(p11) ); \
}
```

7.36.2.12 DECLARE_FUNCTION_12P

```
#define DECLARE_FUNCTION_12P(
    NAME )
```

Value:

```
class NAME##_SERIAL { public: static unsigned serial; }; \
const unsigned NAME##_NPARAMS = 12; \
template< typename T1, typename T2, typename T3, typename T4, typename T5, typename T6, typename T7,
    typename T8, typename T9, typename T10, typename T11, typename T12 > const GiNaC::function NAME( const
    T1 & p1, const T2 & p2, const T3 & p3, const T4 & p4, const T5 & p5, const T6 & p6, const T7 & p7,
    const T8 & p8, const T9 & p9, const T10 & p10, const T11 & p11, const T12 & p12 ) { \
    return GiNaC::function(NAME##_SERIAL::serial, GiNaC::ex(p1), GiNaC::ex(p2), GiNaC::ex(p3),
    GiNaC::ex(p4), GiNaC::ex(p5), GiNaC::ex(p6), GiNaC::ex(p7), GiNaC::ex(p8), GiNaC::ex(p9),
    GiNaC::ex(p10), GiNaC::ex(p11), GiNaC::ex(p12) ); \
}
```

7.36.2.13 DECLARE_FUNCTION_13P

```
#define DECLARE_FUNCTION_13P(
    NAME )
```

Value:

```
class NAME##_SERIAL { public: static unsigned serial; }; \
const unsigned NAME##_NPARAMS = 13; \
template< typename T1, typename T2, typename T3, typename T4, typename T5, typename T6, typename T7,
    typename T8, typename T9, typename T10, typename T11, typename T12, typename T13 > const
    GiNaC::function NAME( const T1 & p1, const T2 & p2, const T3 & p3, const T4 & p4, const T5 & p5, const
    T6 & p6, const T7 & p7, const T8 & p8, const T9 & p9, const T10 & p10, const T11 & p11, const T12 &
    p12, const T13 & p13 ) { \
    return GiNaC::function(NAME##_SERIAL::serial, GiNaC::ex(p1), GiNaC::ex(p2), GiNaC::ex(p3),
    GiNaC::ex(p4), GiNaC::ex(p5), GiNaC::ex(p6), GiNaC::ex(p7), GiNaC::ex(p8), GiNaC::ex(p9),
    GiNaC::ex(p10), GiNaC::ex(p11), GiNaC::ex(p12), GiNaC::ex(p13) ); \
}
```

7.36.2.14 DECLARE_FUNCTION_14P

```
#define DECLARE_FUNCTION_14P(
    NAME )
```

Value:

```
class NAME##_SERIAL { public: static unsigned serial; }; \
const unsigned NAME##_NPARAMS = 14; \
template< typename T1, typename T2, typename T3, typename T4, typename T5, typename T6, typename T7,
    typename T8, typename T9, typename T10, typename T11, typename T12, typename T13, typename T14 > const
    GiNaC::function NAME( const T1 & p1, const T2 & p2, const T3 & p3, const T4 & p4, const T5 & p5, const
    T6 & p6, const T7 & p7, const T8 & p8, const T9 & p9, const T10 & p10, const T11 & p11, const T12 &
    p12, const T13 & p13, const T14 & p14 ) { \
    return GiNaC::function(NAME##_SERIAL::serial, GiNaC::ex(p1), GiNaC::ex(p2), GiNaC::ex(p3),
    GiNaC::ex(p4), GiNaC::ex(p5), GiNaC::ex(p6), GiNaC::ex(p7), GiNaC::ex(p8), GiNaC::ex(p9),
    GiNaC::ex(p10), GiNaC::ex(p11), GiNaC::ex(p12), GiNaC::ex(p13), GiNaC::ex(p14) ); \
}
```

7.36.2.15 REGISTER_FUNCTION

```
#define REGISTER_FUNCTION(
    NAME,
    OPT )
```

Value:

```
unsigned NAME##_SERIAL::serial = \
    GiNaC::function::register_new(GiNaC::function_options(#NAME, NAME##_NPARAMS).OPT);
```

7.36.2.16 is_ex_the_function

```
#define is_ex_the_function(  
    OBJ,  
    FUNCNAME ) (GiNaC::is_the_function<FUNCNAME##_SERIAL> (OBJ) )
```

7.37 ginac.h File Reference

This include file includes all other public [GiNaC](#) headers.

```
#include "version.h"  
#include "basic.h"  
#include "ex.h"  
#include "normal.h"  
#include "archive.h"  
#include "print.h"  
#include "constant.h"  
#include "fail.h"  
#include "integral.h"  
#include "lst.h"  
#include "matrix.h"  
#include "numeric.h"  
#include "power.h"  
#include "relational.h"  
#include "structure.h"  
#include "symbol.h"  
#include "pseries.h"  
#include "wildcard.h"  
#include "symmetry.h"  
#include "expair.h"  
#include "expairseq.h"  
#include "add.h"  
#include "mul.h"  
#include "exprseq.h"  
#include "function.h"  
#include "ncmul.h"  
#include "inifcns.h"  
#include "fderivative.h"  
#include "operators.h"  
#include "hash_map.h"  
#include "idx.h"  
#include "indexed.h"  
#include "tensor.h"  
#include "color.h"  
#include "clifford.h"  
#include "factor.h"  
#include "integration_kernel.h"  
#include "excompiler.h"  
#include "Gt.h"  
#include "parser.h"
```

7.37.1 Detailed Description

This include file includes all other public [GiNaC](#) headers.

7.38 Gt.cpp File Reference

Implementation of [GiNaC](#)'s elliptic multiple polylogarithms.

```
#include <type_traits>
#include <sstream>
#include <limits>
#include <cln/cln.h>
#include "add.h"
#include "mul.h"
#include "power.h"
#include "lst.h"
#include "matrix.h"
#include "symbol.h"
#include "numeric.h"
#include "inifcns.h"
#include "operators.h"
#include "relational.h"
#include "utils_multi_iterator.h"
#include "hash_seed.h"
#include "constant.h"
#include "Gt.h"
#include "Gt_helpers.h"
```

Namespaces

- namespace [GiNaC](#)

Macros

- `#define` [DEBUG_GT](#) 0

Functions

- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([Gt](#), [basic](#), [print_func](#)<[print_dflt](#)>(&[Gt::do_print](#)). [print_func](#)<[print_python](#)>(&[Gt::do_print](#)). [print_func](#)<[print_latex](#)>(&[Gt::do_print_latex](#))) [Gt](#)
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([Gt](#))

7.38.1 Detailed Description

Implementation of [GiNaC](#)'s elliptic multiple polylogarithms.

7.38.2 Macro Definition Documentation

7.38.2.1 DEBUG_GT

```
#define DEBUG_GT 0
```

7.39 Gt.h File Reference

Interface to [GiNaC](#)'s elliptic multiple polylogarithms.

```
#include "archive.h"
#include "basic.h"
#include "ex.h"
```

Classes

- class [GiNaC::Gt](#)
Elliptic Multiple Polylogarithm.
- struct [GiNaC::Gt::kernel](#)

Namespaces

- namespace [GiNaC](#)
- namespace [GiNaC::Gt_detail](#)

Functions

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) (Gt)

7.39.1 Detailed Description

Interface to [GiNaC](#)'s elliptic multiple polylogarithms.

7.40 Gt_helpers.cpp File Reference

Implementation of helper functions for elliptic multiple polylogarithms.

```
#include "add.h"
#include "mul.h"
#include "power.h"
#include "lst.h"
#include "operators.h"
#include "relational.h"
#include "inifcns.h"
#include "function.h"
#include "Gt.h"
#include "Gt_helpers.h"
```

Namespaces

- namespace [GiNaC](#)
- namespace [GiNaC::Gt_detail](#)

Functions

- `std::vector< std::vector< int > > GiNaC::Gt_detail::integer_partition` (const int `n`, const int `m`)
- `template<typename Kernel > ex GiNaC::Gt_detail::deconcatenate_path` (const `std::vector< Kernel > &args`, const `std::vector< ex > &endpoints`, const `std::function< ex(std::vector< Kernel > &new_args, const ex &start, const ex &end)> &construct`)
- `template ex GiNaC::Gt_detail::deconcatenate_path< ex >` (const `std::vector< ex > &args`, const `std::vector< ex > &endpoints`, const `std::function< ex(std::vector< ex > &new_args, const ex &start, const ex &end)> &construct`)
- `template ex GiNaC::Gt_detail::deconcatenate_path< Gt::kernel >` (const `std::vector< Gt::kernel > &args`, const `std::vector< ex > &endpoints`, const `std::function< ex(std::vector< Gt::kernel > &new_args, const ex &start, const ex &end)> &construct`)
- `std::ostream & GiNaC::Gt_detail::operator<<` (std::ostream &`os`, const `pathintegral_term` &`term`)
- `bool GiNaC::Gt_detail::operator<` (const `pathintegral_term` &`a`, const `pathintegral_term` &`b`)

7.40.1 Detailed Description

Implementation of helper functions for elliptic multiple polylogarithms.

7.41 Gt_helpers.h File Reference

Interface to helper functions for elliptic multiple polylogarithms.

```
#include <set>
#include "ex.h"
#include "hash_map.h"
```

Classes

- class `GiNaC::Gt_detail::TransformExpressionWithCache< Type >`
- class `GiNaC::Gt_detail::pathintegral_term`
- class `GiNaC::Gt_detail::pathintegral_term_list`

Namespaces

- namespace `GiNaC`
- namespace `GiNaC::Gt_detail`

Functions

- `std::vector< std::vector< int > > GiNaC::Gt_detail::integer_partition` (const int `n`, const int `m`)
- `template<typename Kernel > ex GiNaC::Gt_detail::deconcatenate_path` (const `std::vector< Kernel > &args`, const `std::vector< ex > &endpoints`, const `std::function< ex(std::vector< Kernel > &new_args, const ex &start, const ex &end)> &construct`)
- `std::ostream & GiNaC::Gt_detail::operator<<` (std::ostream &`os`, const `pathintegral_term` &`term`)
- `bool GiNaC::Gt_detail::operator<` (const `pathintegral_term` &`a`, const `pathintegral_term` &`b`)

7.41.1 Detailed Description

Interface to helper functions for elliptic multiple polylogarithms.

7.42 hash_map.h File Reference

Replacement for map<> using hash tables.

```
#include <unordered_map>
```

Namespaces

- namespace [GiNaC](#)

Typedefs

- template<typename T, class Hash = std::hash<ex>, class KeyEqual = std::equal_to<ex>, class Allocator = std::allocator<std::pair<const ex, T>>>>
using [GiNaC::exhashmap](#) = std::unordered_map< [ex](#), T, Hash, KeyEqual, Allocator >

7.42.1 Detailed Description

Replacement for map<> using hash tables.

7.43 hash_seed.h File Reference

Type-specific hash seed.

```
#include <typeinfo>  
#include <cstring>  
#include "utils.h"
```

Namespaces

- namespace [GiNaC](#)

Functions

- static unsigned [GiNaC::make_hash_seed](#) (const std::type_info &tinfo)
We need a hash function which gives different values for objects of different types.

7.43.1 Detailed Description

Type-specific hash seed.

7.44 idx.cpp File Reference

Implementation of [GiNaC](#)'s indices.

```
#include "idx.h"
#include "symbol.h"
#include "lst.h"
#include "relational.h"
#include "operators.h"
#include "archive.h"
#include "utils.h"
#include "hash_seed.h"
#include <sstream>
#include <stdexcept>
```

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([idx](#), [basic](#), [print_func< print_context >\(&idx::do_print\)](#), [print_func< print_latex >\(&idx::do_print_latex\)](#), [print_func< print_csrc >\(&idx::do_print_csrc\)](#), [print_func< print_tree >\(&idx::do_print_tree\)](#)) [GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#)([varidx](#))
- [GiNaC::print_func< print_context > \(&varidx::do_print\)](#), [print_func< print_latex >\(&varidx](#)
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([idx](#))
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([varidx](#))
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([spinidx](#))
- [bool GiNaC::is_dummy_pair](#) (const [idx](#) &i1, const [idx](#) &i2)
Check whether two indices form a dummy pair.
- [bool GiNaC::is_dummy_pair](#) (const [ex](#) &e1, const [ex](#) &e2)
Check whether two expressions form a dummy index pair.
- [void GiNaC::find_free_and_dummy](#) ([exvector::const_iterator](#) it, [exvector::const_iterator](#) itend, [exvector](#) &out←
_free, [exvector](#) &out_dummy)
Given a vector of indices, split them into two vectors, one containing the free indices, the other containing the dummy indices (numeric indices are neither free nor dummy ones).
- [ex GiNaC::minimal_dim](#) (const [ex](#) &dim1, const [ex](#) &dim2)
Return the minimum of two index dimensions.

Variables

- [GiNaC::idx](#)

7.44.1 Detailed Description

Implementation of [GiNaC](#)'s indices.

7.45 idx.h File Reference

Interface to [GiNaC](#)'s indices.

```
#include "ex.h"
#include "numeric.h"
```

Classes

- class [GiNaC::idx](#)
This class holds one index of an indexed object.
- class [GiNaC::varidx](#)
This class holds an index with a variance (co- or contravariant).
- class [GiNaC::spinidx](#)
This class holds a spinor index that can be dotted or undotted and that also has a variance.

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([idx](#))
- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([varidx](#))
- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([spinidx](#))
- bool [GiNaC::is_dummy_pair](#) (const [idx](#) &i1, const [idx](#) &i2)
Check whether two indices form a dummy pair.
- bool [GiNaC::is_dummy_pair](#) (const [ex](#) &e1, const [ex](#) &e2)
Check whether two expressions form a dummy index pair.
- void [GiNaC::find_free_and_dummy](#) (exvector::const_iterator it, exvector::const_iterator itend, [exvector](#) &out_free, [exvector](#) &out_dummy)
Given a vector of indices, split them into two vectors, one containing the free indices, the other containing the dummy indices (numeric indices are neither free nor dummy ones).
- void [GiNaC::find_free_and_dummy](#) (const [exvector](#) &v, [exvector](#) &out_free, [exvector](#) &out_dummy)
Given a vector of indices, split them into two vectors, one containing the free indices, the other containing the dummy indices (numeric indices are neither free nor dummy ones).
- void [GiNaC::find_dummy_indices](#) (const [exvector](#) &v, [exvector](#) &out_dummy)
Given a vector of indices, find the dummy indices.
- size_t [GiNaC::count_dummy_indices](#) (const [exvector](#) &v)
Count the number of dummy index pairs in an index vector.
- size_t [GiNaC::count_free_indices](#) (const [exvector](#) &v)
Count the number of dummy index pairs in an index vector.
- [ex](#) [GiNaC::minimal_dim](#) (const [ex](#) &dim1, const [ex](#) &dim2)
Return the minimum of two index dimensions.

7.45.1 Detailed Description

Interface to [GiNaC](#)'s indices.

7.46 indexed.cpp File Reference

Implementation of [GiNaC](#)'s indexed expressions.

```
#include "indexed.h"
#include "idx.h"
#include "add.h"
#include "mul.h"
#include "ncmul.h"
#include "power.h"
#include "relational.h"
#include "symmetry.h"
#include "operators.h"
#include "lst.h"
#include "archive.h"
#include "symbol.h"
#include "utils.h"
#include "integral.h"
#include "matrix.h"
#include "inifcns.h"
#include <algorithm>
#include <iostream>
#include <limits>
#include <sstream>
#include <stdexcept>
```

Classes

- struct [GiNaC::idx_is_equal_ignore_dim](#)
- struct [GiNaC::is_summation_idx](#)
- struct [GiNaC::ex_base_is_less](#)
- class [GiNaC::terminfo](#)

This structure stores the original and symmetrized versions of terms obtained during the simplification of sums.

- class [GiNaC::terminfo_is_less](#)
- class [GiNaC::symminfo](#)

This structure stores the individual symmetrized terms obtained during the simplification of sums.

- class [GiNaC::symminfo_is_less_by_symmterm](#)
- class [GiNaC::symminfo_is_less_by_orig](#)

Namespaces

- namespace [GiNaC](#)

Functions

- `GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`indexed`, `exprseq`, `print_func< print_context >(&indexed::do_print)`, `print_func< print_latex >(&indexed::do_print_latex)`, `print_func< print_tree >(&indexed::do_print_tree)`) `indexed`
- `GiNaC::GINAC_BIND_UNARCHIVER` (`indexed`)
- static bool `GiNaC::indices_consistent` (const `exvector` &v1, const `exvector` &v2)
Check whether two sorted index vectors are consistent (i.e.
- template<class T >
`size_t GiNaC::number_of_type` (const `exvector` &v)
- template<class T >
`static ex GiNaC::rename_dummy_indices` (const `ex` &e, `exvector` &global_dummy_indices, `exvector` &local_↔
`_dummy_indices`)
Rename dummy indices in an expression.
- static void `GiNaC::find_variant_indices` (const `exvector` &v, `exvector` &variant_indices)
Given a set of indices, extract those of class varidx.
- bool `GiNaC::reposition_dummy_indices` (`ex` &e, `exvector` &variant_dummy_indices, `exvector` &moved_↔
`indices`)
Raise/lower dummy indices in a single indexed objects to canonicalize their variance.
- static void `GiNaC::product_to_exvector` (const `ex` &e, `exvector` &v, bool &non_commutative)
- template<class T >
`ex GiNaC::idx_symmetrization` (const `ex` &r, const `exvector` &local_dummy_indices)
- `ex GiNaC::simplify_indexed` (const `ex` &e, `exvector` &free_indices, `exvector` &dummy_indices, const
`scalar_products` &sp)
Simplify indexed expression, return list of free indices.
- `ex GiNaC::simplify_indexed_product` (const `ex` &e, `exvector` &free_indices, `exvector` &dummy_indices, const
`scalar_products` &sp)
*Simplify product of indexed expressions (commutative, noncommutative and simple squares), return list of free in-
indices.*
- bool `GiNaC::hasindex` (const `ex` &x, const `ex` &sym)
- `exvector GiNaC::get_all_dummy_indices_safely` (const `ex` &e)
More reliable version of the form.
- `exvector GiNaC::get_all_dummy_indices` (const `ex` &e)
Returns all dummy indices from the exvector.
- `lst GiNaC::rename_dummy_indices_uniquely` (const `exvector` &va, const `exvector` &vb)
Similar to above, where va and vb are the same and the return value is a list of two lists for substitution in b.
- `ex GiNaC::rename_dummy_indices_uniquely` (const `exvector` &va, const `exvector` &vb, const `ex` &b)
Same as above, where va and vb contain the indices of a and b and are sorted.
- `ex GiNaC::rename_dummy_indices_uniquely` (const `ex` &a, const `ex` &b)
Returns b with all dummy indices, which are common with a, renamed.
- `ex GiNaC::rename_dummy_indices_uniquely` (`exvector` &va, const `ex` &b, bool modify_va=false)
*Returns b with all dummy indices, which are listed in va, renamed if modify_va is set to TRUE all dummy indices of b
will be appended to va.*
- `ex GiNaC::expand_dummy_sum` (const `ex` &e, bool subs_idx=false)
*This function returns the given expression with expanded sums for all dummy index summations, where the dimen-
sionality of the dummy index is a nonnegative integer.*

7.46.1 Detailed Description

Implementation of `GiNaC`'s indexed expressions.

7.47 indexed.h File Reference

Interface to [GiNaC](#)'s indexed expressions.

```
#include "exprseq.h"
#include "wildcard.h"
#include <map>
```

Classes

- class [GiNaC::indexed](#)
This class holds an indexed expression.
- class [GiNaC::spmapkey](#)
- class [GiNaC::scalar_products](#)
Helper class for storing information about known scalar products which are to be automatically replaced by [simplify_indexed\(\)](#).

Namespaces

- namespace [GiNaC](#)

Typedefs

- typedef std::map< [spmapkey](#), [ex](#) > [GiNaC::spmap](#)

Functions

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([indexed](#))
- [exvector](#) [GiNaC::get_all_dummy_indices](#) (const [ex](#) &[e](#))
Returns all dummy indices from the exvector.
- [exvector](#) [GiNaC::get_all_dummy_indices_safely](#) (const [ex](#) &[e](#))
More reliable version of the form.
- [ex](#) [GiNaC::rename_dummy_indices_uniquely](#) ([exvector](#) &[va](#), const [ex](#) &[b](#), bool [modify_va](#)=false)
Returns [b](#) with all dummy indices, which are listed in [va](#), renamed if [modify_va](#) is set to TRUE all dummy indices of [b](#) will be appended to [va](#).
- [ex](#) [GiNaC::rename_dummy_indices_uniquely](#) (const [ex](#) &[a](#), const [ex](#) &[b](#))
Returns [b](#) with all dummy indices, which are common with [a](#), renamed.
- [ex](#) [GiNaC::rename_dummy_indices_uniquely](#) (const [exvector](#) &[va](#), const [exvector](#) &[vb](#), const [ex](#) &[b](#))
Same as above, where [va](#) and [vb](#) contain the indices of [a](#) and [b](#) and are sorted.
- [lst](#) [GiNaC::rename_dummy_indices_uniquely](#) (const [exvector](#) &[va](#), const [exvector](#) &[vb](#))
Similar to above, where [va](#) and [vb](#) are the same and the return value is a list of two lists for substitution in [b](#).
- [ex](#) [GiNaC::expand_dummy_sum](#) (const [ex](#) &[e](#), bool [subs_idx](#)=false)
This function returns the given expression with expanded sums for all dummy index summations, where the dimensionality of the dummy index is a nonnegative integer.

7.47.1 Detailed Description

Interface to [GiNaC](#)'s indexed expressions.

7.48 inifcns.cpp File Reference

Implementation of [GiNaC](#)'s initially known functions.

```
#include "inifcns.h"
#include "ex.h"
#include "constant.h"
#include "lst.h"
#include "fderivative.h"
#include "matrix.h"
#include "mul.h"
#include "power.h"
#include "operators.h"
#include "relational.h"
#include "pseries.h"
#include "symbol.h"
#include "symmetry.h"
#include "utils.h"
#include <stdexcept>
#include <vector>
```

Classes

- class [GiNaC::symbolset](#)

Namespaces

- namespace [GiNaC](#)

Functions

- static [ex](#) [GiNaC::conjugate_evalf](#) (const [ex](#) &arg)
- static [ex](#) [GiNaC::conjugate_eval](#) (const [ex](#) &arg)
- static void [GiNaC::conjugate_print_latex](#) (const [ex](#) &arg, const [print_context](#) &c)
- static [ex](#) [GiNaC::conjugate_conjugate](#) (const [ex](#) &arg)
- static [ex](#) [GiNaC::conjugate_expl_derivative](#) (const [ex](#) &arg, const [symbol](#) &s)
- static [ex](#) [GiNaC::conjugate_real_part](#) (const [ex](#) &arg)
- static [ex](#) [GiNaC::conjugate_imag_part](#) (const [ex](#) &arg)
- static bool [GiNaC::func_arg_info](#) (const [ex](#) &arg, unsigned inf)
- static bool [GiNaC::conjugate_info](#) (const [ex](#) &arg, unsigned inf)
- [GiNaC::REGISTER_FUNCTION](#) (conjugate_function, eval_func([conjugate_eval](#)). evalf_func([conjugate_evalf](#)).
expl_derivative_func([conjugate_expl_derivative](#)). info_func([conjugate_info](#)). print_func< [print_latex](#)
>([conjugate_print_latex](#)). conjugate_func([conjugate_conjugate](#)). real_part_func([conjugate_real_part](#)).
imag_part_func([conjugate_imag_part](#)). set_name("conjugate","conjugate"))
- static [ex](#) [GiNaC::real_part_evalf](#) (const [ex](#) &arg)
- static [ex](#) [GiNaC::real_part_eval](#) (const [ex](#) &arg)
- static void [GiNaC::real_part_print_latex](#) (const [ex](#) &arg, const [print_context](#) &c)
- static [ex](#) [GiNaC::real_part_conjugate](#) (const [ex](#) &arg)
- static [ex](#) [GiNaC::real_part_real_part](#) (const [ex](#) &arg)
- static [ex](#) [GiNaC::real_part_imag_part](#) (const [ex](#) &arg)
- static [ex](#) [GiNaC::real_part_expl_derivative](#) (const [ex](#) &arg, const [symbol](#) &s)

- `GiNaC::REGISTER_FUNCTION` (`real_part_function`, `eval_func(real_part_eval)`. `evalf_func(real_part_evalf)`. `expl_derivative_func(real_part_expl_derivative)`. `print_func< print_latex >(real_part_print_latex)`. `conjugate_func(real_part_conjugate)`. `real_part_func(real_part_real_part)`. `imag_part_func(real_part_imag_part)`. `set_name("real_part","real_part")`)
- static `ex` `GiNaC::imag_part_evalf` (`const ex &arg`)
- static `ex` `GiNaC::imag_part_eval` (`const ex &arg`)
- static void `GiNaC::imag_part_print_latex` (`const ex &arg`, `const print_context &c`)
- static `ex` `GiNaC::imag_part_conjugate` (`const ex &arg`)
- static `ex` `GiNaC::imag_part_real_part` (`const ex &arg`)
- static `ex` `GiNaC::imag_part_imag_part` (`const ex &arg`)
- static `ex` `GiNaC::imag_part_expl_derivative` (`const ex &arg`, `const symbol &s`)
- `GiNaC::REGISTER_FUNCTION` (`imag_part_function`, `eval_func(imag_part_eval)`. `evalf_func(imag_part_evalf)`. `expl_derivative_func(imag_part_expl_derivative)`. `print_func< print_latex >(imag_part_print_latex)`. `conjugate_func(imag_part_conjugate)`. `real_part_func(imag_part_real_part)`. `imag_part_func(imag_part_imag_part)`. `set_name("imag_part","imag_part")`)
- static `ex` `GiNaC::abs_evalf` (`const ex &arg`)
- static `ex` `GiNaC::abs_eval` (`const ex &arg`)
- static `ex` `GiNaC::abs_expand` (`const ex &arg`, `unsigned options`)
- static `ex` `GiNaC::abs_expl_derivative` (`const ex &arg`, `const symbol &s`)
- static void `GiNaC::abs_print_latex` (`const ex &arg`, `const print_context &c`)
- static void `GiNaC::abs_print_csrc_float` (`const ex &arg`, `const print_context &c`)
- static `ex` `GiNaC::abs_conjugate` (`const ex &arg`)
- static `ex` `GiNaC::abs_real_part` (`const ex &arg`)
- static `ex` `GiNaC::abs_imag_part` (`const ex &arg`)
- static `ex` `GiNaC::abs_power` (`const ex &arg`, `const ex &exp`)
- bool `GiNaC::abs_info` (`const ex &arg`, `unsigned inf`)
- `GiNaC::REGISTER_FUNCTION` (`abs`, `eval_func(abs_eval)`. `evalf_func(abs_evalf)`. `expand_func(abs_expand)`. `expl_derivative_func(abs_expl_derivative)`. `info_func(abs_info)`. `print_func< print_latex >(abs_print_latex)`. `print_func< print_csrc_float >(abs_print_csrc_float)`. `print_func< print_csrc_double >(abs_print_csrc_double)`. `conjugate_func(abs_conjugate)`. `real_part_func(abs_real_part)`. `imag_part_func(abs_imag_part)`. `power_func(abs_power)`)
- static `ex` `GiNaC::step_evalf` (`const ex &arg`)
- static `ex` `GiNaC::step_eval` (`const ex &arg`)
- static `ex` `GiNaC::step_series` (`const ex &arg`, `const relational &rel`, `int order`, `unsigned options`)
- static `ex` `GiNaC::step_conjugate` (`const ex &arg`)
- static `ex` `GiNaC::step_real_part` (`const ex &arg`)
- static `ex` `GiNaC::step_imag_part` (`const ex &arg`)
- `GiNaC::REGISTER_FUNCTION` (`step`, `eval_func(step_eval)`. `evalf_func(step_evalf)`. `series_func(step_series)`. `conjugate_func(step_conjugate)`. `real_part_func(step_real_part)`. `imag_part_func(step_imag_part)`)
- static `ex` `GiNaC::csgn_evalf` (`const ex &arg`)
- static `ex` `GiNaC::csgn_eval` (`const ex &arg`)
- static `ex` `GiNaC::csgn_series` (`const ex &arg`, `const relational &rel`, `int order`, `unsigned options`)
- static `ex` `GiNaC::csgn_conjugate` (`const ex &arg`)
- static `ex` `GiNaC::csgn_real_part` (`const ex &arg`)
- static `ex` `GiNaC::csgn_imag_part` (`const ex &arg`)
- static `ex` `GiNaC::csgn_power` (`const ex &arg`, `const ex &exp`)
- `GiNaC::REGISTER_FUNCTION` (`csgn`, `eval_func(csgn_eval)`. `evalf_func(csgn_evalf)`. `series_func(csgn_series)`. `conjugate_func(csgn_conjugate)`. `real_part_func(csgn_real_part)`. `imag_part_func(csgn_imag_part)`. `power_func(csgn_power)`)
- static `ex` `GiNaC::eta_evalf` (`const ex &x`, `const ex &y`)
- static `ex` `GiNaC::eta_eval` (`const ex &x`, `const ex &y`)
- static `ex` `GiNaC::eta_series` (`const ex &x`, `const ex &y`, `const relational &rel`, `int order`, `unsigned options`)
- static `ex` `GiNaC::eta_conjugate` (`const ex &x`, `const ex &y`)
- static `ex` `GiNaC::eta_real_part` (`const ex &x`, `const ex &y`)
- static `ex` `GiNaC::eta_imag_part` (`const ex &x`, `const ex &y`)

- `GiNaC::REGISTER_FUNCTION` (`eta`, `eval_func(eta_eval)`). `evalf_func(eta_evalf)`. `series_func(eta_series)`. `latex_name("\\eta")`. `set_symmetry(sy_symm(0, 1))`. `conjugate_func(eta_conjugate)`. `real_part_func(eta_real_part)`. `imag_part_func(eta_imag_part)`)
 - static `ex` `GiNaC::Li2_evalf` (`const ex &x`)
 - static `ex` `GiNaC::Li2_eval` (`const ex &x`)
 - static `ex` `GiNaC::Li2_deriv` (`const ex &x`, unsigned `deriv_param`)
 - static `ex` `GiNaC::Li2_series` (`const ex &x`, `const relational &rel`, int `order`, unsigned `options`)
 - static `ex` `GiNaC::Li2_conjugate` (`const ex &x`)
 - `GiNaC::REGISTER_FUNCTION` (`Li2`, `eval_func(Li2_eval)`). `evalf_func(Li2_evalf)`. `derivative_func(Li2_deriv)`. `series_func(Li2_series)`. `conjugate_func(Li2_conjugate)`. `latex_name("\\mathrm{Li}_2")`)
 - static `ex` `GiNaC::Li3_eval` (`const ex &x`)
 - `GiNaC::REGISTER_FUNCTION` (`Li3`, `eval_func(Li3_eval)`). `latex_name("\\mathrm{Li}_3")`)
 - static `ex` `GiNaC::zetaderiv_eval` (`const ex &n`, `const ex &x`)
 - static `ex` `GiNaC::zetaderiv_deriv` (`const ex &n`, `const ex &x`, unsigned `deriv_param`)
 - `GiNaC::REGISTER_FUNCTION` (`zetaderiv`, `eval_func(zetaderiv_eval)`). `derivative_func(zetaderiv_deriv)`. `latex_name("\\zeta^{\\prime}")`)
 - static `ex` `GiNaC::factorial_evalf` (`const ex &x`)
 - static `ex` `GiNaC::factorial_eval` (`const ex &x`)
 - static void `GiNaC::factorial_print_dflt_latex` (`const ex &x`, `const print_context &c`)
 - static `ex` `GiNaC::factorial_conjugate` (`const ex &x`)
 - static `ex` `GiNaC::factorial_real_part` (`const ex &x`)
 - static `ex` `GiNaC::factorial_imag_part` (`const ex &x`)
 - `GiNaC::REGISTER_FUNCTION` (`factorial`, `eval_func(factorial_eval)`). `evalf_func(factorial_evalf)`. `print_func< print_dflt >(factorial_print_dflt_latex)`. `print_func< print_latex >(factorial_print_dflt_latex)`. `conjugate_func(factorial_conjugate)`. `real_part_func(factorial_real_part)`. `imag_part_func(factorial_imag_part)`)
 - static `ex` `GiNaC::binomial_evalf` (`const ex &x`, `const ex &y`)
 - static `ex` `GiNaC::binomial_sym` (`const ex &x`, `const numeric &y`)
 - static `ex` `GiNaC::binomial_eval` (`const ex &x`, `const ex &y`)
 - static `ex` `GiNaC::binomial_conjugate` (`const ex &x`, `const ex &y`)
 - static `ex` `GiNaC::binomial_real_part` (`const ex &x`, `const ex &y`)
 - static `ex` `GiNaC::binomial_imag_part` (`const ex &x`, `const ex &y`)
 - `GiNaC::REGISTER_FUNCTION` (`binomial`, `eval_func(binomial_eval)`). `evalf_func(binomial_evalf)`. `conjugate_func(binomial_conjugate)`. `real_part_func(binomial_real_part)`. `imag_part_func(binomial_imag_part)`)
 - static `ex` `GiNaC::Order_eval` (`const ex &x`)
 - static `ex` `GiNaC::Order_series` (`const ex &x`, `const relational &r`, int `order`, unsigned `options`)
 - static `ex` `GiNaC::Order_conjugate` (`const ex &x`)
 - static `ex` `GiNaC::Order_real_part` (`const ex &x`)
 - static `ex` `GiNaC::Order_imag_part` (`const ex &x`)
 - static `ex` `GiNaC::Order_power` (`const ex &x`, `const ex &e`)
 - static `ex` `GiNaC::Order_expl_derivative` (`const ex &arg`, `const symbol &s`)
 - `GiNaC::REGISTER_FUNCTION` (`Order`, `eval_func(Order_eval)`). `series_func(Order_series)`. `latex_name("\\mathcal{O}")`. `expl_derivative_func(Order_expl_derivative)`. `conjugate_func(Order_conjugate)`. `real_part_func(Order_real_part)`. `imag_part_func(Order_imag_part)`. `power_func(Order_power)`)
 - `ex` `GiNaC::lsolve` (`const ex &eqns`, `const ex &symbols`, unsigned `options=solve_algo::automatic`)
- Factorial function.*
- `const numeric` `GiNaC::fsolve` (`const ex &f`, `const symbol &x`, `const numeric &x1`, `const numeric &x2`)
- Find a real root of real-valued function f(x) numerically within a given interval.*

Variables

- unsigned `GiNaC::force_include_tgamma` = `tgamma_SERIAL::serial`
- unsigned `GiNaC::force_include_zeta1` = `zeta1_SERIAL::serial`

7.48.1 Detailed Description

Implementation of [GiNaC](#)'s initially known functions.

7.49 inifcns.h File Reference

Interface to [GiNaC](#)'s initially known functions.

```
#include "numeric.h"
#include "function.h"
#include "ex.h"
```

Classes

- class [GiNaC::zeta1_SERIAL](#)
Complex conjugate.
- class [GiNaC::zeta2_SERIAL](#)
Alternating Euler sum or colored MZV.
- class [GiNaC::G2_SERIAL](#)
Generalized multiple polylogarithm.
- class [GiNaC::G3_SERIAL](#)
Generalized multiple polylogarithm with explicit imaginary parts.
- class [GiNaC::psi1_SERIAL](#)
Polylogarithm and multiple polylogarithm.
- class [GiNaC::psi2_SERIAL](#)
Derivatives of Psi-function (aka polygamma-functions).
- class [GiNaC::iterated_integral2_SERIAL](#)
Complete elliptic integral of the first kind.
- class [GiNaC::iterated_integral3_SERIAL](#)
Iterated integral with explicit truncation.

Namespaces

- namespace [GiNaC](#)

Functions

- `template<typename T1 >`
`function GiNaC::zeta (const T1 &p1)`
- `template<typename T1 , typename T2 >`
`function GiNaC::zeta (const T1 &p1, const T2 &p2)`
- `template<> bool GiNaC::is\_the\_function< zeta\_SERIAL > (const ex &x)`
- `template<typename T1 , typename T2 >`
`function GiNaC::G (const T1 &x, const T2 &y)`
- `template<typename T1 , typename T2 , typename T3 >`
`function GiNaC::G (const T1 &x, const T2 &s, const T3 &y)`
- `template<> bool GiNaC::is\_the\_function< G\_SERIAL > (const ex &x)`

- `template<typename T1 >`
`function GiNaC::psi (const T1 &p1)`
- `template<typename T1 , typename T2 >`
`function GiNaC::psi (const T1 &p1, const T2 &p2)`
- `template<> bool GiNaC::is_the_function< psi_SERIAL > (const ex &x)`
- `template<typename T1 , typename T2 >`
`function GiNaC::iterated_integral (const T1 &kernel_lst, const T2 &lambda)`
- `template<typename T1 , typename T2 , typename T3 >`
`function GiNaC::iterated_integral (const T1 &kernel_lst, const T2 &lambda, const T3 &N_trunc)`
- `template<> bool GiNaC::is_the_function< iterated_integral_SERIAL > (const ex &x)`
- `ex GiNaC::lsolve (const ex &eqns, const ex &symbols, unsigned options=solve_algo::automatic)`
Factorial function.
- `const numeric GiNaC::fsolve (const ex &f, const symbol &x, const numeric &x1, const numeric &x2)`
Find a real root of real-valued function f(x) numerically within a given interval.
- `bool GiNaC::is_order_function (const ex &e)`
Check whether a function is the Order (O(n)) function.
- `ex GiNaC::convert_H_to_Li (const ex &parameterlst, const ex &arg)`
Converts a given list containing parameters for H in Remiddi/Vermaseren notation into the corresponding GiNaC functions.

7.49.1 Detailed Description

Interface to [GiNaC](#)'s initially known functions.

7.50 inifcns_elliptic.cpp File Reference

Implementation of some special functions related to elliptic curves.

```
#include "inifcns.h"
#include "add.h"
#include "constant.h"
#include "lst.h"
#include "mul.h"
#include "numeric.h"
#include "operators.h"
#include "power.h"
#include "pseries.h"
#include "relational.h"
#include "symbol.h"
#include "utils.h"
#include "wildcard.h"
#include "integration_kernel.h"
#include "utils_multi_iterator.h"
#include <cln/cln.h>
#include <sstream>
#include <stdexcept>
#include <vector>
#include <cmath>
```

Namespaces

- namespace [GiNaC](#)

Functions

- static `ex` `GiNaC::EllipticK_evalf` (const `ex` &`k`)
- static `ex` `GiNaC::EllipticK_eval` (const `ex` &`k`)
- static `ex` `GiNaC::EllipticK_deriv` (const `ex` &`k`, unsigned `deriv_param`)
- static `ex` `GiNaC::EllipticK_series` (const `ex` &`k`, const `relational` &`rel`, int `order`, unsigned `options`)
- static void `GiNaC::EllipticK_print_latex` (const `ex` &`k`, const `print_context` &`c`)
- `GiNaC::REGISTER_FUNCTION` (`EllipticK`, `evalf_func`(`EllipticK_evalf`). `eval_func`(`EllipticK_eval`). `derivative`↔
_func(`EllipticK_deriv`). `series_func`(`EllipticK_series`). `print_func`< `print_latex` >(`EllipticK_print_latex`). `do_`↔
`not_evalf_params`())
- static `ex` `GiNaC::EllipticE_evalf` (const `ex` &`k`)
- static `ex` `GiNaC::EllipticE_eval` (const `ex` &`k`)
- static `ex` `GiNaC::EllipticE_deriv` (const `ex` &`k`, unsigned `deriv_param`)
- static `ex` `GiNaC::EllipticE_series` (const `ex` &`k`, const `relational` &`rel`, int `order`, unsigned `options`)
- static void `GiNaC::EllipticE_print_latex` (const `ex` &`k`, const `print_context` &`c`)
- `GiNaC::REGISTER_FUNCTION` (`EllipticE`, `evalf_func`(`EllipticE_evalf`). `eval_func`(`EllipticE_eval`). `derivative`↔
_func(`EllipticE_deriv`). `series_func`(`EllipticE_series`). `print_func`< `print_latex` >(`EllipticE_print_latex`). `do_`↔
`not_evalf_params`())
- static `ex` `GiNaC::iterated_integral_evalf_impl` (const `ex` &`kernel_lst`, const `ex` &`lambda`, const `ex` &`N_trunc`)
- static `ex` `GiNaC::iterated_integral2_evalf` (const `ex` &`kernel_lst`, const `ex` &`lambda`)
- static `ex` `GiNaC::iterated_integral3_evalf` (const `ex` &`kernel_lst`, const `ex` &`lambda`, const `ex` &`N_trunc`)
- static `ex` `GiNaC::iterated_integral2_eval` (const `ex` &`kernel_lst`, const `ex` &`lambda`)
- static `ex` `GiNaC::iterated_integral3_eval` (const `ex` &`kernel_lst`, const `ex` &`lambda`, const `ex` &`N_trunc`)

7.50.1 Detailed Description

Implementation of some special functions related to elliptic curves.

The functions are: complete elliptic integral of the first kind `EllipticK(k)` complete elliptic integral of the second kind `EllipticE(k)` iterated integral `iterated_integral(a,y)` or `iterated_integral(a,y,N_trunc)`

Some remarks:

- All formulae used can be looked up in the following publication: [WW] Numerical evaluation of iterated integrals related to elliptic Feynman integrals, M.Walden, S.Weinzierl, arXiv:2010.05271
- When these routines and methods are used for scientific work that leads to publication in a scientific journal, please refer to this program as : M.Walden, S.Weinzierl, "Numerical evaluation of iterated integrals related to elliptic Feynman integrals", arXiv:2010.05271
- As these routines build on the core part of `GiNaC`, it is also polite to acknowledge C. Bauer, A. Frink, R. Kreckel, "Introduction to the GiNaC Framework for Symbolic Computation within the C++ Programming Language", J. Symbolic Computations 33, 1 (2002), cs.sc/0004015

7.51 inifcns_gamma.cpp File Reference

Implementation of Gamma-function, Beta-function, Polygamma-functions, and some related stuff.

```
#include "inifcns.h"
#include "constant.h"
#include "pseries.h"
#include "numeric.h"
#include "power.h"
#include "relational.h"
#include "operators.h"
#include "symbol.h"
#include "symmetry.h"
#include "utils.h"
#include <stdexcept>
#include <vector>
```

Namespaces

- namespace [GiNaC](#)

Functions

- static [ex](#) [GiNaC::lgamma_evalf](#) (const [ex](#) &[x](#))
- static [ex](#) [GiNaC::lgamma_eval](#) (const [ex](#) &[x](#))

Evaluation of $\lgamma(x)$, the natural logarithm of the Gamma function.
- static [ex](#) [GiNaC::lgamma_deriv](#) (const [ex](#) &[x](#), unsigned [deriv_param](#))
- static [ex](#) [GiNaC::lgamma_series](#) (const [ex](#) &[arg](#), const [relational](#) &[rel](#), int [order](#), unsigned [options](#))
- static [ex](#) [GiNaC::lgamma_conjugate](#) (const [ex](#) &[x](#))
- [GiNaC::REGISTER_FUNCTION](#) ([lgamma](#), [eval_func](#)([lgamma_eval](#)). [evalf_func](#)([lgamma_evalf](#)). [derivative_↔_func](#)([lgamma_deriv](#)). [series_func](#)([lgamma_series](#)). [conjugate_func](#)([lgamma_conjugate](#)). [latex_name](#)("\\log \\Gamma"))
- static [ex](#) [GiNaC::tgamma_evalf](#) (const [ex](#) &[x](#))
- static [ex](#) [GiNaC::tgamma_eval](#) (const [ex](#) &[x](#))

Evaluation of $tgamma(x)$, the true Gamma function.
- static [ex](#) [GiNaC::tgamma_deriv](#) (const [ex](#) &[x](#), unsigned [deriv_param](#))
- static [ex](#) [GiNaC::tgamma_series](#) (const [ex](#) &[arg](#), const [relational](#) &[rel](#), int [order](#), unsigned [options](#))
- static [ex](#) [GiNaC::tgamma_conjugate](#) (const [ex](#) &[x](#))
- [GiNaC::REGISTER_FUNCTION](#) ([tgamma](#), [eval_func](#)([tgamma_eval](#)). [evalf_func](#)([tgamma_evalf](#)). [derivative_↔_func](#)([tgamma_deriv](#)). [series_func](#)([tgamma_series](#)). [conjugate_func](#)([tgamma_conjugate](#)). [latex_↔_name](#)("\\Gamma"))
- static [ex](#) [GiNaC::beta_evalf](#) (const [ex](#) &[x](#), const [ex](#) &[y](#))
- static [ex](#) [GiNaC::beta_eval](#) (const [ex](#) &[x](#), const [ex](#) &[y](#))
- static [ex](#) [GiNaC::beta_deriv](#) (const [ex](#) &[x](#), const [ex](#) &[y](#), unsigned [deriv_param](#))
- static [ex](#) [GiNaC::beta_series](#) (const [ex](#) &[arg1](#), const [ex](#) &[arg2](#), const [relational](#) &[rel](#), int [order](#), unsigned [options](#))
- [GiNaC::REGISTER_FUNCTION](#) ([beta](#), [eval_func](#)([beta_eval](#)). [evalf_func](#)([beta_evalf](#)). [derivative_↔_func](#)([beta_deriv](#)). [series_func](#)([beta_series](#)). [latex_name](#)("\\mathrm{B}"). [set_symmetry](#)([sy_symm](#)(0, 1)))
- static [ex](#) [GiNaC::psi1_evalf](#) (const [ex](#) &[x](#))
- static [ex](#) [GiNaC::psi1_eval](#) (const [ex](#) &[x](#))

Evaluation of digamma-function $\psi_1(x)$.
- static [ex](#) [GiNaC::psi1_deriv](#) (const [ex](#) &[x](#), unsigned [deriv_param](#))

- static `ex` `GiNaC::psi1_series` (const `ex` &arg, const `relational` &rel, int `order`, unsigned `options`)
- static `ex` `GiNaC::psi2_evalf` (const `ex` &n, const `ex` &x)
- static `ex` `GiNaC::psi2_eval` (const `ex` &n, const `ex` &x)
Evaluation of polygamma-function $\psi(n,x)$.
- static `ex` `GiNaC::psi2_deriv` (const `ex` &n, const `ex` &x, unsigned `deriv_param`)
- static `ex` `GiNaC::psi2_series` (const `ex` &n, const `ex` &arg, const `relational` &rel, int `order`, unsigned `options`)

7.51.1 Detailed Description

Implementation of Gamma-function, Beta-function, Polygamma-functions, and some related stuff.

7.52 inifcns_nstdsums.cpp File Reference

Implementation of some special functions that have a representation as nested sums.

```
#include "inifcns.h"
#include "add.h"
#include "constant.h"
#include "lst.h"
#include "mul.h"
#include "numeric.h"
#include "operators.h"
#include "power.h"
#include "pseries.h"
#include "relational.h"
#include "symbol.h"
#include "utils.h"
#include "wildcard.h"
#include <cln/cln.h>
#include <sstream>
#include <stdexcept>
#include <vector>
#include <cmath>
```

Namespaces

- namespace `GiNaC`

Functions

- static `ex` `GiNaC::G2_evalf` (const `ex` &x_, const `ex` &y)
- static `ex` `GiNaC::G2_eval` (const `ex` &x_, const `ex` &y)
- static `ex` `GiNaC::G3_evalf` (const `ex` &x_, const `ex` &s_, const `ex` &y)
- static `ex` `GiNaC::G3_eval` (const `ex` &x_, const `ex` &s_, const `ex` &y)
- static `ex` `GiNaC::Li_evalf` (const `ex` &m_, const `ex` &x_)
- static `ex` `GiNaC::Li_eval` (const `ex` &m_, const `ex` &x_)
- static `ex` `GiNaC::Li_series` (const `ex` &m, const `ex` &x, const `relational` &rel, int `order`, unsigned `options`)
- static `ex` `GiNaC::Li_deriv` (const `ex` &m_, const `ex` &x_, unsigned `deriv_param`)
- static void `GiNaC::Li_print_latex` (const `ex` &m_, const `ex` &x_, const `print_context` &c)

- `GiNaC::REGISTER_FUNCTION` (`Li`, `evalf_func(Li_evalf)`. `eval_func(Li_eval)`. `series_func(Li_series)`. `derivative_func(Li_deriv)`. `print_func< print_latex >(Li_print_latex)`. `do_not_evalf_params()`)
- static `ex` `GiNaC::S_evalf` (const `ex` &`n`, const `ex` &`p`, const `ex` &`x`)
- static `ex` `GiNaC::S_eval` (const `ex` &`n`, const `ex` &`p`, const `ex` &`x`)
- static `ex` `GiNaC::S_series` (const `ex` &`n`, const `ex` &`p`, const `ex` &`x`, const `relational` &`rel`, int `order`, unsigned `options`)
- static `ex` `GiNaC::S_deriv` (const `ex` &`n`, const `ex` &`p`, const `ex` &`x`, unsigned `deriv_param`)
- static void `GiNaC::S_print_latex` (const `ex` &`n`, const `ex` &`p`, const `ex` &`x`, const `print_context` &`c`)
- `GiNaC::REGISTER_FUNCTION` (`S`, `evalf_func(S_evalf)`. `eval_func(S_eval)`. `series_func(S_series)`. `derivative_func(S_deriv)`. `print_func< print_latex >(S_print_latex)`. `do_not_evalf_params()`)
- static `ex` `GiNaC::H_evalf` (const `ex` &`x1`, const `ex` &`x2`)
- static `ex` `GiNaC::H_eval` (const `ex` &`m`, const `ex` &`x`)
- static `ex` `GiNaC::H_series` (const `ex` &`m`, const `ex` &`x`, const `relational` &`rel`, int `order`, unsigned `options`)
- static `ex` `GiNaC::H_deriv` (const `ex` &`m`, const `ex` &`x`, unsigned `deriv_param`)
- static void `GiNaC::H_print_latex` (const `ex` &`m`, const `ex` &`x`, const `print_context` &`c`)
- `GiNaC::REGISTER_FUNCTION` (`H`, `evalf_func(H_evalf)`. `eval_func(H_eval)`. `series_func(H_series)`. `derivative_func(H_deriv)`. `print_func< print_latex >(H_print_latex)`. `do_not_evalf_params()`)
- `ex` `GiNaC::convert_H_to_Li` (const `ex` &`parameterlst`, const `ex` &`arg`)
Converts a given list containing parameters for H in Remiddi/Vermaseren notation into the corresponding GiNaC functions.
- static `ex` `GiNaC::zeta1_evalf` (const `ex` &`x`)
- static `ex` `GiNaC::zeta1_eval` (const `ex` &`m`)
- static `ex` `GiNaC::zeta1_deriv` (const `ex` &`m`, unsigned `deriv_param`)
- static void `GiNaC::zeta1_print_latex` (const `ex` &`m`, const `print_context` &`c`)
- static `ex` `GiNaC::zeta2_evalf` (const `ex` &`x`, const `ex` &`s`)
- static `ex` `GiNaC::zeta2_eval` (const `ex` &`m`, const `ex` &`s`)
- static `ex` `GiNaC::zeta2_deriv` (const `ex` &`m`, const `ex` &`s`, unsigned `deriv_param`)
- static void `GiNaC::zeta2_print_latex` (const `ex` &`m`, const `ex` &`s`, const `print_context` &`c`)

7.52.1 Detailed Description

Implementation of some special functions that have a representation as nested sums.

The functions are: classical polylogarithm $\text{Li}(n,x)$ multiple polylogarithm $\text{Li}(\{m_1,\dots,m_k\},\{x_1,\dots,x_k\})$ $G(\{a_1,\dots,a_k\},y)$ or $G(\{a_1,\dots,a_k\},\{s_1,\dots,s_k\},y)$ Nielsen's generalized polylogarithm $S(n,p,x)$ harmonic polylogarithm $H(m,x)$ or $H(\{m_1,\dots,m_k\},x)$ multiple zeta value $\text{zeta}(m)$ or $\text{zeta}(\{m_1,\dots,m_k\})$ alternating Euler sum $\text{zeta}(m,s)$ or $\text{zeta}(\{m_1,\dots,m_k\},\{s_1,\dots,s_k\})$

Some remarks:

- All formulae used can be looked up in the following publications: [Kol] Nielsen's Generalized Polylogarithms, K.S.Kolbig, SIAM J.Math.Anal. 17 (1986), pp. 1232-1258. [Cra] Fast Evaluation of Multiple Zeta Sums, R.E.Crandall, Math.Comp. 67 (1998), pp. 1163-1172. [ReV] Harmonic Polylogarithms, E.Remiddi, J.A. Vermaseren, Int.J.Mod.Phys. A15 (2000), pp. 725-754 [BBB] Special Values of Multiple Polylogarithms, J.Borwein, D.Bradley, D.Broadhurst, P.Lisonek, Trans.Amer.Math.Soc. 353/3 (2001), pp. 907-941 [VSW] Numerical evaluation of multiple polylogarithms, J.Vollinga, S.Weinzierl, hep-ph/0410259
- The order of parameters and arguments of Li and zeta is defined according to the nested sums representation. The parameters for H are understood as in [ReV]. They can be in expanded — only 0, 1 and -1 — or in compactified — a string with zeros in front of 1 or -1 is written as a single number — notation.
- All functions can be numerically evaluated with arguments in the whole complex plane. The parameters for Li , zeta and S must be positive integers. If you want to have an alternating Euler sum, you have to give the signs of the parameters as a second argument s to $\text{zeta}(m,s)$ containing 1 and -1.

- The calculation of classical polylogarithms is speeded up by using Bernoulli numbers and look-up tables. S uses look-up tables as well. The zeta function applies the algorithms in [Cra] and [BBB] for speed up. Multiple polylogarithms use Hoelder convolution [BBB].
- The functions have no means to do a series expansion into nested sums. To do this, you have to convert these functions into the appropriate objects from the nestedsums library, do the expansion and convert the result back.
- Numerical testing of this implementation has been performed by doing a comparison of results between this software and the commercial M..... 4.1. Multiple zeta values have been checked by means of evaluations into simple zeta values. Harmonic polylogarithms have been checked by comparison to $S(n,p,x)$ for corresponding parameter combinations and by continuity checks around $|x|=1$ along with comparisons to corresponding zeta functions. Multiple polylogarithms were checked against H and zeta and by means of shuffle and quasi-shuffle relations.

7.53 inifcns_trans.cpp File Reference

Implementation of transcendental (and trigonometric and hyperbolic) functions.

```
#include "inifcns.h"
#include "ex.h"
#include "constant.h"
#include "add.h"
#include "mul.h"
#include "numeric.h"
#include "power.h"
#include "operators.h"
#include "relational.h"
#include "symbol.h"
#include "pseries.h"
#include "utils.h"
#include <stdexcept>
#include <vector>
```

Namespaces

- namespace [GiNaC](#)

Functions

- static [ex](#) [GiNaC::exp_evalf](#) (const [ex](#) &x)
- static [ex](#) [GiNaC::exp_eval](#) (const [ex](#) &x)
- static [ex](#) [GiNaC::exp_expand](#) (const [ex](#) &arg, unsigned [options](#))
- static [ex](#) [GiNaC::exp_deriv](#) (const [ex](#) &x, unsigned deriv_param)
- static [ex](#) [GiNaC::exp_real_part](#) (const [ex](#) &x)
- static [ex](#) [GiNaC::exp_imag_part](#) (const [ex](#) &x)
- static [ex](#) [GiNaC::exp_conjugate](#) (const [ex](#) &x)
- static [ex](#) [GiNaC::exp_power](#) (const [ex](#) &x, const [ex](#) &a)
- static bool [GiNaC::exp_info](#) (const [ex](#) &x, unsigned inf)
- [GiNaC::REGISTER_FUNCTION](#) ([exp](#), [eval_func](#)([exp_eval](#)). [evalf_func](#)([exp_evalf](#)). [info_func](#)([exp_info](#)). [expand_func](#)([exp_expand](#)). [derivative_func](#)([exp_deriv](#)). [real_part_func](#)([exp_real_part](#)). [imag_part_func](#)([exp_imag_part](#)). [conjugate_func](#)([exp_conjugate](#)). [power_func](#)([exp_power](#)). [latex_name](#)("\\exp"))

- static `ex` `GiNaC::log_evalf` (const `ex` &`x`)
- static `ex` `GiNaC::log_eval` (const `ex` &`x`)
- static `ex` `GiNaC::log_deriv` (const `ex` &`x`, unsigned `deriv_param`)
- static `ex` `GiNaC::log_series` (const `ex` &`arg`, const `relational` &`rel`, int `order`, unsigned `options`)
- static `ex` `GiNaC::log_real_part` (const `ex` &`x`)
- static `ex` `GiNaC::log_imag_part` (const `ex` &`x`)
- static `ex` `GiNaC::log_expand` (const `ex` &`arg`, unsigned `options`)
- static `ex` `GiNaC::log_conjugate` (const `ex` &`x`)
- static bool `GiNaC::log_info` (const `ex` &`x`, unsigned `inf`)
- `GiNaC::REGISTER_FUNCTION` (`log`, `eval_func(log_eval)`. `evalf_func(log_evalf)`. `info_func(log_info)`. `expand_func(log_expand)`. `derivative_func(log_deriv)`. `series_func(log_series)`. `real_part_func(log_real_part)`. `imag_part_func(log_imag_part)`. `conjugate_func(log_conjugate)`. `latex_name("\\ln")`)
- static `ex` `GiNaC::sin_evalf` (const `ex` &`x`)
- static `ex` `GiNaC::sin_eval` (const `ex` &`x`)
- static `ex` `GiNaC::sin_deriv` (const `ex` &`x`, unsigned `deriv_param`)
- static `ex` `GiNaC::sin_real_part` (const `ex` &`x`)
- static `ex` `GiNaC::sin_imag_part` (const `ex` &`x`)
- static `ex` `GiNaC::sin_conjugate` (const `ex` &`x`)
- static bool `GiNaC::trig_info` (const `ex` &`x`, unsigned `inf`)
- `GiNaC::REGISTER_FUNCTION` (`sin`, `eval_func(sin_eval)`. `evalf_func(sin_evalf)`. `info_func(trig_info)`. `derivative_func(sin_deriv)`. `real_part_func(sin_real_part)`. `imag_part_func(sin_imag_part)`. `conjugate_↵
func(sin_conjugate)`. `latex_name("\\sin")`)
- static `ex` `GiNaC::cos_evalf` (const `ex` &`x`)
- static `ex` `GiNaC::cos_eval` (const `ex` &`x`)
- static `ex` `GiNaC::cos_deriv` (const `ex` &`x`, unsigned `deriv_param`)
- static `ex` `GiNaC::cos_real_part` (const `ex` &`x`)
- static `ex` `GiNaC::cos_imag_part` (const `ex` &`x`)
- static `ex` `GiNaC::cos_conjugate` (const `ex` &`x`)
- `GiNaC::REGISTER_FUNCTION` (`cos`, `eval_func(cos_eval)`. `info_func(trig_info)`. `evalf_func(cos_evalf)`. `derivative_func(cos_deriv)`. `real_part_func(cos_real_part)`. `imag_part_func(cos_imag_part)`. `conjugate_↵
func(cos_conjugate)`. `latex_name("\\cos")`)
- static `ex` `GiNaC::tan_evalf` (const `ex` &`x`)
- static `ex` `GiNaC::tan_eval` (const `ex` &`x`)
- static `ex` `GiNaC::tan_deriv` (const `ex` &`x`, unsigned `deriv_param`)
- static `ex` `GiNaC::tan_real_part` (const `ex` &`x`)
- static `ex` `GiNaC::tan_imag_part` (const `ex` &`x`)
- static `ex` `GiNaC::tan_series` (const `ex` &`x`, const `relational` &`rel`, int `order`, unsigned `options`)
- static `ex` `GiNaC::tan_conjugate` (const `ex` &`x`)
- `GiNaC::REGISTER_FUNCTION` (`tan`, `eval_func(tan_eval)`. `evalf_func(tan_evalf)`. `info_func(trig_info)`. `derivative_func(tan_deriv)`. `series_func(tan_series)`. `real_part_func(tan_real_part)`. `imag_part_↵
func(tan_imag_part)`. `conjugate_func(tan_conjugate)`. `latex_name("\\tan")`)
- static `ex` `GiNaC::asin_evalf` (const `ex` &`x`)
- static `ex` `GiNaC::asin_eval` (const `ex` &`x`)
- static `ex` `GiNaC::asin_deriv` (const `ex` &`x`, unsigned `deriv_param`)
- static `ex` `GiNaC::asin_conjugate` (const `ex` &`x`)
- static bool `GiNaC::asin_info` (const `ex` &`x`, unsigned `inf`)
- `GiNaC::REGISTER_FUNCTION` (`asin`, `eval_func(asin_eval)`. `evalf_func(asin_evalf)`. `info_func(asin_info)`. `derivative_func(asin_deriv)`. `conjugate_func(asin_conjugate)`. `latex_name("\\arcsin")`)
- static `ex` `GiNaC::acos_evalf` (const `ex` &`x`)
- static `ex` `GiNaC::acos_eval` (const `ex` &`x`)
- static `ex` `GiNaC::acos_deriv` (const `ex` &`x`, unsigned `deriv_param`)
- static `ex` `GiNaC::acos_conjugate` (const `ex` &`x`)
- `GiNaC::REGISTER_FUNCTION` (`acos`, `eval_func(acos_eval)`. `evalf_func(acos_evalf)`. `info_func(asin_info)`. `derivative_func(acos_deriv)`. `conjugate_func(acos_conjugate)`. `latex_name("\\arccos")`)
- static `ex` `GiNaC::atan_evalf` (const `ex` &`x`)

- static `ex` `GiNaC::atan_eval` (const `ex` &`x`)
- static `ex` `GiNaC::atan_deriv` (const `ex` &`x`, unsigned `deriv_param`)
- static `ex` `GiNaC::atan_series` (const `ex` &`arg`, const `relational` &`rel`, int `order`, unsigned `options`)
- static `ex` `GiNaC::atan_conjugate` (const `ex` &`x`)
- static bool `GiNaC::atan_info` (const `ex` &`x`, unsigned `inf`)
- `GiNaC::REGISTER_FUNCTION` (`atan`, `eval_func(atan_eval)`. `evalf_func(atan_evalf)`. `info_func(atan_info)`. `derivative_func(atan_deriv)`. `series_func(atan_series)`. `conjugate_func(atan_conjugate)`. `latex_name("\\arctan")`)
- static `ex` `GiNaC::atan2_evalf` (const `ex` &`y`, const `ex` &`x`)
- static `ex` `GiNaC::atan2_eval` (const `ex` &`y`, const `ex` &`x`)
- static `ex` `GiNaC::atan2_deriv` (const `ex` &`y`, const `ex` &`x`, unsigned `deriv_param`)
- static bool `GiNaC::atan2_info` (const `ex` &`y`, const `ex` &`x`, unsigned `inf`)
- `GiNaC::REGISTER_FUNCTION` (`atan2`, `eval_func(atan2_eval)`. `evalf_func(atan2_evalf)`. `info_func(atan2_info)`. `evalf_func(atan2_evalf)`. `derivative_func(atan2_deriv)`)
- static `ex` `GiNaC::sinh_evalf` (const `ex` &`x`)
- static `ex` `GiNaC::sinh_eval` (const `ex` &`x`)
- static `ex` `GiNaC::sinh_deriv` (const `ex` &`x`, unsigned `deriv_param`)
- static `ex` `GiNaC::sinh_real_part` (const `ex` &`x`)
- static `ex` `GiNaC::sinh_imag_part` (const `ex` &`x`)
- static `ex` `GiNaC::sinh_conjugate` (const `ex` &`x`)
- `GiNaC::REGISTER_FUNCTION` (`sinh`, `eval_func(sinh_eval)`. `evalf_func(sinh_evalf)`. `info_func(atan_info)`. `derivative_func(sinh_deriv)`. `real_part_func(sinh_real_part)`. `imag_part_func(sinh_imag_part)`. `conjugate_func(sinh_conjugate)`. `latex_name("\\sinh")`)
- static `ex` `GiNaC::cosh_evalf` (const `ex` &`x`)
- static `ex` `GiNaC::cosh_eval` (const `ex` &`x`)
- static `ex` `GiNaC::cosh_deriv` (const `ex` &`x`, unsigned `deriv_param`)
- static `ex` `GiNaC::cosh_real_part` (const `ex` &`x`)
- static `ex` `GiNaC::cosh_imag_part` (const `ex` &`x`)
- static `ex` `GiNaC::cosh_conjugate` (const `ex` &`x`)
- `GiNaC::REGISTER_FUNCTION` (`cosh`, `eval_func(cosh_eval)`. `evalf_func(cosh_evalf)`. `info_func(exp_info)`. `derivative_func(cosh_deriv)`. `real_part_func(cosh_real_part)`. `imag_part_func(cosh_imag_part)`. `conjugate_func(cosh_conjugate)`. `latex_name("\\cosh")`)
- static `ex` `GiNaC::tanh_evalf` (const `ex` &`x`)
- static `ex` `GiNaC::tanh_eval` (const `ex` &`x`)
- static `ex` `GiNaC::tanh_deriv` (const `ex` &`x`, unsigned `deriv_param`)
- static `ex` `GiNaC::tanh_series` (const `ex` &`x`, const `relational` &`rel`, int `order`, unsigned `options`)
- static `ex` `GiNaC::tanh_real_part` (const `ex` &`x`)
- static `ex` `GiNaC::tanh_imag_part` (const `ex` &`x`)
- static `ex` `GiNaC::tanh_conjugate` (const `ex` &`x`)
- `GiNaC::REGISTER_FUNCTION` (`tanh`, `eval_func(tanh_eval)`. `evalf_func(tanh_evalf)`. `info_func(atan_info)`. `derivative_func(tanh_deriv)`. `series_func(tanh_series)`. `real_part_func(tanh_real_part)`. `imag_part_func(tanh_imag_part)`. `conjugate_func(tanh_conjugate)`. `latex_name("\\tanh")`)
- static `ex` `GiNaC::asinh_evalf` (const `ex` &`x`)
- static `ex` `GiNaC::asinh_eval` (const `ex` &`x`)
- static `ex` `GiNaC::asinh_deriv` (const `ex` &`x`, unsigned `deriv_param`)
- static `ex` `GiNaC::asinh_conjugate` (const `ex` &`x`)
- `GiNaC::REGISTER_FUNCTION` (`asinh`, `eval_func(asinh_eval)`. `evalf_func(asinh_evalf)`. `info_func(atan_info)`. `derivative_func(asinh_deriv)`. `conjugate_func(asinh_conjugate)`)
- static `ex` `GiNaC::acosh_evalf` (const `ex` &`x`)
- static `ex` `GiNaC::acosh_eval` (const `ex` &`x`)
- static `ex` `GiNaC::acosh_deriv` (const `ex` &`x`, unsigned `deriv_param`)
- static `ex` `GiNaC::acosh_conjugate` (const `ex` &`x`)
- `GiNaC::REGISTER_FUNCTION` (`acosh`, `eval_func(acosh_eval)`. `evalf_func(acosh_evalf)`. `info_func(asin_info)`. `derivative_func(acosh_deriv)`. `conjugate_func(acosh_conjugate)`)
- static `ex` `GiNaC::atanh_evalf` (const `ex` &`x`)

- static [ex](#) [GiNaC::atanh_eval](#) (const [ex](#) &x)
- static [ex](#) [GiNaC::atanh_deriv](#) (const [ex](#) &x, unsigned deriv_param)
- static [ex](#) [GiNaC::atanh_series](#) (const [ex](#) &arg, const [relational](#) &rel, int [order](#), unsigned [options](#))
- static [ex](#) [GiNaC::atanh_conjugate](#) (const [ex](#) &x)
- [GiNaC::REGISTER_FUNCTION](#) ([atanh](#), [eval_func](#)([atanh_eval](#)). [evalf_func](#)([atanh_evalf](#)). [info_↔](#)
[func](#)([asin_info](#)). [derivative_func](#)([atanh_deriv](#)). [series_func](#)([atanh_series](#)). [conjugate_func](#)([atanh_conjugate](#)))

7.53.1 Detailed Description

Implementation of transcendental (and trigonometric and hyperbolic) functions.

7.54 integral.cpp File Reference

Implementation of [GiNaC](#)'s symbolic integral.

```
#include "integral.h"
#include "numeric.h"
#include "symbol.h"
#include "add.h"
#include "mul.h"
#include "power.h"
#include "inifcns.h"
#include "wildcard.h"
#include "archive.h"
#include "registrar.h"
#include "utils.h"
#include "operators.h"
#include "relational.h"
```

Classes

- struct [GiNaC::error_and_integral](#)
- struct [GiNaC::error_and_integral_is_less](#)

Namespaces

- namespace [GiNaC](#)

Typedefs

- typedef map< [error_and_integral](#), [ex](#), [error_and_integral_is_less](#) > [GiNaC::lookup_map](#)

Functions

- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([integral](#), [basic](#), [print_func](#)< [print_dflt](#)
>(&[integral::do_print](#)). [print_func](#)< [print_python](#) >(&[integral::do_print](#)). [print_func](#)< [print_latex](#)
>(&[integral::do_print_latex](#))) [integral](#)
- [ex](#) [GiNaC::subvalue](#) (const [ex](#) &var, const [ex](#) &value, const [ex](#) &fun)
- [ex](#) [GiNaC::adaptivesimpson](#) (const [ex](#) &x, const [ex](#) &a_in, const [ex](#) &b_in, const [ex](#) &f, const [ex](#) &error)
Numeric integration routine based upon the "Adaptive Quadrature" one in "Numerical Analysis" by Burden and Faires.
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([integral](#))

7.54.1 Detailed Description

Implementation of [GiNaC](#)'s symbolic integral.

7.55 integral.h File Reference

Interface to [GiNaC](#)'s symbolic integral.

```
#include "basic.h"
#include "ex.h"
#include "archive.h"
```

Classes

- class [GiNaC::integral](#)
Symbolic integral.

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([integral](#))
- [ex GiNaC::adaptivesimpson](#) (const [ex](#) &x, const [ex](#) &a_in, const [ex](#) &b_in, const [ex](#) &f, const [ex](#) &error)
Numeric integration routine based upon the "Adaptive Quadrature" one in "Numerical Analysis" by Burden and Faires.

7.55.1 Detailed Description

Interface to [GiNaC](#)'s symbolic integral.

7.56 integration_kernel.cpp File Reference

Implementation of [GiNaC](#)'s integration kernels for iterated integrals.

```
#include "integration_kernel.h"
#include "add.h"
#include "mul.h"
#include "operators.h"
#include "power.h"
#include "relational.h"
#include "symbol.h"
#include "constant.h"
#include "numeric.h"
#include "function.h"
#include "pseries.h"
#include "utils.h"
#include "inifcns.h"
#include <iostream>
#include <stdexcept>
#include <cln/cln.h>
```

Namespaces

- namespace [GiNaC](#)

Functions

- [ex GiNaC::ifactor](#) (const [numeric](#) &n)
Returns the decomposition of the positive integer n into prime numbers in the form $lst(lst(p1,...,pr), lst(a1,...,ar))$ such that $n = p1^{a1} \dots pr^{ar}$.
- [bool GiNaC::is_discriminant_of_quadratic_number_field](#) (const [numeric](#) &n)
Returns true if the integer n is either one or the discriminant of a quadratic number field.
- [numeric GiNaC::kronecker_symbol](#) (const [numeric](#) &a, const [numeric](#) &n)
Returns the Kronecker symbol $a: integer\ n: integer$.
- [numeric GiNaC::primitive_dirichlet_character](#) (const [numeric](#) &n, const [numeric](#) &a)
Defines a primitive Dirichlet character through the Kronecker symbol.
- [numeric GiNaC::dirichlet_character](#) (const [numeric](#) &n, const [numeric](#) &a, const [numeric](#) &N)
Defines a Dirichlet character through the Kronecker symbol.
- [numeric GiNaC::generalised_Bernoulli_number](#) (const [numeric](#) &k, const [numeric](#) &b)
The generalised Bernoulli number.
- [ex GiNaC::Bernoulli_polynomial](#) (const [numeric](#) &k, const [ex](#) &x)
The Bernoulli polynomials.
- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) (integration_kernel, [basic](#), [print_func](#)<[print_context](#)>(&integration_kernel::do_print)) integration_kernel
- [GiNaC::GINAC_BIND_UNARCHIVER](#) (integration_kernel)
- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) (basic_log_kernel, integration_kernel, [print_func](#)<[print_context](#)>(&basic_log_kernel::do_print)) basic_log_kernel
- [GiNaC::GINAC_BIND_UNARCHIVER](#) (basic_log_kernel)
- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) (multiple_polylog_kernel, integration_kernel, [print_func](#)<[print_context](#)>(&multiple_polylog_kernel::do_print)) multiple_polylog_kernel
- [GiNaC::GINAC_BIND_UNARCHIVER](#) (multiple_polylog_kernel)
- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) (ELi_kernel, integration_kernel, [print_func](#)<[print_context](#)>(&ELi_kernel::do_print)) ELi_kernel
- [GiNaC::GINAC_BIND_UNARCHIVER](#) (ELi_kernel)
- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) (Ebar_kernel, integration_kernel, [print_func](#)<[print_context](#)>(&Ebar_kernel::do_print)) Ebar_kernel
- [GiNaC::GINAC_BIND_UNARCHIVER](#) (Ebar_kernel)
- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) (Kronecker_dtau_kernel, integration_kernel, [print_func](#)<[print_context](#)>(&Kronecker_dtau_kernel::do_print)) Kronecker_dtau_kernel
- [GiNaC::GINAC_BIND_UNARCHIVER](#) (Kronecker_dtau_kernel)
- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) (Kronecker_dz_kernel, integration_kernel, [print_func](#)<[print_context](#)>(&Kronecker_dz_kernel::do_print)) Kronecker_dz_kernel
- [GiNaC::GINAC_BIND_UNARCHIVER](#) (Kronecker_dz_kernel)
- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) (Eisenstein_kernel, integration_kernel, [print_func](#)<[print_context](#)>(&Eisenstein_kernel::do_print)) Eisenstein_kernel
- [GiNaC::GINAC_BIND_UNARCHIVER](#) (Eisenstein_kernel)
- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) (Eisenstein_h_kernel, integration_kernel, [print_func](#)<[print_context](#)>(&Eisenstein_h_kernel::do_print)) Eisenstein_h_kernel
- [GiNaC::GINAC_BIND_UNARCHIVER](#) (Eisenstein_h_kernel)
- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) (modular_form_kernel, integration_kernel, [print_func](#)<[print_context](#)>(&modular_form_kernel::do_print)) modular_form_kernel
- [GiNaC::GINAC_BIND_UNARCHIVER](#) (modular_form_kernel)
- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) (user_defined_kernel, integration_kernel, [print_func](#)<[print_context](#)>(&user_defined_kernel::do_print)) user_defined_kernel
- [GiNaC::GINAC_BIND_UNARCHIVER](#) (user_defined_kernel)

7.56.1 Detailed Description

Implementation of [GiNaC](#)'s integration kernels for iterated integrals.

7.56.2 Variable Documentation

7.56.2.1 qbar

```
ex qbar
```

Referenced by [GiNaC::divide_in_z\(\)](#), [GiNaC::ELi_kernel::get_numerical_value\(\)](#), [GiNaC::Ebar_kernel::get_numerical_value\(\)](#), [GiNaC::Kronecker_dtau_kernel::get_numerical_value\(\)](#), [GiNaC::Eisenstein_kernel::get_numerical_value\(\)](#), [GiNaC::Eisenstein_h_kernel::get_numerical_value\(\)](#), [GiNaC::modular_form_kernel::get_numerical_value\(\)](#), [GiNaC::modular_form_kernel::is_numeric\(\)](#), [GiNaC::modular_form_kernel::Laurent_series\(\)](#), [GiNaC::Eisenstein_kernel::series\(\)](#), [GiNaC::Eisenstein_h_kernel::series\(\)](#), [GiNaC::modular_form_kernel::series\(\)](#), and [GiNaC::Kronecker_dz_kernel::series_coeff_impl\(\)](#).

7.56.2.2 order

```
int order
```

Referenced by [GiNaC::atan_series\(\)](#), [GiNaC::atanh_series\(\)](#), [GiNaC::beta_series\(\)](#), [GiNaC::fderivative::do_print_latex\(\)](#), [GiNaC::EllipticE_series\(\)](#), [GiNaC::EllipticK_series\(\)](#), [GiNaC::modular_form_kernel::Laurent_series\(\)](#), [GiNaC::integration_kernel::Laurent_series\(\)](#), [GiNaC::Eisenstein_kernel::Laurent_series\(\)](#), [GiNaC::Eisenstein_h_kernel::Laurent_series\(\)](#), [GiNaC::user_defined_kernel::Laurent_series\(\)](#), [GiNaC::lgamma_series\(\)](#), [GiNaC::Li2_series\(\)](#), [GiNaC::Li_series\(\)](#), [GiNaC::log_series\(\)](#), [GiNaC::Order_series\(\)](#), [GiNaC::psi1_series\(\)](#), [GiNaC::psi2_series\(\)](#), [GiNaC::Eisenstein_kernel::q_expansion_modular_form\(\)](#), [GiNaC::S_series\(\)](#), [GiNaC::ex::series\(\)](#), [GiNaC::series\(\)](#), [GiNaC::basic::series\(\)](#), [GiNaC::add::series\(\)](#), [GiNaC::fderivative::series\(\)](#), [GiNaC::function::series\(\)](#), [GiNaC::integral::series\(\)](#), [GiNaC::integration_kernel::series\(\)](#), [GiNaC::Eisenstein_kernel::series\(\)](#), [GiNaC::Eisenstein_h_kernel::series\(\)](#), [GiNaC::modular_form_kernel::series\(\)](#), [GiNaC::pseries::series\(\)](#), [GiNaC::structure< T, CompA, CompB>::series\(\)](#), [GiNaC::mul::series\(\)](#), [GiNaC::power::series\(\)](#), [GiNaC::symbol::series\(\)](#), [GiNaC::tan_series\(\)](#), [GiNaC::tanh_series\(\)](#), and [GiNaC::tgamma_series\(\)](#).

7.56.2.3 cache_vec

```
std::vector<ex> cache_vec [static], [protected]
```

7.56.2.4 x

```
symbol x [static], [protected]
```

Referenced by [GiNaC::integration_kernel::Laurent_series\(\)](#), and [GiNaC::integration_kernel::series_coeff\(\)](#).

7.57 integration_kernel.h File Reference

Interface to [GiNaC](#)'s integration kernels for iterated integrals.

```
#include "basic.h"
#include "archive.h"
#include "numeric.h"
#include "lst.h"
#include <cln/complex.h>
#include <vector>
```

Classes

- class [GiNaC::integration_kernel](#)
The base class for integration kernels for iterated integrals.
- class [GiNaC::basic_log_kernel](#)
The basic integration kernel with a logarithmic singularity at the origin.
- class [GiNaC::multiple_polylog_kernel](#)
The integration kernel for multiple polylogarithms.
- class [GiNaC::ELi_kernel](#)
The ELi-kernel.
- class [GiNaC::Ebar_kernel](#)
The Ebar-kernel.
- class [GiNaC::Kronecker_dtau_kernel](#)
The kernel corresponding to integrating the Kronecker coefficient function $g^{(n)}(z_j, K\tau)$ in τ (or equivalently in $\bar{q}$).
- class [GiNaC::Kronecker_dz_kernel](#)
The kernel corresponding to integrating the Kronecker coefficient function $g^{(n-1)}(z - z_j, K\tau)$ in z .
- class [GiNaC::Eisenstein_kernel](#)
The kernel corresponding to the Eisenstein series $E_{k,N,a,b,K}(\tau)$.
- class [GiNaC::Eisenstein_h_kernel](#)
The kernel corresponding to the Eisenstein series $h_{k,N,r,s}(\tau)$.
- class [GiNaC::modular_form_kernel](#)
A kernel corresponding to a polynomial in Eisenstein series.
- class [GiNaC::user_defined_kernel](#)
A user-defined integration kernel.

Namespaces

- namespace [GiNaC](#)

Functions

- [ex GiNaC::ifactor](#) (const [numeric](#) &n)
Returns the decomposition of the positive integer n into prime numbers in the form $lst(lst(p1,...,pr), lst(a1,...,ar))$ such that $n = p1^{a1} \dots pr^{ar}$.
- [bool GiNaC::is_discriminant_of_quadratic_number_field](#) (const [numeric](#) &n)
Returns true if the integer n is either one or the discriminant of a quadratic number field.
- [numeric GiNaC::kronecker_symbol](#) (const [numeric](#) &a, const [numeric](#) &n)
Returns the Kronecker symbol a : integer n : integer.
- [numeric GiNaC::primitive_dirichlet_character](#) (const [numeric](#) &n, const [numeric](#) &a)
Defines a primitive Dirichlet character through the Kronecker symbol.
- [numeric GiNaC::dirichlet_character](#) (const [numeric](#) &n, const [numeric](#) &a, const [numeric](#) &N)
Defines a Dirichlet character through the Kronecker symbol.
- [numeric GiNaC::generalised_Bernoulli_number](#) (const [numeric](#) &k, const [numeric](#) &b)
The generalised Bernoulli number.
- [ex GiNaC::Bernoulli_polynomial](#) (const [numeric](#) &k, const [ex](#) &x)
The Bernoulli polynomials.
- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) (integration_kernel)
- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) (basic_log_kernel)
- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) (multiple_polylog_kernel)
- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) (ELi_kernel)

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([Ebar_kernel](#))
- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([Kronecker_dtau_kernel](#))
- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([Kronecker_dz_kernel](#))
- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([Eisenstein_kernel](#))
- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([Eisenstein_h_kernel](#))
- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([modular_form_kernel](#))
- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([user_defined_kernel](#))

7.57.1 Detailed Description

Interface to [GiNaC](#)'s integration kernels for iterated integrals.

7.58 Ist.cpp File Reference

Implementation of [GiNaC](#)'s `Ist`.

```
#include "lst.h"
#include "archive.h"
```

Namespaces

- namespace [GiNaC](#)

Variables

- `template<> GINAC\_IMPLEMENT\_REGISTERED\_CLASS\_OPT\_T(Ist, basic, print_func< print\_context >(&Ist::do_print). print_func< print\_tree >(&Ist::do_print_tree))` `template<> bool Ist GiNaC::GINAC\_BIND\_UNARCHIVER(Ist)`

Specialization of [container::info\(\)](#) for `Ist`.

7.58.1 Detailed Description

Implementation of [GiNaC](#)'s `Ist`.

7.59 Ist.h File Reference

Definition of [GiNaC](#)'s `Ist`.

```
#include "container.h"
#include <list>
```

Namespaces

- namespace [GiNaC](#)

Typedefs

- typedef [container](#)< std::list > [GiNaC::lst](#)

Functions

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([lst](#))

7.59.1 Detailed Description

Definition of [GiNaC](#)'s [lst](#).

7.60 matrix.cpp File Reference

Implementation of symbolic matrices.

```
#include "matrix.h"
#include "numeric.h"
#include "lst.h"
#include "idx.h"
#include "indexed.h"
#include "add.h"
#include "power.h"
#include "symbol.h"
#include "operators.h"
#include "normal.h"
#include "archive.h"
#include "utils.h"
#include <map>
#include <sstream>
#include <stdexcept>
#include <string>
```

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([matrix](#), [basic](#), [print_func](#)< [print_context](#) >(&[matrix::do_print](#)). [print_func](#)< [print_latex](#) >(&[matrix::do_print_latex](#)). [print_func](#)< [print_tree](#) >(&[matrix::do_print_tree](#)). [print_func](#)< [print_python_repr](#) >(&[matrix::do_print_python_repr](#))) [matrix](#)

Default ctor.

- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([matrix](#))
- [ex](#) [GiNaC::lst_to_matrix](#) (const [lst](#) &l)

Convert list of lists to matrix.

- [ex](#) [GiNaC::diag_matrix](#) (const [lst](#) &l)
- Convert list of diagonal elements to matrix.*

- [ex GiNaC::diag_matrix](#) (std::initializer_list< [ex](#) > l)
- [ex GiNaC::unit_matrix](#) (unsigned r, unsigned c)
Create an r times c unit matrix.
- [ex GiNaC::symbolic_matrix](#) (unsigned r, unsigned c, const std::string &base_name, const std::string &tex_↵
base_name)
Create an r times c matrix of newly generated symbols consisting of the given base name plus the numeric row/column position of each element.
- [ex GiNaC::reduced_matrix](#) (const [matrix](#) &m, unsigned r, unsigned c)
Return the reduced matrix that is formed by deleting the rth row and cth column of matrix m.
- [ex GiNaC::sub_matrix](#) (const [matrix](#) &m, unsigned r, unsigned nr, unsigned c, unsigned nc)
Return the nr times nc submatrix starting at position r, c of matrix m.

7.60.1 Detailed Description

Implementation of symbolic matrices.

7.61 matrix.h File Reference

Interface to symbolic matrices.

```
#include "basic.h"
#include "ex.h"
#include "archive.h"
#include "compiler.h"
#include <string>
#include <vector>
```

Classes

- class [GiNaC::matrix](#)
Symbolic matrices.

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([matrix](#))
- [size_t GiNaC::nops](#) (const [matrix](#) &m)
- [ex GiNaC::expand](#) (const [matrix](#) &m, unsigned options=0)
- [ex GiNaC::evalf](#) (const [matrix](#) &m)
- unsigned [GiNaC::rows](#) (const [matrix](#) &m)
- unsigned [GiNaC::cols](#) (const [matrix](#) &m)
- [matrix GiNaC::transpose](#) (const [matrix](#) &m)
- [ex GiNaC::determinant](#) (const [matrix](#) &m, unsigned options=[determinant_algo::automatic](#))
- [ex GiNaC::trace](#) (const [matrix](#) &m)

- `ex GiNaC::charpoly` (const `matrix` &`m`, const `ex` &`lambda`)
- `matrix GiNaC::inverse` (const `matrix` &`m`)
- `matrix GiNaC::inverse` (const `matrix` &`m`, unsigned `algo`)
- unsigned `GiNaC::rank` (const `matrix` &`m`)
- unsigned `GiNaC::rank` (const `matrix` &`m`, unsigned `solve_algo`)
- `ex GiNaC::lst_to_matrix` (const `lst` &`l`)
Convert list of lists to matrix.
- `ex GiNaC::diag_matrix` (const `lst` &`l`)
Convert list of diagonal elements to matrix.
- `ex GiNaC::diag_matrix` (std::initializer_list< `ex` > `l`)
- `ex GiNaC::unit_matrix` (unsigned `r`, unsigned `c`)
Create an r times c unit matrix.
- `ex GiNaC::unit_matrix` (unsigned `x`)
Create a x times x unit matrix.
- `ex GiNaC::symbolic_matrix` (unsigned `r`, unsigned `c`, const std::string &`base_name`, const std::string &`tex_` ↔ `base_name`)
Create an r times c matrix of newly generated symbols consisting of the given base name plus the numeric row/column position of each element.
- `ex GiNaC::reduced_matrix` (const `matrix` &`m`, unsigned `r`, unsigned `c`)
Return the reduced matrix that is formed by deleting the rth row and cth column of matrix m.
- `ex GiNaC::sub_matrix` (const `matrix` &`m`, unsigned `r`, unsigned `nr`, unsigned `c`, unsigned `nc`)
Return the nr times nc submatrix starting at position r, c of matrix m.
- `ex GiNaC::symbolic_matrix` (unsigned `r`, unsigned `c`, const std::string &`base_name`)
Create an r times c matrix of newly generated symbols consisting of the given base name plus the numeric row/column position of each element.

7.61.1 Detailed Description

Interface to symbolic matrices.

7.62 mul.cpp File Reference

Implementation of GiNaC's products of expressions.

```
#include "mul.h"
#include "add.h"
#include "power.h"
#include "operators.h"
#include "matrix.h"
#include "indexed.h"
#include "lst.h"
#include "archive.h"
#include "utils.h"
#include "symbol.h"
#include "compiler.h"
#include <limits>
#include <stdexcept>
#include <vector>
```

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([mul](#), [expairseq](#), [print_func](#)< [print_context](#) >(&[mul::do_print](#)). [print_func](#)< [print_latex](#) >(&[mul::do_print_latex](#)). [print_func](#)< [print_csrc](#) >(&[mul::do_print_csrc](#)). [print_func](#)< [print_tree](#) >(&[mul::do_print_tree](#)). [print_func](#)< [print_python_repr](#) >(&[mul::do_print_python_repr](#))
[mul](#)
- bool [GiNaC::tryfactsubs](#) (const [ex](#) &origfactor, const [ex](#) &patternfactor, int &nummatches, [exmap](#) &repls)
- bool [GiNaC::algebraic_match_mul_with_mul](#) (const [mul](#) &e, const [ex](#) &pat, [exmap](#) &repls, int [factor](#), int &nummatches, const std::vector< bool > &subsed, std::vector< bool > &matched)
Checks whether e matches to the pattern pat and the (possibly to be updated) list of replacements repls.
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([mul](#))

7.62.1 Detailed Description

Implementation of [GiNaC](#)'s products of expressions.

7.63 mul.h File Reference

Interface to [GiNaC](#)'s products of expressions.

```
#include "expairseq.h"
```

Classes

- class [GiNaC::mul](#)
Product of expressions.

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([mul](#))

7.63.1 Detailed Description

Interface to [GiNaC](#)'s products of expressions.

7.64 ncmul.cpp File Reference

Implementation of [GiNaC](#)'s non-commutative products of expressions.

```
#include "ncmul.h"
#include "ex.h"
#include "add.h"
#include "mul.h"
#include "clifford.h"
#include "matrix.h"
#include "archive.h"
#include "indexed.h"
#include "utils.h"
#include <algorithm>
#include <stdexcept>
```

Namespaces

- namespace [GiNaC](#)

Typedefs

- typedef std::vector< std::size_t > [GiNaC::uintvector](#)
- typedef std::vector< unsigned > [GiNaC::unsignedvector](#)
- typedef std::vector< [exvector](#) > [GiNaC::exvectorvector](#)

Functions

- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([ncmul](#), [exprseq](#), [print_func< print_context >\(&ncmul::do_print\)](#), [print_func< print_tree >\(&ncmul::do_print_tree\)](#), [print_func< print_csrc >\(&ncmul::do_print_csrc\)](#), [print_func< print_python_repr >\(&ncmul::do_print_csrc\)](#)) [ncmul](#)
- [ex GiNaC::reeval_ncmul](#) (const [exvector](#) &[v](#))
- [ex GiNaC::hold_ncmul](#) (const [exvector](#) &[v](#))
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([ncmul](#))

7.64.1 Detailed Description

Implementation of [GiNaC](#)'s non-commutative products of expressions.

7.65 ncmul.h File Reference

Interface to [GiNaC](#)'s non-commutative products of expressions.

```
#include "exprseq.h"
#include "archive.h"
```

Classes

- class [GiNaC::ncmul](#)
Non-commutative product of expressions.

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([ncmul](#))
- [ex GiNaC::reeval_ncmul](#) (const [exvector](#) &[v](#))
- [ex GiNaC::hold_ncmul](#) (const [exvector](#) &[v](#))

7.65.1 Detailed Description

Interface to [GiNaC](#)'s non-commutative products of expressions.

7.66 normal.cpp File Reference

This file implements several functions that work on univariate and multivariate polynomials and rational functions.

```
#include "normal.h"
#include "basic.h"
#include "ex.h"
#include "add.h"
#include "constant.h"
#include "expairseq.h"
#include "fail.h"
#include "inifcns.h"
#include "lst.h"
#include "mul.h"
#include "numeric.h"
#include "power.h"
#include "relational.h"
#include "operators.h"
#include "matrix.h"
#include "pseries.h"
#include "symbol.h"
#include "utils.h"
#include "polynomial/chinrem_gcd.h"
#include <algorithm>
#include <map>
```

Classes

- struct [GiNaC::sym_desc](#)
This structure holds information about the highest and lowest degrees in which a symbol appears in two multivariate polynomials "a" and "b".
- class [GiNaC::gcdheu_failed](#)
Exception thrown by [heur_gcd\(\)](#) to signal failure.
- struct [GiNaC::normal_map_function](#)
Function object to be applied by [basic::normal\(\)](#).

Namespaces

- namespace [GiNaC](#)

Macros

- #define [FAST_COMPARE](#) 1
- #define [USE_REMEMBER](#) 0
- #define [USE_TRIAL_DIVISION](#) 0
- #define [STATISTICS](#) 0

Typedefs

- typedef std::vector< [sym_desc](#) > [GiNaC::sym_desc_vec](#)

Functions

- static bool [GiNaC::get_first_symbol](#) (const [ex](#) &e, [ex](#) &x)
Return pointer to first symbol found in expression.
- static void [GiNaC::add_symbol](#) (const [ex](#) &s, [sym_desc_vec](#) &v)
- static void [GiNaC::collect_symbols](#) (const [ex](#) &e, [sym_desc_vec](#) &v)
- static void [GiNaC::get_symbol_stats](#) (const [ex](#) &a, const [ex](#) &b, [sym_desc_vec](#) &v)
Collect statistical information about symbols in polynomials.
- static [numeric](#) [GiNaC::lcmcoeff](#) (const [ex](#) &e, const [numeric](#) &l)
- static [numeric](#) [GiNaC::lcm_of_coefficients_denominators](#) (const [ex](#) &e)
Compute LCM of denominators of coefficients of a polynomial.
- static [ex](#) [GiNaC::multiply_lcm](#) (const [ex](#) &e, const [numeric](#) &lcm)
Bring polynomial from $Q[X]$ to $Z[X]$ by multiplying in the previously determined LCM of the coefficient's denominators.
- [ex](#) [GiNaC::quo](#) (const [ex](#) &a, const [ex](#) &b, const [ex](#) &x, bool check_args)
Quotient $q(x)$ of polynomials $a(x)$ and $b(x)$ in $Q[x]$.
- [ex](#) [GiNaC::rem](#) (const [ex](#) &a, const [ex](#) &b, const [ex](#) &x, bool check_args)
Remainder $r(x)$ of polynomials $a(x)$ and $b(x)$ in $Q[x]$.
- [ex](#) [GiNaC::decomp_rational](#) (const [ex](#) &a, const [ex](#) &x)
Decompose rational function $a(x)=N(x)/D(x)$ into $P(x)+n(x)/D(x)$ with $\text{degree}(n, x) < \text{degree}(D, x)$.
- [ex](#) [GiNaC::prem](#) (const [ex](#) &a, const [ex](#) &b, const [ex](#) &x, bool check_args)
Pseudo-remainder of polynomials $a(x)$ and $b(x)$ in $Q[x]$.
- [ex](#) [GiNaC::sprem](#) (const [ex](#) &a, const [ex](#) &b, const [ex](#) &x, bool check_args)
Sparse pseudo-remainder of polynomials $a(x)$ and $b(x)$ in $Q[x]$.
- bool [GiNaC::divide](#) (const [ex](#) &a, const [ex](#) &b, [ex](#) &q, bool check_args)

- Exact polynomial division of $a(X)$ by $b(X)$ in $Q[X]$.*
 - static bool `GiNaC::divide_in_z` (const `ex` &a, const `ex` &b, `ex` &q, `sym_desc_vec::const_iterator` var)
- Exact polynomial division of $a(X)$ by $b(X)$ in $Z[X]$.*
 - static `ex` `GiNaC::sr_gcd` (const `ex` &a, const `ex` &b, `sym_desc_vec::const_iterator` var)
- Compute GCD of multivariate polynomials using the subresultant PRS algorithm.*
 - static `ex` `GiNaC::interpolate` (const `ex` &gamma, const `numeric` &xi, const `ex` &x, int degree_hint=1)
- ξ -adic polynomial interpolation*
 - static bool `GiNaC::heur_gcd_z` (`ex` &res, const `ex` &a, const `ex` &b, `ex` *ca, `ex` *cb, `sym_desc_vec::const_iterator` var)
- Compute GCD of multivariate polynomials using the heuristic GCD algorithm.*
 - static bool `GiNaC::heur_gcd` (`ex` &res, const `ex` &a, const `ex` &b, `ex` *ca, `ex` *cb, `sym_desc_vec::const_iterator` var)
- Compute GCD of multivariate polynomials using the heuristic GCD algorithm.*
 - static `ex` `GiNaC::gcd_pf_pow` (const `ex` &a, const `ex` &b, `ex` *ca, `ex` *cb)
 - static `ex` `GiNaC::gcd_pf_mul` (const `ex` &a, const `ex` &b, `ex` *ca, `ex` *cb)
 - `ex` `GiNaC::gcd` (const `ex` &a, const `ex` &b, `ex` *ca, `ex` *cb, bool check_args, unsigned `options`)
- Compute GCD (Greatest Common Divisor) of multivariate polynomials $a(X)$ and $b(X)$ in $Z[X]$.*
 - static `ex` `GiNaC::gcd_pf_pow_pow` (const `ex` &a, const `ex` &b, `ex` *ca, `ex` *cb)
 - `ex` `GiNaC::lcm` (const `ex` &a, const `ex` &b, bool check_args)
- Compute LCM (Least Common Multiple) of multivariate polynomials in $Z[X]$.*
 - static `epvector` `GiNaC::sqrfree_yun` (const `ex` &a, const `symbol` &x)
- Compute square-free factorization of multivariate polynomial $a(x)$ using Yun's algorithm.*
 - `ex` `GiNaC::sqrfree` (const `ex` &a, const `lst` &l)
- Compute a square-free factorization of a multivariate polynomial in $Q[X]$.*
 - `ex` `GiNaC::sqrfree_parfrac` (const `ex` &a, const `symbol` &x)
- Compute square-free partial fraction decomposition of rational function $a(x)$.*
 - static `ex` `GiNaC::replace_with_symbol` (const `ex` &e, `exmap` &repl, `exmap` &rev_lookup, `lst` &modifier)
- Create a symbol for replacing the expression "e" (or return a previously assigned symbol).*
 - static `ex` `GiNaC::replace_with_symbol` (const `ex` &e, `exmap` &repl)
- Create a symbol for replacing the expression "e" (or return a previously assigned symbol).*
 - static `ex` `GiNaC::frac_cancel` (const `ex` &n, const `ex` &d)
- Fraction cancellation.*
 - static `ex` `GiNaC::find_common_factor` (const `ex` &e, `ex` &factor, `exmap` &repl)
- Remove the common factor in the terms of a sum 'e' by calculating the GCD, and multiply it into the expression 'factor' (which needs to be initialized to 1, unless you're accumulating factors).*
 - `ex` `GiNaC::collect_common_factors` (const `ex` &e)
- Collect common factors in sums.*
 - `ex` `GiNaC::resultant` (const `ex` &e1, const `ex` &e2, const `ex` &s)
- Resultant of two expressions e_1, e_2 with respect to symbol s .*

7.66.1 Detailed Description

This file implements several functions that work on univariate and multivariate polynomials and rational functions.

These functions include polynomial quotient and remainder, GCD and LCM computation, square-free factorization and rational function normalization.

7.66.2 Macro Definition Documentation

7.66.2.1 FAST_COMPARE

```
#define FAST_COMPARE 1
```


7.66.2.2 USE_REMEMBER

```
#define USE_REMEMBER 0
```

7.66.2.3 USE_TRIAL_DIVISION

```
#define USE_TRIAL_DIVISION 0
```

7.66.2.4 STATISTICS

```
#define STATISTICS 0
```

7.67 normal.h File Reference

This file defines several functions that work on univariate and multivariate polynomials and rational functions.

```
#include "lst.h"
```

Classes

- struct [GiNaC::gcd_options](#)
Flags to control the behavior of [gcd\(\)](#) and friends.

Namespaces

- namespace [GiNaC](#)

Functions

- [ex GiNaC::quo](#) (const [ex](#) &a, const [ex](#) &b, const [ex](#) &x, bool check_args)
Quotient $q(x)$ of polynomials $a(x)$ and $b(x)$ in $Q[x]$.
- [ex GiNaC::rem](#) (const [ex](#) &a, const [ex](#) &b, const [ex](#) &x, bool check_args)
Remainder $r(x)$ of polynomials $a(x)$ and $b(x)$ in $Q[x]$.
- [ex GiNaC::decomp_rational](#) (const [ex](#) &a, const [ex](#) &x)
Decompose rational function $a(x)=N(x)/D(x)$ into $P(x)+n(x)/D(x)$ with $\text{degree}(n, x) < \text{degree}(D, x)$.
- [ex GiNaC::prem](#) (const [ex](#) &a, const [ex](#) &b, const [ex](#) &x, bool check_args)
Pseudo-remainder of polynomials $a(x)$ and $b(x)$ in $Q[x]$.
- [ex GiNaC::sprem](#) (const [ex](#) &a, const [ex](#) &b, const [ex](#) &x, bool check_args)
Sparse pseudo-remainder of polynomials $a(x)$ and $b(x)$ in $Q[x]$.
- [bool GiNaC::divide](#) (const [ex](#) &a, const [ex](#) &b, [ex](#) &q, bool check_args)
Exact polynomial division of $a(X)$ by $b(X)$ in $Q[X]$.
- [ex GiNaC::gcd](#) (const [ex](#) &a, const [ex](#) &b, [ex](#) *ca, [ex](#) *cb, bool check_args, unsigned [options](#))
Compute GCD (Greatest Common Divisor) of multivariate polynomials $a(X)$ and $b(X)$ in $Z[X]$.
- [ex GiNaC::lcm](#) (const [ex](#) &a, const [ex](#) &b, bool check_args)
Compute LCM (Least Common Multiple) of multivariate polynomials in $Z[X]$.
- [ex GiNaC::sqrfree](#) (const [ex](#) &a, const [lst](#) &l)
Compute a square-free factorization of a multivariate polynomial in $Q[X]$.
- [ex GiNaC::sqrfree_parfrac](#) (const [ex](#) &a, const [symbol](#) &x)
Compute square-free partial fraction decomposition of rational function $a(x)$.
- [ex GiNaC::collect_common_factors](#) (const [ex](#) &e)
Collect common factors in sums.
- [ex GiNaC::resultant](#) (const [ex](#) &e1, const [ex](#) &e2, const [ex](#) &s)
Resultant of two expressions $e1, e2$ with respect to symbol s .

7.67.1 Detailed Description

This file defines several functions that work on univariate and multivariate polynomials and rational functions.

These functions include polynomial quotient and remainder, GCD and LCM computation, square-free factorization and rational function normalization.

7.68 numeric.cpp File Reference

This file contains the interface to the underlying bignum package.

```
#include "numeric.h"
#include "ex.h"
#include "operators.h"
#include "archive.h"
#include "utils.h"
#include <limits>
#include <sstream>
#include <stdexcept>
#include <string>
#include <vector>
#include <cln/output.h>
#include <cln/integer_io.h>
#include <cln/integer_ring.h>
#include <cln/rational_io.h>
#include <cln/rational_ring.h>
#include <cln/lfloat_class.h>
#include <cln/lfloat_io.h>
#include <cln/real_io.h>
#include <cln/real_ring.h>
#include <cln/complex_io.h>
#include <cln/complex_ring.h>
#include <cln/numtheory.h>
```

Classes

- class [GiNaC::lanczos_coeffs](#)

Namespaces

- namespace [GiNaC](#)

Functions

- `GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`numeric`, `basic`, `print_func< print_context >(&numeric::do_print)`, `print_func< print_latex >(&numeric::do_print_latex)`, `print_func< print_csrc >(&numeric::do_print_csrc)`, `print_func< print_csrc_cl_N >(&numeric::do_print_csrc_cl_N)`, `print_func< print_tree >(&numeric::do_print_tree)`, `print_func< print_python_repr >(&numeric::do_print_python_repr)`)
`numeric`
default ctor.
- static const `cln::cl_F` `GiNaC::make_real_float` (`const cln::cl_idcoded_float &dec`)
Construct a floating point number from sign, mantissa, and exponent.
- static const `cln::cl_F` `GiNaC::read_real_float` (`std::istream &s`)
Read serialized floating point number.
- `GiNaC::GINAC_BIND_UNARCHIVER` (`numeric`)
- static void `GiNaC::write_real_float` (`std::ostream &s`, `const cln::cl_R &n`)
- static void `GiNaC::print_real_number` (`const print_context &c`, `const cln::cl_R &x`)
Helper function to print a real number in a nicer way than is CLN's default.
- static void `GiNaC::print_integer_csrc` (`const print_context &c`, `const cln::cl_I &x`)
Helper function to print integer number in C++ source format.
- static void `GiNaC::print_real_csrc` (`const print_context &c`, `const cln::cl_R &x`)
Helper function to print real number in C++ source format.
- `template<typename T1 , typename T2 >`
static bool `GiNaC::coerce` (`T1 &dst`, `const T2 &arg`)
- `template<> bool` `GiNaC::coerce< int, cln::cl_I >` (`int &dst`, `const cln::cl_I &arg`)
Check if CLN integer can be converted into int.
- `template<> bool` `GiNaC::coerce< unsigned int, cln::cl_I >` (`unsigned int &dst`, `const cln::cl_I &arg`)
- static void `GiNaC::print_real_cl_N` (`const print_context &c`, `const cln::cl_R &x`)
Helper function to print real number in C++ source format using cl_N types.
- const `numeric` `GiNaC::exp` (`const numeric &x`)
Exponential function.
- const `numeric` `GiNaC::log` (`const numeric &x`)
Natural logarithm.
- const `numeric` `GiNaC::sin` (`const numeric &x`)
Numeric sine (trigonometric function).
- const `numeric` `GiNaC::cos` (`const numeric &x`)
Numeric cosine (trigonometric function).
- const `numeric` `GiNaC::tan` (`const numeric &x`)
Numeric tangent (trigonometric function).
- const `numeric` `GiNaC::asin` (`const numeric &x`)
Numeric inverse sine (trigonometric function).
- const `numeric` `GiNaC::acos` (`const numeric &x`)
Numeric inverse cosine (trigonometric function).
- const `numeric` `GiNaC::atan` (`const numeric &x`)
Numeric arcustangent.
- const `numeric` `GiNaC::atan` (`const numeric &y`, `const numeric &x`)
Numeric arcustangent of two arguments, analytically continued in a suitable way.
- const `numeric` `GiNaC::sinh` (`const numeric &x`)
Numeric hyperbolic sine (trigonometric function).
- const `numeric` `GiNaC::cosh` (`const numeric &x`)
Numeric hyperbolic cosine (trigonometric function).
- const `numeric` `GiNaC::tanh` (`const numeric &x`)
Numeric hyperbolic tangent (trigonometric function).
- const `numeric` `GiNaC::asinh` (`const numeric &x`)

- Numeric inverse hyperbolic sine (trigonometric function).*
- const [numeric GiNaC::acosh](#) (const [numeric &x](#))
- Numeric inverse hyperbolic cosine (trigonometric function).*
- const [numeric GiNaC::atanh](#) (const [numeric &x](#))
- Numeric inverse hyperbolic tangent (trigonometric function).*
- static [cln::cl_N GiNaC::Li2_series](#) (const [cln::cl_N &x](#), const [cln::float_format_t &prec](#))
- Numeric evaluation of Dilogarithm within circle of convergence (unit circle) using a power series.*
- static [cln::cl_N GiNaC::Li2_projection](#) (const [cln::cl_N &x](#), const [cln::float_format_t &prec](#))
- Folds Li2's argument inside a small rectangle to enhance convergence.*
- const [cln::cl_N GiNaC::Li2_](#) (const [cln::cl_N &value](#))
- Numeric evaluation of Dilogarithm.*
- const [numeric GiNaC::Li2](#) (const [numeric &x](#))
- const [numeric GiNaC::zeta](#) (const [numeric &x](#))
- Numeric evaluation of Riemann's Zeta function.*
- static [cln::float_format_t GiNaC::guess_precision](#) (const [cln::cl_N &x](#))
- const [cln::cl_N GiNaC::lgamma](#) (const [cln::cl_N &x](#))
- The Gamma function.*
- const [numeric GiNaC::lgamma](#) (const [numeric &x](#))
- const [cln::cl_N GiNaC::tgamma](#) (const [cln::cl_N &x](#))
- const [numeric GiNaC::tgamma](#) (const [numeric &x](#))
- const [numeric GiNaC::psi](#) (const [numeric &x](#))
- The psi function (aka polygamma function).*
- const [numeric GiNaC::psi](#) (const [numeric &n](#), const [numeric &x](#))
- The psi functions (aka polygamma functions).*
- const [numeric GiNaC::factorial](#) (const [numeric &n](#))
- Factorial combinatorial function.*
- const [numeric GiNaC::doublefactorial](#) (const [numeric &n](#))
- The double factorial combinatorial function.*
- const [numeric GiNaC::binomial](#) (const [numeric &n](#), const [numeric &k](#))
- The Binomial coefficients.*
- const [numeric GiNaC::bernoulli](#) (const [numeric &nn](#))
- Bernoulli number.*
- const [numeric GiNaC::fibonacci](#) (const [numeric &n](#))
- Fibonacci number.*
- const [numeric GiNaC::abs](#) (const [numeric &x](#))
- Absolute value.*
- const [numeric GiNaC::mod](#) (const [numeric &a](#), const [numeric &b](#))
- Modulus (in positive representation).*
- const [numeric GiNaC::smod](#) (const [numeric &a_](#), const [numeric &b_](#))
- Modulus (in symmetric representation).*
- const [numeric GiNaC::irem](#) (const [numeric &a](#), const [numeric &b](#))
- Numeric integer remainder.*
- const [numeric GiNaC::irem](#) (const [numeric &a](#), const [numeric &b](#), [numeric &q](#))
- Numeric integer remainder.*
- const [numeric GiNaC::iquo](#) (const [numeric &a](#), const [numeric &b](#))
- Numeric integer quotient.*
- const [numeric GiNaC::iquo](#) (const [numeric &a](#), const [numeric &b](#), [numeric &r](#))
- Numeric integer quotient.*
- const [numeric GiNaC::gcd](#) (const [numeric &a](#), const [numeric &b](#))
- Greatest Common Divisor.*
- const [numeric GiNaC::lcm](#) (const [numeric &a](#), const [numeric &b](#))

- Least Common Multiple.*
- `const numeric GiNaC::sqrt` (`const numeric &x`)
Numeric square root.
- `const numeric GiNaC::isqrt` (`const numeric &x`)
Integer numeric square root.
- `ex GiNaC::PiEvalf` ()
Floating point evaluation of Archimedes' constant Pi.
- `ex GiNaC::EulerEvalf` ()
Floating point evaluation of Euler's constant gamma.
- `ex GiNaC::CatalanEvalf` ()
Floating point evaluation of Catalan's constant.
- `std::ostream & GiNaC::operator<<` (`std::ostream &os`, `const _numeric_digits &e`)

Variables

- `const numeric GiNaC::I = numeric`(`cln::complex`(`cln::cl_I`(0),`cln::cl_I`(1)))
Imaginary unit.
- `_numeric_digits GiNaC::Digits`
Accuracy in decimal digits.

7.68.1 Detailed Description

This file contains the interface to the underlying bignum package.

Its most important design principle is to completely hide the inner working of that other package from the user of `GiNaC`. It must either provide implementation of arithmetic operators and numerical evaluation of special functions or implement the interface to the bignum package.

7.69 numeric.h File Reference

Makes the interface to the underlying bignum package available.

```
#include "basic.h"
#include "ex.h"
#include "archive.h"
#include <cln/complex.h>
#include <stdexcept>
#include <vector>
```

Classes

- class `GiNaC::_numeric_digits`
This class is used to instantiate a global singleton object Digits which behaves just like Maple's Digits.
- class `GiNaC::pole_error`
Exception class thrown when a singularity is encountered.
- class `GiNaC::numeric`
This class is a wrapper around CLN-numbers within the `GiNaC` class hierarchy.

Namespaces

- namespace [GiNaC](#)

Typedefs

- typedef void(* [GiNaC::digits_changed_callback](#)) (long)
Function pointer to implement callbacks in the case 'Digits' gets changed.

Functions

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) (numeric)
- const [numeric GiNaC::exp](#) (const [numeric &x](#))
Exponential function.
- const [numeric GiNaC::log](#) (const [numeric &x](#))
Natural logarithm.
- const [numeric GiNaC::sin](#) (const [numeric &x](#))
Numeric sine (trigonometric function).
- const [numeric GiNaC::cos](#) (const [numeric &x](#))
Numeric cosine (trigonometric function).
- const [numeric GiNaC::tan](#) (const [numeric &x](#))
Numeric tangent (trigonometric function).
- const [numeric GiNaC::asin](#) (const [numeric &x](#))
Numeric inverse sine (trigonometric function).
- const [numeric GiNaC::acos](#) (const [numeric &x](#))
Numeric inverse cosine (trigonometric function).
- const [numeric GiNaC::atan](#) (const [numeric &x](#))
Numeric arcustangent.
- const [numeric GiNaC::atan](#) (const [numeric &y](#), const [numeric &x](#))
Numeric arcustangent of two arguments, analytically continued in a suitable way.
- const [numeric GiNaC::sinh](#) (const [numeric &x](#))
Numeric hyperbolic sine (trigonometric function).
- const [numeric GiNaC::cosh](#) (const [numeric &x](#))
Numeric hyperbolic cosine (trigonometric function).
- const [numeric GiNaC::tanh](#) (const [numeric &x](#))
Numeric hyperbolic tangent (trigonometric function).
- const [numeric GiNaC::asinh](#) (const [numeric &x](#))
Numeric inverse hyperbolic sine (trigonometric function).
- const [numeric GiNaC::acosh](#) (const [numeric &x](#))
Numeric inverse hyperbolic cosine (trigonometric function).
- const [numeric GiNaC::atanh](#) (const [numeric &x](#))
Numeric inverse hyperbolic tangent (trigonometric function).
- const [numeric GiNaC::Li2](#) (const [numeric &x](#))
- const [numeric GiNaC::zeta](#) (const [numeric &x](#))
Numeric evaluation of Riemann's Zeta function.
- const [numeric GiNaC::lgamma](#) (const [numeric &x](#))
- const [numeric GiNaC::tgamma](#) (const [numeric &x](#))
- const [numeric GiNaC::psi](#) (const [numeric &x](#))
The psi function (aka polygamma function).
- const [numeric GiNaC::psi](#) (const [numeric &n](#), const [numeric &x](#))

The psi functions (aka polygamma functions).

- const [numeric GiNaC::factorial](#) (const [numeric](#) &n)

Factorial combinatorial function.

- const [numeric GiNaC::doublefactorial](#) (const [numeric](#) &n)

The double factorial combinatorial function.

- const [numeric GiNaC::binomial](#) (const [numeric](#) &n, const [numeric](#) &k)

The Binomial coefficients.

- const [numeric GiNaC::bernoulli](#) (const [numeric](#) &nn)

Bernoulli number.

- const [numeric GiNaC::fibonacci](#) (const [numeric](#) &n)

Fibonacci number.

- const [numeric GiNaC::isqrt](#) (const [numeric](#) &x)

Integer numeric square root.

- const [numeric GiNaC::sqrt](#) (const [numeric](#) &x)

Numeric square root.

- const [numeric GiNaC::abs](#) (const [numeric](#) &x)

Absolute value.

- const [numeric GiNaC::mod](#) (const [numeric](#) &a, const [numeric](#) &b)

Modulus (in positive representation).

- const [numeric GiNaC::smod](#) (const [numeric](#) &a_, const [numeric](#) &b_)

Modulus (in symmetric representation).

- const [numeric GiNaC::irem](#) (const [numeric](#) &a, const [numeric](#) &b)

Numeric integer remainder.

- const [numeric GiNaC::irem](#) (const [numeric](#) &a, const [numeric](#) &b, [numeric](#) &q)

Numeric integer remainder.

- const [numeric GiNaC::iquo](#) (const [numeric](#) &a, const [numeric](#) &b)

Numeric integer quotient.

- const [numeric GiNaC::iquo](#) (const [numeric](#) &a, const [numeric](#) &b, [numeric](#) &r)

Numeric integer quotient.

- const [numeric GiNaC::gcd](#) (const [numeric](#) &a, const [numeric](#) &b)

Greatest Common Divisor.

- const [numeric GiNaC::lcm](#) (const [numeric](#) &a, const [numeric](#) &b)

Least Common Multiple.

- const [numeric GiNaC::pow](#) (const [numeric](#) &x, const [numeric](#) &y)

- const [numeric GiNaC::inverse](#) (const [numeric](#) &x)

- [numeric GiNaC::step](#) (const [numeric](#) &x)

- int [GiNaC::csgn](#) (const [numeric](#) &x)

- bool [GiNaC::is_zero](#) (const [numeric](#) &x)

- bool [GiNaC::is_positive](#) (const [numeric](#) &x)

- bool [GiNaC::is_negative](#) (const [numeric](#) &x)

- bool [GiNaC::is_integer](#) (const [numeric](#) &x)

- bool [GiNaC::is_pos_integer](#) (const [numeric](#) &x)

- bool [GiNaC::is_nonneg_integer](#) (const [numeric](#) &x)

- bool [GiNaC::is_even](#) (const [numeric](#) &x)

- bool [GiNaC::is_odd](#) (const [numeric](#) &x)

- bool [GiNaC::is_prime](#) (const [numeric](#) &x)

- bool [GiNaC::is_rational](#) (const [numeric](#) &x)

- bool [GiNaC::is_real](#) (const [numeric](#) &x)

- bool [GiNaC::is_cinteger](#) (const [numeric](#) &x)

- bool [GiNaC::is_crational](#) (const [numeric](#) &x)

- int [GiNaC::to_int](#) (const [numeric](#) &x)

- long [GiNaC::to_long](#) (const [numeric](#) &x)

- double `GiNaC::to_double` (const numeric &x)
- const numeric `GiNaC::real` (const numeric &x)
- const numeric `GiNaC::imag` (const numeric &x)
- const numeric `GiNaC::numer` (const numeric &x)
- const numeric `GiNaC::denom` (const numeric &x)
- ex `GiNaC::PiEvalf` ()
Floating point evaluation of Archimedes' constant Pi.
- ex `GiNaC::EulerEvalf` ()
Floating point evaluation of Euler's constant gamma.
- ex `GiNaC::CatalanEvalf` ()
Floating point evaluation of Catalan's constant.

7.69.1 Detailed Description

Makes the interface to the underlying bignum package available.

7.70 operators.cpp File Reference

Implementation of `GiNaC`'s overloaded operators.

```
#include "operators.h"
#include "numeric.h"
#include "add.h"
#include "mul.h"
#include "power.h"
#include "ncmul.h"
#include "relational.h"
#include "print.h"
#include "utils.h"
```

Namespaces

- namespace `GiNaC`

Enumerations

- enum { `GiNaC::callback_registered` = 1 }

Functions

- static const [ex](#) [GiNaC::exadd](#) (const [ex](#) &lh, const [ex](#) &rh)
Used internally by [operator+\(\)](#) to add two ex objects.
- static const [ex](#) [GiNaC::exmul](#) (const [ex](#) &lh, const [ex](#) &rh)
Used internally by [operator\(\)](#) to multiply two ex objects.*
- static const [ex](#) [GiNaC::exminus](#) (const [ex](#) &lh)
Used internally by [operator-\(\)](#) and friends to change the sign of an argument.
- const [ex](#) [GiNaC::operator+](#) (const [ex](#) &lh, const [ex](#) &rh)
- const [ex](#) [GiNaC::operator-](#) (const [ex](#) &lh, const [ex](#) &rh)
- const [ex](#) [GiNaC::operator*](#) (const [ex](#) &lh, const [ex](#) &rh)
- const [ex](#) [GiNaC::operator/](#) (const [ex](#) &lh, const [ex](#) &rh)
- const [numeric](#) [GiNaC::operator+](#) (const [numeric](#) &lh, const [numeric](#) &rh)
- const [numeric](#) [GiNaC::operator-](#) (const [numeric](#) &lh, const [numeric](#) &rh)
- const [numeric](#) [GiNaC::operator*](#) (const [numeric](#) &lh, const [numeric](#) &rh)
- const [numeric](#) [GiNaC::operator/](#) (const [numeric](#) &lh, const [numeric](#) &rh)
- [ex](#) & [GiNaC::operator+=](#) ([ex](#) &lh, const [ex](#) &rh)
- [ex](#) & [GiNaC::operator-=](#) ([ex](#) &lh, const [ex](#) &rh)
- [ex](#) & [GiNaC::operator*=](#) ([ex](#) &lh, const [ex](#) &rh)
- [ex](#) & [GiNaC::operator/=](#) ([ex](#) &lh, const [ex](#) &rh)
- [numeric](#) & [GiNaC::operator+=](#) ([numeric](#) &lh, const [numeric](#) &rh)
- [numeric](#) & [GiNaC::operator-=](#) ([numeric](#) &lh, const [numeric](#) &rh)
- [numeric](#) & [GiNaC::operator*=](#) ([numeric](#) &lh, const [numeric](#) &rh)
- [numeric](#) & [GiNaC::operator/=](#) ([numeric](#) &lh, const [numeric](#) &rh)
- const [ex](#) [GiNaC::operator+](#) (const [ex](#) &lh)
- const [ex](#) [GiNaC::operator-](#) (const [ex](#) &lh)
- const [numeric](#) [GiNaC::operator+](#) (const [numeric](#) &lh)
- const [numeric](#) [GiNaC::operator-](#) (const [numeric](#) &lh)
- [ex](#) & [GiNaC::operator++](#) ([ex](#) &rh)
Expression prefix increment.
- [ex](#) & [GiNaC::operator--](#) ([ex](#) &rh)
Expression prefix decrement.
- const [ex](#) [GiNaC::operator++](#) ([ex](#) &lh, int)
Expression postfix increment.
- const [ex](#) [GiNaC::operator--](#) ([ex](#) &lh, int)
Expression postfix decrement.
- [numeric](#) & [GiNaC::operator++](#) ([numeric](#) &rh)
Numeric prefix increment.
- [numeric](#) & [GiNaC::operator--](#) ([numeric](#) &rh)
Numeric prefix decrement.
- const [numeric](#) [GiNaC::operator++](#) ([numeric](#) &lh, int)
Numeric postfix increment.
- const [numeric](#) [GiNaC::operator--](#) ([numeric](#) &lh, int)
Numeric postfix decrement.
- const [relational](#) [GiNaC::operator==](#) (const [ex](#) &lh, const [ex](#) &rh)
- const [relational](#) [GiNaC::operator!=](#) (const [ex](#) &lh, const [ex](#) &rh)
- const [relational](#) [GiNaC::operator<](#) (const [ex](#) &lh, const [ex](#) &rh)
- const [relational](#) [GiNaC::operator<=](#) (const [ex](#) &lh, const [ex](#) &rh)
- const [relational](#) [GiNaC::operator>](#) (const [ex](#) &lh, const [ex](#) &rh)
- const [relational](#) [GiNaC::operator>=](#) (const [ex](#) &lh, const [ex](#) &rh)
- static int [GiNaC::my_ios_index](#) ()
- static void [GiNaC::my_ios_callback](#) (std::ios_base::event ev, std::ios_base &s, int i)
- static [print_context](#) * [GiNaC::get_print_context](#) (std::ios_base &s)

- static void [GiNaC::set_print_context](#) (std::ios_base &s, const [print_context](#) &c)
- static unsigned [GiNaC::get_print_options](#) (std::ios_base &s)
- static void [GiNaC::set_print_options](#) (std::ostream &s, unsigned [options](#))
- std::ostream & [GiNaC::operator<<](#) (std::ostream &os, const [ex](#) &e)
- std::ostream & [GiNaC::operator<<](#) (std::ostream &os, const [exvector](#) &e)
- std::ostream & [GiNaC::operator<<](#) (std::ostream &os, const [exset](#) &e)
- std::ostream & [GiNaC::operator<<](#) (std::ostream &os, const [exmap](#) &e)
- std::istream & [GiNaC::operator>>](#) (std::istream &is, [ex](#) &e)
- std::ostream & [GiNaC::dflt](#) (std::ostream &os)
- std::ostream & [GiNaC::latex](#) (std::ostream &os)
- std::ostream & [GiNaC::python](#) (std::ostream &os)
- std::ostream & [GiNaC::python_repr](#) (std::ostream &os)
- std::ostream & [GiNaC::tree](#) (std::ostream &os)
- std::ostream & [GiNaC::csrc](#) (std::ostream &os)
- std::ostream & [GiNaC::csrc_float](#) (std::ostream &os)
- std::ostream & [GiNaC::csrc_double](#) (std::ostream &os)
- std::ostream & [GiNaC::csrc_cl_N](#) (std::ostream &os)
- std::ostream & [GiNaC::index_dimensions](#) (std::ostream &os)
- std::ostream & [GiNaC::no_index_dimensions](#) (std::ostream &os)

7.70.1 Detailed Description

Implementation of [GiNaC](#)'s overloaded operators.

7.71 operators.h File Reference

Interface to [GiNaC](#)'s overloaded operators.

```
#include <iosfwd>
```

Namespaces

- namespace [GiNaC](#)

Functions

- const [ex](#) [GiNaC::operator+](#) (const [ex](#) &lh, const [ex](#) &rh)
- const [ex](#) [GiNaC::operator-](#) (const [ex](#) &lh, const [ex](#) &rh)
- const [ex](#) [GiNaC::operator*](#) (const [ex](#) &lh, const [ex](#) &rh)
- const [ex](#) [GiNaC::operator/](#) (const [ex](#) &lh, const [ex](#) &rh)
- const [numeric](#) [GiNaC::operator+](#) (const [numeric](#) &lh, const [numeric](#) &rh)
- const [numeric](#) [GiNaC::operator-](#) (const [numeric](#) &lh, const [numeric](#) &rh)
- const [numeric](#) [GiNaC::operator*](#) (const [numeric](#) &lh, const [numeric](#) &rh)
- const [numeric](#) [GiNaC::operator/](#) (const [numeric](#) &lh, const [numeric](#) &rh)
- [ex](#) & [GiNaC::operator+=](#) ([ex](#) &lh, const [ex](#) &rh)
- [ex](#) & [GiNaC::operator-=](#) ([ex](#) &lh, const [ex](#) &rh)
- [ex](#) & [GiNaC::operator*=](#) ([ex](#) &lh, const [ex](#) &rh)
- [ex](#) & [GiNaC::operator/=](#) ([ex](#) &lh, const [ex](#) &rh)

- `numeric & GiNaC::operator+= (numeric &lh, const numeric &rh)`
- `numeric & GiNaC::operator-= (numeric &lh, const numeric &rh)`
- `numeric & GiNaC::operator*= (numeric &lh, const numeric &rh)`
- `numeric & GiNaC::operator/= (numeric &lh, const numeric &rh)`
- `const ex GiNaC::operator+ (const ex &lh)`
- `const ex GiNaC::operator- (const ex &lh)`
- `const numeric GiNaC::operator+ (const numeric &lh)`
- `const numeric GiNaC::operator- (const numeric &lh)`
- `ex & GiNaC::operator++ (ex &rh)`
Expression prefix increment.
- `ex & GiNaC::operator-- (ex &rh)`
Expression prefix decrement.
- `const ex GiNaC::operator++ (ex &lh, int)`
Expression postfix increment.
- `const ex GiNaC::operator-- (ex &lh, int)`
Expression postfix decrement.
- `numeric & GiNaC::operator++ (numeric &rh)`
Numeric prefix increment.
- `numeric & GiNaC::operator-- (numeric &rh)`
Numeric prefix decrement.
- `const numeric GiNaC::operator++ (numeric &lh, int)`
Numeric postfix increment.
- `const numeric GiNaC::operator-- (numeric &lh, int)`
Numeric postfix decrement.
- `const relational GiNaC::operator== (const ex &lh, const ex &rh)`
- `const relational GiNaC::operator!= (const ex &lh, const ex &rh)`
- `const relational GiNaC::operator< (const ex &lh, const ex &rh)`
- `const relational GiNaC::operator<= (const ex &lh, const ex &rh)`
- `const relational GiNaC::operator> (const ex &lh, const ex &rh)`
- `const relational GiNaC::operator>= (const ex &lh, const ex &rh)`
- `std::ostream & GiNaC::operator<< (std::ostream &os, const ex &e)`
- `std::istream & GiNaC::operator>> (std::istream &is, ex &e)`
- `std::ostream & GiNaC::dflt (std::ostream &os)`
- `std::ostream & GiNaC::latex (std::ostream &os)`
- `std::ostream & GiNaC::python (std::ostream &os)`
- `std::ostream & GiNaC::python_repr (std::ostream &os)`
- `std::ostream & GiNaC::tree (std::ostream &os)`
- `std::ostream & GiNaC::csrc (std::ostream &os)`
- `std::ostream & GiNaC::csrc_float (std::ostream &os)`
- `std::ostream & GiNaC::csrc_double (std::ostream &os)`
- `std::ostream & GiNaC::csrc_cl_N (std::ostream &os)`
- `std::ostream & GiNaC::index_dimensions (std::ostream &os)`
- `std::ostream & GiNaC::no_index_dimensions (std::ostream &os)`

7.71.1 Detailed Description

Interface to `GiNaC`'s overloaded operators.

7.72 power.cpp File Reference

Implementation of [GiNaC](#)'s symbolic exponentiation ($\text{basis}^{\text{exponent}}$).

```
#include "power.h"
#include "expairseq.h"
#include "add.h"
#include "mul.h"
#include "ncmul.h"
#include "numeric.h"
#include "constant.h"
#include "operators.h"
#include "inifcns.h"
#include "matrix.h"
#include "indexed.h"
#include "symbol.h"
#include "lst.h"
#include "archive.h"
#include "utils.h"
#include "relational.h"
#include "compiler.h"
#include <limits>
#include <stdexcept>
#include <vector>
#include <algorithm>
```

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([power](#), [basic](#), [print_func](#)<[print_dflt](#)>(&[power::do_print_dflt](#)), [print_func](#)<[print_latex](#)>(&[power::do_print_latex](#)), [print_func](#)<[print_csrc](#)>(&[power::do_print_csrc](#)), [print_func](#)<[print_python](#)>(&[power::do_print_python](#)), [print_func](#)<[print_python_repr](#)>(&[power::do_print_python_repr](#)), [print_func](#)<[print_csrc_cl_N](#)>(&[power::do_print_csrc_cl_N](#))) [power](#)
- static void [GiNaC::print_sym_pow](#) (const [print_context](#) &[c](#), const [symbol](#) &[x](#), int [exp](#))
- bool [GiNaC::tryfactsubs](#) (const [ex](#) &[origfactor](#), const [ex](#) &[patternfactor](#), int &[nummatches](#), [exmap](#) &[repls](#))
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([power](#))

7.72.1 Detailed Description

Implementation of [GiNaC](#)'s symbolic exponentiation ($\text{basis}^{\text{exponent}}$).

7.73 power.h File Reference

Interface to [GiNaC](#)'s symbolic exponentiation ($\text{basis}^{\text{exponent}}$).

```
#include "basic.h"
#include "ex.h"
#include "archive.h"
```

Classes

- class [GiNaC::power](#)

This class holds a two-component object, a basis and an exponent representing exponentiation.

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([power](#))
- [ex GiNaC::pow](#) (const [ex](#) &b, const [ex](#) &e)
Symbolic exponentiation.
- `template<typename T1 , typename T2 >`
[ex GiNaC::pow](#) (const T1 &b, const T2 &e)
- [ex GiNaC::sqrt](#) (const [ex](#) &a)
Square root expression.

7.73.1 Detailed Description

Interface to [GiNaC](#)'s symbolic exponentiation ($\text{basis}^{\text{exponent}}$).

7.74 print.cpp File Reference

Implementation of helper classes for expression output.

```
#include "print.h"
#include <iostream>
```

Namespaces

- namespace [GiNaC](#)

Variables

- unsigned [GiNaC::next_print_context_id](#) = 0
Next free ID for [print_context](#) types.

7.74.1 Detailed Description

Implementation of helper classes for expression output.

7.75 print.h File Reference

Definition of helper classes for expression output.

```
#include "class_info.h"
#include <iosfwd>
#include <memory>
```

Classes

- class [GiNaC::print_context_options](#)
This class stores information about a registered [print_context](#) class.
- class [GiNaC::print_options](#)
Flags to control the behavior of a [print_context](#).
- class [GiNaC::print_context](#)
Base class for [print_contexts](#).
- class [GiNaC::print_dflt](#)
Context for default (ginsh-parsable) output.
- class [GiNaC::print_latex](#)
Context for latex-parsable output.
- class [GiNaC::print_python](#)
Context for python pretty-print output.
- class [GiNaC::print_python_repr](#)
Context for python-parsable output.
- class [GiNaC::print_tree](#)
Context for tree-like output for debugging.
- class [GiNaC::print_csrc](#)
Base context for C source output.
- class [GiNaC::print_csrc_float](#)
Context for C source output using float precision.
- class [GiNaC::print_csrc_double](#)
Context for C source output using double precision.
- class [GiNaC::print_csrc_cl_N](#)
Context for C source output using CLN numbers.
- class [GiNaC::print_functor_impl](#)
Base class for [print_functor](#) handlers.
- class [GiNaC::print_ptrfun_handler< T, C >](#)
*[print_functor](#) handler for pointer-to-functions of class *T*, context type *C**
- class [GiNaC::print_memfun_handler< T, C >](#)
*[print_functor](#) handler for member functions of class *T*, context type *C**
- class [GiNaC::print_functor](#)
This class represents a print method for a certain algebraic class and [print_context](#) type.

Namespaces

- namespace [GiNaC](#)

Macros

- `#define GINAC_DECLARE_PRINT_CONTEXT_COMMON(classname)`
Common part of GINAC_DECLARE_PRINT_CONTEXT_BASE and GINAC_DECLARE_PRINT_CONTEXT_DERIVED.
- `#define GINAC_DECLARE_PRINT_CONTEXT_BASE(classname)`
- `#define GINAC_DECLARE_PRINT_CONTEXT(classname, surname)`
Macro for inclusion in the declaration of a print_context class.
- `#define GINAC_IMPLEMENT_PRINT_CONTEXT(classname, surname)`
Macro for inclusion in the implementation of each print_context class.

Typedefs

- `typedef class_info< print_context_options > GiNaC::print_context_class_info`

Functions

- `template<class T >`
`bool GiNaC::is_a (const print_context &obj)`
Check if obj is a T, including base classes.

7.75.1 Detailed Description

Definition of helper classes for expression output.

7.75.2 Macro Definition Documentation

7.75.2.1 GINAC_DECLARE_PRINT_CONTEXT_COMMON

```
#define GINAC_DECLARE_PRINT_CONTEXT_COMMON(  
    classname )
```

Value:

```
public: \
    friend class function_options; \
    friend class registered_class_options; \
    static const GiNaC::print_context_class_info &get_class_info_static(); \
    classname();
```

Common part of GINAC_DECLARE_PRINT_CONTEXT_BASE and GINAC_DECLARE_PRINT_CONTEXT_DERIVED.

7.75.2.2 GINAC_DECLARE_PRINT_CONTEXT_BASE

```
#define GINAC_DECLARE_PRINT_CONTEXT_BASE(  
    classname )
```

Value:

```
GINAC_DECLARE_PRINT_CONTEXT_COMMON(classname) \
virtual const GiNaC::print_context_class_info &get_class_info() const { return  
    classname::get_class_info_static(); } \
virtual const char *class_name() const { return classname::get_class_info_static().options.get_name(); } \
virtual classname * duplicate() const { return new classname(*this); } \
private:
```

7.75.2.3 GINAC_DECLARE_PRINT_CONTEXT

```
#define GINAC_DECLARE_PRINT_CONTEXT(
    classname,
    supertype )
```

Value:

```
GINAC_DECLARE_PRINT_CONTEXT_COMMON(classname) \
typedef supertype inherited; \
const GiNaC::print_context_class_info &get_class_info() const override { return
    classname::get_class_info_static(); } \
const char *class_name() const override { return classname::get_class_info_static().options.get_name();
    } \
classname * duplicate() const override { return new classname(*this); } \
private:
```

Macro for inclusion in the declaration of a `print_context` class.

It declares some functions that are common to all classes derived from 'print_context' as well as all required stuff for the [GiNaC](#) registry.

7.75.2.4 GINAC_IMPLEMENT_PRINT_CONTEXT

```
#define GINAC_IMPLEMENT_PRINT_CONTEXT(
    classname,
    supertype )
```

Value:

```
const GiNaC::print_context_class_info &classname::get_class_info_static() \
{ \
    static GiNaC::print_context_class_info reg_info =
        GiNaC::print_context_class_info(GiNaC::print_context_options(#classname, #supertype,
        GiNaC::next_print_context_id++)); \
    return reg_info; \
}
```

Macro for inclusion in the implementation of each `print_context` class.

7.76 pseries.cpp File Reference

Implementation of class for extended truncated power series and methods for series expansion.

```
#include "pseries.h"
#include "add.h"
#include "inifcns.h"
#include "lst.h"
#include "mul.h"
#include "power.h"
#include "relational.h"
#include "operators.h"
#include "symbol.h"
#include "integral.h"
#include "archive.h"
#include "utils.h"
#include <limits>
#include <numeric>
#include <stdexcept>
```


Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([pseries](#), [basic](#), [print_func](#)< [print_context](#)>(&[pseries::do_print](#)). [print_func](#)< [print_latex](#)>(&[pseries::do_print_latex](#)). [print_func](#)< [print_tree](#)>(&[pseries::do_print_tree](#)). [print_func](#)< [print_python](#)>(&[pseries::do_print_python](#)). [print_func](#)< [print_python_repr](#)>(&[pseries::do_print_python_repr](#)) [pseries](#)
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([pseries](#))

7.76.1 Detailed Description

Implementation of class for extended truncated power series and methods for series expansion.

7.77 pseries.h File Reference

Interface to class for extended truncated power series.

```
#include "basic.h"
#include "expairseq.h"
```

Classes

- class [GiNaC::pseries](#)
This class holds a extended truncated power series (positive and negative integer powers).

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([pseries](#))
- [ex GiNaC::series_to_poly](#) (const [ex](#) &e)
Convert the pseries object embedded in an expression to an ordinary polynomial in the expansion variable.
- bool [GiNaC::is_terminating](#) (const [pseries](#) &s)

7.77.1 Detailed Description

Interface to class for extended truncated power series.

7.78 ptr.h File Reference

Reference-counted pointer template.

```
#include "assertion.h"
#include <cstddef>
#include <functional>
#include <iosfwd>
```

Classes

- class [GiNaC::refcounted](#)
Base class for reference-counted objects.
- class [GiNaC::ptr< T >](#)
Class of (intrusively) reference-counted pointers that support copy-on-write semantics.
- struct [std::less< GiNaC::ptr< T > >](#)
Specialization of std::less for ptr< T > to enable ordering of ptr< T > objects (e.g.

Namespaces

- namespace [GiNaC](#)
- namespace [std](#)

7.78.1 Detailed Description

Reference-counted pointer template.

7.79 registrar.cpp File Reference

[GiNaC](#)'s class registrar (for class basic and all classes derived from it).

```
#include "registrar.h"
```

Namespaces

- namespace [GiNaC](#)

7.79.1 Detailed Description

[GiNaC](#)'s class registrar (for class basic and all classes derived from it).

7.80 registrar.h File Reference

GiNaC's class registrar (for class basic and all classes derived from it).

```
#include "class_info.h"
#include "print.h"
#include <list>
#include <typeinfo>
#include <vector>
```

Classes

- struct [GiNaC::return_type_t](#)
To distinguish between different kinds of non-commutative objects.
- class [GiNaC::registered_class_options](#)
This class stores information about a registered [GiNaC](#) class.

Namespaces

- namespace [GiNaC](#)

Macros

- #define [GINAC_DECLARE_REGISTERED_CLASS_COMMON](#)(classname)
*Common part of GINAC_DECLARE_REGISTERED_CLASS_NO_CTORS and GINAC_DECLARE_REGISTERED↵
_CLASS.*
- #define [GINAC_DECLARE_REGISTERED_CLASS_NO_CTORS](#)(classname, supertype)
Primary macro for inclusion in the declaration of each registered class.
- #define [GINAC_DECLARE_REGISTERED_CLASS](#)(classname, supertype)
Macro for inclusion in the declaration of each registered class.
- #define [GINAC_IMPLEMENT_REGISTERED_CLASS](#)(classname, supertype) [GiNaC::registered_class_info](#)
classname::reg_info = [GiNaC::registered_class_info](#)([GiNaC::registered_class_options](#)(#classname, #super-
name, typeid(classname)));
Macro for inclusion in the implementation of each registered class.
- #define [GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#)(classname, supertype, options) [GiNaC::registered_class_info](#)
classname::reg_info = [GiNaC::registered_class_info](#)([GiNaC::registered_class_options](#)(#classname, #super-
name, typeid(classname)).options);
Macro for inclusion in the implementation of each registered class.
- #define [GINAC_IMPLEMENT_REGISTERED_CLASS_OPT_T](#)(classname, supertype, options) [GiNaC::registered_class_info](#)
classname::reg_info = [GiNaC::registered_class_info](#)([GiNaC::registered_class_options](#)(#classname, #super-
name, typeid(classname)).options);
Macro for inclusion in the implementation of each registered class.

Typedefs

- typedef [class_info](#)< [registered_class_options](#) > [GiNaC::registered_class_info](#)

Functions

- `template<typename T>`
`return_type_t GiNaC::make_return_type_t (const unsigned rl=0)`
- `template<class Alg, class Ctx, class T, class C>`
`void GiNaC::set_print_func (void f(const T &, const C &c, unsigned))`
Add or replace a print method.
- `template<class Alg, class Ctx, class T, class C>`
`void GiNaC::set_print_func (void(T::*f)(const C &, unsigned))`
Add or replace a print method.

7.80.1 Detailed Description

GiNaC's class registrar (for class basic and all classes derived from it).

7.80.2 Macro Definition Documentation

7.80.2.1 GINAC_DECLARE_REGISTERED_CLASS_COMMON

```
#define GINAC_DECLARE_REGISTERED_CLASS_COMMON(  
    classname )
```

Value:

```
private: \
    static GiNaC::registered_class_info reg_info; \
public: \
    static GiNaC::registered_class_info &get_class_info_static() { return reg_info; } \
    class visitor { \
    public: \
        virtual void visit(const classname &) = 0; \
        virtual ~visitor() {}; \
    };
```

Common part of GINAC_DECLARE_REGISTERED_CLASS_NO_CTORS and GINAC_DECLARE_REGISTERED_↵
_CLASS.

7.80.2.2 GINAC_DECLARE_REGISTERED_CLASS_NO_CTORS

```
#define GINAC_DECLARE_REGISTERED_CLASS_NO_CTORS(  
    classname,  
    supertype )
```

Value:

```
GINAC_DECLARE_REGISTERED_CLASS_COMMON(classname) \
typedef supertype inherited; \
virtual const GiNaC::registered_class_info &get_class_info() const { return \
    classname::get_class_info_static(); } \
virtual GiNaC::registered_class_info &get_class_info() { return classname::get_class_info_static(); } \
virtual const char *class_name() const { return classname::get_class_info_static().options.get_name(); } \
private:
```

Primary macro for inclusion in the declaration of each registered class.

7.80.2.3 GINAC_DECLARE_REGISTERED_CLASS

```
#define GINAC_DECLARE_REGISTERED_CLASS(  
    classname,  
    supertype )
```

Value:

```
GINAC_DECLARE_REGISTERED_CLASS_COMMON(classname) \  
template<class B, typename... Args> friend B & dynallocate(Args &&... args); \  
typedef supertype inherited; \  
classname(); \  
classname * duplicate() const override { \  
    classname * bp = new classname(*this); \  
    bp->setflag(GiNaC::status_flags::dynallocated); \  
    return bp; \  
} \  
void accept(GiNaC::visitor & v) const override \  
{ \  
    if (visitor *p = dynamic_cast<visitor *>(&v)) \  
        p->visit(*this); \  
    else \  
        inherited::accept(v); \  
} \  
const GiNaC::registered_class_info &get_class_info() const override { return \  
    classname::get_class_info_static(); } \  
GiNaC::registered_class_info &get_class_info() override { return classname::get_class_info_static(); } \  
const char *class_name() const override { return classname::get_class_info_static().options.get_name(); \  
} \  
protected: \  
    int compare_same_type(const GiNaC::basic & other) const override; \  
private:
```

Macro for inclusion in the declaration of each registered class.

It declares some functions that are common to all classes derived from 'basic' as well as all required stuff for the [GiNaC](#) class registry (mainly needed for archiving).

7.80.2.4 GINAC_IMPLEMENT_REGISTERED_CLASS

```
#define GINAC_IMPLEMENT_REGISTERED_CLASS(  
    classname,  
    supertype ) GiNaC::registered_class_info classname::reg_info = GiNaC::registered_class_info(GiNaC::get_class_info_static(), \  
#supertype, typeid(classname));
```

Macro for inclusion in the implementation of each registered class.

7.80.2.5 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT

```
#define GINAC_IMPLEMENT_REGISTERED_CLASS_OPT(  
    classname,  
    supertype,  
    options ) GiNaC::registered_class_info classname::reg_info = GiNaC::registered_class_info(GiNaC::get_class_info_static(), \  
#supertype, typeid(classname)).options);
```

Macro for inclusion in the implementation of each registered class.

Additional options can be specified.

7.80.2.6 GINAC_IMPLEMENT_REGISTERED_CLASS_OPT_T

```
#define GINAC_IMPLEMENT_REGISTERED_CLASS_OPT_T(  
    classname,  
    supename,  
    options ) GiNaC::registered_class_info classname::reg_info = GiNaC::registered_class_info(GiNaC  
#supename, typeid(classname)).options);
```

Macro for inclusion in the implementation of each registered class.

Additional options can be specified.

7.81 relational.cpp File Reference

Implementation of relations between expressions.

```
#include "relational.h"  
#include "operators.h"  
#include "numeric.h"  
#include "archive.h"  
#include "utils.h"  
#include "hash_seed.h"  
#include <stdexcept>
```

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([relational](#), [basic](#), [print_func](#)< [print_context](#)>(&[relational::do_print](#)). [print_func](#)< [print_tree](#)>(&[relational::do_print_tree](#)). [print_func](#)< [print_python_repr](#)>(&[relational::do_print_python_repr](#))) [relational](#)
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([relational](#))
- static void [GiNaC::print_operator](#) (const [print_context](#) &c, [relational::operators](#) o)

7.81.1 Detailed Description

Implementation of relations between expressions.

7.82 relational.h File Reference

Interface to relations between expressions.

```
#include "basic.h"  
#include "ex.h"  
#include "archive.h"
```

Classes

- class [GiNaC::relational](#)
This class holds a relation consisting of two expressions and a logical relation between them.
- struct [GiNaC::relational::safe_bool_helper](#)

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([relational](#))

7.82.1 Detailed Description

Interface to relations between expressions.

7.83 remember.cpp File Reference

Implementation of helper classes for using the remember option in [GiNaC](#) functions.

```
#include "function.h"
#include "utils.h"
#include "remember.h"
#include <stdexcept>
```

Namespaces

- namespace [GiNaC](#)

7.83.1 Detailed Description

Implementation of helper classes for using the remember option in [GiNaC](#) functions.

7.84 remember.h File Reference

Interface to helper classes for using the remember option in [GiNaC](#) functions.

```
#include <iosfwd>
#include <list>
#include <vector>
```

Classes

- class [GiNaC::remember_table_entry](#)
A single entry in the remember table of a function.
- class [GiNaC::remember_table_list](#)
A list of entries in the remember table having some least significant bits of the hashvalue in common.
- class [GiNaC::remember_table](#)
The remember table is organized like an n-fold associative cache in a microprocessor.

Namespaces

- namespace [GiNaC](#)

7.84.1 Detailed Description

Interface to helper classes for using the remember option in [GiNaC](#) functions.

7.85 structure.h File Reference

Wrapper template for making [GiNaC](#) classes out of C++ structures.

```
#include "ex.h"
#include "ncmul.h"
#include "numeric.h"
#include "operators.h"
#include "print.h"
#include <functional>
```

Classes

- class [GiNaC::compare_all_equal< T >](#)
Comparison policy: all structures of one type are equal.
- class [GiNaC::compare_std_less< T >](#)
Comparison policy: use std::equal_to/std::less (defaults to operators == and <) to compare structures.
- class [GiNaC::compare_bitwise< T >](#)
Comparison policy: use bit-wise comparison to compare structures.
- class [GiNaC::structure< T, ComparisonPolicy >](#)
Wrapper template for making [GiNaC](#) classes out of C++ structures.

Namespaces

- namespace [GiNaC](#)

7.85.1 Detailed Description

Wrapper template for making [GiNaC](#) classes out of C++ structures.

7.86 symbol.cpp File Reference

Implementation of [GiNaC](#)'s symbolic objects.

```
#include "symbol.h"
#include "lst.h"
#include "archive.h"
#include "utils.h"
#include "hash_seed.h"
#include "inifcns.h"
#include <map>
#include <stdexcept>
#include <string>
```

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([symbol](#), [basic](#), [print_func< print_context >\(&symbol::do_print\)](#), [print_func< print_latex >\(&symbol::do_print_latex\)](#), [print_func< print_tree >\(&symbol::do_print_tree\)](#), [print_func< print_python_repr >\(&symbol::do_print_python_repr\)](#)) [symbol](#)
- static const std::string & [GiNaC::get_default_TeX_name](#) (const std::string &name)
Return default TeX name for symbol.
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([symbol](#))
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([realsymbol](#))
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([possymbol](#))

7.86.1 Detailed Description

Implementation of [GiNaC](#)'s symbolic objects.

7.87 symbol.h File Reference

Interface to [GiNaC](#)'s symbolic objects.

```
#include "basic.h"
#include "ex.h"
#include "ptr.h"
#include "archive.h"
#include <string>
#include <typeinfo>
```

Classes

- class [GiNaC::symbol](#)
Basic CAS symbol.
- class [GiNaC::realsymbol](#)
Specialization of symbol to real domain.
- class [GiNaC::possymbol](#)
Specialization of symbol to real positive domain.

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([symbol](#))
- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([realsymbol](#))
- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([possymbol](#))

7.87.1 Detailed Description

Interface to [GiNaC](#)'s symbolic objects.

7.88 symmetry.cpp File Reference

Implementation of [GiNaC](#)'s symmetry definitions.

```
#include "symmetry.h"
#include "lst.h"
#include "add.h"
#include "numeric.h"
#include "operators.h"
#include "archive.h"
#include "utils.h"
#include "hash_seed.h"
#include <functional>
#include <limits>
#include <stdexcept>
```

Classes

- class [GiNaC::sy_is_less](#)
- class [GiNaC::sy_swap](#)

Namespaces

- namespace [GiNaC](#)

Functions

- `GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT` (`symmetry`, `basic`, `print_func` < `print_context` >(&`symmetry::do_print`). `print_func` < `print_tree` >(&`symmetry::do_print_tree`)) `symmetry`
- `GiNaC::GINAC_BIND_UNARCHIVER` (`symmetry`)
- static const `symmetry` & `GiNaC::index0` ()
- static const `symmetry` & `GiNaC::index1` ()
- static const `symmetry` & `GiNaC::index2` ()
- static const `symmetry` & `GiNaC::index3` ()
- const `symmetry` & `GiNaC::not_symmetric` ()
- const `symmetry` & `GiNaC::symmetric2` ()
- const `symmetry` & `GiNaC::symmetric3` ()
- const `symmetry` & `GiNaC::symmetric4` ()
- const `symmetry` & `GiNaC::antisymmetric2` ()
- const `symmetry` & `GiNaC::antisymmetric3` ()
- const `symmetry` & `GiNaC::antisymmetric4` ()
- int `GiNaC::canonicalize` (`exvector::iterator` v, const `symmetry` &`symm`)
Canonicalize the order of elements of an expression vector, according to the symmetry properties defined in a symmetry tree.
- static `ex` `GiNaC::symm` (const `ex` &e, `exvector::const_iterator` first, `exvector::const_iterator` last, bool asymmetric)
- `ex` `GiNaC::symmetrize` (const `ex` &e, `exvector::const_iterator` first, `exvector::const_iterator` last)
Symmetrize expression over a set of objects (symbols, indices).
- `ex` `GiNaC::antisymmetrize` (const `ex` &e, `exvector::const_iterator` first, `exvector::const_iterator` last)
Antisymmetrize expression over a set of objects (symbols, indices).
- `ex` `GiNaC::symmetrize_cyclic` (const `ex` &e, `exvector::const_iterator` first, `exvector::const_iterator` last)
Symmetrize expression by cyclic permutation over a set of objects (symbols, indices).

7.88.1 Detailed Description

Implementation of `GiNaC`'s symmetry definitions.

7.89 symmetry.h File Reference

Interface to `GiNaC`'s symmetry definitions.

```
#include "ex.h"
#include "archive.h"
#include <set>
```

Classes

- class `GiNaC::symmetry`
This class describes the symmetry of a group of indices.

Namespaces

- namespace `GiNaC`

Functions

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([symmetry](#))
- [symmetry GiNaC::sy_none](#) ()
- [symmetry GiNaC::sy_none](#) (const [symmetry](#) &c1, const [symmetry](#) &c2)
- [symmetry GiNaC::sy_none](#) (const [symmetry](#) &c1, const [symmetry](#) &c2, const [symmetry](#) &c3)
- [symmetry GiNaC::sy_none](#) (const [symmetry](#) &c1, const [symmetry](#) &c2, const [symmetry](#) &c3, const [symmetry](#) &c4)
- [symmetry GiNaC::sy_symm](#) ()
- [symmetry GiNaC::sy_symm](#) (const [symmetry](#) &c1, const [symmetry](#) &c2)
- [symmetry GiNaC::sy_symm](#) (const [symmetry](#) &c1, const [symmetry](#) &c2, const [symmetry](#) &c3)
- [symmetry GiNaC::sy_symm](#) (const [symmetry](#) &c1, const [symmetry](#) &c2, const [symmetry](#) &c3, const [symmetry](#) &c4)
- [symmetry GiNaC::sy_anti](#) ()
- [symmetry GiNaC::sy_anti](#) (const [symmetry](#) &c1, const [symmetry](#) &c2)
- [symmetry GiNaC::sy_anti](#) (const [symmetry](#) &c1, const [symmetry](#) &c2, const [symmetry](#) &c3)
- [symmetry GiNaC::sy_anti](#) (const [symmetry](#) &c1, const [symmetry](#) &c2, const [symmetry](#) &c3, const [symmetry](#) &c4)
- [symmetry GiNaC::sy_cycl](#) ()
- [symmetry GiNaC::sy_cycl](#) (const [symmetry](#) &c1, const [symmetry](#) &c2)
- [symmetry GiNaC::sy_cycl](#) (const [symmetry](#) &c1, const [symmetry](#) &c2, const [symmetry](#) &c3)
- [symmetry GiNaC::sy_cycl](#) (const [symmetry](#) &c1, const [symmetry](#) &c2, const [symmetry](#) &c3, const [symmetry](#) &c4)
- const [symmetry](#) & [GiNaC::not_symmetric](#) ()
- const [symmetry](#) & [GiNaC::symmetric2](#) ()
- const [symmetry](#) & [GiNaC::symmetric3](#) ()
- const [symmetry](#) & [GiNaC::symmetric4](#) ()
- const [symmetry](#) & [GiNaC::antisymmetric2](#) ()
- const [symmetry](#) & [GiNaC::antisymmetric3](#) ()
- const [symmetry](#) & [GiNaC::antisymmetric4](#) ()
- int [GiNaC::canonicalize](#) (exvector::iterator v, const [symmetry](#) &symm)
Canonicalize the order of elements of an expression vector, according to the symmetry properties defined in a symmetry tree.
- [ex GiNaC::symmetrize](#) (const [ex](#) &e, exvector::const_iterator first, exvector::const_iterator last)
Symmetrize expression over a set of objects (symbols, indices).
- [ex GiNaC::symmetrize](#) (const [ex](#) &e, const [exvector](#) &v)
Symmetrize expression over a set of objects (symbols, indices).
- [ex GiNaC::antisymmetrize](#) (const [ex](#) &e, exvector::const_iterator first, exvector::const_iterator last)
Antisymmetrize expression over a set of objects (symbols, indices).
- [ex GiNaC::antisymmetrize](#) (const [ex](#) &e, const [exvector](#) &v)
Antisymmetrize expression over a set of objects (symbols, indices).
- [ex GiNaC::symmetrize_cyclic](#) (const [ex](#) &e, exvector::const_iterator first, exvector::const_iterator last)
Symmetrize expression by cyclic permutation over a set of objects (symbols, indices).
- [ex GiNaC::symmetrize_cyclic](#) (const [ex](#) &e, const [exvector](#) &v)
Symmetrize expression by cyclic permutation over a set of objects (symbols, indices).

7.89.1 Detailed Description

Interface to [GiNaC](#)'s symmetry definitions.

7.90 tensor.cpp File Reference

Implementation of [GiNaC](#)'s special tensors.

```
#include "tensor.h"
#include "idx.h"
#include "indexed.h"
#include "symmetry.h"
#include "relational.h"
#include "operators.h"
#include "lst.h"
#include "numeric.h"
#include "matrix.h"
#include "archive.h"
#include "utils.h"
#include <stdexcept>
#include <vector>
```

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([tensdelta](#), [tensor](#), [print_func< print_dflt >\(&tensdelta::do_print\)](#), [print_func< print_latex >\(&tensdelta::do_print_latex\)](#)) [GINAC_IMPLEMENT_REGISTERED_CLASS_OPT\(tensmetric\)](#)
- [GiNaC::print_func< print_dflt > \(&tensmetric::do_print\)](#), [print_func< print_latex >\(&tensmetric\)](#)
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([minkmetric](#))
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([tensepsilon](#))
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([tensdelta](#))
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([tensmetric](#))
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([spinmetric](#))
- [ex GiNaC::delta_tensor](#) (const [ex](#) &i1, const [ex](#) &i2)
Create a delta tensor with specified indices.
- [ex GiNaC::metric_tensor](#) (const [ex](#) &i1, const [ex](#) &i2)
Create a symmetric metric tensor with specified indices.
- [ex GiNaC::lorentz_g](#) (const [ex](#) &i1, const [ex](#) &i2, bool pos_sig=false)
Create a Minkowski metric tensor with specified indices.
- [ex GiNaC::spinor_metric](#) (const [ex](#) &i1, const [ex](#) &i2)
Create a spinor metric tensor with specified indices.
- [ex GiNaC::epsilon_tensor](#) (const [ex](#) &i1, const [ex](#) &i2)
Create an epsilon tensor in a Euclidean space with two indices.
- [ex GiNaC::epsilon_tensor](#) (const [ex](#) &i1, const [ex](#) &i2, const [ex](#) &i3)
Create an epsilon tensor in a Euclidean space with three indices.
- [ex GiNaC::lorentz_eps](#) (const [ex](#) &i1, const [ex](#) &i2, const [ex](#) &i3, const [ex](#) &i4, bool pos_sig=false)
Create an epsilon tensor in a Minkowski space with four indices.

7.90.1 Detailed Description

Implementation of [GiNaC](#)'s special tensors.

7.91 tensor.h File Reference

Interface to [GiNaC](#)'s special tensors.

```
#include "ex.h"
#include "archive.h"
```

Classes

- class [GiNaC::tensor](#)
This class holds one of [GiNaC](#)'s predefined special tensors such as the delta and the metric tensors.
- class [GiNaC::tensdelta](#)
This class represents the delta tensor.
- class [GiNaC::tensmetric](#)
This class represents a general metric tensor which can be used to raise/lower indices.
- class [GiNaC::minkmetric](#)
This class represents a Minkowski metric tensor.
- class [GiNaC::spinmetric](#)
This class represents an antisymmetric spinor metric tensor which can be used to raise/lower indices of 2-component Weyl spinors.
- class [GiNaC::tensepsilon](#)
This class represents the totally antisymmetric epsilon tensor.

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([tensdelta](#))
- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([tensmetric](#))
- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([minkmetric](#))
- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([spinmetric](#))
- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([tensepsilon](#))
- [ex GiNaC::delta_tensor](#) (const [ex](#) &i1, const [ex](#) &i2)
Create a delta tensor with specified indices.
- [ex GiNaC::metric_tensor](#) (const [ex](#) &i1, const [ex](#) &i2)
Create a symmetric metric tensor with specified indices.
- [ex GiNaC::lorentz_g](#) (const [ex](#) &i1, const [ex](#) &i2, bool pos_sig=false)
Create a Minkowski metric tensor with specified indices.
- [ex GiNaC::spinor_metric](#) (const [ex](#) &i1, const [ex](#) &i2)
Create a spinor metric tensor with specified indices.
- [ex GiNaC::epsilon_tensor](#) (const [ex](#) &i1, const [ex](#) &i2)
Create an epsilon tensor in a Euclidean space with two indices.
- [ex GiNaC::epsilon_tensor](#) (const [ex](#) &i1, const [ex](#) &i2, const [ex](#) &i3)
Create an epsilon tensor in a Euclidean space with three indices.
- [ex GiNaC::lorentz_eps](#) (const [ex](#) &i1, const [ex](#) &i2, const [ex](#) &i3, const [ex](#) &i4, bool pos_sig=false)
Create an epsilon tensor in a Minkowski space with four indices.

7.91.1 Detailed Description

Interface to [GiNaC](#)'s special tensors.

7.92 utils.cpp File Reference

Implementation of several small and furry utilities needed within [GiNaC](#) but not of any interest to the user of the library.

```
#include "ex.h"
#include "numeric.h"
#include "utils.h"
#include "version.h"
```

Namespaces

- namespace [GiNaC](#)

Functions

- unsigned [GiNaC::log2](#) (unsigned n)
Integer binary logarithm.
- const [numeric GiNaC::multinomial_coefficient](#) (const std::vector< unsigned > &p)
Compute the multinomial coefficient $n!/(p_1! * p_2! * \dots * p_k!)$ where $n = p_1 + p_2 + \dots + p_k$, i.e.

Variables

- const int [GiNaC::version_major](#) = GINACLIB_MAJOR_VERSION
- const int [GiNaC::version_minor](#) = GINACLIB_MINOR_VERSION
- const int [GiNaC::version_micro](#) = GINACLIB_MICRO_VERSION
- const [numeric](#) * [GiNaC::_num_120_p](#)
- const [ex](#) [GiNaC::_ex_120](#) = [ex](#)(*[_num_120_p](#))
- const [numeric](#) * [GiNaC::_num_60_p](#)
- const [ex](#) [GiNaC::_ex_60](#) = [ex](#)(*[_num_60_p](#))
- const [numeric](#) * [GiNaC::_num_48_p](#)
- const [ex](#) [GiNaC::_ex_48](#) = [ex](#)(*[_num_48_p](#))
- const [numeric](#) * [GiNaC::_num_30_p](#)
- const [ex](#) [GiNaC::_ex_30](#) = [ex](#)(*[_num_30_p](#))
- const [numeric](#) * [GiNaC::_num_25_p](#)
- const [ex](#) [GiNaC::_ex_25](#) = [ex](#)(*[_num_25_p](#))
- const [numeric](#) * [GiNaC::_num_24_p](#)
- const [ex](#) [GiNaC::_ex_24](#) = [ex](#)(*[_num_24_p](#))
- const [numeric](#) * [GiNaC::_num_20_p](#)
- const [ex](#) [GiNaC::_ex_20](#) = [ex](#)(*[_num_20_p](#))
- const [numeric](#) * [GiNaC::_num_18_p](#)
- const [ex](#) [GiNaC::_ex_18](#) = [ex](#)(*[_num_18_p](#))
- const [numeric](#) * [GiNaC::_num_15_p](#)
- const [ex](#) [GiNaC::_ex_15](#) = [ex](#)(*[_num_15_p](#))
- const [numeric](#) * [GiNaC::_num_12_p](#)

- `const ex GiNaC::_ex_12 = ex(*_num_12_p)`
- `const numeric * GiNaC::_num_11_p`
- `const ex GiNaC::_ex_11 = ex(*_num_11_p)`
- `const numeric * GiNaC::_num_10_p`
- `const ex GiNaC::_ex_10 = ex(*_num_10_p)`
- `const numeric * GiNaC::_num_9_p`
- `const ex GiNaC::_ex_9 = ex(*_num_9_p)`
- `const numeric * GiNaC::_num_8_p`
- `const ex GiNaC::_ex_8 = ex(*_num_8_p)`
- `const numeric * GiNaC::_num_7_p`
- `const ex GiNaC::_ex_7 = ex(*_num_7_p)`
- `const numeric * GiNaC::_num_6_p`
- `const ex GiNaC::_ex_6 = ex(*_num_6_p)`
- `const numeric * GiNaC::_num_5_p`
- `const ex GiNaC::_ex_5 = ex(*_num_5_p)`
- `const numeric * GiNaC::_num_4_p`
- `const ex GiNaC::_ex_4 = ex(*_num_4_p)`
- `const numeric * GiNaC::_num_3_p`
- `const ex GiNaC::_ex_3 = ex(*_num_3_p)`
- `const numeric * GiNaC::_num_2_p`
- `const ex GiNaC::_ex_2 = ex(*_num_2_p)`
- `const numeric * GiNaC::_num_1_p`
- `const ex GiNaC::_ex_1 = ex(*_num_1_p)`
- `const numeric * GiNaC::_num_1_2_p`
- `const ex GiNaC::_ex_1_2 = ex(*_num_1_2_p)`
- `const numeric * GiNaC::_num_1_3_p`
- `const ex GiNaC::_ex_1_3 = ex(*_num_1_3_p)`
- `const numeric * GiNaC::_num_1_4_p`
- `const ex GiNaC::_ex_1_4 = ex(*_num_1_4_p)`
- `const numeric * GiNaC::_num0_p`
- `const ex GiNaC::_ex0 = ex(*_num0_p)`
- `const numeric * GiNaC::_num1_4_p`
- `const ex GiNaC::_ex1_4 = ex(*_num1_4_p)`
- `const numeric * GiNaC::_num1_3_p`
- `const ex GiNaC::_ex1_3 = ex(*_num1_3_p)`
- `const numeric * GiNaC::_num1_2_p`
- `const ex GiNaC::_ex1_2 = ex(*_num1_2_p)`
- `const numeric * GiNaC::_num1_p`
- `const ex GiNaC::_ex1 = ex(*_num1_p)`
- `const numeric * GiNaC::_num2_p`
- `const ex GiNaC::_ex2 = ex(*_num2_p)`
- `const numeric * GiNaC::_num3_p`
- `const ex GiNaC::_ex3 = ex(*_num3_p)`
- `const numeric * GiNaC::_num4_p`
- `const ex GiNaC::_ex4 = ex(*_num4_p)`
- `const numeric * GiNaC::_num5_p`
- `const ex GiNaC::_ex5 = ex(*_num5_p)`
- `const numeric * GiNaC::_num6_p`
- `const ex GiNaC::_ex6 = ex(*_num6_p)`
- `const numeric * GiNaC::_num7_p`
- `const ex GiNaC::_ex7 = ex(*_num7_p)`
- `const numeric * GiNaC::_num8_p`
- `const ex GiNaC::_ex8 = ex(*_num8_p)`
- `const numeric * GiNaC::_num9_p`
- `const ex GiNaC::_ex9 = ex(*_num9_p)`

- const [numeric](#) * [GiNaC::_num10_p](#)
- const [ex](#) [GiNaC::_ex10](#) = [ex](#)(*[_num10_p](#))
- const [numeric](#) * [GiNaC::_num11_p](#)
- const [ex](#) [GiNaC::_ex11](#) = [ex](#)(*[_num11_p](#))
- const [numeric](#) * [GiNaC::_num12_p](#)
- const [ex](#) [GiNaC::_ex12](#) = [ex](#)(*[_num12_p](#))
- const [numeric](#) * [GiNaC::_num15_p](#)
- const [ex](#) [GiNaC::_ex15](#) = [ex](#)(*[_num15_p](#))
- const [numeric](#) * [GiNaC::_num18_p](#)
- const [ex](#) [GiNaC::_ex18](#) = [ex](#)(*[_num18_p](#))
- const [numeric](#) * [GiNaC::_num20_p](#)
- const [ex](#) [GiNaC::_ex20](#) = [ex](#)(*[_num20_p](#))
- const [numeric](#) * [GiNaC::_num24_p](#)
- const [ex](#) [GiNaC::_ex24](#) = [ex](#)(*[_num24_p](#))
- const [numeric](#) * [GiNaC::_num25_p](#)
- const [ex](#) [GiNaC::_ex25](#) = [ex](#)(*[_num25_p](#))
- const [numeric](#) * [GiNaC::_num30_p](#)
- const [ex](#) [GiNaC::_ex30](#) = [ex](#)(*[_num30_p](#))
- const [numeric](#) * [GiNaC::_num48_p](#)
- const [ex](#) [GiNaC::_ex48](#) = [ex](#)(*[_num48_p](#))
- const [numeric](#) * [GiNaC::_num60_p](#)
- const [ex](#) [GiNaC::_ex60](#) = [ex](#)(*[_num60_p](#))
- const [numeric](#) * [GiNaC::_num120_p](#)
- const [ex](#) [GiNaC::_ex120](#) = [ex](#)(*[_num120_p](#))

7.92.1 Detailed Description

Implementation of several small and furry utilities needed within [GiNaC](#) but not of any interest to the user of the library.

7.93 utils.h File Reference

Interface to several small and furry utilities needed within [GiNaC](#) but not of any interest to the user of the library.

```
#include "assertion.h"
#include <functional>
#include <cstdint>
#include <string>
```

Classes

- class [GiNaC::dunno](#)
Exception class thrown by functions to signal unimplemented functionality so the expression may just be .hold()
- class [GiNaC::basic_partition_generator](#)
Base class for generating all bounded combinatorial partitions of an integer n with exactly m parts in non-decreasing order.
- struct [GiNaC::basic_partition_generator::mpartition2](#)
- class [GiNaC::partition_with_zero_parts_generator](#)

Generate all bounded combinatorial partitions of an integer n with exactly m parts (including zero parts) in non-decreasing order.

- class [GiNaC::partition_generator](#)

Generate all bounded combinatorial partitions of an integer n with exactly m parts (not including zero parts) in non-decreasing order.

- class [GiNaC::composition_generator](#)

Generate all compositions of a partition of an integer n , starting with the compositions which has non-decreasing order.

- struct [GiNaC::composition_generator::coolmulti](#)
- struct [GiNaC::composition_generator::coolmulti::element](#)

Namespaces

- namespace [GiNaC](#)

Macros

- #define [DEFAULT_CTOR](#)(classname) classname::classname() { setflag(status_flags::evaluated | status_↔ flags::expanded); }
- #define [DEFAULT_COMPARE](#)(classname)
- #define [DEFAULT_PRINT](#)(classname, text)
- #define [DEFAULT_PRINT_LATEX](#)(classname, text, latex)

Functions

- unsigned [GiNaC::log2](#) (unsigned n)
Integer binary logarithm.
- unsigned [GiNaC::rotate_left](#) (unsigned n)
Rotate bits of unsigned value by one bit to the left.
- template<class T >
int [GiNaC::compare_pointers](#) (const T *a, const T *b)
Compare two pointers (just to establish some sort of canonical order).
- unsigned [GiNaC::golden_ratio_hash](#) (uintptr_t n)
Truncated multiplication with golden ratio, for computing hash values.
- template<class It >
int [GiNaC::permutation_sign](#) (It first, It last)
- template<class It, class Cmp, class Swap >
int [GiNaC::permutation_sign](#) (It first, It last, Cmp comp, Swap swapit)
- template<class It, class Cmp, class Swap >
void [GiNaC::shaker_sort](#) (It first, It last, Cmp comp, Swap swapit)
- template<class It, class Swap >
void [GiNaC::cyclic_permutation](#) (It first, It last, It new_first, Swap swapit)
- const numeric [GiNaC::multinomial_coefficient](#) (const std::vector< unsigned > &p)
Compute the multinomial coefficient $n!/(p_1! * p_2! * \dots * p_k!)$ where $n = p_1 + p_2 + \dots + p_k$, i.e.

7.93.1 Detailed Description

Interface to several small and furry utilities needed within [GiNaC](#) but not of any interest to the user of the library.

7.93.2 Macro Definition Documentation

7.93.2.1 DEFAULT_CTOR

```
#define DEFAULT_CTOR(  
    classname ) classname::classname() { setflag(status_flags::evaluated | status_↵  
flags::expanded); }
```

7.93.2.2 DEFAULT_COMPARE

```
#define DEFAULT_COMPARE(  
    classname )
```

Value:

```
int classname::compare_same_type(const basic & other) const \
{ \
    /* by default, the objects are always identical */ \
    return 0; \
}
```

7.93.2.3 DEFAULT_PRINT

```
#define DEFAULT_PRINT(  
    classname,  
    text )
```

Value:

```
void classname::do_print(const print_context & c, unsigned level) const \
{ \
    c.s << text; \
}
```

7.93.2.4 DEFAULT_PRINT_LATEX

```
#define DEFAULT_PRINT_LATEX(  
    classname,  
    text,  
    latex )
```

Value:

```
DEFAULT_PRINT(classname, text) \
void classname::do_print_latex(const print_latex & c, unsigned level) const \
{ \
    c.s << latex; \
}
```

7.94 utils_multi_iterator.h File Reference

Utilities for summing over multiple indices.

```
#include <cstddef>  
#include <vector>  
#include <ostream>  
#include <iterator>
```

Classes

- class [GiNaC::has_distance< T >](#)
SFINAE test for distance.
- class [GiNaC::basic_multi_iterator< T >](#)
basic_multi_iterator is a base class.
- class [GiNaC::multi_iterator_ordered< T >](#)
The class multi_iterator_ordered defines a multi_iterator $(i_1, i_2, \dots, i_k)$, such that.
- class [GiNaC::multi_iterator_ordered_eq< T >](#)
The class multi_iterator_ordered_eq defines a multi_iterator $(i_1, i_2, \dots, i_k)$, such that.
- class [GiNaC::multi_iterator_ordered_eq_indv< T >](#)
The class multi_iterator_ordered_eq_indv defines a multi_iterator $(i_1, i_2, \dots, i_k)$, such that.
- class [GiNaC::multi_iterator_counter< T >](#)
The class multi_iterator_counter defines a multi_iterator $(i_1, i_2, \dots, i_k)$, such that.
- class [GiNaC::multi_iterator_counter_indv< T >](#)
The class multi_iterator_counter_indv defines a multi_iterator $(i_1, i_2, \dots, i_k)$, such that.
- class [GiNaC::multi_iterator_permutation< T >](#)
The class multi_iterator_permutation defines a multi_iterator $(i_1, i_2, \dots, i_k)$, for which.
- class [GiNaC::multi_iterator_shuffle< T >](#)
The class multi_iterator_shuffle defines a multi_iterator, which runs over all shuffles of a and b.
- class [GiNaC::multi_iterator_shuffle_prime< T >](#)
The class multi_iterator_shuffle_prime defines a multi_iterator, which runs over all shuffles of a and b, excluding the first one (a,b).

Namespaces

- namespace [GiNaC](#)

Functions

- `template<typename T >`
`std::enable_if< has\_distance< T >::value, typename std::iterator_traits< T >::difference_type >::type`
`GiNaC::format\_index\_value (const T &a, const T &b)`
For printing a multi-index: If the templates are used, where T is an iterator, printing the address where the iterator points to is not meaningful.
- `template<typename T >`
`std::enable_if<!has\_distance< T >::value, T >::type GiNaC::format\_index\_value (const T &a, const T &b)`
For all other cases we simply print the value.
- `template<class T >`
`std::ostream & GiNaC::operator<< (std::ostream &os, const basic\_multi\_iterator< T > &v)`
Output operator.
- `template<class T >`
`std::ostream & GiNaC::operator<< (std::ostream &os, const multi\_iterator\_ordered< T > &v)`
Output operator.
- `template<class T >`
`std::ostream & GiNaC::operator<< (std::ostream &os, const multi\_iterator\_ordered\_eq< T > &v)`
Output operator.
- `template<class T >`
`std::ostream & GiNaC::operator<< (std::ostream &os, const multi\_iterator\_ordered\_eq\_indv< T > &v)`
Output operator.

- `template<class T >`
`std::ostream & GiNaC::operator<< (std::ostream &os, const multi_iterator_counter< T > &v)`
Output operator.
- `template<class T >`
`std::ostream & GiNaC::operator<< (std::ostream &os, const multi_iterator_counter_indv< T > &v)`
Output operator.
- `template<class T >`
`std::ostream & GiNaC::operator<< (std::ostream &os, const multi_iterator_permutation< T > &v)`
Output operator.
- `template<class T >`
`std::ostream & GiNaC::operator<< (std::ostream &os, const multi_iterator_shuffle< T > &v)`
Output operator.
- `template<class T >`
`std::ostream & GiNaC::operator<< (std::ostream &os, const multi_iterator_shuffle_prime< T > &v)`
Output operator.

7.94.1 Detailed Description

Utilities for summing over multiple indices.

7.95 version.h File Reference

[GiNaC](#) library version information.

Namespaces

- namespace [GiNaC](#)

Macros

- `#define GINACLIB_MAJOR_VERSION 1`
- `#define GINACLIB_MINOR_VERSION 8`
- `#define GINACLIB_MICRO_VERSION 10`
- `#define GINAC_LT_CURRENT 14`
- `#define GINAC_LT_REVISION 2`
- `#define GINAC_LT_AGE 1`
- `#define GINACLIB_ARCHIVE_VERSION 3`
- `#define GINACLIB_ARCHIVE_AGE 3`
- `#define GINACLIB_STR_HELPER(x) #x`
- `#define GINACLIB_STR(x) GINACLIB_STR_HELPER(x)`
- `#define GINACLIB_VERSION`

7.95.1 Detailed Description

[GiNaC](#) library version information.

7.95.2 Macro Definition Documentation

7.95.2.1 GINACLIB_MAJOR_VERSION

```
#define GINACLIB_MAJOR_VERSION 1
```

7.95.2.2 GINACLIB_MINOR_VERSION

```
#define GINACLIB_MINOR_VERSION 8
```

7.95.2.3 GINACLIB_MICRO_VERSION

```
#define GINACLIB_MICRO_VERSION 10
```

7.95.2.4 GINAC_LT_CURRENT

```
#define GINAC_LT_CURRENT 14
```

7.95.2.5 GINAC_LT_REVISION

```
#define GINAC_LT_REVISION 2
```

7.95.2.6 GINAC_LT_AGE

```
#define GINAC_LT_AGE 1
```

7.95.2.7 GINACLIB_ARCHIVE_VERSION

```
#define GINACLIB_ARCHIVE_VERSION 3
```

7.95.2.8 GINACLIB_ARCHIVE_AGE

```
#define GINACLIB_ARCHIVE_AGE 3
```

7.95.2.9 GINACLIB_STR_HELPER

```
#define GINACLIB_STR_HELPER(  
    x ) #x
```

7.95.2.10 GINACLIB_STR

```
#define GINACLIB_STR(  
    x ) GINACLIB_STR_HELPER(x)
```

7.95.2.11 GINACLIB_VERSION

```
#define GINACLIB_VERSION
```

Value:

```
GINACLIB_STR(GINACLIB_MAJOR_VERSION) "." \
GINACLIB_STR(GINACLIB_MINOR_VERSION) "." \
GINACLIB_STR(GINACLIB_MICRO_VERSION)
```

7.96 wildcard.cpp File Reference

Implementation of [GiNaC](#)'s wildcard objects.

```
#include "wildcard.h"
#include "archive.h"
#include "utils.h"
#include "hash_seed.h"
```

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_IMPLEMENT_REGISTERED_CLASS_OPT](#) ([wildcard](#), [basic](#), [print_func< print_context >\(&wildcard::do_print\)](#), [print_func< print_tree >\(&wildcard::do_print_tree\)](#), [print_func< print_python_repr >\(&wildcard::do_print_python_repr\)](#)) [wildcard](#)
- [GiNaC::GINAC_BIND_UNARCHIVER](#) ([wildcard](#))
- [bool GiNaC::haswild](#) (const [ex](#) &[x](#))

Check whether x has a wildcard anywhere as a subexpression.

7.96.1 Detailed Description

Implementation of [GiNaC](#)'s wildcard objects.

7.97 wildcard.h File Reference

Interface to [GiNaC](#)'s wildcard objects.

```
#include "ex.h"
#include "archive.h"
```

Classes

- class [GiNaC::wildcard](#)

This class acts as a wildcard for [subs\(\)](#), [match\(\)](#), [has\(\)](#) and [find\(\)](#).

Namespaces

- namespace [GiNaC](#)

Functions

- [GiNaC::GINAC_DECLARE_UNARCHIVER](#) ([wildcard](#))
- [ex GiNaC::wild](#) (unsigned label=0)
Create a wildcard object with the specified label.
- bool [GiNaC::haswild](#) (const [ex](#) &[x](#))
Check whether x has a wildcard anywhere as a subexpression.

7.97.1 Detailed Description

Interface to [GiNaC](#)'s wildcard objects.

Index

[_ex0](#)
 [GiNaC, 265](#)

[_ex1](#)
 [GiNaC, 266](#)

[_ex10](#)
 [GiNaC, 269](#)

[_ex11](#)
 [GiNaC, 270](#)

[_ex12](#)
 [GiNaC, 270](#)

[_ex120](#)
 [GiNaC, 272](#)

[_ex15](#)
 [GiNaC, 270](#)

[_ex18](#)
 [GiNaC, 271](#)

[_ex1_2](#)
 [GiNaC, 266](#)

[_ex1_3](#)
 [GiNaC, 266](#)

[_ex1_4](#)
 [GiNaC, 266](#)

[_ex2](#)
 [GiNaC, 267](#)

[_ex20](#)
 [GiNaC, 271](#)

[_ex24](#)
 [GiNaC, 271](#)

[_ex25](#)
 [GiNaC, 271](#)

[_ex3](#)
 [GiNaC, 267](#)

[_ex30](#)
 [GiNaC, 272](#)

[_ex4](#)
 [GiNaC, 268](#)

[_ex48](#)
 [GiNaC, 272](#)

[_ex5](#)
 [GiNaC, 268](#)

[_ex6](#)
 [GiNaC, 268](#)

[_ex60](#)
 [GiNaC, 272](#)

[_ex7](#)
 [GiNaC, 269](#)

[_ex8](#)
 [GiNaC, 269](#)

[_ex9](#)
 [GiNaC, 269](#)

[_ex_1](#)
 [GiNaC, 264](#)

[_ex_10](#)
 [GiNaC, 261](#)

[_ex_11](#)
 [GiNaC, 261](#)

[_ex_12](#)
 [GiNaC, 261](#)

[_ex_120](#)
 [GiNaC, 259](#)

[_ex_15](#)
 [GiNaC, 261](#)

[_ex_18](#)
 [GiNaC, 260](#)

[_ex_1_2](#)
 [GiNaC, 264](#)

[_ex_1_3](#)
 [GiNaC, 264](#)

[_ex_1_4](#)
 [GiNaC, 265](#)

[_ex_2](#)
 [GiNaC, 263](#)

[_ex_20](#)
 [GiNaC, 260](#)

[_ex_24](#)
 [GiNaC, 260](#)

[_ex_25](#)
 [GiNaC, 260](#)

[_ex_3](#)
 [GiNaC, 263](#)

[_ex_30](#)
 [GiNaC, 259](#)

[_ex_4](#)
 [GiNaC, 263](#)

[_ex_48](#)
 [GiNaC, 259](#)

[_ex_5](#)
 [GiNaC, 263](#)

[_ex_6](#)
 [GiNaC, 262](#)

[_ex_60](#)
 [GiNaC, 259](#)

[_ex_7](#)
 [GiNaC, 262](#)

[_ex_8](#)
 [GiNaC, 262](#)

[_ex_9](#)
 [GiNaC, 262](#)

_iter_rep
 GiNaC::internal::_iter_rep, 277
 _num0_bp
 GiNaC, 257
 _num0_p
 GiNaC, 265
 _num10_p
 GiNaC, 269
 _num11_p
 GiNaC, 270
 _num120_p
 GiNaC, 272
 _num12_p
 GiNaC, 270
 _num15_p
 GiNaC, 270
 _num18_p
 GiNaC, 270
 _num1_2_p
 GiNaC, 266
 _num1_3_p
 GiNaC, 266
 _num1_4_p
 GiNaC, 265
 _num1_p
 GiNaC, 266
 _num20_p
 GiNaC, 271
 _num24_p
 GiNaC, 271
 _num25_p
 GiNaC, 271
 _num2_p
 GiNaC, 267
 _num30_p
 GiNaC, 271
 _num3_p
 GiNaC, 267
 _num48_p
 GiNaC, 272
 _num4_p
 GiNaC, 268
 _num5_p
 GiNaC, 268
 _num60_p
 GiNaC, 272
 _num6_p
 GiNaC, 268
 _num7_p
 GiNaC, 268
 _num8_p
 GiNaC, 269
 _num9_p
 GiNaC, 269
 _num_10_p
 GiNaC, 261
 _num_11_p
 GiNaC, 261
 _num_120_p
 GiNaC, 258
 _num_12_p
 GiNaC, 261
 _num_15_p
 GiNaC, 260
 _num_18_p
 GiNaC, 260
 _num_1_2_p
 GiNaC, 264
 _num_1_3_p
 GiNaC, 264
 _num_1_4_p
 GiNaC, 265
 _num_1_p
 GiNaC, 264
 _num_20_p
 GiNaC, 260
 _num_24_p
 GiNaC, 260
 _num_25_p
 GiNaC, 259
 _num_2_p
 GiNaC, 263
 _num_30_p
 GiNaC, 259
 _num_3_p
 GiNaC, 263
 _num_48_p
 GiNaC, 259
 _num_4_p
 GiNaC, 263
 _num_5_p
 GiNaC, 262
 _num_60_p
 GiNaC, 259
 _num_6_p
 GiNaC, 262
 _num_7_p
 GiNaC, 262
 _num_8_p
 GiNaC, 262
 _num_9_p
 GiNaC, 261
 _numeric_digits
 GiNaC::_numeric_digits, 279
 ~archive
 GiNaC::archive, 297
 ~basic
 GiNaC::basic, 317
 ~basic_multi_iterator
 GiNaC::basic_multi_iterator< T >, 344
 ~compare_all_equal
 GiNaC::compare_all_equal< T >, 380
 ~compare_bitwise
 GiNaC::compare_bitwise< T >, 381
 ~compare_std_less
 GiNaC::compare_std_less< T >, 382

- ~container_storage
 - GiNaC::container_storage< C >, 421
- ~coolmulti
 - GiNaC::composition_generator::coolmulti, 424
- ~element
 - GiNaC::composition_generator::coolmulti::element, 475
- ~function_options
 - GiNaC::function_options, 586
- ~library_init
 - GiNaC::library_init, 712
- ~map_function
 - GiNaC::map_function, 716
- ~print_context
 - GiNaC::print_context, 888
- ~print_functor_impl
 - GiNaC::print_functor_impl, 899
- ~ptr
 - GiNaC::ptr< T >, 929
- ~unarchive_table_t
 - GiNaC::unarchive_table_t, 1077
- ~visitor
 - GiNaC::visitor, 1093
- a
 - GiNaC::archive_node, 310
 - GiNaC::Eisenstein_kernel, 474
 - GiNaC::integral, 682
- abs
 - GiNaC, 215
- abs_conjugate
 - GiNaC, 129
- abs_eval
 - GiNaC, 128
- abs_evalf
 - GiNaC, 128
- abs_expand
 - GiNaC, 129
- abs_expl_derivative
 - GiNaC, 129
- abs_imag_part
 - GiNaC, 130
- abs_info
 - GiNaC, 130
- abs_power
 - GiNaC, 130
- abs_print_csrc_float
 - GiNaC, 129
- abs_print_latex
 - GiNaC, 129
- abs_real_part
 - GiNaC, 129
- accept
 - GiNaC::basic, 323
 - GiNaC::ex, 501
- access_counter
 - GiNaC::remember_table_entry, 959
- acos
 - GiNaC, 207
- acos_conjugate
 - GiNaC, 165
- acos_deriv
 - GiNaC, 165
- acos_eval
 - GiNaC, 165
- acos_evalf
 - GiNaC, 165
- acosh
 - GiNaC, 209
- acosh_conjugate
 - GiNaC, 173
- acosh_deriv
 - GiNaC, 173
- acosh_eval
 - GiNaC, 172
- acosh_evalf
 - GiNaC, 172
- adaptivesimpson
 - GiNaC, 174
- add
 - GiNaC::add, 286, 287
 - GiNaC::Gt_detail::pathintegral_term_list, 851
 - GiNaC::matrix, 727
 - GiNaC::mul, 770
 - GiNaC::numeric, 829
 - GiNaC::scalar_products, 964
 - GiNaC::symmetry, 1045
- add.cpp, 1101
- add.h, 1102
- add_bool
 - GiNaC::archive_node, 305
- add_callback
 - GiNaC::numeric_digits, 280
- add_child
 - GiNaC::class_info< OPT >::tree_node, 1076
- add_dyn
 - GiNaC::numeric, 831
- add_entry
 - GiNaC::remember_table, 955
 - GiNaC::remember_table_list, 960
- add_ex
 - GiNaC::archive_node, 306
- add_indexed
 - GiNaC::basic, 328
 - GiNaC::matrix, 724
 - GiNaC::structure< T, ComparisonPolicy >, 998
- add_node
 - GiNaC::archive, 300
- add_reference
 - GiNaC::refcounted, 939
- add_series
 - GiNaC::pseries, 922
- add_string
 - GiNaC::archive_node, 306
- add_symbol
 - GiNaC, 188
- add_unsigned

- GiNaC::archive_node, 306
- add_vectors
 - GiNaC::scalar_products, 964
- after_i
 - GiNaC::composition_generator::coolmulti, 424
- algebraic
 - GiNaC::has_options, 641
 - GiNaC::subs_options, 1021
- algebraic_match_mul_with_mul
 - GiNaC, 186
- algebraic_subs_mul
 - GiNaC::mul, 768
- all
 - GiNaC::factor_options, 546
- all_index_values_are
 - GiNaC::indexed, 667
- antisymmetric
 - GiNaC::symmetry, 1043
- antisymmetric2
 - GiNaC, 240
- antisymmetric3
 - GiNaC, 240
- antisymmetric4
 - GiNaC, 240
- antisymmetrize
 - GiNaC, 108, 241, 244
 - GiNaC::ex, 513, 514
- append
 - GiNaC::container< C >, 417
- append_factors
 - GiNaC::ncmul, 814
- apply_function_recursive
 - GiNaC::Gt, 636
- applyIntegrationPath
 - GiNaC::Gt, 632
- archive
 - GiNaC::archive, 297, 298
 - GiNaC::basic, 332
 - GiNaC::clifford, 360
 - GiNaC::color, 377
 - GiNaC::constant, 401
 - GiNaC::container< C >, 414
 - GiNaC::expairseq, 533
 - GiNaC::fderivative, 559
 - GiNaC::function, 576
 - GiNaC::Gt, 631
 - GiNaC::idx, 648
 - GiNaC::indexed, 665
 - GiNaC::integral, 679
 - GiNaC::matrix, 725
 - GiNaC::minkmetric, 742
 - GiNaC::numeric, 828
 - GiNaC::power, 882
 - GiNaC::pseries, 920
 - GiNaC::relational, 949
 - GiNaC::spinidx, 974
 - GiNaC::symbol, 1033
 - GiNaC::symmetry, 1044
 - GiNaC::tensepsilon, 1060
 - GiNaC::varidx, 1090
 - GiNaC::wildcard, 1097
- archive.cpp, 1102
- archive.h, 1103
 - GINAC_BIND_UNARCHIVER, 1105
 - GINAC_DECLARE_UNARCHIVER, 1104
- archive_atom
 - GiNaC, 58
- archive_ex
 - GiNaC::archive, 298
- archive_node
 - GiNaC::archive_node, 305
 - GiNaC::ex, 517
- archive_node_cit
 - GiNaC::archive_node, 304
- archive_node_id
 - GiNaC, 58
- archived_ex
 - GiNaC::archive::archived_ex, 312
- are_ex_trivially_equal
 - GiNaC, 101
 - GiNaC::ex, 517
- arg1
 - GiNaC::pointer_to_map_function_1arg< T1 >, 854
 - GiNaC::pointer_to_map_function_2args< T1, T2 >, 856
 - GiNaC::pointer_to_map_function_3args< T1, T2, T3 >, 857
 - GiNaC::pointer_to_member_to_map_function_1arg< C, T1 >, 861
 - GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 >, 863
 - GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >, 865
- arg2
 - GiNaC::pointer_to_map_function_2args< T1, T2 >, 856
 - GiNaC::pointer_to_map_function_3args< T1, T2, T3 >, 858
 - GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 >, 863
 - GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >, 865
- arg3
 - GiNaC::pointer_to_map_function_3args< T1, T2, T3 >, 858
 - GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >, 865
- args
 - GiNaC::Gt, 638
- argument_type
 - GiNaC::map_function, 716
- asin
 - GiNaC, 207
- asin_conjugate
 - GiNaC, 164

- asin_deriv
 - GiNaC, [164](#)
- asin_eval
 - GiNaC, [164](#)
- asin_evalf
 - GiNaC, [164](#)
- asin_info
 - GiNaC, [165](#)
- asinh
 - GiNaC, [209](#)
- asinh_conjugate
 - GiNaC, [172](#)
- asinh_deriv
 - GiNaC, [172](#)
- asinh_eval
 - GiNaC, [172](#)
- asinh_evalf
 - GiNaC, [171](#)
- assertion.h, [1105](#)
 - GINAC_ASSERT, [1106](#)
- atan
 - GiNaC, [207](#), [208](#)
- atan2_deriv
 - GiNaC, [167](#)
- atan2_eval
 - GiNaC, [167](#)
- atan2_evalf
 - GiNaC, [167](#)
- atan2_info
 - GiNaC, [167](#)
- atan_conjugate
 - GiNaC, [166](#)
- atan_deriv
 - GiNaC, [166](#)
- atan_eval
 - GiNaC, [166](#)
- atan_evalf
 - GiNaC, [166](#)
- atan_info
 - GiNaC, [167](#)
- atan_series
 - GiNaC, [166](#)
- atanh
 - GiNaC, [210](#)
- atanh_conjugate
 - GiNaC, [174](#)
- atanh_deriv
 - GiNaC, [173](#)
- atanh_eval
 - GiNaC, [173](#)
- atanh_evalf
 - GiNaC, [173](#)
- atanh_series
 - GiNaC, [174](#)
- atend
 - GiNaC::composition_generator, [384](#)
- atomize
 - GiNaC::archive, [301](#)
- atoms
 - GiNaC::archive, [302](#)
- attribute_pure
 - compiler.h, [1116](#)
- auto_clear_polylog_cache
 - GiNaC::Gt, [637](#)
- automatic
 - GiNaC::determinant_algo, [427](#)
 - GiNaC::solve_algo, [967](#)
- B
 - GiNaC::basic_multi_iterator< T >, [347](#)
- b
 - GiNaC::Eisenstein_kernel, [474](#)
 - GiNaC::integral, [682](#)
- bareiss
 - GiNaC::determinant_algo, [427](#)
 - GiNaC::solve_algo, [967](#)
- base_and_index
 - GiNaC, [86](#)
- basic
 - GiNaC::basic, [317](#)
- basic.cpp, [1106](#)
- basic.h, [1107](#)
- basic_multi_iterator
 - GiNaC::basic_multi_iterator< T >, [344](#)
- basic_partition_generator
 - GiNaC::basic_partition_generator, [348](#)
- basis
 - GiNaC::power, [887](#)
- begin
 - GiNaC::archive_node::archive_node_cit_range, [311](#)
 - GiNaC::container< C >, [418](#)
 - GiNaC::ex, [493](#)
 - GiNaC::Gt_detail::pathintegral_term_list, [850](#)
- bernoulli
 - GiNaC, [214](#)
- Bernoulli_polynomial
 - GiNaC, [177](#)
- beta_deriv
 - GiNaC, [149](#)
- beta_eval
 - GiNaC, [148](#)
- beta_evalf
 - GiNaC, [148](#)
- beta_series
 - GiNaC, [149](#)
- binomial
 - GiNaC, [214](#)
- binomial_conjugate
 - GiNaC, [138](#)
- binomial_eval
 - GiNaC, [137](#)
- binomial_evalf
 - GiNaC, [137](#)
- binomial_imag_part
 - GiNaC, [138](#)
- binomial_real_part

- GiNaC, [138](#)
- binomial_sym
 - GiNaC, [137](#)
- bp
 - GiNaC::ex, [518](#)
- c
 - factor.cpp, [1129](#)
 - GiNaC::pointer_to_member_to_map_function< C
>, [859](#)
 - GiNaC::pointer_to_member_to_map_function_1arg<check
C, T1 >, [861](#)
 - GiNaC::pointer_to_member_to_map_function_2args<check_type
C, T1, T2 >, [863](#)
 - GiNaC::pointer_to_member_to_map_function_3args<
C, T1, T2, T3 >, [865](#)
- C_norm
 - GiNaC::Eisenstein_h_kernel, [466](#)
 - GiNaC::Eisenstein_kernel, [474](#)
 - GiNaC::Kronecker_dtau_kernel, [702](#)
 - GiNaC::Kronecker_dz_kernel, [709](#)
 - GiNaC::modular_form_kernel, [751](#)
- cache
 - factor.cpp, [1131](#)
 - GiNaC::Gt_detail::TransformExpressionWithCache<
Type >, [1075](#)
- cache_step_size
 - GiNaC::integration_kernel, [690](#)
- cache_vec
 - integration_kernel.cpp, [1176](#)
- calc_lanczos_A
 - GiNaC::lanczos_coefs, [710](#)
- calchash
 - GiNaC::basic, [331](#)
 - GiNaC::constant, [402](#)
 - GiNaC::expairseq, [535](#)
 - GiNaC::function, [574](#)
 - GiNaC::Gt, [630](#)
 - GiNaC::idx, [649](#)
 - GiNaC::numeric, [829](#)
 - GiNaC::relational, [951](#)
 - GiNaC::structure< T, ComparisonPolicy >, [1001](#)
 - GiNaC::symbol, [1034](#)
 - GiNaC::symmetry, [1044](#)
 - GiNaC::wildcard, [1098](#)
- callback_registered
 - GiNaC, [82](#)
- callbacklist
 - GiNaC::_numeric_digits, [280](#)
- can_be_further_expanded
 - GiNaC::mul, [769](#)
- can_make_flat
 - GiNaC::expairseq, [538](#)
 - GiNaC::mul, [767](#)
- canonical
 - GiNaC::relational, [950](#)
- canonicalize
 - GiNaC, [240](#)
 - GiNaC::expairseq, [540](#)
 - GiNaC::symmetry, [1046](#)
- canonicalize_clifford
 - GiNaC, [90](#)
- Catalan
 - GiNaC, [256](#)
- CatalanEvalf
 - GiNaC, [219](#)
- charpoly
 - GiNaC, [185](#)
 - GiNaC::matrix, [731](#)
- GiNaC::Gt::kernel, [694](#)
- GiNaC::Gt_detail::TransformExpressionWithCache<
Type >, [1074](#), [1075](#)
- children
 - GiNaC::class_info< OPT >::tree_node, [1076](#)
 - GiNaC::symmetry, [1047](#)
- cinteger
 - GiNaC::info_flags, [672](#)
- cinteger_polynomial
 - GiNaC::info_flags, [672](#)
- class_info
 - GiNaC::class_info< OPT >, [349](#)
- class_info.h, [1108](#)
- clear
 - GiNaC::archive, [300](#)
 - GiNaC::Gt_detail::pathintegral_term_list, [850](#)
 - GiNaC::scalar_products, [964](#)
- clear_all_entries
 - GiNaC::remember_table, [955](#)
- clear_cache
 - GiNaC::Gt_detail::TransformExpressionWithCache<
Type >, [1074](#)
- clear_polylog_cache
 - GiNaC::Gt, [636](#)
- clearflag
 - GiNaC::basic, [335](#)
- clifford
 - GiNaC::clifford, [359](#)
- clifford.cpp, [1109](#)
- clifford.h, [1111](#)
- clifford_bar
 - GiNaC, [95](#)
- clifford_inverse
 - GiNaC, [92](#)
- clifford_max_label
 - GiNaC, [91](#)
- clifford_moebius_map
 - GiNaC, [93](#), [94](#)
- clifford_norm
 - GiNaC, [92](#)
- clifford_prime
 - GiNaC, [91](#)
- clifford_star
 - GiNaC, [96](#)
- clifford_star_bar
 - GiNaC, [90](#)

- clifford_to_lst
 - GiNaC, [93](#)
- clifford_unit
 - GiNaC, [87](#)
- cmgen
 - GiNaC::composition_generator, [384](#)
- CMPINDICES
 - color.cpp, [1114](#)
- coeff
 - GiNaC, [104](#)
 - GiNaC::add, [288](#)
 - GiNaC::basic, [324](#)
 - GiNaC::ex, [503](#)
 - GiNaC::expair, [523](#)
 - GiNaC::mul, [760](#)
 - GiNaC::ncmul, [810](#)
 - GiNaC::numeric, [824](#)
 - GiNaC::power, [878](#)
 - GiNaC::pseries, [917](#)
 - GiNaC::structure< T, ComparisonPolicy >, [994](#)
 - GiNaC::symminfo, [1048](#)
- coefficient_a0
 - GiNaC::Eisenstein_h_kernel, [464](#)
- coefficient_an
 - GiNaC::Eisenstein_h_kernel, [465](#)
- coeffop
 - GiNaC::pseries, [922](#)
- coeffs
 - GiNaC::lanczos_coeffs, [710](#)
- coerce
 - GiNaC, [204](#)
- coerce< int, cln::cl_I >
 - GiNaC, [204](#)
- coerce< unsigned int, cln::cl_I >
 - GiNaC, [204](#)
- col
 - GiNaC::matrix, [737](#)
- collect
 - GiNaC, [106](#)
 - GiNaC::basic, [325](#)
 - GiNaC::ex, [504](#)
 - GiNaC::pseries, [917](#)
 - GiNaC::structure< T, ComparisonPolicy >, [994](#)
- collect_common_factors
 - GiNaC, [201](#)
- collect_symbols
 - GiNaC, [188](#)
- color
 - GiNaC::color, [376](#), [377](#)
- color.cpp, [1113](#)
 - CMPINDICES, [1114](#)
 - TEST_PERMUTATION, [1114](#)
- color.h, [1115](#)
- color_d
 - GiNaC, [98](#)
- color_f
 - GiNaC, [98](#)
- color_h
 - GiNaC, [99](#)
- color_ONE
 - GiNaC, [97](#)
- color_T
 - GiNaC, [97](#)
- color_trace
 - GiNaC, [99](#), [100](#)
- cols
 - GiNaC, [184](#)
 - GiNaC::matrix, [726](#)
- combine_ex_with_coeff_to_pair
 - GiNaC::add, [293](#)
 - GiNaC::expairseq, [536](#)
 - GiNaC::mul, [766](#)
- combine_indices
 - GiNaC::make_flat_inserter, [714](#)
- combine_overall_coeff
 - GiNaC::expairseq, [538](#)
 - GiNaC::mul, [767](#)
- combine_pair_with_coeff_to_pair
 - GiNaC::add, [293](#)
 - GiNaC::expairseq, [537](#)
 - GiNaC::mul, [766](#)
- combine_same_terms_sorted_seq
 - GiNaC::expairseq, [541](#)
- commutative
 - GiNaC::return_types, [963](#)
- commutator_sign
 - GiNaC::clifford, [364](#)
- compare
 - GiNaC::basic, [334](#)
 - GiNaC::ex, [512](#)
 - GiNaC::expair, [522](#)
 - GiNaC::numeric, [833](#)
- compare_pointers
 - GiNaC, [250](#)
- compare_same_type
 - GiNaC::basic, [331](#)
- compile_ex
 - GiNaC, [111](#), [112](#)
- compiler.h, [1116](#)
 - attribute_pure, [1116](#)
 - likely, [1116](#)
 - unlikely, [1116](#)
- complex
 - GiNaC::domain, [450](#)
- composition
 - GiNaC::composition_generator, [384](#)
- composition_generator
 - GiNaC::composition_generator, [383](#)
- conjugate
 - GiNaC, [103](#)
 - GiNaC::add, [291](#)
 - GiNaC::basic, [330](#)
 - GiNaC::constant, [401](#)
 - GiNaC::container< C >, [414](#)
 - GiNaC::diracgamma5, [436](#)
 - GiNaC::diracgammaL, [440](#)

- GiNaC::diracgammaR, [444](#)
 - GiNaC::ex, [498](#)
 - GiNaC::expair, [523](#)
 - GiNaC::expairseq, [533](#)
 - GiNaC::function, [575](#)
 - GiNaC::integral, [679](#)
 - GiNaC::matrix, [725](#)
 - GiNaC::mul, [764](#)
 - GiNaC::ncmul, [812](#)
 - GiNaC::numeric, [827](#)
 - GiNaC::power, [882](#)
 - GiNaC::pseries, [919](#)
 - GiNaC::realsymbol, [936](#)
 - GiNaC::spinidx, [973](#)
 - GiNaC::symbol, [1032](#)
- conjugate_conjugate
 - GiNaC, [125](#)
- conjugate_eval
 - GiNaC, [124](#)
- conjugate_evalf
 - GiNaC, [124](#)
- conjugate_expl_derivative
 - GiNaC, [125](#)
- conjugate_f
 - GiNaC::function_options, [616](#)
- conjugate_func
 - GiNaC::function_options, [591–593](#), [609](#)
- conjugate_funcp
 - GiNaC, [60](#)
- conjugate_funcp_1
 - GiNaC, [61](#)
- conjugate_funcp_10
 - GiNaC, [73](#)
- conjugate_funcp_11
 - GiNaC, [74](#)
- conjugate_funcp_12
 - GiNaC, [75](#)
- conjugate_funcp_13
 - GiNaC, [77](#)
- conjugate_funcp_14
 - GiNaC, [78](#)
- conjugate_funcp_2
 - GiNaC, [62](#)
- conjugate_funcp_3
 - GiNaC, [63](#)
- conjugate_funcp_4
 - GiNaC, [65](#)
- conjugate_funcp_5
 - GiNaC, [66](#)
- conjugate_funcp_6
 - GiNaC, [67](#)
- conjugate_funcp_7
 - GiNaC, [69](#)
- conjugate_funcp_8
 - GiNaC, [70](#)
- conjugate_funcp_9
 - GiNaC, [71](#)
- conjugate_funcp_exvector
 - GiNaC, [80](#)
- conjugate_imag_part
 - GiNaC, [125](#)
- conjugate_info
 - GiNaC, [126](#)
- conjugate_print_latex
 - GiNaC, [125](#)
- conjugate_real_part
 - GiNaC, [125](#)
- conjugate_use_exvector_args
 - GiNaC::function_options, [619](#)
- conjugateepvector
 - GiNaC, [114](#)
- const_iterator
 - GiNaC::const_iterator, [386](#)
 - GiNaC::container< C >, [410](#)
- const_postorder_iterator
 - GiNaC::const_iterator, [389](#)
 - GiNaC::const_postorder_iterator, [391](#)
- const_preorder_iterator
 - GiNaC::const_iterator, [389](#)
 - GiNaC::const_preorder_iterator, [394](#)
- const_reverse_iterator
 - GiNaC::container< C >, [411](#)
- constant
 - GiNaC::constant, [400](#)
- constant.cpp, [1117](#)
- constant.h, [1117](#)
- construct_from_2_ex
 - GiNaC::expairseq, [539](#)
- construct_from_2_expairseq
 - GiNaC::expairseq, [539](#)
- construct_from_basic
 - GiNaC::ex, [515](#)
- construct_from_double
 - GiNaC::ex, [516](#)
- construct_from_epvector
 - GiNaC::expairseq, [539](#), [540](#)
- construct_from_expairseq_ex
 - GiNaC::expairseq, [539](#)
- construct_from_exvector
 - GiNaC::expairseq, [539](#)
- construct_from_int
 - GiNaC::ex, [515](#)
- construct_from_long
 - GiNaC::ex, [516](#)
- construct_from_longlong
 - GiNaC::ex, [516](#)
- construct_from_string_and_lst
 - GiNaC::ex, [516](#)
- construct_from_uint
 - GiNaC::ex, [515](#)
- construct_from_ulong
 - GiNaC::ex, [516](#)
- construct_from_ulonglong
 - GiNaC::ex, [516](#)
- cont
 - factor.cpp, [1134](#)

- container
 - GiNaC::container< C >, 411
- container.h, 1118
- container_storage
 - GiNaC::container_storage< C >, 421
- content
 - GiNaC::ex, 508
- contract_with
 - GiNaC::basic, 329
 - GiNaC::cliffordunit, 369
 - GiNaC::diracgamma, 431
 - GiNaC::matrix, 724
 - GiNaC::spinmetric, 981
 - GiNaC::structure< T, ComparisonPolicy >, 999
 - GiNaC::su3d, 1006
 - GiNaC::su3f, 1011
 - GiNaC::su3t, 1020
 - GiNaC::tensdelta, 1054
 - GiNaC::tensepsilon, 1060
 - GiNaC::tensmetric, 1066
- convert_H_to_Li
 - GiNaC, 143
- convert_to_poly
 - GiNaC::pseries, 921
- coolmulti
 - GiNaC::composition_generator::coolmulti, 424
- cos
 - GiNaC, 206
- cos_conjugate
 - GiNaC, 162
- cos_deriv
 - GiNaC, 162
- cos_eval
 - GiNaC, 162
- cos_evalf
 - GiNaC, 161
- cos_imag_part
 - GiNaC, 162
- cos_real_part
 - GiNaC, 162
- cosh
 - GiNaC, 209
- cosh_conjugate
 - GiNaC, 170
- cosh_deriv
 - GiNaC, 169
- cosh_eval
 - GiNaC, 169
- cosh_evalf
 - GiNaC, 169
- cosh_imag_part
 - GiNaC, 170
- cosh_real_part
 - GiNaC, 169
- count
 - GiNaC::archive_node::property_info, 911
 - GiNaC::library_init, 713
- count_dummy_indices
 - GiNaC, 119
- count_factors
 - GiNaC::ncmul, 813
- count_free_indices
 - GiNaC, 120
- covariant
 - GiNaC::varidx, 1092
- crational
 - GiNaC::info_flags, 672
- crational_polynomial
 - GiNaC::info_flags, 672
- crc32
 - GiNaC, 101
- crc32.h, 1119
- crctab
 - GiNaC, 256
- csgn
 - GiNaC, 220
 - GiNaC::numeric, 833
- csgn_conjugate
 - GiNaC, 132
- csgn_eval
 - GiNaC, 132
- csgn_evalf
 - GiNaC, 131
- csgn_imag_part
 - GiNaC, 132
- csgn_power
 - GiNaC, 132
- csgn_real_part
 - GiNaC, 132
- csgn_series
 - GiNaC, 132
- csrc
 - GiNaC, 232
- csrc_cl_N
 - GiNaC, 233
- csrc_double
 - GiNaC, 233
- csrc_float
 - GiNaC, 233
- current_serial
 - GiNaC::function, 580
- current_updated
 - GiNaC::composition_generator, 384
 - GiNaC::partition_generator, 844
 - GiNaC::partition_with_zero_parts_generator, 846
- current_vector
 - GiNaC::lanczos_coeffs, 710
- cutIntegrationPath_threshold_imag
 - GiNaC::Gt, 637
- cutIntegrationPath_threshold_real
 - GiNaC::Gt, 637
- cyclic
 - GiNaC::symmetry, 1043
- cyclic_permutation
 - GiNaC, 251
- dbgprint

- GiNaC::basic, [319](#)
 - GiNaC::ex, [495](#)
- dbgprinttree
 - GiNaC::basic, [319](#)
 - GiNaC::ex, [496](#)
- DCOUT
 - factor.cpp, [1129](#)
- DCOUT2
 - factor.cpp, [1129](#)
- DCOUTVAR
 - factor.cpp, [1129](#)
- DEBUG_GT
 - Gt.cpp, [1152](#)
- debugprint
 - GiNaC::scalar_products, [965](#)
 - GiNaC::smapkey, [983](#)
- DECLARE_FUNCTION_10P
 - function.h, [1149](#)
- DECLARE_FUNCTION_11P
 - function.h, [1149](#)
- DECLARE_FUNCTION_12P
 - function.h, [1149](#)
- DECLARE_FUNCTION_13P
 - function.h, [1150](#)
- DECLARE_FUNCTION_14P
 - function.h, [1150](#)
- DECLARE_FUNCTION_1P
 - function.h, [1147](#)
- DECLARE_FUNCTION_2P
 - function.h, [1147](#)
- DECLARE_FUNCTION_3P
 - function.h, [1147](#)
- DECLARE_FUNCTION_4P
 - function.h, [1147](#)
- DECLARE_FUNCTION_5P
 - function.h, [1148](#)
- DECLARE_FUNCTION_6P
 - function.h, [1148](#)
- DECLARE_FUNCTION_7P
 - function.h, [1148](#)
- DECLARE_FUNCTION_8P
 - function.h, [1148](#)
- DECLARE_FUNCTION_9P
 - function.h, [1149](#)
- decomp_rational
 - GiNaC, [191](#)
- decomposeIntegrationPath
 - GiNaC::Gt, [632](#)
- deconcatenate_path
 - GiNaC::Gt_detail, [273](#)
- deconcatenate_path< ex >
 - GiNaC::Gt_detail, [273](#)
- deconcatenate_path< Gt::kernel >
 - GiNaC::Gt_detail, [273](#)
- DEFAULT_COMPARE
 - utils.h, [1221](#)
- DEFAULT_CTOR
 - utils.h, [1221](#)
- default_overall_coeff
 - GiNaC::expairseq, [537](#)
 - GiNaC::mul, [767](#)
- DEFAULT_PRINT
 - utils.h, [1221](#)
- DEFAULT_PRINT_LATEX
 - utils.h, [1221](#)
- deform
 - GiNaC::Gt::kernel, [694](#)
- deg
 - GiNaC::pole_error, [866](#)
- deg_a
 - GiNaC::sym_desc, [1025](#)
- deg_b
 - GiNaC::sym_desc, [1025](#)
- degree
 - GiNaC, [104](#)
 - GiNaC::add, [288](#)
 - GiNaC::basic, [324](#)
 - GiNaC::ex, [502](#)
 - GiNaC::integral, [677](#)
 - GiNaC::mul, [760](#)
 - GiNaC::ncmul, [810](#)
 - GiNaC::numeric, [824](#)
 - GiNaC::pole_error, [866](#)
 - GiNaC::power, [878](#)
 - GiNaC::pseries, [916](#)
 - GiNaC::structure< T, ComparisonPolicy >, [994](#)
- delete_cyclic
 - GiNaC::remember_strategies, [954](#)
- delete_lfu
 - GiNaC::remember_strategies, [954](#)
- delete_lru
 - GiNaC::remember_strategies, [954](#)
- delete_never
 - GiNaC::remember_strategies, [954](#)
- delta_indent
 - GiNaC::print_tree, [908](#)
- delta_tensor
 - GiNaC, [246](#)
- denom
 - GiNaC, [105](#), [224](#)
 - GiNaC::ex, [506](#)
 - GiNaC::Gt_detail::pathintegral_term, [849](#)
 - GiNaC::numeric, [840](#)
- derivative
 - GiNaC::add, [292](#)
 - GiNaC::basic, [325](#)
 - GiNaC::constant, [402](#)
 - GiNaC::fderivative, [560](#)
 - GiNaC::function, [577](#)
 - GiNaC::idx, [648](#)
 - GiNaC::indexed, [666](#)
 - GiNaC::integral, [680](#)
 - GiNaC::mul, [764](#)
 - GiNaC::ncmul, [812](#)
 - GiNaC::numeric, [828](#)
 - GiNaC::power, [883](#)

- GiNaC::pseries, [920](#)
- GiNaC::structure< T, ComparisonPolicy >, [996](#)
- GiNaC::symbol, [1034](#)
- derivative_f
 - GiNaC::function_options, [617](#)
- derivative_func
 - GiNaC::function_options, [599–601](#), [610](#)
- derivative_funcp
 - GiNaC, [60](#)
- derivative_funcp_1
 - GiNaC, [61](#)
- derivative_funcp_10
 - GiNaC, [73](#)
- derivative_funcp_11
 - GiNaC, [74](#)
- derivative_funcp_12
 - GiNaC, [76](#)
- derivative_funcp_13
 - GiNaC, [77](#)
- derivative_funcp_14
 - GiNaC, [79](#)
- derivative_funcp_2
 - GiNaC, [63](#)
- derivative_funcp_3
 - GiNaC, [64](#)
- derivative_funcp_4
 - GiNaC, [65](#)
- derivative_funcp_5
 - GiNaC, [66](#)
- derivative_funcp_6
 - GiNaC, [68](#)
- derivative_funcp_7
 - GiNaC, [69](#)
- derivative_funcp_8
 - GiNaC, [70](#)
- derivative_funcp_9
 - GiNaC, [72](#)
- derivative_funcp_exvector
 - GiNaC, [80](#)
- derivative_map_function
 - GiNaC::derivative_map_function, [426](#)
- derivative_use_exvector_args
 - GiNaC::function_options, [620](#)
- derivatives
 - GiNaC::fderivative, [561](#)
- descend
 - GiNaC::const_postorder_iterator, [392](#)
- determinant
 - GiNaC, [184](#)
 - GiNaC::matrix, [729](#)
- determinant_minor
 - GiNaC::matrix, [733](#)
- dfft
 - GiNaC, [232](#)
- diag_matrix
 - GiNaC, [182](#)
- diff
 - GiNaC, [107](#)
- GiNaC::basic, [333](#)
- GiNaC::ex, [504](#)
- difference_type
 - GiNaC::const_iterator, [386](#)
 - GiNaC::const_postorder_iterator, [391](#)
 - GiNaC::const_preorder_iterator, [394](#)
- Digits
 - GiNaC, [258](#)
- digits
 - GiNaC::numeric_digits, [280](#)
- digits_changed_callback
 - GiNaC, [82](#)
- dim
 - GiNaC::idx, [652](#)
 - GiNaC::spmapkey, [984](#)
- dirac_gamma
 - GiNaC, [87](#)
- dirac_gamma5
 - GiNaC, [88](#)
- dirac_gammaL
 - GiNaC, [88](#)
- dirac_gammaR
 - GiNaC, [88](#)
- dirac_ONE
 - GiNaC, [86](#)
- dirac_slash
 - GiNaC, [89](#)
- dirac_trace
 - GiNaC, [89](#), [90](#)
- dirichlet_character
 - GiNaC, [176](#)
- div
 - GiNaC::numeric, [830](#)
- div_dyn
 - GiNaC::numeric, [831](#)
- divfree
 - GiNaC::determinant_algo, [427](#)
 - GiNaC::solve_algo, [967](#)
- divide
 - GiNaC, [193](#)
- divide_in_z
 - GiNaC, [193](#)
- division_free_elimination
 - GiNaC::matrix, [734](#)
- do_not_evalf_params
 - GiNaC::function_options, [613](#)
- do_print
 - GiNaC::add, [294](#)
 - GiNaC::basic, [336](#)
 - GiNaC::basic_log_kernel, [342](#)
 - GiNaC::cliffordunit, [369](#)
 - GiNaC::constant, [403](#)
 - GiNaC::container< C >, [419](#)
 - GiNaC::diracgamma, [431](#)
 - GiNaC::diracgamma5, [436](#)
 - GiNaC::diracgammaL, [440](#)
 - GiNaC::diracgammaR, [444](#)
 - GiNaC::diracone, [449](#)

- GiNaC::Ebar_kernel, 456
- GiNaC::Eisenstein_h_kernel, 465
- GiNaC::Eisenstein_kernel, 473
- GiNaC::ELi_kernel, 481
- GiNaC::expairseq, 538
- GiNaC::fail, 549
- GiNaC::fderivative, 561
- GiNaC::Gt, 636
- GiNaC::idx, 651
- GiNaC::indexed, 669
- GiNaC::integral, 681
- GiNaC::integration_kernel, 690
- GiNaC::Kronecker_dtau_kernel, 701
- GiNaC::Kronecker_dz_kernel, 708
- GiNaC::matrix, 736
- GiNaC::minkmetric, 743
- GiNaC::modular_form_kernel, 751
- GiNaC::mul, 768
- GiNaC::multiple_polylog_kernel, 802
- GiNaC::ncmul, 813
- GiNaC::numeric, 841
- GiNaC::pseries, 924
- GiNaC::relational, 951
- GiNaC::spinidx, 976
- GiNaC::spinmetric, 982
- GiNaC::su3d, 1006
- GiNaC::su3f, 1011
- GiNaC::su3one, 1016
- GiNaC::su3t, 1020
- GiNaC::symbol, 1036
- GiNaC::symmetry, 1046
- GiNaC::tensdelta, 1055
- GiNaC::tensepsilon, 1061
- GiNaC::tensmetric, 1067
- GiNaC::user_defined_kernel, 1084
- GiNaC::varidx, 1091
- GiNaC::wildcard, 1098
- do_print_csrc
 - GiNaC::add, 295
 - GiNaC::fderivative, 561
 - GiNaC::idx, 652
 - GiNaC::mul, 769
 - GiNaC::ncmul, 813
 - GiNaC::numeric, 841
 - GiNaC::power, 884
- do_print_csrc_cl_N
 - GiNaC::numeric, 841
 - GiNaC::power, 885
- do_print_dflt
 - GiNaC::clifford, 363
 - GiNaC::power, 884
- do_print_latex
 - GiNaC::add, 295
 - GiNaC::clifford, 364
 - GiNaC::cliffordunit, 369
 - GiNaC::constant, 403
 - GiNaC::diracgamma, 432
 - GiNaC::diracgamma5, 436
 - GiNaC::diracgammaL, 440
 - GiNaC::diracgammaR, 445
 - GiNaC::diracone, 449
 - GiNaC::fderivative, 561
 - GiNaC::Gt, 636
 - GiNaC::idx, 652
 - GiNaC::indexed, 669
 - GiNaC::integral, 681
 - GiNaC::matrix, 736
 - GiNaC::minkmetric, 743
 - GiNaC::mul, 769
 - GiNaC::numeric, 841
 - GiNaC::power, 884
 - GiNaC::pseries, 924
 - GiNaC::spinidx, 976
 - GiNaC::spinmetric, 982
 - GiNaC::su3d, 1006
 - GiNaC::su3f, 1011
 - GiNaC::su3one, 1016
 - GiNaC::su3t, 1020
 - GiNaC::symbol, 1036
 - GiNaC::tensdelta, 1055
 - GiNaC::tensepsilon, 1061
- do_print_python
 - GiNaC::container< C >, 419
 - GiNaC::power, 885
 - GiNaC::pseries, 925
- do_print_python_repr
 - GiNaC::add, 295
 - GiNaC::basic, 336
 - GiNaC::constant, 403
 - GiNaC::container< C >, 419
 - GiNaC::matrix, 736
 - GiNaC::mul, 769
 - GiNaC::numeric, 842
 - GiNaC::power, 885
 - GiNaC::pseries, 925
 - GiNaC::relational, 951
 - GiNaC::symbol, 1036
 - GiNaC::wildcard, 1098
- do_print_tree
 - GiNaC::basic, 336
 - GiNaC::clifford, 364
 - GiNaC::constant, 403
 - GiNaC::container< C >, 419
 - GiNaC::expairseq, 538
 - GiNaC::fderivative, 562
 - GiNaC::idx, 652
 - GiNaC::indexed, 669
 - GiNaC::numeric, 841
 - GiNaC::pseries, 925
 - GiNaC::spinidx, 976
 - GiNaC::symbol, 1036
 - GiNaC::symmetry, 1046
 - GiNaC::varidx, 1092
 - GiNaC::wildcard, 1098
- do_renaming
 - GiNaC::make_flat_inserter, 714

- domain
 - GiNaC::constant, [405](#)
- dotted
 - GiNaC::spinidx, [976](#)
- doublefactorial
 - GiNaC, [213](#)
- dummy
 - GiNaC::function_options, [587](#)
- dump_hierarchy
 - GiNaC::class_info< OPT >, [350](#)
- dump_tree
 - GiNaC::class_info< OPT >, [350](#)
- duplicate
 - GiNaC::basic, [317](#)
 - GiNaC::possymbol, [871](#)
 - GiNaC::print_functor_impl, [899](#)
 - GiNaC::print_memfun_handler< T, C >, [901](#)
 - GiNaC::print_ptrfun_handler< T, C >, [904](#)
 - GiNaC::realsymbol, [937](#)
- dynallocate
 - GiNaC, [84](#), [85](#)
- dynallocated
 - GiNaC::status_flags, [985](#)
- e
 - GiNaC::archive_node, [311](#)
 - GiNaC::const_iterator, [389](#)
 - GiNaC::internal::_iter_rep, [278](#)
- Ebar_kernel
 - GiNaC::Ebar_kernel, [455](#)
- echelon_form
 - GiNaC::matrix, [733](#)
- ef
 - GiNaC::constant, [404](#)
- Eisenstein_h_kernel
 - GiNaC::Eisenstein_h_kernel, [462](#)
- Eisenstein_kernel
 - GiNaC::Eisenstein_kernel, [471](#)
- element
 - GiNaC::composition_generator::coolmulti::element, [475](#)
- ELi_kernel
 - GiNaC::ELi_kernel, [480](#)
- EllipticE_deriv
 - GiNaC, [145](#)
- EllipticE_eval
 - GiNaC, [144](#)
- EllipticE_evalf
 - GiNaC, [144](#)
- EllipticE_print_latex
 - GiNaC, [145](#)
- EllipticE_series
 - GiNaC, [145](#)
- EllipticK_deriv
 - GiNaC, [144](#)
- EllipticK_eval
 - GiNaC, [143](#)
- EllipticK_evalf
 - GiNaC, [143](#)
- EllipticK_print_latex
 - GiNaC, [144](#)
- EllipticK_series
 - GiNaC, [144](#)
- empty
 - GiNaC::Gt_detail::pathintegral_term_list, [850](#)
- enable_tauToFundamental
 - GiNaC::Gt, [638](#)
- end
 - GiNaC::archive_node::archive_node_cit_range, [311](#)
 - GiNaC::container< C >, [418](#)
 - GiNaC::ex, [493](#)
 - GiNaC::Gt_detail::pathintegral_term_list, [850](#)
- ensure_if_modifiable
 - GiNaC::basic, [336](#)
- epp
 - GiNaC, [59](#)
- epsilon_tensor
 - GiNaC, [247](#), [248](#)
- epvector
 - GiNaC, [59](#)
- equal
 - GiNaC::relational, [947](#)
- error
 - GiNaC::error_and_integral, [483](#)
- error_and_integral
 - GiNaC::error_and_integral, [483](#)
- eta_conjugate
 - GiNaC, [133](#)
- eta_eval
 - GiNaC, [133](#)
- eta_evalf
 - GiNaC, [133](#)
- eta_imag_part
 - GiNaC, [134](#)
- eta_real_part
 - GiNaC, [134](#)
- eta_series
 - GiNaC, [133](#)
- Euler
 - GiNaC, [256](#)
- EulerEvalf
 - GiNaC, [219](#)
- eval
 - GiNaC, [106](#)
 - GiNaC::add, [288](#)
 - GiNaC::basic, [317](#)
 - GiNaC::ex, [494](#)
 - GiNaC::expairseq, [532](#)
 - GiNaC::fderivative, [558](#)
 - GiNaC::function, [574](#)
 - GiNaC::Gt, [629](#)
 - GiNaC::indexed, [665](#)
 - GiNaC::integral, [677](#)
 - GiNaC::mul, [761](#)
 - GiNaC::ncmul, [811](#)
 - GiNaC::numeric, [825](#)

- GiNaC::power, [878](#)
 - GiNaC::pseries, [918](#)
 - GiNaC::structure< T, ComparisonPolicy >, [989](#)
 - GiNaC::symbol, [1031](#)
- eval_f
 - GiNaC::function_options, [616](#)
- eval_func
 - GiNaC::function_options, [587–589](#), [609](#)
- eval_funcp
 - GiNaC, [60](#)
- eval_funcp_1
 - GiNaC, [61](#)
- eval_funcp_10
 - GiNaC, [72](#)
- eval_funcp_11
 - GiNaC, [74](#)
- eval_funcp_12
 - GiNaC, [75](#)
- eval_funcp_13
 - GiNaC, [77](#)
- eval_funcp_14
 - GiNaC, [78](#)
- eval_funcp_2
 - GiNaC, [62](#)
- eval_funcp_3
 - GiNaC, [63](#)
- eval_funcp_4
 - GiNaC, [64](#)
- eval_funcp_5
 - GiNaC, [66](#)
- eval_funcp_6
 - GiNaC, [67](#)
- eval_funcp_7
 - GiNaC, [68](#)
- eval_funcp_8
 - GiNaC, [70](#)
- eval_funcp_9
 - GiNaC, [71](#)
- eval_funcp_exvector
 - GiNaC, [80](#)
- eval_indexed
 - GiNaC::basic, [319](#)
 - GiNaC::matrix, [724](#)
 - GiNaC::minkmetric, [742](#)
 - GiNaC::spinmetric, [981](#)
 - GiNaC::structure< T, ComparisonPolicy >, [990](#)
 - GiNaC::su3d, [1006](#)
 - GiNaC::su3f, [1011](#)
 - GiNaC::tensdelta, [1054](#)
 - GiNaC::tensepsilon, [1060](#)
 - GiNaC::tensmetric, [1066](#)
- eval_integ
 - GiNaC, [106](#)
 - GiNaC::basic, [318](#)
 - GiNaC::ex, [495](#)
 - GiNaC::integral, [679](#)
 - GiNaC::pseries, [920](#)
- eval_ncmul
 - GiNaC::add, [292](#)
 - GiNaC::basic, [318](#)
 - GiNaC::clifford, [360](#)
 - GiNaC::color, [378](#)
 - GiNaC::ex, [494](#)
 - GiNaC::function, [574](#)
 - GiNaC::integral, [677](#)
 - GiNaC::mul, [764](#)
 - GiNaC::power, [883](#)
 - GiNaC::relational, [950](#)
 - GiNaC::structure< T, ComparisonPolicy >, [990](#)
- eval_use_exvector_args
 - GiNaC::function_options, [619](#)
- evalchildren
 - GiNaC::expairseq, [541](#)
- evalf
 - GiNaC, [106](#), [184](#)
 - GiNaC::basic, [318](#)
 - GiNaC::constant, [400](#)
 - GiNaC::ex, [494](#)
 - GiNaC::function, [574](#)
 - GiNaC::Gt, [629](#)
 - GiNaC::idx, [647](#)
 - GiNaC::integral, [677](#)
 - GiNaC::mul, [761](#)
 - GiNaC::numeric, [825](#)
 - GiNaC::power, [879](#)
 - GiNaC::pseries, [918](#)
 - GiNaC::symbol, [1031](#)
- evalf_f
 - GiNaC::function_options, [616](#)
- evalf_func
 - GiNaC::function_options, [589–591](#), [609](#)
- evalf_funcp
 - GiNaC, [60](#)
- evalf_funcp_1
 - GiNaC, [61](#)
- evalf_funcp_10
 - GiNaC, [72](#)
- evalf_funcp_11
 - GiNaC, [74](#)
- evalf_funcp_12
 - GiNaC, [75](#)
- evalf_funcp_13
 - GiNaC, [77](#)
- evalf_funcp_14
 - GiNaC, [78](#)
- evalf_funcp_2
 - GiNaC, [62](#)
- evalf_funcp_3
 - GiNaC, [63](#)
- evalf_funcp_4
 - GiNaC, [65](#)
- evalf_funcp_5
 - GiNaC, [66](#)
- evalf_funcp_6
 - GiNaC, [67](#)
- evalf_funcp_7

- GiNaC, 68
- evalf_funcp_8
 - GiNaC, 70
- evalf_funcp_9
 - GiNaC, 71
- evalf_funcp_exvector
 - GiNaC, 80
- evalf_params_first
 - GiNaC::function_options, 618
- evalf_use_exvector_args
 - GiNaC::function_options, 619
- evalffunctype
 - GiNaC, 59
- evalm
 - GiNaC, 106
 - GiNaC::add, 289
 - GiNaC::basic, 318
 - GiNaC::ex, 494
 - GiNaC::matrix, 723
 - GiNaC::mul, 762
 - GiNaC::ncmul, 811
 - GiNaC::power, 879
 - GiNaC::pseries, 920
 - GiNaC::structure< T, ComparisonPolicy >, 989
- evalpoint
 - factor.cpp, 1134
- evaluate
 - GiNaC::Gt, 633
 - GiNaC::Gt_detail::pathintegral_term, 848
 - GiNaC::scalar_products, 965
- evaluated
 - GiNaC::status_flags, 985
- even
 - GiNaC::info_flags, 672
- ex
 - GiNaC::basic, 337
 - GiNaC::const_iterator, 389
 - GiNaC::ex, 491, 492
- ex.cpp, 1119
- ex.h, 1120
- ex_cutIntegrationPath
 - GiNaC::Gt, 633, 635
- ex_evaluate
 - GiNaC::Gt, 634, 635
- ex_prepare
 - GiNaC::Gt, 634, 635
- ex_qExpand
 - GiNaC::Gt, 634, 635
- ex_regularise
 - GiNaC::Gt, 633, 635
- ex_tauToFundamental
 - GiNaC::Gt, 633, 635
- ex_to
 - GiNaC, 111
 - GiNaC::ex, 517
- ex_zisToFundamental
 - GiNaC::Gt, 633, 634
- exadd
 - GiNaC, 224
- excompiler.cpp, 1122
- excompiler.h, 1123
- exhashmap
 - GiNaC, 81
- exmap
 - GiNaC, 59
- exminus
 - GiNaC, 224
- exmul
 - GiNaC, 224
- exp
 - GiNaC, 205
- exp_conjugate
 - GiNaC, 158
- exp_deriv
 - GiNaC, 157
- exp_eval
 - GiNaC, 157
- exp_evalf
 - GiNaC, 157
- exp_expand
 - GiNaC, 157
- exp_imag_part
 - GiNaC, 157
- exp_info
 - GiNaC, 158
- exp_power
 - GiNaC, 158
- exp_real_part
 - GiNaC, 157
- expair
 - GiNaC::expair, 521
- expair.cpp, 1124
- expair.h, 1124
- expair_needs_further_processing
 - GiNaC::expairseq, 537
 - GiNaC::mul, 766
- expairseq
 - GiNaC::expairseq, 530
- expairseq.cpp, 1125
- expairseq.h, 1126
- expand
 - GiNaC, 103, 183
 - GiNaC::add, 294
 - GiNaC::basic, 324
 - GiNaC::ex, 503
 - GiNaC::expairseq, 535
 - GiNaC::function, 573
 - GiNaC::indexed, 667
 - GiNaC::integral, 678
 - GiNaC::mul, 767
 - GiNaC::ncmul, 810
 - GiNaC::power, 883
 - GiNaC::pseries, 919
 - GiNaC::structure< T, ComparisonPolicy >, 994
- expand_add
 - GiNaC::power, 885

- expand_add_2
 - GiNaC::power, 885
- expand_dummy_sum
 - GiNaC, 124
- expand_f
 - GiNaC::function_options, 617
- expand_func
 - GiNaC::function_options, 597–599, 610
- expand_funcp
 - GiNaC, 60
- expand_funcp_1
 - GiNaC, 61
- expand_funcp_10
 - GiNaC, 73
- expand_funcp_11
 - GiNaC, 74
- expand_funcp_12
 - GiNaC, 76
- expand_funcp_13
 - GiNaC, 77
- expand_funcp_14
 - GiNaC, 79
- expand_funcp_2
 - GiNaC, 63
- expand_funcp_3
 - GiNaC, 64
- expand_funcp_4
 - GiNaC, 65
- expand_funcp_5
 - GiNaC, 66
- expand_funcp_6
 - GiNaC, 68
- expand_funcp_7
 - GiNaC, 69
- expand_funcp_8
 - GiNaC, 70
- expand_funcp_9
 - GiNaC, 72
- expand_funcp_exvector
 - GiNaC, 80
- expand_function_args
 - GiNaC::expand_options, 545
- expand_indexed
 - GiNaC::expand_options, 545
- expand_map_function
 - GiNaC::expand_map_function, 544
- expand_mul
 - GiNaC::power, 886
- expand_rename_idx
 - GiNaC::expand_options, 545
- expand_transcendental
 - GiNaC::expand_options, 545
- expand_use_exvector_args
 - GiNaC::function_options, 620
- expandchildren
 - GiNaC::expairseq, 541
 - GiNaC::mul, 769
 - GiNaC::ncmul, 814
- expanded
 - GiNaC::info_flags, 672
 - GiNaC::status_flags, 985
- expl_derivative
 - GiNaC::function, 578
- expl_derivative_f
 - GiNaC::function_options, 617
- expl_derivative_func
 - GiNaC::function_options, 601–603, 610
- expl_derivative_funcp
 - GiNaC, 60
- expl_derivative_funcp_1
 - GiNaC, 62
- expl_derivative_funcp_10
 - GiNaC, 73
- expl_derivative_funcp_11
 - GiNaC, 74
- expl_derivative_funcp_12
 - GiNaC, 76
- expl_derivative_funcp_13
 - GiNaC, 77
- expl_derivative_funcp_14
 - GiNaC, 79
- expl_derivative_funcp_2
 - GiNaC, 63
- expl_derivative_funcp_3
 - GiNaC, 64
- expl_derivative_funcp_4
 - GiNaC, 65
- expl_derivative_funcp_5
 - GiNaC, 66
- expl_derivative_funcp_6
 - GiNaC, 68
- expl_derivative_funcp_7
 - GiNaC, 69
- expl_derivative_funcp_8
 - GiNaC, 70
- expl_derivative_funcp_9
 - GiNaC, 72
- expl_derivative_funcp_exvector
 - GiNaC, 80
- expl_derivative_use_exvector_args
 - GiNaC::function_options, 620
- exponent
 - GiNaC::power, 887
- exponop
 - GiNaC::pseries, 922
- exprs
 - GiNaC::archive, 302
- exprseq
 - GiNaC, 59
 - GiNaC::info_flags, 672
- exprseq.cpp, 1127
- exprseq.h, 1127
- exprtable
 - GiNaC::archive, 302
- exset
 - GiNaC, 58

- exvector
 - GiNaC, [58](#)
- exvectorvector
 - GiNaC, [81](#)
- F
 - GiNaC::print_memfun_handler< T, C >, [901](#)
 - GiNaC::print_ptrfun_handler< T, C >, [904](#)
- f
 - GiNaC::integral, [682](#)
 - GiNaC::print_memfun_handler< T, C >, [902](#)
 - GiNaC::print_ptrfun_handler< T, C >, [904](#)
 - GiNaC::user_defined_kernel, [1084](#)
- factor
 - GiNaC, [114](#)
- factor.cpp, [1128](#)
 - c, [1129](#)
 - cache, [1131](#)
 - cont, [1134](#)
 - DCOUT, [1129](#)
 - DCOUT2, [1129](#)
 - DCOUTVAR, [1129](#)
 - evalpoint, [1134](#)
 - factors, [1131](#)
 - k, [1132](#)
 - last, [1132](#)
 - len, [1132](#)
 - lr, [1131](#)
 - m, [1130](#)
 - modulus, [1135](#)
 - n, [1131](#)
 - one, [1131](#)
 - options, [1135](#)
 - poly, [1133](#)
 - pp, [1134](#)
 - R, [1134](#)
 - r, [1129](#)
 - syms, [1135](#)
 - syms_wox, [1134](#)
 - unit, [1134](#)
 - USE_SAME_DEGREE_FACTOR, [1129](#)
 - value, [1129](#)
 - vn, [1135](#)
 - vnlst, [1135](#)
 - x, [1133](#)
- factor.h, [1136](#)
- factorial
 - GiNaC, [213](#)
- factorial_conjugate
 - GiNaC, [136](#)
- factorial_eval
 - GiNaC, [136](#)
- factorial_evalf
 - GiNaC, [136](#)
- factorial_imag_part
 - GiNaC, [137](#)
- factorial_print_dflt_latex
 - GiNaC, [136](#)
- factorial_real_part
 - GiNaC, [137](#)
- factors
 - factor.cpp, [1131](#)
- fail.cpp, [1136](#)
- fail.h, [1137](#)
- FAST_COMPARE
 - normal.cpp, [1186](#)
- fderivative
 - GiNaC::fderivative, [557](#), [558](#)
 - GiNaC::function_options, [615](#)
- fderivative.cpp, [1137](#)
- fderivative.h, [1138](#)
- fibonacci
 - GiNaC, [214](#)
- find
 - GiNaC, [104](#)
 - GiNaC::class_info< OPT >, [350](#)
 - GiNaC::ex, [499](#)
 - GiNaC::unarchive_table_t, [1077](#)
- find_bool
 - GiNaC::archive_node, [306](#)
- find_common_factor
 - GiNaC, [201](#)
- find_dummy_indices
 - GiNaC, [119](#)
- find_ex
 - GiNaC::archive_node, [308](#)
- find_ex_by_loc
 - GiNaC::archive_node, [308](#)
- find_ex_node
 - GiNaC::archive_node, [308](#)
- find_factory_fcn
 - GiNaC, [84](#)
- find_first
 - GiNaC::archive_node, [307](#)
- find_free_and_dummy
 - GiNaC, [118](#), [119](#)
- find_function
 - GiNaC::function, [579](#)
- find_last
 - GiNaC::archive_node, [307](#)
- find_property_range
 - GiNaC::archive_node, [307](#)
- find_real_imag
 - GiNaC::mul, [768](#)
- find_string
 - GiNaC::archive_node, [307](#)
- find_unsigned
 - GiNaC::archive_node, [306](#)
- find_variant_indices
 - GiNaC, [121](#)
- findMoebiusTransform
 - GiNaC::Gt, [632](#)
- finished
 - GiNaC::composition_generator::coolmulti, [424](#)
- first
 - GiNaC::class_info< OPT >, [351](#)
- flag_overflow

- GiNaC::basic_multi_iterator< T >, 347
- flags
 - GiNaC::basic, 337
- flags.h, 1138
- force_include_tgamma
 - GiNaC, 257
- force_include_zeta1
 - GiNaC, 257
- forget
 - GiNaC::archive, 300
 - GiNaC::archive_node, 309
- format_index_value
 - GiNaC, 252
- frac_cancel
 - GiNaC, 201
- fraction_free_elimination
 - GiNaC::matrix, 735
- fsolve
 - GiNaC, 140
- func_arg_info
 - GiNaC, 125
- func_default
 - GiNaC::Gt_detail::TransformExpressionWithCache< Type >, 1075
- func_obj
 - GiNaC::Gt_detail::TransformExpressionWithCache< Type >, 1075
- FUNCP_1P
 - GiNaC, 59
- FUNCP_2P
 - GiNaC, 59
- FUNCP_CUBA
 - GiNaC, 59
- function
 - GiNaC::function, 569–573
 - GiNaC::function_options, 615
- function.cpp, 1139
- function.h, 1140
 - DECLARE_FUNCTION_10P, 1149
 - DECLARE_FUNCTION_11P, 1149
 - DECLARE_FUNCTION_12P, 1149
 - DECLARE_FUNCTION_13P, 1150
 - DECLARE_FUNCTION_14P, 1150
 - DECLARE_FUNCTION_1P, 1147
 - DECLARE_FUNCTION_2P, 1147
 - DECLARE_FUNCTION_3P, 1147
 - DECLARE_FUNCTION_4P, 1147
 - DECLARE_FUNCTION_5P, 1148
 - DECLARE_FUNCTION_6P, 1148
 - DECLARE_FUNCTION_7P, 1148
 - DECLARE_FUNCTION_8P, 1148
 - DECLARE_FUNCTION_9P, 1149
 - is_ex_the_function, 1150
 - REGISTER_FUNCTION, 1150
- function_options
 - GiNaC::function_options, 586
- functions_with_same_name
 - GiNaC::function_options, 621
- G
 - GiNaC, 141
- G2_eval
 - GiNaC, 151
- G2_evalf
 - GiNaC, 151
- G3_eval
 - GiNaC, 151
- G3_evalf
 - GiNaC, 151
- G_path
 - GiNaC::Gt_detail::pathintegral_term, 848
- gauss
 - GiNaC::determinant_algo, 427
 - GiNaC::solve_algo, 967
- gauss_elimination
 - GiNaC::matrix, 734
- gcd
 - GiNaC, 197, 218
- gcd_pf_mul
 - GiNaC, 196
- gcd_pf_pow
 - GiNaC, 196
- gcd_pf_pow_pow
 - GiNaC, 197
- generalised_Bernoulli_number
 - GiNaC, 176
- get
 - GiNaC::composition_generator, 383
 - GiNaC::partition_generator, 844
 - GiNaC::partition_with_zero_parts_generator, 846
- get_all_dummy_indices
 - GiNaC, 122
- get_all_dummy_indices_safely
 - GiNaC, 122
- get_cache
 - GiNaC::Gt_detail::TransformExpressionWithCache< Type >, 1074
- get_cache_size
 - GiNaC::integration_kernel, 689
- get_class_name
 - GiNaC::structure< T, ComparisonPolicy >, 989
- get_clifford_comp
 - GiNaC, 93
- get_close_delim
 - GiNaC::container< C >, 412
- get_commutator_sign
 - GiNaC::clifford, 362
- get_default_flags
 - GiNaC::container< C >, 412
- get_default_TeX_name
 - GiNaC, 237
- get_dim
 - GiNaC::idx, 650
- get_dim_uint
 - GiNaC, 87
- get_domain
 - GiNaC::possymbol, 871

- GiNaC::realsymbol, 936
 - GiNaC::symbol, 1035
- get_dummy_indices
 - GiNaC::indexed, 668
- get_ex
 - GiNaC::archive_node, 309
- get_factors
 - GiNaC::ncmul, 814
- get_first_symbol
 - GiNaC, 188
- get_free_indices
 - GiNaC::add, 291
 - GiNaC::basic, 328
 - GiNaC::ex, 511
 - GiNaC::indexed, 665
 - GiNaC::integral, 678
 - GiNaC::mul, 764
 - GiNaC::ncmul, 811
 - GiNaC::structure< T, ComparisonPolicy >, 998
- get_id
 - GiNaC::print_context_options, 890
 - GiNaC::registered_class_options, 941
- get_indices
 - GiNaC::indexed, 667
- get_label
 - GiNaC::wildcard, 1098
- get_last_access
 - GiNaC::remember_table_entry, 958
- get_metric
 - GiNaC::clifford, 362
- get_name
 - GiNaC::constant, 403
 - GiNaC::function, 580
 - GiNaC::function_options, 614
 - GiNaC::print_context_options, 889
 - GiNaC::registered_class_options, 941
 - GiNaC::symbol, 1035
- get_node
 - GiNaC::archive, 300
- get_nparams
 - GiNaC::function_options, 614
- get_numerical_value
 - GiNaC::Ebar_kernel, 456
 - GiNaC::Eisenstein_h_kernel, 464
 - GiNaC::Eisenstein_kernel, 472
 - GiNaC::ELi_kernel, 481
 - GiNaC::integration_kernel, 688
 - GiNaC::Kronecker_dtau_kernel, 700
 - GiNaC::Kronecker_dz_kernel, 707
 - GiNaC::modular_form_kernel, 750
- get_numerical_value_impl
 - GiNaC::integration_kernel, 689
- get_open_delim
 - GiNaC::container< C >, 412
- get_order
 - GiNaC::lanczos_coeffs, 710
- get_parent
 - GiNaC::class_info< OPT >, 350
- get_parent_name
 - GiNaC::print_context_options, 889
 - GiNaC::registered_class_options, 941
- get_point
 - GiNaC::pseries, 921
- get_pointer
 - GiNaC::ptr< T >, 931
- get_print_context
 - GiNaC, 231
- get_print_dispatch_table
 - GiNaC::registered_class_options, 941
- get_print_options
 - GiNaC, 231
- get_properties
 - GiNaC::archive_node, 308
- get_refcount
 - GiNaC::refcounted, 939
- get_registered_functions
 - GiNaC::function, 580
- get_representation_label
 - GiNaC, 89, 99
 - GiNaC::clifford, 362
 - GiNaC::color, 379
- get_result
 - GiNaC::remember_table_entry, 958
- get_serial
 - GiNaC::function, 580
- get_series_coeff
 - GiNaC::integration_kernel, 689
- get_sign
 - GiNaC::multi_iterator_permutation< T >, 789
- get_struct
 - GiNaC::structure< T, ComparisonPolicy >, 1001
- get_successful_hits
 - GiNaC::remember_table_entry, 958
- get_symbol_stats
 - GiNaC, 189
- get_symmetry
 - GiNaC::indexed, 668
- get_TeX_name
 - GiNaC::constant, 403
 - GiNaC::symbol, 1035
- get_top_node
 - GiNaC::archive, 299
- get_type
 - GiNaC::symmetry, 1044
- get_value
 - GiNaC::idx, 650
- get_var
 - GiNaC::pseries, 921
- get_vector
 - GiNaC::basic_multi_iterator< T >, 345
- getArgs
 - GiNaC::Gt, 629
- gethash
 - GiNaC::basic, 335
 - GiNaC::ex, 515
- getTau

GiNaC::Gt, [628](#)
 getZ
 GiNaC::Gt, [629](#)
 GiNaC, [19](#)
 _ex0, [265](#)
 _ex1, [266](#)
 _ex10, [269](#)
 _ex11, [270](#)
 _ex12, [270](#)
 _ex120, [272](#)
 _ex15, [270](#)
 _ex18, [271](#)
 _ex1_2, [266](#)
 _ex1_3, [266](#)
 _ex1_4, [266](#)
 _ex2, [267](#)
 _ex20, [271](#)
 _ex24, [271](#)
 _ex25, [271](#)
 _ex3, [267](#)
 _ex30, [272](#)
 _ex4, [268](#)
 _ex48, [272](#)
 _ex5, [268](#)
 _ex6, [268](#)
 _ex60, [272](#)
 _ex7, [269](#)
 _ex8, [269](#)
 _ex9, [269](#)
 _ex_1, [264](#)
 _ex_10, [261](#)
 _ex_11, [261](#)
 _ex_12, [261](#)
 _ex_120, [259](#)
 _ex_15, [261](#)
 _ex_18, [260](#)
 _ex_1_2, [264](#)
 _ex_1_3, [264](#)
 _ex_1_4, [265](#)
 _ex_2, [263](#)
 _ex_20, [260](#)
 _ex_24, [260](#)
 _ex_25, [260](#)
 _ex_3, [263](#)
 _ex_30, [259](#)
 _ex_4, [263](#)
 _ex_48, [259](#)
 _ex_5, [263](#)
 _ex_6, [262](#)
 _ex_60, [259](#)
 _ex_7, [262](#)
 _ex_8, [262](#)
 _ex_9, [262](#)
 _num0_bp, [257](#)
 _num0_p, [265](#)
 _num10_p, [269](#)
 _num11_p, [270](#)
 _num120_p, [272](#)
 _num12_p, [270](#)
 _num15_p, [270](#)
 _num18_p, [270](#)
 _num1_2_p, [266](#)
 _num1_3_p, [266](#)
 _num1_4_p, [265](#)
 _num1_p, [266](#)
 _num20_p, [271](#)
 _num24_p, [271](#)
 _num25_p, [271](#)
 _num2_p, [267](#)
 _num30_p, [271](#)
 _num3_p, [267](#)
 _num48_p, [272](#)
 _num4_p, [268](#)
 _num5_p, [268](#)
 _num60_p, [272](#)
 _num6_p, [268](#)
 _num7_p, [268](#)
 _num8_p, [269](#)
 _num9_p, [269](#)
 _num_10_p, [261](#)
 _num_11_p, [261](#)
 _num_120_p, [258](#)
 _num_12_p, [261](#)
 _num_15_p, [260](#)
 _num_18_p, [260](#)
 _num_1_2_p, [264](#)
 _num_1_3_p, [264](#)
 _num_1_4_p, [265](#)
 _num_1_p, [264](#)
 _num_20_p, [260](#)
 _num_24_p, [260](#)
 _num_25_p, [259](#)
 _num_2_p, [263](#)
 _num_30_p, [259](#)
 _num_3_p, [263](#)
 _num_48_p, [259](#)
 _num_4_p, [263](#)
 _num_5_p, [262](#)
 _num_60_p, [259](#)
 _num_6_p, [262](#)
 _num_7_p, [262](#)
 _num_8_p, [262](#)
 _num_9_p, [261](#)
 abs, [215](#)
 abs_conjugate, [129](#)
 abs_eval, [128](#)
 abs_evalf, [128](#)
 abs_expand, [129](#)
 abs_expl_derivative, [129](#)
 abs_imag_part, [130](#)
 abs_info, [130](#)
 abs_power, [130](#)
 abs_print_csrc_float, [129](#)
 abs_print_latex, [129](#)
 abs_real_part, [129](#)
 acos, [207](#)

`acos_conjugate`, 165
`acos_deriv`, 165
`acos_eval`, 165
`acos_evalf`, 165
`acosh`, 209
`acosh_conjugate`, 173
`acosh_deriv`, 173
`acosh_eval`, 172
`acosh_evalf`, 172
`adaptivesimpson`, 174
`add_symbol`, 188
`algebraic_match_mul_with_mul`, 186
`antisymmetric2`, 240
`antisymmetric3`, 240
`antisymmetric4`, 240
`antisymmetrize`, 108, 241, 244
`archive_atom`, 58
`archive_node_id`, 58
`are_ex_trivially_equal`, 101
`asin`, 207
`asin_conjugate`, 164
`asin_deriv`, 164
`asin_eval`, 164
`asin_evalf`, 164
`asin_info`, 165
`asinh`, 209
`asinh_conjugate`, 172
`asinh_deriv`, 172
`asinh_eval`, 172
`asinh_evalf`, 171
`atan`, 207, 208
`atan2_deriv`, 167
`atan2_eval`, 167
`atan2_evalf`, 167
`atan2_info`, 167
`atan_conjugate`, 166
`atan_deriv`, 166
`atan_eval`, 166
`atan_evalf`, 166
`atan_info`, 167
`atan_series`, 166
`atanh`, 210
`atanh_conjugate`, 174
`atanh_deriv`, 173
`atanh_eval`, 173
`atanh_evalf`, 173
`atanh_series`, 174
`base_and_index`, 86
`bernoulli`, 214
`Bernoulli_polynomial`, 177
`beta_deriv`, 149
`beta_eval`, 148
`beta_evalf`, 148
`beta_series`, 149
`binomial`, 214
`binomial_conjugate`, 138
`binomial_eval`, 137
`binomial_evalf`, 137
`binomial_imag_part`, 138
`binomial_real_part`, 138
`binomial_sym`, 137
`callback_registered`, 82
`canonicalize`, 240
`canonicalize_clifford`, 90
`Catalan`, 256
`CatalanEvalf`, 219
`charpoly`, 185
`clifford_bar`, 95
`clifford_inverse`, 92
`clifford_max_label`, 91
`clifford_moebius_map`, 93, 94
`clifford_norm`, 92
`clifford_prime`, 91
`clifford_star`, 96
`clifford_star_bar`, 90
`clifford_to_lst`, 93
`clifford_unit`, 87
`coeff`, 104
`coerce`, 204
`coerce< int, cln::cl_I >`, 204
`coerce< unsigned int, cln::cl_I >`, 204
`collect`, 106
`collect_common_factors`, 201
`collect_symbols`, 188
`color_d`, 98
`color_f`, 98
`color_h`, 99
`color_ONE`, 97
`color_T`, 97
`color_trace`, 99, 100
`cols`, 184
`compare_pointers`, 250
`compile_ex`, 111, 112
`conjugate`, 103
`conjugate_conjugate`, 125
`conjugate_eval`, 124
`conjugate_evalf`, 124
`conjugate_expl_derivative`, 125
`conjugate_funcp`, 60
`conjugate_funcp_1`, 61
`conjugate_funcp_10`, 73
`conjugate_funcp_11`, 74
`conjugate_funcp_12`, 75
`conjugate_funcp_13`, 77
`conjugate_funcp_14`, 78
`conjugate_funcp_2`, 62
`conjugate_funcp_3`, 63
`conjugate_funcp_4`, 65
`conjugate_funcp_5`, 66
`conjugate_funcp_6`, 67
`conjugate_funcp_7`, 69
`conjugate_funcp_8`, 70
`conjugate_funcp_9`, 71
`conjugate_funcp_exvector`, 80
`conjugate_imag_part`, 125
`conjugate_info`, 126

conjugate_print_latex, 125
 conjugate_real_part, 125
 conjugateepvector, 114
 convert_H_to_Li, 143
 cos, 206
 cos_conjugate, 162
 cos_deriv, 162
 cos_eval, 162
 cos_evalf, 161
 cos_imag_part, 162
 cos_real_part, 162
 cosh, 209
 cosh_conjugate, 170
 cosh_deriv, 169
 cosh_eval, 169
 cosh_evalf, 169
 cosh_imag_part, 170
 cosh_real_part, 169
 count_dummy_indices, 119
 count_free_indices, 120
 crc32, 101
 crctab, 256
 csgn, 220
 csgn_conjugate, 132
 csgn_eval, 132
 csgn_evalf, 131
 csgn_imag_part, 132
 csgn_power, 132
 csgn_real_part, 132
 csgn_series, 132
 csrc, 232
 csrc_cl_N, 233
 csrc_double, 233
 csrc_float, 233
 cyclic_permutation, 251
 decomp_rational, 191
 degree, 104
 delta_tensor, 246
 denom, 105, 224
 derivative_funcp, 60
 derivative_funcp_1, 61
 derivative_funcp_10, 73
 derivative_funcp_11, 74
 derivative_funcp_12, 76
 derivative_funcp_13, 77
 derivative_funcp_14, 79
 derivative_funcp_2, 63
 derivative_funcp_3, 64
 derivative_funcp_4, 65
 derivative_funcp_5, 66
 derivative_funcp_6, 68
 derivative_funcp_7, 69
 derivative_funcp_8, 70
 derivative_funcp_9, 72
 derivative_funcp_exvector, 80
 determinant, 184
 dflt, 232
 diag_matrix, 182
 diff, 107
 Digits, 258
 digits_changed_callback, 82
 dirac_gamma, 87
 dirac_gamma5, 88
 dirac_gammaL, 88
 dirac_gammaR, 88
 dirac_ONE, 86
 dirac_slash, 89
 dirac_trace, 89, 90
 dirichlet_character, 176
 divide, 193
 divide_in_z, 193
 doublefactorial, 213
 dynallocate, 84, 85
 EllipticE_deriv, 145
 EllipticE_eval, 144
 EllipticE_evalf, 144
 EllipticE_print_latex, 145
 EllipticE_series, 145
 EllipticK_deriv, 144
 EllipticK_eval, 143
 EllipticK_evalf, 143
 EllipticK_print_latex, 144
 EllipticK_series, 144
 epp, 59
 epsilon_tensor, 247, 248
 epvector, 59
 eta_conjugate, 133
 eta_eval, 133
 eta_evalf, 133
 eta_imag_part, 134
 eta_real_part, 134
 eta_series, 133
 Euler, 256
 EulerEvalf, 219
 eval, 106
 eval_funcp, 60
 eval_funcp_1, 61
 eval_funcp_10, 72
 eval_funcp_11, 74
 eval_funcp_12, 75
 eval_funcp_13, 77
 eval_funcp_14, 78
 eval_funcp_2, 62
 eval_funcp_3, 63
 eval_funcp_4, 64
 eval_funcp_5, 66
 eval_funcp_6, 67
 eval_funcp_7, 68
 eval_funcp_8, 70
 eval_funcp_9, 71
 eval_funcp_exvector, 80
 eval_integ, 106
 evalf, 106, 184
 evalf_funcp, 60
 evalf_funcp_1, 61
 evalf_funcp_10, 72

- [evalf_funcp_11](#), [74](#)
- [evalf_funcp_12](#), [75](#)
- [evalf_funcp_13](#), [77](#)
- [evalf_funcp_14](#), [78](#)
- [evalf_funcp_2](#), [62](#)
- [evalf_funcp_3](#), [63](#)
- [evalf_funcp_4](#), [65](#)
- [evalf_funcp_5](#), [66](#)
- [evalf_funcp_6](#), [67](#)
- [evalf_funcp_7](#), [68](#)
- [evalf_funcp_8](#), [70](#)
- [evalf_funcp_9](#), [71](#)
- [evalf_funcp_exvector](#), [80](#)
- [evalfunctype](#), [59](#)
- [evalm](#), [106](#)
- [ex_to](#), [111](#)
- [exadd](#), [224](#)
- [exhashmap](#), [81](#)
- [exmap](#), [59](#)
- [exminus](#), [224](#)
- [exmul](#), [224](#)
- [exp](#), [205](#)
- [exp_conjugate](#), [158](#)
- [exp_deriv](#), [157](#)
- [exp_eval](#), [157](#)
- [exp_evalf](#), [157](#)
- [exp_expand](#), [157](#)
- [exp_imag_part](#), [157](#)
- [exp_info](#), [158](#)
- [exp_power](#), [158](#)
- [exp_real_part](#), [157](#)
- [expand](#), [103](#), [183](#)
- [expand_dummy_sum](#), [124](#)
- [expand_funcp](#), [60](#)
- [expand_funcp_1](#), [61](#)
- [expand_funcp_10](#), [73](#)
- [expand_funcp_11](#), [74](#)
- [expand_funcp_12](#), [76](#)
- [expand_funcp_13](#), [77](#)
- [expand_funcp_14](#), [79](#)
- [expand_funcp_2](#), [63](#)
- [expand_funcp_3](#), [64](#)
- [expand_funcp_4](#), [65](#)
- [expand_funcp_5](#), [66](#)
- [expand_funcp_6](#), [68](#)
- [expand_funcp_7](#), [69](#)
- [expand_funcp_8](#), [70](#)
- [expand_funcp_9](#), [72](#)
- [expand_funcp_exvector](#), [80](#)
- [expl_derivative_funcp](#), [60](#)
- [expl_derivative_funcp_1](#), [62](#)
- [expl_derivative_funcp_10](#), [73](#)
- [expl_derivative_funcp_11](#), [74](#)
- [expl_derivative_funcp_12](#), [76](#)
- [expl_derivative_funcp_13](#), [77](#)
- [expl_derivative_funcp_14](#), [79](#)
- [expl_derivative_funcp_2](#), [63](#)
- [expl_derivative_funcp_3](#), [64](#)
- [expl_derivative_funcp_4](#), [65](#)
- [expl_derivative_funcp_5](#), [66](#)
- [expl_derivative_funcp_6](#), [68](#)
- [expl_derivative_funcp_7](#), [69](#)
- [expl_derivative_funcp_8](#), [70](#)
- [expl_derivative_funcp_9](#), [72](#)
- [expl_derivative_funcp_exvector](#), [80](#)
- [exprseq](#), [59](#)
- [exset](#), [58](#)
- [exvector](#), [58](#)
- [exvectorvector](#), [81](#)
- [factor](#), [114](#)
- [factorial](#), [213](#)
- [factorial_conjugate](#), [136](#)
- [factorial_eval](#), [136](#)
- [factorial_evalf](#), [136](#)
- [factorial_imag_part](#), [137](#)
- [factorial_print_dflt_latex](#), [136](#)
- [factorial_real_part](#), [137](#)
- [fibonacci](#), [214](#)
- [find](#), [104](#)
- [find_common_factor](#), [201](#)
- [find_dummy_indices](#), [119](#)
- [find_factory_fcn](#), [84](#)
- [find_free_and_dummy](#), [118](#), [119](#)
- [find_variant_indices](#), [121](#)
- [force_include_tgamma](#), [257](#)
- [force_include_zeta1](#), [257](#)
- [format_index_value](#), [252](#)
- [frac_cancel](#), [201](#)
- [fsolve](#), [140](#)
- [func_arg_info](#), [125](#)
- [FUNCP_1P](#), [59](#)
- [FUNCP_2P](#), [59](#)
- [FUNCP_CUBA](#), [59](#)
- [G](#), [141](#)
- [G2_eval](#), [151](#)
- [G2_evalf](#), [151](#)
- [G3_eval](#), [151](#)
- [G3_evalf](#), [151](#)
- [gcd](#), [197](#), [218](#)
- [gcd_pf_mul](#), [196](#)
- [gcd_pf_pow](#), [196](#)
- [gcd_pf_pow_pow](#), [197](#)
- [generalised_Bernoulli_number](#), [176](#)
- [get_all_dummy_indices](#), [122](#)
- [get_all_dummy_indices_safely](#), [122](#)
- [get_clifford_comp](#), [93](#)
- [get_default_TeX_name](#), [237](#)
- [get_dim_uint](#), [87](#)
- [get_first_symbol](#), [188](#)
- [get_print_context](#), [231](#)
- [get_print_options](#), [231](#)
- [get_representation_label](#), [89](#), [99](#)
- [get_symbol_stats](#), [189](#)
- [GINAC_BIND_UNARCHIVER](#), [82](#), [85](#), [86](#), [96](#), [101](#), [115–117](#), [120](#), [175](#), [177–180](#), [182](#), [187](#), [188](#), [203](#), [234](#), [235](#), [237](#), [238](#), [245](#), [255](#), [257](#)

- GINAC_DECLARE_UNARCHIVER, [82](#), [94](#), [95](#),
[100](#), [101](#), [115](#), [116](#), [118](#), [119](#), [124](#), [175](#), [180](#),
[181](#), [183](#), [187](#), [188](#), [219](#), [234](#), [235](#), [237](#), [238](#),
[241](#), [249](#), [255](#)
- GINAC_IMPLEMENT_REGISTERED_CLASS_OPT,
[82](#), [84](#), [85](#), [96](#), [101](#), [114–116](#), [120](#), [174](#), [177–](#)
[181](#), [186](#), [187](#), [202](#), [233](#), [235–238](#), [245](#), [254](#)
- golden_ratio_hash, [250](#)
- guess_precision, [211](#)
- H_deriv, [155](#)
- H_eval, [154](#)
- H_evalf, [154](#)
- H_print_latex, [155](#)
- H_series, [154](#)
- has, [103](#)
- hasindex, [122](#)
- haswild, [255](#)
- heur_gcd, [196](#)
- heur_gcd_z, [195](#)
- hold_ncmul, [187](#)
- I, [257](#)
- idx, [257](#)
- idx_symmetrization, [121](#)
- ifactor, [175](#)
- imag, [223](#)
- imag_part, [103](#)
- imag_part_conjugate, [128](#)
- imag_part_eval, [127](#)
- imag_part_evalf, [127](#)
- imag_part_expl_derivative, [128](#)
- imag_part_funcp, [60](#)
- imag_part_funcp_1, [61](#)
- imag_part_funcp_10, [73](#)
- imag_part_funcp_11, [74](#)
- imag_part_funcp_12, [76](#)
- imag_part_funcp_13, [77](#)
- imag_part_funcp_14, [79](#)
- imag_part_funcp_2, [62](#)
- imag_part_funcp_3, [64](#)
- imag_part_funcp_4, [65](#)
- imag_part_funcp_5, [66](#)
- imag_part_funcp_6, [67](#)
- imag_part_funcp_7, [69](#)
- imag_part_funcp_8, [70](#)
- imag_part_funcp_9, [71](#)
- imag_part_funcp_exvector, [80](#)
- imag_part_imag_part, [128](#)
- imag_part_print_latex, [127](#)
- imag_part_real_part, [128](#)
- index0, [238](#)
- index1, [239](#)
- index2, [239](#)
- index3, [239](#)
- index_dimensions, [233](#)
- indices_consistent, [120](#)
- info_funcp, [61](#)
- info_funcp_1, [62](#)
- info_funcp_10, [74](#)
- info_funcp_11, [75](#)
- info_funcp_12, [76](#)
- info_funcp_13, [78](#)
- info_funcp_14, [79](#)
- info_funcp_2, [63](#)
- info_funcp_3, [64](#)
- info_funcp_4, [66](#)
- info_funcp_5, [67](#)
- info_funcp_6, [68](#)
- info_funcp_7, [70](#)
- info_funcp_8, [71](#)
- info_funcp_9, [72](#)
- info_funcp_exvector, [81](#)
- interpolate, [194](#)
- inverse, [185](#), [220](#)
- iquo, [217](#)
- irem, [216](#)
- is_a, [84](#), [110](#), [235](#)
- is_cinteger, [222](#)
- is_clifford_tinfo, [95](#)
- is_color_tinfo, [99](#)
- is_crational, [222](#)
- is_dirac_slash, [86](#)
- is_discriminant_of_quadratic_number_field, [175](#)
- is_dummy_pair, [117](#)
- is_even, [221](#)
- is_exactly_a, [84](#), [110](#)
- is_integer, [221](#)
- is_negative, [221](#)
- is_nonneg_integer, [221](#)
- is_odd, [222](#)
- is_order_function, [143](#)
- is_polynomial, [104](#)
- is_pos_integer, [221](#)
- is_positive, [221](#)
- is_prime, [222](#)
- is_rational, [222](#)
- is_real, [222](#)
- is_terminating, [236](#)
- is_the_function, [116](#)
- is_the_function< G_SERIAL >, [141](#)
- is_the_function< iterated_integral_SERIAL >, [143](#)
- is_the_function< psi_SERIAL >, [142](#)
- is_the_function< zeta_SERIAL >, [141](#)
- is_zero, [109](#), [220](#)
- isqrt, [219](#)
- iterated_integral, [142](#)
- iterated_integral2_eval, [146](#)
- iterated_integral2_evalf, [145](#)
- iterated_integral3_eval, [146](#)
- iterated_integral3_evalf, [146](#)
- iterated_integral_evalf_impl, [145](#)
- kronecker_symbol, [175](#)
- latex, [232](#)
- lcm, [198](#), [218](#)
- lcm_of_coefficients_denominators, [189](#)
- lcmcoeff, [189](#)
- ldegree, [104](#)

lgamma, 211, 212
lgamma_conjugate, 147
lgamma_deriv, 147
lgamma_eval, 146
lgamma_evalf, 146
lgamma_series, 147
lhs, 109
Li2, 211
Li2_, 210
Li2_conjugate, 135
Li2_deriv, 134
Li2_eval, 134
Li2_evalf, 134
Li2_projection, 210
Li2_series, 135, 210
Li3_eval, 135
Li_deriv, 152
Li_eval, 152
Li_evalf, 152
Li_print_latex, 152
Li_series, 152
library_initializer, 257
link_ex, 112, 113
log, 205
log2, 249
log_conjugate, 160
log_deriv, 159
log_eval, 158
log_evalf, 158
log_expand, 159
log_imag_part, 159
log_info, 160
log_real_part, 159
log_series, 159
lookup_map, 81
lorentz_eps, 248
lorentz_g, 247
lsolve, 140
lst, 81
lst_to_clifford, 92
lst_to_matrix, 182
make_hash_seed, 116
make_real_float, 202
make_return_type_t, 236
map_eval_integ, 255
map_evalm, 255
match, 107
metric_tensor, 246
minimal_dim, 118
mod, 215
multinomial_coefficient, 250
multiply_lcm, 190
my_ios_callback, 230
my_ios_index, 230
next_print_context_id, 258
no_index_dimensions, 233
nops, 102, 183
normal, 105
not_symmetric, 239
number_of_type, 120
numer, 105, 223
numer_denom, 105
op, 109
operator!=, 229
operator<, 230
operator<<, 83, 102, 219, 231, 252–254
operator<=, 230
operator>, 230
operator>>, 83, 232
operator>=, 230
operator+, 224, 225, 227
operator++, 228, 229
operator+=", 226
operator-, 225, 227
operator--, 228, 229
operator-=, 226, 227
operator/, 225, 226
operator/=", 226, 227
operator==, 229
operator*, 225
operator*=", 226, 227
Order_conjugate, 139
Order_eval, 138
Order_expl_derivative, 139
Order_imag_part, 139
Order_power, 139
Order_real_part, 139
Order_series, 138
paramset, 60
permutation_sign, 251
permute_free_index_to_front, 97
Pi, 256
PiEvalf, 219
pow, 220, 234
power_funcp, 60
power_funcp_1, 62
power_funcp_10, 73
power_funcp_11, 75
power_funcp_12, 76
power_funcp_13, 78
power_funcp_14, 79
power_funcp_2, 63
power_funcp_3, 64
power_funcp_4, 65
power_funcp_5, 67
power_funcp_6, 68
power_funcp_7, 69
power_funcp_8, 71
power_funcp_9, 72
power_funcp_exvector, 80
prem, 192
primitive_dirichlet_character, 176
print_context_class_info, 82
print_func< print_context >, 117
print_func< print_dflt >, 85, 96, 245
print_funcp, 61

- print_funcp_1, 62
- print_funcp_10, 73
- print_funcp_11, 75
- print_funcp_12, 76
- print_funcp_13, 78
- print_funcp_14, 79
- print_funcp_2, 63
- print_funcp_3, 64
- print_funcp_4, 65
- print_funcp_5, 67
- print_funcp_6, 68
- print_funcp_7, 69
- print_funcp_8, 71
- print_funcp_9, 72
- print_funcp_exvector, 81
- print_integer_csrc, 203
- print_operator, 237
- print_real_cl_N, 205
- print_real_csrc, 204
- print_real_number, 203
- print_sym_pow, 233
- product_to_exvector, 121
- psi, 142, 212
- psi1_deriv, 150
- psi1_eval, 149
- psi1_evalf, 149
- psi1_series, 150
- psi2_deriv, 150
- psi2_eval, 150
- psi2_evalf, 150
- psi2_series, 151
- python, 232
- python_repr, 232
- quo, 190
- rank, 185
- read_real_float, 202
- read_unsigned, 83
- real, 223
- real_part, 103
- real_part_conjugate, 126
- real_part_eval, 126
- real_part_evalf, 126
- real_part_expl_derivative, 127
- real_part_funcp, 60
- real_part_funcp_1, 61
- real_part_funcp_10, 73
- real_part_funcp_11, 74
- real_part_funcp_12, 75
- real_part_funcp_13, 77
- real_part_funcp_14, 78
- real_part_funcp_2, 62
- real_part_funcp_3, 64
- real_part_funcp_4, 65
- real_part_funcp_5, 66
- real_part_funcp_6, 67
- real_part_funcp_7, 69
- real_part_funcp_8, 70
- real_part_funcp_9, 71
- real_part_funcp_exvector, 80
- real_part_imag_part, 127
- real_part_print_latex, 126
- real_part_real_part, 127
- reduced_matrix, 183
- reeval_ncmul, 187
- REGISTER_FUNCTION, 126–128, 130, 131, 133–139, 144, 145, 147–149, 153–155, 158, 160–162, 164–174
- registered_class_info, 82
- rem, 191
- remove_dirac_ONE, 91
- rename_dummy_indices, 120
- rename_dummy_indices_uniquely, 123
- replace_with_symbol, 200
- reposition_dummy_indices, 121
- resultant, 202
- rhs, 109
- rotate_left, 250
- rows, 184
- S_deriv, 153
- S_eval, 153
- S_evalf, 153
- S_print_latex, 154
- S_series, 153
- series, 107
- series_funcp, 61
- series_funcp_1, 62
- series_funcp_10, 73
- series_funcp_11, 75
- series_funcp_12, 76
- series_funcp_13, 78
- series_funcp_14, 79
- series_funcp_2, 63
- series_funcp_3, 64
- series_funcp_4, 65
- series_funcp_5, 67
- series_funcp_6, 68
- series_funcp_7, 69
- series_funcp_8, 71
- series_funcp_9, 72
- series_funcp_exvector, 80
- series_to_poly, 235
- set_print_context, 231
- set_print_func, 236
- set_print_options, 231
- shaker_sort, 251
- simplify_indexed, 107, 122
- simplify_indexed_product, 122
- sin, 206
- sin_conjugate, 161
- sin_deriv, 160
- sin_eval, 160
- sin_evalf, 160
- sin_imag_part, 161
- sin_real_part, 161
- sinh, 208
- sinh_conjugate, 169

- [sinh_deriv](#), 168
- [sinh_eval](#), 168
- [sinh_evalf](#), 168
- [sinh_imag_part](#), 168
- [sinh_real_part](#), 168
- [smod](#), 216
- [spinor_metric](#), 247
- [spmap](#), 81
- [sprem](#), 192
- [sqrfree](#), 199
- [sqrfree_parfrac](#), 199
- [sqrfree_yun](#), 198
- [sqrt](#), 218, 234
- [sr_gcd](#), 194
- [step](#), 220
- [step_conjugate](#), 131
- [step_eval](#), 130
- [step_evalf](#), 130
- [step_imag_part](#), 131
- [step_real_part](#), 131
- [step_series](#), 131
- [sub_matrix](#), 183
- [subs](#), 110
- [subsvalue](#), 174
- [swap](#), 109, 114
- [sy_anti](#), 243
- [sy_cycl](#), 243, 244
- [sy_none](#), 241, 242
- [sy_symm](#), 242, 243
- [sym_desc_vec](#), 81
- [symbolic_matrix](#), 182, 186
- [symm](#), 240
- [symmetric2](#), 239
- [symmetric3](#), 239
- [symmetric4](#), 239
- [symmetrize](#), 108, 241, 244
- [symmetrize_cyclic](#), 108, 241, 244
- [synthesize_func](#), 58
- [tan](#), 206
- [tan_conjugate](#), 164
- [tan_deriv](#), 163
- [tan_eval](#), 163
- [tan_evalf](#), 163
- [tan_imag_part](#), 163
- [tan_real_part](#), 163
- [tan_series](#), 163
- [tanh](#), 209
- [tanh_conjugate](#), 171
- [tanh_deriv](#), 170
- [tanh_eval](#), 170
- [tanh_evalf](#), 170
- [tanh_imag_part](#), 171
- [tanh_real_part](#), 171
- [tanh_series](#), 171
- [tensor](#), 256
- [tgamma](#), 212
- [tgamma_conjugate](#), 148
- [tgamma_deriv](#), 148
- [tgamma_eval](#), 147
- [tgamma_evalf](#), 147
- [tgamma_series](#), 148
- [to_double](#), 223
- [to_int](#), 223
- [to_long](#), 223
- [to_polynomial](#), 106
- [to_rational](#), 105
- [trace](#), 184
- [trace_string](#), 89
- [transpose](#), 184
- [tree](#), 232
- [trig_info](#), 161
- [tryfactsubs](#), 186
- [uintvector](#), 81
- [unarch_table_instance](#), 255
- [unarchive_map_t](#), 58
- [unit_matrix](#), 182, 185
- [unlink_ex](#), 113
- [unsignedvector](#), 81
- [version_major](#), 258
- [version_micro](#), 258
- [version_minor](#), 258
- [wild](#), 255
- [write_real_float](#), 203
- [write_unsigned](#), 83
- [zeta](#), 141, 211
- [zeta1_deriv](#), 156
- [zeta1_eval](#), 155
- [zeta1_evalf](#), 155
- [zeta1_print_latex](#), 156
- [zeta2_deriv](#), 156
- [zeta2_eval](#), 156
- [zeta2_evalf](#), 156
- [zeta2_print_latex](#), 156
- [zetaderiv_deriv](#), 136
- [zetaderiv_eval](#), 135
- [ginac.h](#), 1151
- [GiNaC::numeric_digits](#), 278
 - [_numeric_digits](#), 279
 - [add_callback](#), 280
 - [callbacklist](#), 280
 - [digits](#), 280
 - [operator long](#), 279
 - [operator=](#), 279
 - [print](#), 279
 - [too_late](#), 280
- [GiNaC::add](#), 281
 - [add](#), 286, 287
 - [coeff](#), 288
 - [combine_ex_with_coeff_to_pair](#), 293
 - [combine_pair_with_coeff_to_pair](#), 293
 - [conjugate](#), 291
 - [degree](#), 288
 - [derivative](#), 292
 - [do_print](#), 294
 - [do_print_csrc](#), 295
 - [do_print_latex](#), 295

- do_print_python_repr, 295
- eval, 288
- eval_ncmul, 292
- evalm, 289
- expand, 294
- get_free_indices, 291
- imag_part, 291
- info, 287
- integer_content, 290
- is_polynomial, 287
- ldegree, 288
- max_coefficient, 291
- mul, 295
- normal, 289
- power, 295
- precedence, 287
- print_add, 294
- real_part, 291
- recombine_pair_to_ex, 294
- return_type, 292
- return_type_tinfo, 292
- series, 289
- smod, 290
- split_ex_to_pair, 293
- thisexpairseq, 292, 293
- GiNaC::archive, 296
 - ~archive, 297
 - add_node, 300
 - archive, 297, 298
 - archive_ex, 298
 - atomize, 301
 - atoms, 302
 - clear, 300
 - exprs, 302
 - exprtable, 302
 - forget, 300
 - get_node, 300
 - get_top_node, 299
 - inv_at_cit, 297
 - inverse_atoms, 302
 - nodes, 302
 - num_expressions, 299
 - operator<<, 301
 - operator>>, 301
 - printraw, 300
 - unarchive_ex, 298, 299
 - unatomize, 301
- GiNaC::archive::archived_ex, 312
 - archived_ex, 312
 - name, 313
 - root, 313
- GiNaC::archive_node, 303
 - a, 310
 - add_bool, 305
 - add_ex, 306
 - add_string, 306
 - add_unsigned, 306
 - archive_node, 305
 - archive_node_cit, 304
 - e, 311
 - find_bool, 306
 - find_ex, 308
 - find_ex_by_loc, 308
 - find_ex_node, 308
 - find_first, 307
 - find_last, 307
 - find_property_range, 307
 - find_string, 307
 - find_unsigned, 306
 - forget, 309
 - get_ex, 309
 - get_properties, 308
 - has_ex, 309
 - has_expression, 311
 - has_same_ex_as, 309
 - operator<<, 310
 - operator>>, 310
 - operator=, 305
 - printraw, 310
 - property_type, 305
 - propinfovector, 304
 - props, 310
 - PTYPE_BOOL, 305
 - PTYPE_NODE, 305
 - PTYPE_STRING, 305
 - PTYPE_UNSIGNED, 305
 - unarchive, 309
- GiNaC::archive_node::archive_node_cit_range, 311
 - begin, 311
 - end, 311
- GiNaC::archive_node::property, 908
 - name, 909
 - property, 909
 - type, 909
 - value, 909
- GiNaC::archive_node::property_info, 909
 - count, 911
 - name, 910
 - property_info, 910
 - type, 910
- GiNaC::basic, 313
 - ~basic, 317
 - accept, 323
 - add_indexed, 328
 - archive, 332
 - basic, 317
 - calchash, 331
 - clearflag, 335
 - coeff, 324
 - collect, 325
 - compare, 334
 - compare_same_type, 331
 - conjugate, 330
 - contract_with, 329
 - dbgprint, 319
 - dbgprinttree, 319

- degree, [324](#)
- derivative, [325](#)
- diff, [333](#)
- do_print, [336](#)
- do_print_python_repr, [336](#)
- do_print_tree, [336](#)
- duplicate, [317](#)
- ensure_if_modifiable, [336](#)
- eval, [317](#)
- eval_indexed, [319](#)
- eval_integ, [318](#)
- eval_ncmul, [318](#)
- evalf, [318](#)
- evalm, [318](#)
- ex, [337](#)
- expand, [324](#)
- flags, [337](#)
- get_free_indices, [328](#)
- gethash, [335](#)
- has, [322](#)
- hashvalue, [337](#)
- hold, [334](#)
- imag_part, [330](#)
- info, [320](#)
- integer_content, [327](#)
- is_equal, [334](#)
- is_equal_same_type, [331](#)
- is_polynomial, [324](#)
- ldegree, [324](#)
- let_op, [321](#)
- map, [323](#)
- match, [322](#)
- match_same_type, [322](#)
- max_coefficient, [328](#)
- nops, [320](#)
- normal, [326](#)
- op, [320](#)
- operator=, [317](#)
- operator[], [321](#), [322](#)
- precedence, [320](#)
- print, [319](#)
- print_dispatch, [331](#), [332](#)
- read_archive, [332](#)
- real_part, [330](#)
- return_type, [330](#)
- return_type_tinfo, [330](#)
- scalar_mul_indexed, [329](#)
- series, [325](#)
- setflag, [335](#)
- smod, [327](#)
- subs, [323](#)
- subs_one_level, [333](#)
- to_polynomial, [327](#)
- to_rational, [326](#)
- GiNaC::basic_log_kernel, [338](#)
 - do_print, [342](#)
 - series_coeff_impl, [342](#)
- GiNaC::basic_multi_iterator< T >, [342](#)
 - ~basic_multi_iterator, [344](#)
 - B, [347](#)
 - basic_multi_iterator, [344](#)
 - flag_overflow, [347](#)
 - get_vector, [345](#)
 - init, [346](#)
 - N, [347](#)
 - operator<<, [347](#)
 - operator(), [346](#)
 - operator++, [346](#)
 - operator[], [345](#)
 - overflow, [345](#)
 - size, [345](#)
 - v, [347](#)
- GiNaC::basic_partition_generator, [348](#)
 - basic_partition_generator, [348](#)
 - mpgen, [348](#)
- GiNaC::basic_partition_generator::mpartition2, [751](#)
 - m, [752](#)
 - mpartition2, [752](#)
 - n, [752](#)
 - next_partition, [752](#)
 - x, [752](#)
- GiNaC::class_info< OPT >, [349](#)
 - class_info, [349](#)
 - dump_hierarchy, [350](#)
 - dump_tree, [350](#)
 - find, [350](#)
 - first, [351](#)
 - get_parent, [350](#)
 - identify_parents, [350](#)
 - next, [351](#)
 - options, [351](#)
 - parent, [351](#)
 - parents_identified, [351](#)
- GiNaC::class_info< OPT >::tree_node, [1076](#)
 - add_child, [1076](#)
 - children, [1076](#)
 - info, [1076](#)
 - tree_node, [1076](#)
- GiNaC::clifford, [352](#)
 - archive, [360](#)
 - clifford, [359](#)
 - commutator_sign, [364](#)
 - do_print_dflt, [363](#)
 - do_print_latex, [364](#)
 - do_print_tree, [364](#)
 - eval_ncmul, [360](#)
 - get_commutator_sign, [362](#)
 - get_metric, [362](#)
 - get_representation_label, [362](#)
 - let_op, [363](#)
 - match_same_type, [361](#)
 - metric, [364](#)
 - nops, [362](#)
 - op, [363](#)
 - precedence, [360](#)
 - read_archive, [360](#)

- representation_label, 364
- return_type, 361
- return_type_tinfo, 361
- same_metric, 362
- subs, 363
- thiscontainer, 361
- GiNaC::cliffordunit, 365
 - contract_with, 369
 - do_print, 369
 - do_print_latex, 369
- GiNaC::color, 370
 - archive, 377
 - color, 376, 377
 - eval_ncmul, 378
 - get_representation_label, 379
 - match_same_type, 378
 - read_archive, 377
 - representation_label, 379
 - return_type, 379
 - return_type_tinfo, 379
 - thiscontainer, 378, 379
- GiNaC::compare_all_equal< T >, 380
 - ~compare_all_equal, 380
 - struct_compare, 380
 - struct_is_equal, 380
- GiNaC::compare_bitwise< T >, 381
 - ~compare_bitwise, 381
 - struct_compare, 381
 - struct_is_equal, 381
- GiNaC::compare_std_less< T >, 382
 - ~compare_std_less, 382
 - struct_compare, 382
 - struct_is_equal, 382
- GiNaC::composition_generator, 383
 - atend, 384
 - cmgen, 384
 - composition, 384
 - composition_generator, 383
 - current_updated, 384
 - get, 383
 - next, 383
 - trivial, 384
- GiNaC::composition_generator::coolmulti, 423
 - ~coolmulti, 424
 - after_i, 424
 - coolmulti, 424
 - finished, 424
 - head, 424
 - i, 424
 - next_permutation, 424
- GiNaC::composition_generator::coolmulti::element, 474
 - ~element, 475
 - element, 475
 - next, 475
 - value, 475
- GiNaC::const_iterator, 385
 - const_iterator, 386
 - const_postorder_iterator, 389
- const_preorder_iterator, 389
- difference_type, 386
- e, 389
- ex, 389
- i, 389
- iterator_category, 386
- operator!=, 388
- operator<, 388
- operator<=, 388
- operator>, 388
- operator>=, 389
- operator+, 387, 389
- operator++, 387
- operator+=", 387
- operator-, 388, 389
- operator->, 386
- operator--, 387
- operator-=, 388
- operator==, 388
- operator[], 387
- operator*, 386
- pointer, 386
- reference, 386
- value_type, 386
- GiNaC::const_postorder_iterator, 390
 - const_postorder_iterator, 391
 - descend, 392
 - difference_type, 391
 - increment, 392
 - iterator_category, 390
 - operator!=, 392
 - operator++, 391, 392
 - operator->, 391
 - operator==, 392
 - operator*, 391
 - pointer, 391
 - reference, 391
 - s, 393
 - value_type, 390
- GiNaC::const_preorder_iterator, 393
 - const_preorder_iterator, 394
 - difference_type, 394
 - increment, 395
 - iterator_category, 394
 - operator!=, 395
 - operator++, 395
 - operator->, 394
 - operator==, 395
 - operator*, 394
 - pointer, 394
 - reference, 394
 - s, 395
 - value_type, 394
- GiNaC::constant, 396
 - archive, 401
 - calchash, 402
 - conjugate, 401
 - constant, 400

- derivative, 402
- do_print, 403
- do_print_latex, 403
- do_print_python_repr, 403
- do_print_tree, 403
- domain, 405
- ef, 404
- evalf, 400
- get_name, 403
- get_TeX_name, 403
- imag_part, 401
- info, 400
- is_equal_same_type, 402
- is_polynomial, 400
- name, 404
- next_serial, 404
- number, 404
- read_archive, 401
- real_part, 401
- serial, 404
- TeX_name, 404
- GiNaC::container< C >, 405
 - append, 417
 - archive, 414
 - begin, 418
 - conjugate, 414
 - const_iterator, 410
 - const_reverse_iterator, 411
 - container, 411
 - do_print, 419
 - do_print_python, 419
 - do_print_python_repr, 419
 - do_print_tree, 419
 - end, 418
 - get_close_delim, 412
 - get_default_flags, 412
 - get_open_delim, 412
 - imag_part, 415
 - info, 412
 - is_equal_same_type, 415
 - let_op, 413
 - nops, 412
 - op, 413
 - precedence, 412
 - prepend, 417
 - printseq, 416
 - rbegin, 418
 - read_archive, 414
 - real_part, 415
 - remove_all, 417
 - remove_first, 417
 - remove_last, 417
 - rend, 418
 - sort, 418
 - sort_, 416
 - STLT, 410
 - subs, 413
 - subchildren, 419
 - thiscontainer, 415, 416
 - unique, 418
 - unique_, 417, 419
- GiNaC::container_storage< C >, 420
 - ~container_storage, 421
 - container_storage, 421
 - reserve, 422
 - seq, 423
 - STLT, 421
- GiNaC::derivative_map_function, 425
 - derivative_map_function, 426
 - operator(), 426
 - s, 426
- GiNaC::determinant_algo, 426
 - automatic, 427
 - bareiss, 427
 - divfree, 427
 - gauss, 427
 - laplace, 427
- GiNaC::diracgamma, 427
 - contract_with, 431
 - do_print, 431
 - do_print_latex, 432
- GiNaC::diracgamma5, 432
 - conjugate, 436
 - do_print, 436
 - do_print_latex, 436
- GiNaC::diracgammaL, 436
 - conjugate, 440
 - do_print, 440
 - do_print_latex, 440
- GiNaC::diracgammaR, 441
 - conjugate, 444
 - do_print, 444
 - do_print_latex, 445
- GiNaC::diracone, 445
 - do_print, 449
 - do_print_latex, 449
- GiNaC::do_taylor, 449
- GiNaC::domain, 450
 - complex, 450
 - positive, 450
 - real, 450
- GiNaC::dunno, 450
- GiNaC::Ebar_kernel, 451
 - do_print, 456
 - Ebar_kernel, 455
 - get_numerical_value, 456
 - is_numeric, 456
 - let_op, 455
 - m, 457
 - n, 457
 - nops, 455
 - op, 455
 - series_coeff_impl, 456
 - x, 457
 - y, 457
- GiNaC::Eisenstein_h_kernel, 458

- C_norm, 466
- coefficient_a0, 464
- coefficient_an, 465
- do_print, 465
- Eisenstein_h_kernel, 462
- get_numerical_value, 464
- is_numeric, 463
- k, 466
- Laurent_series, 464
- let_op, 463
- N, 466
- nops, 463
- op, 463
- q_expansion_modular_form, 465
- r, 466
- s, 466
- series, 463
- uses_Laurent_series, 464
- GiNaC::Eisenstein_kernel, 466
 - a, 474
 - b, 474
 - C_norm, 474
 - do_print, 473
 - Eisenstein_kernel, 471
 - get_numerical_value, 472
 - is_numeric, 472
 - K, 474
 - k, 473
 - Laurent_series, 472
 - let_op, 472
 - N, 473
 - nops, 471
 - op, 472
 - q_expansion_modular_form, 473
 - series, 471
 - uses_Laurent_series, 473
- GiNaC::ELi_kernel, 476
 - do_print, 481
 - ELi_kernel, 480
 - get_numerical_value, 481
 - is_numeric, 481
 - let_op, 480
 - m, 482
 - n, 482
 - nops, 480
 - op, 480
 - series_coeff_impl, 481
 - x, 482
 - y, 482
- GiNaC::error_and_integral, 483
 - error, 483
 - error_and_integral, 483
 - integral, 483
- GiNaC::error_and_integral_is_less, 484
 - operator(), 484
- GiNaC::eval_integ_map_function, 484
 - operator(), 485
- GiNaC::evalf_map_function, 485
 - operator(), 486
- GiNaC::evalm_map_function, 486
 - operator(), 487
- GiNaC::ex, 487
 - accept, 501
 - antisymmetrize, 513, 514
 - archive_node, 517
 - are_ex_trivially_equal, 517
 - begin, 493
 - bp, 518
 - coeff, 503
 - collect, 504
 - compare, 512
 - conjugate, 498
 - construct_from_basic, 515
 - construct_from_double, 516
 - construct_from_int, 515
 - construct_from_long, 516
 - construct_from_longlong, 516
 - construct_from_string_and_lst, 516
 - construct_from_uint, 515
 - construct_from_ulong, 516
 - construct_from_ulonglong, 516
 - content, 508
 - dbgprint, 495
 - dbgprinttree, 496
 - degree, 502
 - denom, 506
 - diff, 504
 - end, 493
 - eval, 494
 - eval_integ, 495
 - eval_ncmul, 494
 - evalf, 494
 - evalm, 494
 - ex, 491, 492
 - ex_to, 517
 - expand, 503
 - find, 499
 - get_free_indices, 511
 - gethash, 515
 - has, 499
 - imag_part, 499
 - info, 496
 - integer_content, 508
 - is_a, 518
 - is_equal, 512
 - is_exactly_a, 518
 - is_polynomial, 502
 - is_zero, 512
 - is_zero_matrix, 513
 - lcoeff, 503
 - ldegree, 502
 - let_op, 498
 - lhs, 498
 - makewriteable, 517
 - map, 501
 - match, 500

- max_coefficient, [510](#)
- nops, [496](#)
- normal, [505](#)
- numer, [506](#)
- numer_denom, [507](#)
- op, [497](#)
- operator[], [497](#), [498](#)
- postorder_begin, [494](#)
- postorder_end, [494](#)
- preorder_begin, [493](#)
- preorder_end, [493](#)
- primpart, [509](#)
- print, [495](#)
- real_part, [499](#)
- return_type, [514](#)
- return_type_tinfo, [514](#)
- rhs, [498](#)
- series, [504](#)
- share, [517](#)
- simplify_indexed, [511](#)
- smod, [510](#)
- subs, [500](#), [501](#)
- swap, [493](#)
- symmetrize, [513](#)
- symmetrize_cyclic, [514](#)
- tcoeff, [503](#)
- to_polynomial, [506](#)
- to_rational, [505](#)
- traverse, [502](#)
- traverse_postorder, [502](#)
- traverse_preorder, [501](#)
- unit, [507](#)
- unitcontprim, [510](#)
- GiNaC::ex_base_is_less, [519](#)
- operator(), [519](#)
- GiNaC::ex_is_equal, [519](#)
- operator(), [519](#)
- GiNaC::ex_is_less, [519](#)
- operator(), [520](#)
- GiNaC::ex_swap, [520](#)
- operator(), [520](#)
- GiNaC::expair, [520](#)
- coeff, [523](#)
- compare, [522](#)
- conjugate, [523](#)
- expair, [521](#)
- is_canonical_numeric, [522](#)
- is_equal, [522](#)
- is_less, [522](#)
- print, [522](#)
- rest, [523](#)
- swap, [522](#)
- GiNaC::expair_is_less, [524](#)
- operator(), [524](#)
- GiNaC::expair_rest_is_less, [524](#)
- operator(), [525](#)
- GiNaC::expair_swap, [525](#)
- operator(), [525](#)
- GiNaC::expairseq, [525](#)
- archive, [533](#)
- calchash, [535](#)
- can_make_flat, [538](#)
- canonicalize, [540](#)
- combine_ex_with_coeff_to_pair, [536](#)
- combine_overall_coeff, [538](#)
- combine_pair_with_coeff_to_pair, [537](#)
- combine_same_terms_sorted_seq, [541](#)
- conjugate, [533](#)
- construct_from_2_ex, [539](#)
- construct_from_2_expairseq, [539](#)
- construct_from_epvector, [539](#), [540](#)
- construct_from_expairseq_ex, [539](#)
- construct_from_exvector, [539](#)
- default_overall_coeff, [537](#)
- do_print, [538](#)
- do_print_tree, [538](#)
- eval, [532](#)
- evalchildren, [541](#)
- expair_needs_further_processing, [537](#)
- expairseq, [530](#)
- expand, [535](#)
- expandchildren, [541](#)
- info, [531](#)
- is_canonical, [541](#)
- is_equal_same_type, [534](#)
- make_flat, [540](#)
- map, [532](#)
- match, [533](#)
- nops, [531](#)
- op, [531](#)
- overall_coeff, [542](#)
- precedence, [531](#)
- printpair, [536](#)
- printseq, [536](#)
- read_archive, [534](#)
- recombine_pair_to_ex, [537](#)
- return_type, [534](#)
- seq, [542](#)
- split_ex_to_pair, [536](#)
- subs, [533](#)
- subschilren, [542](#)
- thisexpairseq, [535](#)
- to_polynomial, [532](#)
- to_rational, [532](#)
- GiNaC::expand_map_function, [543](#)
- expand_map_function, [544](#)
- operator(), [544](#)
- options, [544](#)
- GiNaC::expand_options, [544](#)
- expand_function_args, [545](#)
- expand_indexed, [545](#)
- expand_rename_idx, [545](#)
- expand_transcendental, [545](#)
- GiNaC::factor_options, [545](#)
- all, [546](#)
- polynomial, [546](#)

- GiNaC::fail, 546
 - do_print, 549
 - return_type, 549
- GiNaC::fderivative, 550
 - archive, 559
 - derivative, 560
 - derivatives, 561
 - do_print, 561
 - do_print_csrc, 561
 - do_print_latex, 561
 - do_print_tree, 562
 - eval, 558
 - fderivative, 557, 558
 - is_equal_same_type, 560
 - match_same_type, 560
 - parameter_set, 562
 - print, 558
 - read_archive, 559
 - series, 558
 - thiscontainer, 559
- GiNaC::function, 562
 - archive, 576
 - calchash, 574
 - conjugate, 575
 - current_serial, 580
 - derivative, 577
 - eval, 574
 - eval_ncmul, 574
 - evalf, 574
 - expand, 573
 - expl_derivative, 578
 - find_function, 579
 - function, 569–573
 - get_name, 580
 - get_registered_functions, 580
 - get_serial, 580
 - imag_part, 576
 - info, 577
 - is_equal_same_type, 577
 - lookup_remember_table, 579
 - match_same_type, 577
 - pderivative, 578
 - power, 579
 - precedence, 573
 - print, 573
 - read_archive, 576
 - real_part, 576
 - register_new, 579
 - registered_functions, 579
 - remember_table_entry, 580
 - return_type, 578
 - return_type_tinfo, 578
 - serial, 580
 - series, 575
 - store_remember_table, 579
 - thiscontainer, 575
- GiNaC::function_options, 581
 - ~function_options, 586
- conjugate_f, 616
- conjugate_func, 591–593, 609
- conjugate_use_exvector_args, 619
- derivative_f, 617
- derivative_func, 599–601, 610
- derivative_use_exvector_args, 620
- do_not_evalf_params, 613
- dummy, 587
- eval_f, 616
- eval_func, 587–589, 609
- eval_use_exvector_args, 619
- evalf_f, 616
- evalf_func, 589–591, 609
- evalf_params_first, 618
- evalf_use_exvector_args, 619
- expand_f, 617
- expand_func, 597–599, 610
- expand_use_exvector_args, 620
- expl_derivative_f, 617
- expl_derivative_func, 601–603, 610
- expl_derivative_use_exvector_args, 620
- fderivative, 615
- function, 615
- function_options, 586
- functions_with_same_name, 621
- get_name, 614
- get_nparams, 614
- has_derivative, 614
- has_power, 614
- imag_part_f, 617
- imag_part_func, 595–597, 610
- imag_part_use_exvector_args, 620
- info_f, 618
- info_func, 607–609, 611
- info_use_exvector_args, 620
- initialize, 587
- latex_name, 587
- name, 616
- nparams, 616
- overloaded, 614
- power_f, 617
- power_func, 603–605, 610
- power_use_exvector_args, 620
- print_dispatch_table, 618
- print_func, 611–613
- print_use_exvector_args, 620
- real_part_f, 617
- real_part_func, 593–595, 610
- real_part_use_exvector_args, 619
- remember, 614
- remember_assoc_size, 619
- remember_size, 619
- remember_strategy, 619
- return_type, 618
- return_type_tinfo, 618
- series_f, 618
- series_func, 605–607, 610
- series_use_exvector_args, 620

- set_name, 587
- set_print_func, 615
- set_return_type, 613
- set_symmetry, 614
- symtree, 621
- test_and_set_nparams, 615
- TeX_name, 616
- use_remember, 619
- use_return_type, 618
- GiNaC::G2_SERIAL, 621
 - serial, 621
- GiNaC::G3_SERIAL, 622
 - serial, 622
- GiNaC::gcd_options, 622
 - no_heur_gcd, 623
 - no_part_factored, 623
 - use_sr_gcd, 623
- GiNaC::gcdheu_failed, 623
- GiNaC::Gt, 623
 - apply_function_recursive, 636
 - applyIntegrationPath, 632
 - archive, 631
 - args, 638
 - auto_clear_polylog_cache, 637
 - calchash, 630
 - clear_polylog_cache, 636
 - cutIntegrationPath_threshold_imag, 637
 - cutIntegrationPath_threshold_real, 637
 - decomposeIntegrationPath, 632
 - do_print, 636
 - do_print_latex, 636
 - enable_tauToFundamental, 638
 - eval, 629
 - evalf, 629
 - evaluate, 633
 - ex_cutIntegrationPath, 633, 635
 - ex_evaluate, 634, 635
 - ex_prepare, 634, 635
 - ex_qExpand, 634, 635
 - ex_regularise, 633, 635
 - ex_tauToFundamental, 633, 635
 - ex_zisToFundamental, 633, 634
 - findMoebiusTransform, 632
 - getArgs, 629
 - getTau, 628
 - getZ, 629
 - Gt, 628
 - has, 630
 - lst_evaluate, 634, 636
 - match, 630
 - nargs, 638
 - nops, 629
 - nqexand_transformer, 638
 - precedence, 629
 - qExpand, 632
 - qExpand_minOrder, 637
 - qExpand_stepSize, 637
 - read_archive, 631
 - regularise, 631
 - setArgs, 628
 - subs, 630
 - tau, 638
 - tauToFundamental, 632
 - to_integer, 637
 - to_numeric, 636
 - z, 638
 - zisToFundamental, 631
- GiNaC::Gt::kernel, 693
 - check, 694
 - deform, 694
 - Gt, 694
 - kernel, 693, 694
 - ni, 694
 - zi, 694
- GiNaC::Gt_detail, 273
 - deconcatenate_path, 273
 - deconcatenate_path< ex >, 273
 - deconcatenate_path< Gt::kernel >, 273
 - integer_partition, 273
 - operator<, 274
 - operator<<, 274
- GiNaC::Gt_detail::pathintegral_term, 846
 - denom, 849
 - evaluate, 848
 - G_path, 848
 - integrate, 848
 - operator<, 849
 - operator<<, 848
 - pathintegral_term, 847
 - pathintegral_term_list, 848
 - polylog, 849
 - power, 849
 - prefactor, 849
- GiNaC::Gt_detail::pathintegral_term_list, 849
 - add, 851
 - begin, 850
 - clear, 850
 - empty, 850
 - end, 850
 - pop, 851
 - size, 850
 - terms, 851
- GiNaC::Gt_detail::TransformExpressionWithCache< Type >, 1073
 - cache, 1075
 - check_type, 1074, 1075
 - clear_cache, 1074
 - func_default, 1075
 - func_obj, 1075
 - get_cache, 1074
 - impl, 1075
 - operator(), 1074
 - print_cache, 1074
 - TransformExpressionWithCache, 1074
- GiNaC::has_distance< T >, 639
 - no_type, 640

- test, 640
- value, 640
- yes_type, 640
- GiNaC::has_options, 641
 - algebraic, 641
- GiNaC::idx, 642
 - archive, 648
 - calchash, 649
 - derivative, 648
 - dim, 652
 - do_print, 651
 - do_print_csrc, 652
 - do_print_latex, 652
 - do_print_tree, 652
 - evalf, 647
 - get_dim, 650
 - get_value, 650
 - idx, 646
 - info, 646
 - is_dim_numeric, 650
 - is_dim_symbolic, 651
 - is_dummy_pair_same_type, 649
 - is_numeric, 650
 - is_symbolic, 650
 - map, 647
 - match_same_type, 649
 - minimal_dim, 651
 - nops, 647
 - op, 647
 - print_index, 651
 - read_archive, 648
 - replace_dim, 651
 - subs, 647
 - value, 652
- GiNaC::idx_is_equal_ignore_dim, 653
 - operator(), 653
- GiNaC::indexed, 653
 - all_index_values_are, 667
 - archive, 665
 - derivative, 666
 - do_print, 669
 - do_print_latex, 669
 - do_print_tree, 669
 - eval, 665
 - expand, 667
 - get_dummy_indices, 668
 - get_free_indices, 665
 - get_indices, 667
 - get_symmetry, 668
 - has_dummy_index_for, 668
 - imag_part, 665
 - indexed, 660–664
 - info, 664
 - precedence, 664
 - print_indexed, 669
 - printindices, 668
 - read_archive, 666
 - real_part, 665
 - reposition_dummy_indices, 670
 - return_type, 667
 - return_type_tinfo, 667
 - simplify_indexed, 670
 - simplify_indexed_product, 670
 - symtree, 671
 - thiscontainer, 666
 - validate, 669
- GiNaC::info_flags, 671
 - cinteger, 672
 - cinteger_polynomial, 672
 - crational, 672
 - crational_polynomial, 672
 - even, 672
 - expanded, 672
 - exprseq, 672
 - has_indices, 672
 - idx, 672
 - indefinite, 672
 - indexed, 672
 - integer, 672
 - integer_polynomial, 672
 - list, 672
 - negative, 672
 - negint, 672
 - nonnegative, 672
 - nonnegint, 672
 - numeric, 672
 - odd, 672
 - polynomial, 672
 - posint, 672
 - positive, 672
 - prime, 672
 - rational, 672
 - rational_function, 672
 - rational_polynomial, 672
 - real, 672
 - relation, 672
 - relation_equal, 672
 - relation_greater, 672
 - relation_greater_or_equal, 672
 - relation_less, 672
 - relation_less_or_equal, 672
 - relation_not_equal, 672
 - symbol, 672
- GiNaC::integral, 672
 - a, 682
 - archive, 679
 - b, 682
 - conjugate, 679
 - degree, 677
 - derivative, 680
 - do_print, 681
 - do_print_latex, 681
 - eval, 677
 - eval_integ, 679
 - eval_ncmul, 677
 - evalf, 677

- expand, 678
- f, 682
- get_free_indices, 678
- integral, 676
- ldegree, 677
- let_op, 678
- max_integration_level, 681
- nops, 678
- op, 678
- precedence, 677
- read_archive, 680
- relative_integration_error, 681
- return_type, 679
- return_type_tinfo, 679
- series, 680
- x, 681
- GiNaC::integration_kernel, 683
 - cache_step_size, 690
 - do_print, 690
 - get_cache_size, 689
 - get_numerical_value, 688
 - get_numerical_value_impl, 689
 - get_series_coeff, 689
 - has_trailing_zero, 687
 - is_numeric, 687
 - Laurent_series, 688
 - series, 687
 - series_coeff, 689
 - series_coeff_impl, 688
 - series_vec, 690
 - set_cache_step, 689
 - uses_Laurent_series, 688
- GiNaC::internal, 274
- GiNaC::internal::_iter_rep, 277
 - _iter_rep, 277
 - e, 278
 - i, 278
 - i_end, 278
 - operator!=, 277
 - operator==, 277
- GiNaC::is_not_a_clifford, 690
 - operator(), 691
- GiNaC::is_summation_idx, 691
 - operator(), 691
- GiNaC::iterated_integral2_SERIAL, 691
 - serial, 692
- GiNaC::iterated_integral3_SERIAL, 692
 - serial, 693
- GiNaC::Kronecker_dtau_kernel, 695
 - C_norm, 702
 - do_print, 701
 - get_numerical_value, 700
 - is_numeric, 700
 - K, 701
 - Kronecker_dtau_kernel, 700
 - let_op, 700
 - n, 701
 - nops, 700
 - op, 700
 - series_coeff_impl, 701
 - z, 701
- GiNaC::Kronecker_dz_kernel, 702
 - C_norm, 709
 - do_print, 708
 - get_numerical_value, 707
 - is_numeric, 707
 - K, 709
 - Kronecker_dz_kernel, 707
 - let_op, 707
 - n, 708
 - nops, 707
 - op, 707
 - series_coeff_impl, 708
 - tau, 708
 - z_j, 708
- GiNaC::lanczos_coeffs, 709
 - calc_lanczos_A, 710
 - coeffs, 710
 - current_vector, 710
 - get_order, 710
 - lanczos_coeffs, 709
 - sufficiently_accurate, 710
- GiNaC::library_init, 711
 - ~library_init, 712
 - count, 713
 - init_unarchivers, 713
 - library_init, 712
- GiNaC::make_flat_inserter, 713
 - combine_indices, 714
 - do_renaming, 714
 - handle_factor, 714
 - make_flat_inserter, 714
 - used_indices, 714
- GiNaC::map_function, 715
 - ~map_function, 716
 - argument_type, 716
 - operator(), 716
 - result_type, 716
- GiNaC::matrix, 716
 - add, 727
 - add_indexed, 724
 - archive, 725
 - charpoly, 731
 - col, 737
 - cols, 726
 - conjugate, 725
 - contract_with, 724
 - determinant, 729
 - determinant_minor, 733
 - division_free_elimination, 734
 - do_print, 736
 - do_print_latex, 736
 - do_print_python_repr, 736
 - echelon_form, 733
 - eval_indexed, 724
 - evalm, 723

- fraction_free_elimination, 735
- gauss_elimination, 734
- imag_part, 725
- inverse, 731
- is_zero_matrix, 733
- let_op, 723
- m, 737
- markowitz_elimination, 735
- match_same_type, 726
- matrix, 721, 722
- mul, 727, 728
- mul_scalar, 728
- nops, 723
- op, 723
- operator(), 728, 729
- pivot, 735
- pow, 728
- print_elements, 736
- rank, 732, 733
- read_archive, 725
- real_part, 725
- return_type, 726
- row, 737
- rows, 726
- scalar_mul_indexed, 724
- set, 729
- solve, 732
- sub, 727
- subs, 723
- trace, 730
- transpose, 729
- GiNaC::minkmetric, 738
 - archive, 742
 - do_print, 743
 - do_print_latex, 743
 - eval_indexed, 742
 - info, 742
 - minkmetric, 742
 - pos_sig, 744
 - read_archive, 743
 - return_type, 743
- GiNaC::modular_form_kernel, 744
 - C_norm, 751
 - do_print, 751
 - get_numerical_value, 750
 - is_numeric, 749
 - k, 751
 - Laurent_series, 750
 - let_op, 749
 - modular_form_kernel, 748
 - nops, 749
 - op, 749
 - P, 751
 - q_expansion_modular_form, 750
 - series, 749
 - uses_Laurent_series, 750
- GiNaC::mul, 753
 - add, 770
 - algebraic_subs_mul, 768
 - can_be_further_expanded, 769
 - can_make_flat, 767
 - coeff, 760
 - combine_ex_with_coeff_to_pair, 766
 - combine_overall_coeff, 767
 - combine_pair_with_coeff_to_pair, 766
 - conjugate, 764
 - default_overall_coeff, 767
 - degree, 760
 - derivative, 764
 - do_print, 768
 - do_print_csrc, 769
 - do_print_latex, 769
 - do_print_python_repr, 769
 - eval, 761
 - eval_ncmul, 764
 - evalf, 761
 - evalm, 762
 - expair_needs_further_processing, 766
 - expand, 767
 - expandchildren, 769
 - find_real_imag, 768
 - get_free_indices, 764
 - has, 761
 - imag_part, 762
 - info, 759
 - integer_content, 763
 - is_polynomial, 760
 - ldegree, 760
 - max_coefficient, 763
 - mul, 758, 759
 - ncmul, 770
 - normal, 762
 - power, 770
 - precedence, 759
 - print_overall_coeff, 768
 - real_part, 762
 - recombine_pair_to_ex, 766
 - return_type, 765
 - return_type_tinfo, 765
 - series, 762
 - smod, 763
 - split_ex_to_pair, 765
 - thisexpairseq, 765
- GiNaC::multi_iterator_counter< T >, 771
 - init, 773
 - multi_iterator_counter, 772
 - operator<<, 773
 - operator++, 773
- GiNaC::multi_iterator_counter_indv< T >, 774
 - init, 776
 - multi_iterator_counter_indv, 775
 - Nv, 777
 - operator<<, 776
 - operator++, 776
- GiNaC::multi_iterator_ordered< T >, 777
 - init, 779

- multi_iterator_ordered, 779
- operator<<, 780
- operator++, 779
- GiNaC::multi_iterator_ordered_eq< T >, 780
 - init, 782
 - multi_iterator_ordered_eq, 782
 - operator<<, 783
 - operator++, 782
- GiNaC::multi_iterator_ordered_eq_indv< T >, 783
 - init, 785
 - multi_iterator_ordered_eq_indv, 785
 - Nv, 786
 - operator<<, 786
 - operator++, 785
- GiNaC::multi_iterator_permutation< T >, 786
 - get_sign, 789
 - init, 789
 - multi_iterator_permutation, 788
 - operator<<, 790
 - operator++, 789
- GiNaC::multi_iterator_shuffle< T >, 790
 - init, 792
 - multi_iterator_shuffle, 792
 - N_internal, 793
 - operator<<, 793
 - operator++, 792
 - v_internal, 793
 - v_orig, 793
- GiNaC::multi_iterator_shuffle_prime< T >, 793
 - init, 796
 - multi_iterator_shuffle_prime, 795
 - operator<<, 796
- GiNaC::multiple_polylog_kernel, 796
 - do_print, 802
 - is_numeric, 801
 - let_op, 801
 - multiple_polylog_kernel, 801
 - nops, 801
 - op, 801
 - series_coeff_impl, 802
 - z, 802
- GiNaC::ncmul, 803
 - append_factors, 814
 - coeff, 810
 - conjugate, 812
 - count_factors, 813
 - degree, 810
 - derivative, 812
 - do_print, 813
 - do_print_csrc, 813
 - eval, 811
 - evalm, 811
 - expand, 810
 - expandchildren, 814
 - get_factors, 814
 - get_free_indices, 811
 - hold_ncmul, 814
 - imag_part, 812
 - info, 809
 - ldegree, 810
 - ncmul, 808, 809
 - power, 814
 - precedence, 809
 - real_part, 812
 - reeval_ncmul, 814
 - return_type, 813
 - return_type_tinfo, 813
 - thiscontainer, 812
- GiNaC::normal_map_function, 815
 - operator(), 816
- GiNaC::numeric, 816
 - add, 829
 - add_dyn, 831
 - archive, 828
 - calchash, 829
 - coeff, 824
 - compare, 833
 - conjugate, 827
 - csgn, 833
 - degree, 824
 - denom, 840
 - derivative, 828
 - div, 830
 - div_dyn, 831
 - do_print, 841
 - do_print_csrc, 841
 - do_print_csrc_cl_N, 841
 - do_print_latex, 841
 - do_print_python_repr, 842
 - do_print_tree, 841
 - eval, 825
 - evalf, 825
 - has, 825
 - imag, 839
 - imag_part, 828
 - info, 823
 - int_length, 840
 - integer_content, 826
 - inverse, 833
 - is_cinteger, 836
 - is_crational, 837
 - is_equal, 834
 - is_equal_same_type, 829
 - is_even, 835
 - is_integer, 835
 - is_negative, 834
 - is_nonneg_integer, 835
 - is_odd, 835
 - is_polynomial, 824
 - is_pos_integer, 835
 - is_positive, 834
 - is_prime, 836
 - is_rational, 836
 - is_real, 836
 - is_zero, 834
 - ldegree, 824

- max_coefficient, [827](#)
- mul, [830](#)
- mul_dyn, [831](#)
- normal, [826](#)
- numer, [840](#)
- numeric, [822](#), [823](#)
- operator!=, [837](#)
- operator<, [837](#)
- operator<=, [837](#)
- operator>, [838](#)
- operator>=, [838](#)
- operator=, [832](#), [833](#)
- operator==, [837](#)
- power, [830](#)
- power_dyn, [832](#)
- precedence, [823](#)
- print_numeric, [840](#)
- read_archive, [828](#)
- real, [839](#)
- real_part, [828](#)
- smod, [827](#)
- step, [833](#)
- sub, [829](#)
- sub_dyn, [831](#)
- subs, [825](#)
- to_cl_N, [839](#)
- to_double, [839](#)
- to_int, [838](#)
- to_long, [838](#)
- to_polynomial, [826](#)
- to_rational, [826](#)
- value, [842](#)
- GiNaC::op0_is_equal, [842](#)
- operator(), [843](#)
- GiNaC::partition_generator, [843](#)
 - current_updated, [844](#)
 - get, [844](#)
 - next, [844](#)
 - partition, [844](#)
 - partition_generator, [844](#)
- GiNaC::partition_with_zero_parts_generator, [845](#)
 - current_updated, [846](#)
 - get, [846](#)
 - m, [846](#)
 - next, [846](#)
 - partition, [846](#)
 - partition_with_zero_parts_generator, [845](#)
- GiNaC::pointer_to_map_function, [852](#)
 - operator(), [852](#)
 - pointer_to_map_function, [852](#)
 - ptr, [853](#)
- GiNaC::pointer_to_map_function_1arg< T1 >, [853](#)
 - arg1, [854](#)
 - operator(), [854](#)
 - pointer_to_map_function_1arg, [854](#)
 - ptr, [854](#)
- GiNaC::pointer_to_map_function_2args< T1, T2 >, [854](#)
 - arg1, [856](#)
 - arg2, [856](#)
 - operator(), [855](#)
 - pointer_to_map_function_2args, [855](#)
 - ptr, [856](#)
- GiNaC::pointer_to_map_function_3args< T1, T2, T3 >, [856](#)
 - arg1, [857](#)
 - arg2, [858](#)
 - arg3, [858](#)
 - operator(), [857](#)
 - pointer_to_map_function_3args, [857](#)
 - ptr, [857](#)
- GiNaC::pointer_to_member_to_map_function< C >, [858](#)
 - c, [859](#)
 - operator(), [859](#)
 - pointer_to_member_to_map_function, [859](#)
 - ptr, [859](#)
- GiNaC::pointer_to_member_to_map_function_1arg< C, T1 >, [860](#)
 - arg1, [861](#)
 - c, [861](#)
 - operator(), [861](#)
 - pointer_to_member_to_map_function_1arg, [860](#)
 - ptr, [861](#)
- GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 >, [861](#)
 - arg1, [863](#)
 - arg2, [863](#)
 - c, [863](#)
 - operator(), [862](#)
 - pointer_to_member_to_map_function_2args, [862](#)
 - ptr, [863](#)
- GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >, [863](#)
 - arg1, [865](#)
 - arg2, [865](#)
 - arg3, [865](#)
 - c, [865](#)
 - operator(), [864](#)
 - pointer_to_member_to_map_function_3args, [864](#)
 - ptr, [865](#)
- GiNaC::pole_error, [865](#)
 - deg, [866](#)
 - degree, [866](#)
 - pole_error, [866](#)
- GiNaC::possymbol, [867](#)
 - duplicate, [871](#)
 - get_domain, [871](#)
 - possymbol, [871](#)
- GiNaC::power, [872](#)
 - archive, [882](#)
 - basis, [887](#)
 - coeff, [878](#)
 - conjugate, [882](#)
 - degree, [878](#)
 - derivative, [883](#)

- do_print_csrc, 884
- do_print_csrc_cl_N, 885
- do_print_dflt, 884
- do_print_latex, 884
- do_print_python, 885
- do_print_python_repr, 885
- eval, 878
- eval_ncmul, 883
- evalf, 879
- evalm, 879
- expand, 883
- expand_add, 885
- expand_add_2, 885
- expand_mul, 886
- exponent, 887
- has, 880
- imag_part, 882
- info, 876
- is_polynomial, 877
- ldegree, 878
- map, 877
- mul, 886
- nops, 877
- normal, 881
- op, 877
- power, 876
- precedence, 876
- print_power, 884
- read_archive, 882
- real_part, 882
- return_type, 883
- return_type_tinfo, 883
- series, 880
- subs, 880
- to_polynomial, 881
- to_rational, 881
- GiNaC::print_context, 887
 - ~print_context, 888
 - options, 888
 - print_context, 888
 - s, 888
- GiNaC::print_context_options, 889
 - get_id, 890
 - get_name, 889
 - get_parent_name, 889
 - id, 890
 - name, 890
 - parent_name, 890
 - print_context_options, 889
- GiNaC::print_csrc, 891
 - print_csrc, 891
- GiNaC::print_csrc_cl_N, 892
 - print_csrc_cl_N, 893
- GiNaC::print_csrc_double, 893
 - print_csrc_double, 894
- GiNaC::print_csrc_float, 894
 - print_csrc_float, 895
- GiNaC::print_dflt, 895
 - print_dflt, 896
- GiNaC::print_funcor, 896
 - impl, 898
 - is_valid, 898
 - operator(), 897
 - operator=, 897
 - print_funcor, 897
- GiNaC::print_funcor_impl, 898
 - ~print_funcor_impl, 899
 - duplicate, 899
 - operator(), 899
- GiNaC::print_latex, 899
 - print_latex, 900
- GiNaC::print_memfun_handler< T, C >, 900
 - duplicate, 901
 - F, 901
 - f, 902
 - operator(), 901
 - print_memfun_handler, 901
- GiNaC::print_options, 902
 - print_index_dimensions, 903
- GiNaC::print_ptrfun_handler< T, C >, 903
 - duplicate, 904
 - F, 904
 - f, 904
 - operator(), 904
 - print_ptrfun_handler, 904
- GiNaC::print_python, 905
 - print_python, 905
- GiNaC::print_python_repr, 906
 - print_python_repr, 906
- GiNaC::print_tree, 907
 - delta_indent, 908
 - print_tree, 907
- GiNaC::pseries, 911
 - add_series, 922
 - archive, 920
 - coeff, 917
 - coeffop, 922
 - collect, 917
 - conjugate, 919
 - convert_to_poly, 921
 - degree, 916
 - derivative, 920
 - do_print, 924
 - do_print_latex, 924
 - do_print_python, 925
 - do_print_python_repr, 925
 - do_print_tree, 925
 - eval, 918
 - eval_integ, 920
 - evalf, 918
 - evalm, 920
 - expand, 919
 - exponop, 922
 - get_point, 921
 - get_var, 921
 - imag_part, 919

- is_compatible_to, [921](#)
- is_terminating, [922](#)
- is_zero, [922](#)
- ldegree, [917](#)
- mul_const, [923](#)
- mul_series, [923](#)
- nops, [916](#)
- normal, [918](#)
- op, [916](#)
- point, [926](#)
- power_const, [924](#)
- precedence, [916](#)
- print_series, [924](#)
- pseries, [915](#), [916](#)
- read_archive, [920](#)
- real_part, [919](#)
- seq, [925](#)
- series, [918](#)
- shift_exponents, [924](#)
- subs, [918](#)
- var, [925](#)
- GiNaC::psi1_SERIAL, [926](#)
 - serial, [927](#)
- GiNaC::psi2_SERIAL, [927](#)
 - serial, [927](#)
- GiNaC::ptr< T >, [928](#)
 - ~ptr, [929](#)
 - get_pointer, [931](#)
 - makewritable, [930](#)
 - operator!=, [930](#), [931](#)
 - operator<<, [931](#)
 - operator->, [930](#)
 - operator=, [929](#)
 - operator==, [930](#), [931](#)
 - operator*, [929](#)
 - p, [932](#)
 - ptr, [929](#)
 - std::less< ptr< T > >, [931](#)
 - swap, [930](#)
- GiNaC::realsymbol, [932](#)
 - conjugate, [936](#)
 - duplicate, [937](#)
 - get_domain, [936](#)
 - imag_part, [937](#)
 - real_part, [937](#)
 - realsymbol, [936](#)
- GiNaC::refcounted, [938](#)
 - add_reference, [939](#)
 - get_refcount, [939](#)
 - refcount, [939](#)
 - refcounted, [939](#)
 - remove_reference, [939](#)
 - set_refcount, [939](#)
- GiNaC::registered_class_options, [940](#)
 - get_id, [941](#)
 - get_name, [941](#)
 - get_parent_name, [941](#)
 - get_print_dispatch_table, [941](#)
 - name, [942](#)
 - parent_name, [942](#)
 - print_dispatch_table, [942](#)
 - print_func, [941](#)
 - registered_class_options, [940](#)
 - set_print_func, [942](#)
 - tinfo_key, [942](#)
- GiNaC::relational, [943](#)
 - archive, [949](#)
 - calchash, [951](#)
 - canonical, [950](#)
 - do_print, [951](#)
 - do_print_python_repr, [951](#)
 - equal, [947](#)
 - eval_ncmul, [950](#)
 - greater, [947](#)
 - greater_or_equal, [947](#)
 - info, [948](#)
 - less, [947](#)
 - less_or_equal, [947](#)
 - lh, [953](#)
 - lhs, [951](#)
 - make_safe_bool, [952](#)
 - map, [949](#)
 - match_same_type, [950](#)
 - nops, [948](#)
 - not_equal, [947](#)
 - o, [953](#)
 - op, [948](#)
 - operator safe_bool, [952](#)
 - operator!, [952](#)
 - operators, [947](#)
 - precedence, [948](#)
 - read_archive, [949](#)
 - relational, [948](#)
 - return_type, [950](#)
 - return_type_tinfo, [951](#)
 - rh, [953](#)
 - rhs, [952](#)
 - safe_bool, [947](#)
 - subs, [949](#)
- GiNaC::relational::safe_bool_helper, [963](#)
 - nonnull, [963](#)
- GiNaC::remember_strategies, [953](#)
 - delete_cyclic, [954](#)
 - delete_lfu, [954](#)
 - delete_lru, [954](#)
 - delete_never, [954](#)
- GiNaC::remember_table, [954](#)
 - add_entry, [955](#)
 - clear_all_entries, [955](#)
 - init_table, [956](#)
 - lookup_entry, [955](#)
 - max_assoc_size, [956](#)
 - remember_strategy, [956](#)
 - remember_table, [955](#)
 - remember_tables, [956](#)
 - show_statistics, [956](#)

- table_size, 956
- GiNaC::remember_table_entry, 957
 - access_counter, 959
 - get_last_access, 958
 - get_result, 958
 - get_successful_hits, 958
 - hashvalue, 958
 - is_equal, 958
 - last_access, 959
 - remember_table_entry, 957
 - result, 958
 - seq, 958
 - successful_hits, 959
- GiNaC::remember_table_list, 959
 - add_entry, 960
 - lookup_entry, 960
 - max_assoc_size, 960
 - remember_strategy, 960
 - remember_table_list, 960
- GiNaC::return_type_t, 961
 - operator!=, 962
 - operator<, 961
 - operator==, 961
 - rl, 962
 - tinfo, 962
- GiNaC::return_types, 962
 - commutative, 963
 - noncommutative, 963
 - noncommutative_composite, 963
- GiNaC::scalar_products, 963
 - add, 964
 - add_vectors, 964
 - clear, 964
 - debugprint, 965
 - evaluate, 965
 - is_defined, 965
 - spm, 965
- GiNaC::series_options, 966
 - suppress_branchcut, 966
- GiNaC::solve_algo, 966
 - automatic, 967
 - bareiss, 967
 - divfree, 967
 - gauss, 967
 - markowitz, 967
- GiNaC::spinidx, 967
 - archive, 974
 - conjugate, 973
 - do_print, 976
 - do_print_latex, 976
 - do_print_tree, 976
 - dotted, 976
 - is_dotted, 975
 - is_dummy_pair_same_type, 973
 - is_undotted, 975
 - match_same_type, 974
 - read_archive, 974
 - spinidx, 973
 - toggle_dot, 975
 - toggle_variance_dot, 975
- GiNaC::spinmetric, 977
 - contract_with, 981
 - do_print, 982
 - do_print_latex, 982
 - eval_indexed, 981
 - info, 981
- GiNaC::spmapkey, 982
 - debugprint, 983
 - dim, 984
 - operator<, 983
 - operator==, 983
 - spmapkey, 983
 - v1, 983
 - v2, 983
- GiNaC::status_flags, 984
 - dynamallocated, 985
 - evaluated, 985
 - expanded, 985
 - has_indices, 985
 - has_no_indices, 985
 - hash_calculated, 985
 - is_negative, 985
 - is_positive, 985
 - not_shareable, 985
 - purely_indefinite, 985
- GiNaC::structure< T, ComparisonPolicy >, 985
 - add_indexed, 998
 - calchash, 1001
 - coeff, 994
 - collect, 994
 - contract_with, 999
 - degree, 994
 - derivative, 996
 - eval, 989
 - eval_indexed, 990
 - eval_ncmul, 990
 - evalm, 989
 - expand, 994
 - get_class_name, 989
 - get_free_indices, 998
 - get_struct, 1001
 - has, 992
 - info, 991
 - integer_content, 997
 - is_equal_same_type, 1000
 - ldegree, 994
 - let_op, 992
 - map, 993
 - match, 992
 - match_same_type, 993
 - max_coefficient, 998
 - nops, 991
 - normal, 996
 - obj, 1002
 - op, 991
 - operator->, 1001

- operator[], [991](#), [992](#)
- precedence, [990](#)
- print, [990](#)
- return_type, [1000](#)
- return_type_tinfo, [1000](#)
- scalar_mul_indexed, [999](#)
- series, [996](#)
- smod, [997](#)
- structure, [989](#)
- subs, [993](#)
- to_polynomial, [997](#)
- to_rational, [997](#)
- GiNaC::su3d, [1002](#)
 - contract_with, [1006](#)
 - do_print, [1006](#)
 - do_print_latex, [1006](#)
 - eval_indexed, [1006](#)
 - return_type, [1006](#)
- GiNaC::su3f, [1007](#)
 - contract_with, [1011](#)
 - do_print, [1011](#)
 - do_print_latex, [1011](#)
 - eval_indexed, [1011](#)
 - return_type, [1011](#)
- GiNaC::su3one, [1012](#)
 - do_print, [1016](#)
 - do_print_latex, [1016](#)
- GiNaC::su3t, [1016](#)
 - contract_with, [1020](#)
 - do_print, [1020](#)
 - do_print_latex, [1020](#)
- GiNaC::subs_options, [1021](#)
 - algebraic, [1021](#)
 - no_index_renaming, [1021](#)
 - no_pattern, [1021](#)
 - pattern_is_not_product, [1021](#)
 - pattern_is_product, [1021](#)
 - really_subs_idx, [1021](#)
 - subs_algebraic, [1021](#)
 - subs_no_pattern, [1021](#)
- GiNaC::sy_is_less, [1022](#)
 - operator(), [1022](#)
 - sy_is_less, [1022](#)
 - v, [1022](#)
- GiNaC::sy_swap, [1023](#)
 - operator(), [1023](#)
 - swapped, [1023](#)
 - sy_swap, [1023](#)
 - v, [1023](#)
- GiNaC::sym_desc, [1024](#)
 - deg_a, [1025](#)
 - deg_b, [1025](#)
 - ldeg_a, [1025](#)
 - ldeg_b, [1025](#)
 - max_deg, [1025](#)
 - max_lcnops, [1026](#)
 - operator<, [1025](#)
 - sym, [1025](#)
 - sym_desc, [1024](#)
- GiNaC::symbol, [1026](#)
 - archive, [1033](#)
 - calchash, [1034](#)
 - conjugate, [1032](#)
 - derivative, [1034](#)
 - do_print, [1036](#)
 - do_print_latex, [1036](#)
 - do_print_python_repr, [1036](#)
 - do_print_tree, [1036](#)
 - eval, [1031](#)
 - evalf, [1031](#)
 - get_domain, [1035](#)
 - get_name, [1035](#)
 - get_TeX_name, [1035](#)
 - imag_part, [1033](#)
 - info, [1031](#)
 - is_equal_same_type, [1034](#)
 - is_polynomial, [1033](#)
 - name, [1036](#)
 - next_serial, [1037](#)
 - normal, [1032](#)
 - read_archive, [1033](#)
 - real_part, [1033](#)
 - serial, [1036](#)
 - series, [1031](#)
 - set_name, [1035](#)
 - set_TeX_name, [1035](#)
 - subs, [1031](#)
 - symbol, [1030](#)
 - TeX_name, [1037](#)
 - to_polynomial, [1032](#)
 - to_rational, [1032](#)
- GiNaC::symbolset, [1037](#)
 - has, [1038](#)
 - insert_symbols, [1038](#)
 - s, [1038](#)
 - symbolset, [1038](#)
- GiNaC::symmetry, [1039](#)
 - add, [1045](#)
 - antisymmetric, [1043](#)
 - archive, [1044](#)
 - calchash, [1044](#)
 - canonicalize, [1046](#)
 - children, [1047](#)
 - cyclic, [1043](#)
 - do_print, [1046](#)
 - do_print_tree, [1046](#)
 - get_type, [1044](#)
 - has_cyclic, [1046](#)
 - has_nonsymmetric, [1045](#)
 - has_symmetry, [1045](#)
 - indices, [1047](#)
 - none, [1043](#)
 - read_archive, [1044](#)
 - set_type, [1045](#)
 - sy_is_less, [1046](#)
 - sy_swap, [1046](#)

- symmetric, [1043](#)
- symmetry, [1043](#)
- symmetry_type, [1043](#)
- type, [1047](#)
- validate, [1045](#)
- GiNaC::symminfo, [1047](#)
 - coeff, [1048](#)
 - num, [1049](#)
 - orig, [1049](#)
 - symminfo, [1048](#)
 - symmterm, [1048](#)
- GiNaC::symminfo_is_less_by_orig, [1049](#)
 - operator(), [1049](#)
- GiNaC::symminfo_is_less_by_symmterm, [1050](#)
 - operator(), [1050](#)
- GiNaC::tensdelta, [1050](#)
 - contract_with, [1054](#)
 - do_print, [1055](#)
 - do_print_latex, [1055](#)
 - eval_indexed, [1054](#)
 - info, [1054](#)
 - return_type, [1055](#)
- GiNaC::tensepsilon, [1055](#)
 - archive, [1060](#)
 - contract_with, [1060](#)
 - do_print, [1061](#)
 - do_print_latex, [1061](#)
 - eval_indexed, [1060](#)
 - info, [1060](#)
 - minkowski, [1061](#)
 - pos_sig, [1061](#)
 - read_archive, [1060](#)
 - return_type, [1061](#)
 - tensepsilon, [1059](#)
- GiNaC::tensmetric, [1062](#)
 - contract_with, [1066](#)
 - do_print, [1067](#)
 - eval_indexed, [1066](#)
 - info, [1066](#)
 - return_type, [1066](#)
- GiNaC::tensor, [1067](#)
 - replace_contr_index, [1071](#)
 - return_type, [1071](#)
- GiNaC::terminfo, [1071](#)
 - orig, [1072](#)
 - symm, [1072](#)
 - terminfo, [1072](#)
- GiNaC::terminfo_is_less, [1073](#)
 - operator(), [1073](#)
- GiNaC::unarchive_table_t, [1077](#)
 - ~unarchive_table_t, [1077](#)
 - find, [1077](#)
 - insert, [1077](#)
 - unarch_map, [1078](#)
 - unarchive_table_t, [1077](#)
 - usecount, [1078](#)
- GiNaC::user_defined_kernel, [1078](#)
 - do_print, [1084](#)
- f, [1084](#)
- is_numeric, [1083](#)
- Laurent_series, [1083](#)
- let_op, [1083](#)
- nops, [1083](#)
- op, [1083](#)
- user_defined_kernel, [1083](#)
- uses_Laurent_series, [1084](#)
- x, [1084](#)
- GiNaC::varidx, [1085](#)
 - archive, [1090](#)
 - covariant, [1092](#)
 - do_print, [1091](#)
 - do_print_tree, [1092](#)
 - is_contravariant, [1091](#)
 - is_covariant, [1091](#)
 - is_dummy_pair_same_type, [1090](#)
 - match_same_type, [1090](#)
 - read_archive, [1090](#)
 - toggle_variance, [1091](#)
 - varidx, [1089](#)
- GiNaC::visitor, [1092](#)
 - ~visitor, [1093](#)
- GiNaC::wildcard, [1093](#)
 - archive, [1097](#)
 - calchash, [1098](#)
 - do_print, [1098](#)
 - do_print_python_repr, [1098](#)
 - do_print_tree, [1098](#)
 - get_label, [1098](#)
 - label, [1099](#)
 - match, [1097](#)
 - read_archive, [1097](#)
 - wildcard, [1097](#)
- GiNaC::zeta1_SERIAL, [1099](#)
 - serial, [1100](#)
- GiNaC::zeta2_SERIAL, [1100](#)
 - serial, [1100](#)
- GINAC_ASSERT
 - assertion.h, [1106](#)
- GINAC_BIND_UNARCHIVER
 - archive.h, [1105](#)
 - GiNaC, [82](#), [85](#), [86](#), [96](#), [101](#), [115–117](#), [120](#), [175](#), [177–180](#), [182](#), [187](#), [188](#), [203](#), [234](#), [235](#), [237](#), [238](#), [245](#), [255](#), [257](#)
- GINAC_DECLARE_PRINT_CONTEXT
 - print.h, [1201](#)
- GINAC_DECLARE_PRINT_CONTEXT_BASE
 - print.h, [1201](#)
- GINAC_DECLARE_PRINT_CONTEXT_COMMON
 - print.h, [1201](#)
- GINAC_DECLARE_REGISTERED_CLASS
 - registrar.h, [1206](#)
- GINAC_DECLARE_REGISTERED_CLASS_COMMON
 - registrar.h, [1206](#)
- GINAC_DECLARE_REGISTERED_CLASS_NO_CTORS
 - registrar.h, [1206](#)
- GINAC_DECLARE_UNARCHIVER

- archive.h, [1104](#)
- GiNaC, [82](#), [94](#), [95](#), [100](#), [101](#), [115](#), [116](#), [118](#), [119](#), [124](#), [175](#), [180](#), [181](#), [183](#), [187](#), [188](#), [219](#), [234](#), [235](#), [237](#), [238](#), [241](#), [249](#), [255](#)
- GINAC_IMPLEMENT_PRINT_CONTEXT
 - print.h, [1202](#)
- GINAC_IMPLEMENT_REGISTERED_CLASS
 - registrar.h, [1207](#)
- GINAC_IMPLEMENT_REGISTERED_CLASS_OPT
 - GiNaC, [82](#), [84](#), [85](#), [96](#), [101](#), [114–116](#), [120](#), [174](#), [177–181](#), [186](#), [187](#), [202](#), [233](#), [235–238](#), [245](#), [254](#)
 - registrar.h, [1207](#)
- GINAC_IMPLEMENT_REGISTERED_CLASS_OPT_T
 - registrar.h, [1207](#)
- GINAC_LT_AGE
 - version.h, [1224](#)
- GINAC_LT_CURRENT
 - version.h, [1224](#)
- GINAC_LT_REVISION
 - version.h, [1224](#)
- GINACLIB_ARCHIVE_AGE
 - version.h, [1224](#)
- GINACLIB_ARCHIVE_VERSION
 - version.h, [1224](#)
- GINACLIB_MAJOR_VERSION
 - version.h, [1224](#)
- GINACLIB_MICRO_VERSION
 - version.h, [1224](#)
- GINACLIB_MINOR_VERSION
 - version.h, [1224](#)
- GINACLIB_STR
 - version.h, [1224](#)
- GINACLIB_STR_HELPER
 - version.h, [1224](#)
- GINACLIB_VERSION
 - version.h, [1224](#)
- golden_ratio_hash
 - GiNaC, [250](#)
- greater
 - GiNaC::relational, [947](#)
- greater_or_equal
 - GiNaC::relational, [947](#)
- Gt
 - GiNaC::Gt, [628](#)
 - GiNaC::Gt::kernel, [694](#)
- Gt.cpp, [1152](#)
 - DEBUG_GT, [1152](#)
- Gt.h, [1153](#)
- Gt_helpers.cpp, [1153](#)
- Gt_helpers.h, [1154](#)
- guess_precision
 - GiNaC, [211](#)
- H_deriv
 - GiNaC, [155](#)
- H_eval
 - GiNaC, [154](#)
- H_evalf
 - GiNaC, [154](#)
- H_print_latex
 - GiNaC, [155](#)
- H_series
 - GiNaC, [154](#)
- handle_factor
 - GiNaC::make_flat_inserter, [714](#)
- has
 - GiNaC, [103](#)
 - GiNaC::basic, [322](#)
 - GiNaC::ex, [499](#)
 - GiNaC::Gt, [630](#)
 - GiNaC::mul, [761](#)
 - GiNaC::numeric, [825](#)
 - GiNaC::power, [880](#)
 - GiNaC::structure< T, ComparisonPolicy >, [992](#)
 - GiNaC::symbolset, [1038](#)
- has_cyclic
 - GiNaC::symmetry, [1046](#)
- has_derivative
 - GiNaC::function_options, [614](#)
- has_dummy_index_for
 - GiNaC::indexed, [668](#)
- has_ex
 - GiNaC::archive_node, [309](#)
- has_expression
 - GiNaC::archive_node, [311](#)
- has_indices
 - GiNaC::info_flags, [672](#)
 - GiNaC::status_flags, [985](#)
- has_no_indices
 - GiNaC::status_flags, [985](#)
- has_nonsymmetric
 - GiNaC::symmetry, [1045](#)
- has_power
 - GiNaC::function_options, [614](#)
- has_same_ex_as
 - GiNaC::archive_node, [309](#)
- has_symmetry
 - GiNaC::symmetry, [1045](#)
- has_trailing_zero
 - GiNaC::integration_kernel, [687](#)
- hash_calculated
 - GiNaC::status_flags, [985](#)
- hash_map.h, [1155](#)
- hash_seed.h, [1155](#)
- hashvalue
 - GiNaC::basic, [337](#)
 - GiNaC::remember_table_entry, [958](#)
- hasindex
 - GiNaC, [122](#)
- haswild
 - GiNaC, [255](#)
- head
 - GiNaC::composition_generator::coolmulti, [424](#)
- heur_gcd
 - GiNaC, [196](#)
- heur_gcd_z

- GiNaC, [195](#)
- hold
 - GiNaC::basic, [334](#)
- hold_ncmul
 - GiNaC, [187](#)
 - GiNaC::ncmul, [814](#)
- I
 - GiNaC, [257](#)
- i
 - GiNaC::composition_generator::coolmulti, [424](#)
 - GiNaC::const_iterator, [389](#)
 - GiNaC::internal::_iter_rep, [278](#)
- i_end
 - GiNaC::internal::_iter_rep, [278](#)
- id
 - GiNaC::print_context_options, [890](#)
- identify_parents
 - GiNaC::class_info< OPT >, [350](#)
- idx
 - GiNaC, [257](#)
 - GiNaC::idx, [646](#)
 - GiNaC::info_flags, [672](#)
- idx.cpp, [1156](#)
- idx.h, [1157](#)
- idx_symmetrization
 - GiNaC, [121](#)
- ifactor
 - GiNaC, [175](#)
- imag
 - GiNaC, [223](#)
 - GiNaC::numeric, [839](#)
- imag_part
 - GiNaC, [103](#)
 - GiNaC::add, [291](#)
 - GiNaC::basic, [330](#)
 - GiNaC::constant, [401](#)
 - GiNaC::container< C >, [415](#)
 - GiNaC::ex, [499](#)
 - GiNaC::function, [576](#)
 - GiNaC::indexed, [665](#)
 - GiNaC::matrix, [725](#)
 - GiNaC::mul, [762](#)
 - GiNaC::ncmul, [812](#)
 - GiNaC::numeric, [828](#)
 - GiNaC::power, [882](#)
 - GiNaC::pseries, [919](#)
 - GiNaC::realsymbol, [937](#)
 - GiNaC::symbol, [1033](#)
- imag_part_conjugate
 - GiNaC, [128](#)
- imag_part_eval
 - GiNaC, [127](#)
- imag_part_evalf
 - GiNaC, [127](#)
- imag_part_expl_derivative
 - GiNaC, [128](#)
- imag_part_f
 - GiNaC::function_options, [617](#)
- imag_part_func
 - GiNaC::function_options, [595–597](#), [610](#)
- imag_part_funcp
 - GiNaC, [60](#)
- imag_part_funcp_1
 - GiNaC, [61](#)
- imag_part_funcp_10
 - GiNaC, [73](#)
- imag_part_funcp_11
 - GiNaC, [74](#)
- imag_part_funcp_12
 - GiNaC, [76](#)
- imag_part_funcp_13
 - GiNaC, [77](#)
- imag_part_funcp_14
 - GiNaC, [79](#)
- imag_part_funcp_2
 - GiNaC, [62](#)
- imag_part_funcp_3
 - GiNaC, [64](#)
- imag_part_funcp_4
 - GiNaC, [65](#)
- imag_part_funcp_5
 - GiNaC, [66](#)
- imag_part_funcp_6
 - GiNaC, [67](#)
- imag_part_funcp_7
 - GiNaC, [69](#)
- imag_part_funcp_8
 - GiNaC, [70](#)
- imag_part_funcp_9
 - GiNaC, [71](#)
- imag_part_funcp_exvector
 - GiNaC, [80](#)
- imag_part_imag_part
 - GiNaC, [128](#)
- imag_part_print_latex
 - GiNaC, [127](#)
- imag_part_real_part
 - GiNaC, [128](#)
- imag_part_use_exvector_args
 - GiNaC::function_options, [620](#)
- impl
 - GiNaC::Gt_detail::TransformExpressionWithCache< Type >, [1075](#)
 - GiNaC::print_funcfor, [898](#)
- increment
 - GiNaC::const_postorder_iterator, [392](#)
 - GiNaC::const_preorder_iterator, [395](#)
- indefinite
 - GiNaC::info_flags, [672](#)
- index0
 - GiNaC, [238](#)
- index1
 - GiNaC, [239](#)
- index2
 - GiNaC, [239](#)
- index3

- GiNaC, 239
- index_dimensions
 - GiNaC, 233
- indexed
 - GiNaC::indexed, 660–664
 - GiNaC::info_flags, 672
- indexed.cpp, 1158
- indexed.h, 1160
- indices
 - GiNaC::symmetry, 1047
- indices_consistent
 - GiNaC, 120
- info
 - GiNaC::add, 287
 - GiNaC::basic, 320
 - GiNaC::class_info< OPT >::tree_node, 1076
 - GiNaC::constant, 400
 - GiNaC::container< C >, 412
 - GiNaC::ex, 496
 - GiNaC::expairseq, 531
 - GiNaC::function, 577
 - GiNaC::idx, 646
 - GiNaC::indexed, 664
 - GiNaC::minkmetric, 742
 - GiNaC::mul, 759
 - GiNaC::ncmul, 809
 - GiNaC::numeric, 823
 - GiNaC::power, 876
 - GiNaC::relational, 948
 - GiNaC::spinmetric, 981
 - GiNaC::structure< T, ComparisonPolicy >, 991
 - GiNaC::symbol, 1031
 - GiNaC::tensdelta, 1054
 - GiNaC::tensepsilon, 1060
 - GiNaC::tensmetric, 1066
- info_f
 - GiNaC::function_options, 618
- info_func
 - GiNaC::function_options, 607–609, 611
- info_funcp
 - GiNaC, 61
- info_funcp_1
 - GiNaC, 62
- info_funcp_10
 - GiNaC, 74
- info_funcp_11
 - GiNaC, 75
- info_funcp_12
 - GiNaC, 76
- info_funcp_13
 - GiNaC, 78
- info_funcp_14
 - GiNaC, 79
- info_funcp_2
 - GiNaC, 63
- info_funcp_3
 - GiNaC, 64
- info_funcp_4
 - GiNaC, 66
- info_funcp_5
 - GiNaC, 67
- info_funcp_6
 - GiNaC, 68
- info_funcp_7
 - GiNaC, 70
- info_funcp_8
 - GiNaC, 71
- info_funcp_9
 - GiNaC, 72
- info_funcp_exvector
 - GiNaC, 81
- info_use_exvector_args
 - GiNaC::function_options, 620
- inifcns.cpp, 1161
- inifcns.h, 1164
- inifcns_elliptic.cpp, 1165
- inifcns_gamma.cpp, 1167
- inifcns_nstdsums.cpp, 1168
- inifcns_trans.cpp, 1170
- init
 - GiNaC::basic_multi_iterator< T >, 346
 - GiNaC::multi_iterator_counter< T >, 773
 - GiNaC::multi_iterator_counter_indv< T >, 776
 - GiNaC::multi_iterator_ordered< T >, 779
 - GiNaC::multi_iterator_ordered_eq< T >, 782
 - GiNaC::multi_iterator_ordered_eq_indv< T >, 785
 - GiNaC::multi_iterator_permutation< T >, 789
 - GiNaC::multi_iterator_shuffle< T >, 792
 - GiNaC::multi_iterator_shuffle_prime< T >, 796
- init_table
 - GiNaC::remember_table, 956
- init_unarchivers
 - GiNaC::library_init, 713
- initialize
 - GiNaC::function_options, 587
- insert
 - GiNaC::unarchive_table_t, 1077
- insert_symbols
 - GiNaC::symbolset, 1038
- int_length
 - GiNaC::numeric, 840
- integer
 - GiNaC::info_flags, 672
- integer_content
 - GiNaC::add, 290
 - GiNaC::basic, 327
 - GiNaC::ex, 508
 - GiNaC::mul, 763
 - GiNaC::numeric, 826
 - GiNaC::structure< T, ComparisonPolicy >, 997
- integer_partition
 - GiNaC::Gt_detail, 273
- integer_polynomial
 - GiNaC::info_flags, 672
- integral
 - GiNaC::error_and_integral, 483

- GiNaC::integral, [676](#)
- integral.cpp, [1173](#)
- integral.h, [1174](#)
- integrate
 - GiNaC::Gt_detail::pathintegral_term, [848](#)
- integration_kernel.cpp, [1174](#)
 - cache_vec, [1176](#)
 - order, [1176](#)
 - qbar, [1176](#)
 - x, [1176](#)
- integration_kernel.h, [1176](#)
- interpolate
 - GiNaC, [194](#)
- inv_at_cit
 - GiNaC::archive, [297](#)
- inverse
 - GiNaC, [185](#), [220](#)
 - GiNaC::matrix, [731](#)
 - GiNaC::numeric, [833](#)
- inverse_atoms
 - GiNaC::archive, [302](#)
- iquo
 - GiNaC, [217](#)
- irem
 - GiNaC, [216](#)
- is_a
 - GiNaC, [84](#), [110](#), [235](#)
 - GiNaC::ex, [518](#)
- is_canonical
 - GiNaC::expairseq, [541](#)
- is_canonical_numeric
 - GiNaC::expair, [522](#)
- is_cinteger
 - GiNaC, [222](#)
 - GiNaC::numeric, [836](#)
- is_clifford_tinfo
 - GiNaC, [95](#)
- is_color_tinfo
 - GiNaC, [99](#)
- is_compatible_to
 - GiNaC::pseries, [921](#)
- is_contravariant
 - GiNaC::varidx, [1091](#)
- is_covariant
 - GiNaC::varidx, [1091](#)
- is_crational
 - GiNaC, [222](#)
 - GiNaC::numeric, [837](#)
- is_defined
 - GiNaC::scalar_products, [965](#)
- is_dim_numeric
 - GiNaC::idx, [650](#)
- is_dim_symbolic
 - GiNaC::idx, [651](#)
- is_dirac_slash
 - GiNaC, [86](#)
- is_discriminant_of_quadratic_number_field
 - GiNaC, [175](#)
- is_dotted
 - GiNaC::spinidx, [975](#)
- is_dummy_pair
 - GiNaC, [117](#)
- is_dummy_pair_same_type
 - GiNaC::idx, [649](#)
 - GiNaC::spinidx, [973](#)
 - GiNaC::varidx, [1090](#)
- is_equal
 - GiNaC::basic, [334](#)
 - GiNaC::ex, [512](#)
 - GiNaC::expair, [522](#)
 - GiNaC::numeric, [834](#)
 - GiNaC::remember_table_entry, [958](#)
- is_equal_same_type
 - GiNaC::basic, [331](#)
 - GiNaC::constant, [402](#)
 - GiNaC::container< C >, [415](#)
 - GiNaC::expairseq, [534](#)
 - GiNaC::fderivative, [560](#)
 - GiNaC::function, [577](#)
 - GiNaC::numeric, [829](#)
 - GiNaC::structure< T, ComparisonPolicy >, [1000](#)
 - GiNaC::symbol, [1034](#)
- is_even
 - GiNaC, [221](#)
 - GiNaC::numeric, [835](#)
- is_ex_the_function
 - function.h, [1150](#)
- is_exactly_a
 - GiNaC, [84](#), [110](#)
 - GiNaC::ex, [518](#)
- is_integer
 - GiNaC, [221](#)
 - GiNaC::numeric, [835](#)
- is_less
 - GiNaC::expair, [522](#)
- is_negative
 - GiNaC, [221](#)
 - GiNaC::numeric, [834](#)
 - GiNaC::status_flags, [985](#)
- is_nonneg_integer
 - GiNaC, [221](#)
 - GiNaC::numeric, [835](#)
- is_numeric
 - GiNaC::Ebar_kernel, [456](#)
 - GiNaC::Eisenstein_h_kernel, [463](#)
 - GiNaC::Eisenstein_kernel, [472](#)
 - GiNaC::ELi_kernel, [481](#)
 - GiNaC::idx, [650](#)
 - GiNaC::integration_kernel, [687](#)
 - GiNaC::Kronecker_dtau_kernel, [700](#)
 - GiNaC::Kronecker_dz_kernel, [707](#)
 - GiNaC::modular_form_kernel, [749](#)
 - GiNaC::multiple_polylog_kernel, [801](#)
 - GiNaC::user_defined_kernel, [1083](#)
- is_odd
 - GiNaC, [222](#)

- GiNaC::numeric, [835](#)
- is_order_function
 - GiNaC, [143](#)
- is_polynomial
 - GiNaC, [104](#)
 - GiNaC::add, [287](#)
 - GiNaC::basic, [324](#)
 - GiNaC::constant, [400](#)
 - GiNaC::ex, [502](#)
 - GiNaC::mul, [760](#)
 - GiNaC::numeric, [824](#)
 - GiNaC::power, [877](#)
 - GiNaC::symbol, [1033](#)
- is_pos_integer
 - GiNaC, [221](#)
 - GiNaC::numeric, [835](#)
- is_positive
 - GiNaC, [221](#)
 - GiNaC::numeric, [834](#)
 - GiNaC::status_flags, [985](#)
- is_prime
 - GiNaC, [222](#)
 - GiNaC::numeric, [836](#)
- is_rational
 - GiNaC, [222](#)
 - GiNaC::numeric, [836](#)
- is_real
 - GiNaC, [222](#)
 - GiNaC::numeric, [836](#)
- is_symbolic
 - GiNaC::idx, [650](#)
- is_terminating
 - GiNaC, [236](#)
 - GiNaC::pseries, [922](#)
- is_the_function
 - GiNaC, [116](#)
- is_the_function< G_SERIAL >
 - GiNaC, [141](#)
- is_the_function< iterated_integral_SERIAL >
 - GiNaC, [143](#)
- is_the_function< psi_SERIAL >
 - GiNaC, [142](#)
- is_the_function< zeta_SERIAL >
 - GiNaC, [141](#)
- is_undotted
 - GiNaC::spinidx, [975](#)
- is_valid
 - GiNaC::print_functor, [898](#)
- is_zero
 - GiNaC, [109](#), [220](#)
 - GiNaC::ex, [512](#)
 - GiNaC::numeric, [834](#)
 - GiNaC::pseries, [922](#)
- is_zero_matrix
 - GiNaC::ex, [513](#)
 - GiNaC::matrix, [733](#)
- isqrt
 - GiNaC, [219](#)
- iterated_integral
 - GiNaC, [142](#)
- iterated_integral2_eval
 - GiNaC, [146](#)
- iterated_integral2_evalf
 - GiNaC, [145](#)
- iterated_integral3_eval
 - GiNaC, [146](#)
- iterated_integral3_evalf
 - GiNaC, [146](#)
- iterated_integral_evalf_impl
 - GiNaC, [145](#)
- iterator_category
 - GiNaC::const_iterator, [386](#)
 - GiNaC::const_postorder_iterator, [390](#)
 - GiNaC::const_preorder_iterator, [394](#)
- K
 - GiNaC::Eisenstein_kernel, [474](#)
 - GiNaC::Kronecker_dtau_kernel, [701](#)
 - GiNaC::Kronecker_dz_kernel, [709](#)
- k
 - factor.cpp, [1132](#)
 - GiNaC::Eisenstein_h_kernel, [466](#)
 - GiNaC::Eisenstein_kernel, [473](#)
 - GiNaC::modular_form_kernel, [751](#)
- kernel
 - GiNaC::Gt::kernel, [693](#), [694](#)
- Kronecker_dtau_kernel
 - GiNaC::Kronecker_dtau_kernel, [700](#)
- Kronecker_dz_kernel
 - GiNaC::Kronecker_dz_kernel, [707](#)
- kronecker_symbol
 - GiNaC, [175](#)
- label
 - GiNaC::wildcard, [1099](#)
- lanczos_coeffs
 - GiNaC::lanczos_coeffs, [709](#)
- laplace
 - GiNaC::determinant_algo, [427](#)
- last
 - factor.cpp, [1132](#)
- last_access
 - GiNaC::remember_table_entry, [959](#)
- latex
 - GiNaC, [232](#)
- latex_name
 - GiNaC::function_options, [587](#)
- Laurent_series
 - GiNaC::Eisenstein_h_kernel, [464](#)
 - GiNaC::Eisenstein_kernel, [472](#)
 - GiNaC::integration_kernel, [688](#)
 - GiNaC::modular_form_kernel, [750](#)
 - GiNaC::user_defined_kernel, [1083](#)
- lcm
 - GiNaC, [198](#), [218](#)
- lcm_of_coefficients_denominators
 - GiNaC, [189](#)

- lcmcoeff
 - GiNaC, 189
- lcoeff
 - GiNaC::ex, 503
- ldeg_a
 - GiNaC::sym_desc, 1025
- ldeg_b
 - GiNaC::sym_desc, 1025
- ldegree
 - GiNaC, 104
 - GiNaC::add, 288
 - GiNaC::basic, 324
 - GiNaC::ex, 502
 - GiNaC::integral, 677
 - GiNaC::mul, 760
 - GiNaC::ncmul, 810
 - GiNaC::numeric, 824
 - GiNaC::power, 878
 - GiNaC::pseries, 917
 - GiNaC::structure< T, ComparisonPolicy >, 994
- len
 - factor.cpp, 1132
- less
 - GiNaC::relational, 947
- less_or_equal
 - GiNaC::relational, 947
- let_op
 - GiNaC::basic, 321
 - GiNaC::clifford, 363
 - GiNaC::container< C >, 413
 - GiNaC::Ebar_kernel, 455
 - GiNaC::Eisenstein_h_kernel, 463
 - GiNaC::Eisenstein_kernel, 472
 - GiNaC::ELi_kernel, 480
 - GiNaC::ex, 498
 - GiNaC::integral, 678
 - GiNaC::Kronecker_dtau_kernel, 700
 - GiNaC::Kronecker_dz_kernel, 707
 - GiNaC::matrix, 723
 - GiNaC::modular_form_kernel, 749
 - GiNaC::multiple_polylog_kernel, 801
 - GiNaC::structure< T, ComparisonPolicy >, 992
 - GiNaC::user_defined_kernel, 1083
- lgamma
 - GiNaC, 211, 212
- lgamma_conjugate
 - GiNaC, 147
- lgamma_deriv
 - GiNaC, 147
- lgamma_eval
 - GiNaC, 146
- lgamma_evalf
 - GiNaC, 146
- lgamma_series
 - GiNaC, 147
- lh
 - GiNaC::relational, 953
- lhs
 - GiNaC, 109
 - GiNaC::ex, 498
 - GiNaC::relational, 951
- Li2
 - GiNaC, 211
- Li2_
 - GiNaC, 210
- Li2_conjugate
 - GiNaC, 135
- Li2_deriv
 - GiNaC, 134
- Li2_eval
 - GiNaC, 134
- Li2_evalf
 - GiNaC, 134
- Li2_projection
 - GiNaC, 210
- Li2_series
 - GiNaC, 135, 210
- Li3_eval
 - GiNaC, 135
- Li_deriv
 - GiNaC, 152
- Li_eval
 - GiNaC, 152
- Li_evalf
 - GiNaC, 152
- Li_print_latex
 - GiNaC, 152
- Li_series
 - GiNaC, 152
- library_init
 - GiNaC::library_init, 712
- library_initializer
 - GiNaC, 257
- likely
 - compiler.h, 1116
- link_ex
 - GiNaC, 112, 113
- list
 - GiNaC::info_flags, 672
- log
 - GiNaC, 205
- log2
 - GiNaC, 249
- log_conjugate
 - GiNaC, 160
- log_deriv
 - GiNaC, 159
- log_eval
 - GiNaC, 158
- log_evalf
 - GiNaC, 158
- log_expand
 - GiNaC, 159
- log_imag_part
 - GiNaC, 159
- log_info

- GiNaC, 160
- log_real_part
 - GiNaC, 159
- log_series
 - GiNaC, 159
- lookup_entry
 - GiNaC::remember_table, 955
 - GiNaC::remember_table_list, 960
- lookup_map
 - GiNaC, 81
- lookup_remember_table
 - GiNaC::function, 579
- lorentz_eps
 - GiNaC, 248
- lorentz_g
 - GiNaC, 247
- lr
 - factor.cpp, 1131
- lsolve
 - GiNaC, 140
- lst
 - GiNaC, 81
- lst.cpp, 1178
- lst.h, 1178
- lst_evaluate
 - GiNaC::Gt, 634, 636
- lst_to_clifford
 - GiNaC, 92
- lst_to_matrix
 - GiNaC, 182
- m
 - factor.cpp, 1130
 - GiNaC::basic_partition_generator::mpartition2, 752
 - GiNaC::Ebar_kernel, 457
 - GiNaC::ELi_kernel, 482
 - GiNaC::matrix, 737
 - GiNaC::partition_with_zero_parts_generator, 846
- make_flat
 - GiNaC::expairseq, 540
- make_flat_inserter
 - GiNaC::make_flat_inserter, 714
- make_hash_seed
 - GiNaC, 116
- make_real_float
 - GiNaC, 202
- make_return_type_t
 - GiNaC, 236
- make_safe_bool
 - GiNaC::relational, 952
- makewritable
 - GiNaC::ptr< T >, 930
- makewriteable
 - GiNaC::ex, 517
- map
 - GiNaC::basic, 323
 - GiNaC::ex, 501
 - GiNaC::expairseq, 532
 - GiNaC::idx, 647
 - GiNaC::power, 877
 - GiNaC::relational, 949
 - GiNaC::structure< T, ComparisonPolicy >, 993
- map_eval_integ
 - GiNaC, 255
- map_evalm
 - GiNaC, 255
- markowitz
 - GiNaC::solve_algo, 967
- markowitz_elimination
 - GiNaC::matrix, 735
- match
 - GiNaC, 107
 - GiNaC::basic, 322
 - GiNaC::ex, 500
 - GiNaC::expairseq, 533
 - GiNaC::Gt, 630
 - GiNaC::structure< T, ComparisonPolicy >, 992
 - GiNaC::wildcard, 1097
- match_same_type
 - GiNaC::basic, 322
 - GiNaC::clifford, 361
 - GiNaC::color, 378
 - GiNaC::fderivative, 560
 - GiNaC::function, 577
 - GiNaC::idx, 649
 - GiNaC::matrix, 726
 - GiNaC::relational, 950
 - GiNaC::spinidx, 974
 - GiNaC::structure< T, ComparisonPolicy >, 993
 - GiNaC::varidx, 1090
- matrix
 - GiNaC::matrix, 721, 722
- matrix.cpp, 1179
- matrix.h, 1180
- max_assoc_size
 - GiNaC::remember_table, 956
 - GiNaC::remember_table_list, 960
- max_coefficient
 - GiNaC::add, 291
 - GiNaC::basic, 328
 - GiNaC::ex, 510
 - GiNaC::mul, 763
 - GiNaC::numeric, 827
 - GiNaC::structure< T, ComparisonPolicy >, 998
- max_deg
 - GiNaC::sym_desc, 1025
- max_integration_level
 - GiNaC::integral, 681
- max_lcnops
 - GiNaC::sym_desc, 1026
- metric
 - GiNaC::clifford, 364
- metric_tensor
 - GiNaC, 246
- minimal_dim
 - GiNaC, 118

- GiNaC::idx, [651](#)
- minkmetric
 - GiNaC::minkmetric, [742](#)
- minkowski
 - GiNaC::tensepsilon, [1061](#)
- mod
 - GiNaC, [215](#)
- modular_form_kernel
 - GiNaC::modular_form_kernel, [748](#)
- modulus
 - factor.cpp, [1135](#)
- mpartition2
 - GiNaC::basic_partition_generator::mpartition2, [752](#)
- mpgen
 - GiNaC::basic_partition_generator, [348](#)
- mul
 - GiNaC::add, [295](#)
 - GiNaC::matrix, [727](#), [728](#)
 - GiNaC::mul, [758](#), [759](#)
 - GiNaC::numeric, [830](#)
 - GiNaC::power, [886](#)
- mul.cpp, [1181](#)
- mul.h, [1182](#)
- mul_const
 - GiNaC::pseries, [923](#)
- mul_dyn
 - GiNaC::numeric, [831](#)
- mul_scalar
 - GiNaC::matrix, [728](#)
- mul_series
 - GiNaC::pseries, [923](#)
- multi_iterator_counter
 - GiNaC::multi_iterator_counter< T >, [772](#)
- multi_iterator_counter_indv
 - GiNaC::multi_iterator_counter_indv< T >, [775](#)
- multi_iterator_ordered
 - GiNaC::multi_iterator_ordered< T >, [779](#)
- multi_iterator_ordered_eq
 - GiNaC::multi_iterator_ordered_eq< T >, [782](#)
- multi_iterator_ordered_eq_indv
 - GiNaC::multi_iterator_ordered_eq_indv< T >, [785](#)
- multi_iterator_permutation
 - GiNaC::multi_iterator_permutation< T >, [788](#)
- multi_iterator_shuffle
 - GiNaC::multi_iterator_shuffle< T >, [792](#)
- multi_iterator_shuffle_prime
 - GiNaC::multi_iterator_shuffle_prime< T >, [795](#)
- multinomial_coefficient
 - GiNaC, [250](#)
- multiple_polylog_kernel
 - GiNaC::multiple_polylog_kernel, [801](#)
- multiply_lcm
 - GiNaC, [190](#)
- my_ios_callback
 - GiNaC, [230](#)
- my_ios_index
 - GiNaC, [230](#)
- N
 - GiNaC::basic_multi_iterator< T >, [347](#)
 - GiNaC::Eisenstein_h_kernel, [466](#)
 - GiNaC::Eisenstein_kernel, [473](#)
- n
 - factor.cpp, [1131](#)
 - GiNaC::basic_partition_generator::mpartition2, [752](#)
 - GiNaC::Ebar_kernel, [457](#)
 - GiNaC::ELi_kernel, [482](#)
 - GiNaC::Kronecker_dtau_kernel, [701](#)
 - GiNaC::Kronecker_dz_kernel, [708](#)
- N_internal
 - GiNaC::multi_iterator_shuffle< T >, [793](#)
- name
 - GiNaC::archive::archived_ex, [313](#)
 - GiNaC::archive_node::property, [909](#)
 - GiNaC::archive_node::property_info, [910](#)
 - GiNaC::constant, [404](#)
 - GiNaC::function_options, [616](#)
 - GiNaC::print_context_options, [890](#)
 - GiNaC::registered_class_options, [942](#)
 - GiNaC::symbol, [1036](#)
- nargs
 - GiNaC::Gt, [638](#)
- ncmul
 - GiNaC::mul, [770](#)
 - GiNaC::ncmul, [808](#), [809](#)
- ncmul.cpp, [1183](#)
- ncmul.h, [1183](#)
- negative
 - GiNaC::info_flags, [672](#)
- negint
 - GiNaC::info_flags, [672](#)
- next
 - GiNaC::class_info< OPT >, [351](#)
 - GiNaC::composition_generator, [383](#)
 - GiNaC::composition_generator::coolmulti::element, [475](#)
 - GiNaC::partition_generator, [844](#)
 - GiNaC::partition_with_zero_parts_generator, [846](#)
- next_partition
 - GiNaC::basic_partition_generator::mpartition2, [752](#)
- next_permutation
 - GiNaC::composition_generator::coolmulti, [424](#)
- next_print_context_id
 - GiNaC, [258](#)
- next_serial
 - GiNaC::constant, [404](#)
 - GiNaC::symbol, [1037](#)
- ni
 - GiNaC::Gt::kernel, [694](#)
- no_heur_gcd
 - GiNaC::gcd_options, [623](#)
- no_index_dimensions
 - GiNaC, [233](#)
- no_index_renaming

- GiNaC::subs_options, 1021
- no_part_factored
 - GiNaC::gcd_options, 623
- no_pattern
 - GiNaC::subs_options, 1021
- no_type
 - GiNaC::has_distance< T >, 640
- nodes
 - GiNaC::archive, 302
- noncommutative
 - GiNaC::return_types, 963
- noncommutative_composite
 - GiNaC::return_types, 963
- none
 - GiNaC::symmetry, 1043
- nonnegative
 - GiNaC::info_flags, 672
- nonnegint
 - GiNaC::info_flags, 672
- nonnull
 - GiNaC::relational::safe_bool_helper, 963
- nops
 - GiNaC, 102, 183
 - GiNaC::basic, 320
 - GiNaC::clifford, 362
 - GiNaC::container< C >, 412
 - GiNaC::Ebar_kernel, 455
 - GiNaC::Eisenstein_h_kernel, 463
 - GiNaC::Eisenstein_kernel, 471
 - GiNaC::ELi_kernel, 480
 - GiNaC::ex, 496
 - GiNaC::expairseq, 531
 - GiNaC::Gt, 629
 - GiNaC::idx, 647
 - GiNaC::integral, 678
 - GiNaC::Kronecker_dtau_kernel, 700
 - GiNaC::Kronecker_dz_kernel, 707
 - GiNaC::matrix, 723
 - GiNaC::modular_form_kernel, 749
 - GiNaC::multiple_polylog_kernel, 801
 - GiNaC::power, 877
 - GiNaC::pseries, 916
 - GiNaC::relational, 948
 - GiNaC::structure< T, ComparisonPolicy >, 991
 - GiNaC::user_defined_kernel, 1083
- normal
 - GiNaC, 105
 - GiNaC::add, 289
 - GiNaC::basic, 326
 - GiNaC::ex, 505
 - GiNaC::mul, 762
 - GiNaC::numeric, 826
 - GiNaC::power, 881
 - GiNaC::pseries, 918
 - GiNaC::structure< T, ComparisonPolicy >, 996
 - GiNaC::symbol, 1032
- normal.cpp, 1184
 - FAST_COMPARE, 1186
 - STATISTICS, 1187
 - USE_REMEMBER, 1186
 - USE_TRIAL_DIVISION, 1187
- normal.h, 1187
- not_equal
 - GiNaC::relational, 947
- not_shareable
 - GiNaC::status_flags, 985
- not_symmetric
 - GiNaC, 239
- nparams
 - GiNaC::function_options, 616
- nqexand_transformer
 - GiNaC::Gt, 638
- num
 - GiNaC::symminfo, 1049
- num_expressions
 - GiNaC::archive, 299
- number
 - GiNaC::constant, 404
- number_of_type
 - GiNaC, 120
- numer
 - GiNaC, 105, 223
 - GiNaC::ex, 506
 - GiNaC::numeric, 840
- numer_denom
 - GiNaC, 105
 - GiNaC::ex, 507
- numeric
 - GiNaC::info_flags, 672
 - GiNaC::numeric, 822, 823
- numeric.cpp, 1188
- numeric.h, 1191
- Nv
 - GiNaC::multi_iterator_counter_indv< T >, 777
 - GiNaC::multi_iterator_ordered_eq_indv< T >, 786
- o
 - GiNaC::relational, 953
- obj
 - GiNaC::structure< T, ComparisonPolicy >, 1002
- odd
 - GiNaC::info_flags, 672
- one
 - factor.cpp, 1131
- op
 - GiNaC, 109
 - GiNaC::basic, 320
 - GiNaC::clifford, 363
 - GiNaC::container< C >, 413
 - GiNaC::Ebar_kernel, 455
 - GiNaC::Eisenstein_h_kernel, 463
 - GiNaC::Eisenstein_kernel, 472
 - GiNaC::ELi_kernel, 480
 - GiNaC::ex, 497
 - GiNaC::expairseq, 531
 - GiNaC::idx, 647
 - GiNaC::integral, 678

- GiNaC::Kronecker_dtau_kernel, 700
 - GiNaC::Kronecker_dz_kernel, 707
 - GiNaC::matrix, 723
 - GiNaC::modular_form_kernel, 749
 - GiNaC::multiple_polylog_kernel, 801
 - GiNaC::power, 877
 - GiNaC::pseries, 916
 - GiNaC::relational, 948
 - GiNaC::structure< T, ComparisonPolicy >, 991
 - GiNaC::user_defined_kernel, 1083
- operator long
 - GiNaC::numeric_digits, 279
- operator safe_bool
 - GiNaC::relational, 952
- operator!
 - GiNaC::relational, 952
- operator!=
 - GiNaC, 229
 - GiNaC::const_iterator, 388
 - GiNaC::const_postorder_iterator, 392
 - GiNaC::const_preorder_iterator, 395
 - GiNaC::internal::_iter_rep, 277
 - GiNaC::numeric, 837
 - GiNaC::ptr< T >, 930, 931
 - GiNaC::return_type_t, 962
- operator<
 - GiNaC, 230
 - GiNaC::const_iterator, 388
 - GiNaC::Gt_detail, 274
 - GiNaC::Gt_detail::pathintegral_term, 849
 - GiNaC::numeric, 837
 - GiNaC::return_type_t, 961
 - GiNaC::spmapkey, 983
 - GiNaC::sym_desc, 1025
- operator<<
 - GiNaC, 83, 102, 219, 231, 252–254
 - GiNaC::archive, 301
 - GiNaC::archive_node, 310
 - GiNaC::basic_multi_iterator< T >, 347
 - GiNaC::Gt_detail, 274
 - GiNaC::Gt_detail::pathintegral_term, 848
 - GiNaC::multi_iterator_counter< T >, 773
 - GiNaC::multi_iterator_counter_indv< T >, 776
 - GiNaC::multi_iterator_ordered< T >, 780
 - GiNaC::multi_iterator_ordered_eq< T >, 783
 - GiNaC::multi_iterator_ordered_eq_indv< T >, 786
 - GiNaC::multi_iterator_permutation< T >, 790
 - GiNaC::multi_iterator_shuffle< T >, 793
 - GiNaC::multi_iterator_shuffle_prime< T >, 796
 - GiNaC::ptr< T >, 931
- operator<=
 - GiNaC, 230
 - GiNaC::const_iterator, 388
 - GiNaC::numeric, 837
- operator>
 - GiNaC, 230
 - GiNaC::const_iterator, 388
 - GiNaC::numeric, 838
- operator>>
 - GiNaC, 83, 232
 - GiNaC::archive, 301
 - GiNaC::archive_node, 310
- operator>=
 - GiNaC, 230
 - GiNaC::const_iterator, 389
 - GiNaC::numeric, 838
- operator()
 - GiNaC::basic_multi_iterator< T >, 346
 - GiNaC::derivative_map_function, 426
 - GiNaC::error_and_integral_is_less, 484
 - GiNaC::eval_integ_map_function, 485
 - GiNaC::evalf_map_function, 486
 - GiNaC::evalm_map_function, 487
 - GiNaC::ex_base_is_less, 519
 - GiNaC::ex_is_equal, 519
 - GiNaC::ex_is_less, 520
 - GiNaC::ex_swap, 520
 - GiNaC::expair_is_less, 524
 - GiNaC::expair_rest_is_less, 525
 - GiNaC::expair_swap, 525
 - GiNaC::expand_map_function, 544
 - GiNaC::Gt_detail::TransformExpressionWithCache< Type >, 1074
 - GiNaC::idx_is_equal_ignore_dim, 653
 - GiNaC::is_not_a_clifford, 691
 - GiNaC::is_summation_idx, 691
 - GiNaC::map_function, 716
 - GiNaC::matrix, 728, 729
 - GiNaC::normal_map_function, 816
 - GiNaC::op0_is_equal, 843
 - GiNaC::pointer_to_map_function, 852
 - GiNaC::pointer_to_map_function_1arg< T1 >, 854
 - GiNaC::pointer_to_map_function_2args< T1, T2 >, 855
 - GiNaC::pointer_to_map_function_3args< T1, T2, T3 >, 857
 - GiNaC::pointer_to_member_to_map_function< C >, 859
 - GiNaC::pointer_to_member_to_map_function_1arg< C, T1 >, 861
 - GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 >, 862
 - GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >, 864
 - GiNaC::print_functor, 897
 - GiNaC::print_functor_impl, 899
 - GiNaC::print_memfun_handler< T, C >, 901
 - GiNaC::print_ptrfun_handler< T, C >, 904
 - GiNaC::sy_is_less, 1022
 - GiNaC::sy_swap, 1023
 - GiNaC::symminfo_is_less_by_orig, 1049
 - GiNaC::symminfo_is_less_by_symmterm, 1050
 - GiNaC::terminfo_is_less, 1073
 - std::equal_to< GiNaC::ex >, 483
 - std::hash< GiNaC::ex >, 642

- std::less< GiNaC::ptr< T > >, 711
- operator+
 - GiNaC, 224, 225, 227
 - GiNaC::const_iterator, 387, 389
- operator++
 - GiNaC, 228, 229
 - GiNaC::basic_multi_iterator< T >, 346
 - GiNaC::const_iterator, 387
 - GiNaC::const_postorder_iterator, 391, 392
 - GiNaC::const_preorder_iterator, 395
 - GiNaC::multi_iterator_counter< T >, 773
 - GiNaC::multi_iterator_counter_indv< T >, 776
 - GiNaC::multi_iterator_ordered< T >, 779
 - GiNaC::multi_iterator_ordered_eq< T >, 782
 - GiNaC::multi_iterator_ordered_eq_indv< T >, 785
 - GiNaC::multi_iterator_permutation< T >, 789
 - GiNaC::multi_iterator_shuffle< T >, 792
- operator+=
 - GiNaC, 226
 - GiNaC::const_iterator, 387
- operator-
 - GiNaC, 225, 227
 - GiNaC::const_iterator, 388, 389
- operator->
 - GiNaC::const_iterator, 386
 - GiNaC::const_postorder_iterator, 391
 - GiNaC::const_preorder_iterator, 394
 - GiNaC::ptr< T >, 930
 - GiNaC::structure< T, ComparisonPolicy >, 1001
- operator--
 - GiNaC, 228, 229
 - GiNaC::const_iterator, 387
- operator-=
 - GiNaC, 226, 227
 - GiNaC::const_iterator, 388
- operator/
 - GiNaC, 225, 226
- operator/=
 - GiNaC, 226, 227
- operator=
 - GiNaC::_numeric_digits, 279
 - GiNaC::archive_node, 305
 - GiNaC::basic, 317
 - GiNaC::numeric, 832, 833
 - GiNaC::print_functor, 897
 - GiNaC::ptr< T >, 929
- operator==
 - GiNaC, 229
 - GiNaC::const_iterator, 388
 - GiNaC::const_postorder_iterator, 392
 - GiNaC::const_preorder_iterator, 395
 - GiNaC::internal::_iter_rep, 277
 - GiNaC::numeric, 837
 - GiNaC::ptr< T >, 930, 931
 - GiNaC::return_type_t, 961
 - GiNaC::spmapkey, 983
- operator[]
 - GiNaC::basic, 321, 322
- GiNaC::basic_multi_iterator< T >, 345
- GiNaC::const_iterator, 387
- GiNaC::ex, 497, 498
- GiNaC::structure< T, ComparisonPolicy >, 991, 992
- operator*
 - GiNaC, 225
 - GiNaC::const_iterator, 386
 - GiNaC::const_postorder_iterator, 391
 - GiNaC::const_preorder_iterator, 394
 - GiNaC::ptr< T >, 929
- operator*=
 - GiNaC, 226, 227
- operators
 - GiNaC::relational, 947
- operators.cpp, 1194
- operators.h, 1196
- options
 - factor.cpp, 1135
 - GiNaC::class_info< OPT >, 351
 - GiNaC::expand_map_function, 544
 - GiNaC::print_context, 888
- order
 - integration_kernel.cpp, 1176
- Order_conjugate
 - GiNaC, 139
- Order_eval
 - GiNaC, 138
- Order_expl_derivative
 - GiNaC, 139
- Order_imag_part
 - GiNaC, 139
- Order_power
 - GiNaC, 139
- Order_real_part
 - GiNaC, 139
- Order_series
 - GiNaC, 138
- orig
 - GiNaC::symminfo, 1049
 - GiNaC::terminfo, 1072
- overall_coeff
 - GiNaC::expairseq, 542
- overflow
 - GiNaC::basic_multi_iterator< T >, 345
- overloaded
 - GiNaC::function_options, 614
- P
 - GiNaC::modular_form_kernel, 751
- p
 - GiNaC::ptr< T >, 932
- parameter_set
 - GiNaC::fderivative, 562
- paramset
 - GiNaC, 60
- parent
 - GiNaC::class_info< OPT >, 351
- parent_name

- GiNaC::print_context_options, [890](#)
 - GiNaC::registered_class_options, [942](#)
- parents_identified
 - GiNaC::class_info< OPT >, [351](#)
- partition
 - GiNaC::partition_generator, [844](#)
 - GiNaC::partition_with_zero_parts_generator, [846](#)
- partition_generator
 - GiNaC::partition_generator, [844](#)
- partition_with_zero_parts_generator
 - GiNaC::partition_with_zero_parts_generator, [845](#)
- pathintegral_term
 - GiNaC::Gt_detail::pathintegral_term, [847](#)
- pathintegral_term_list
 - GiNaC::Gt_detail::pathintegral_term, [848](#)
- pattern_is_not_product
 - GiNaC::subs_options, [1021](#)
- pattern_is_product
 - GiNaC::subs_options, [1021](#)
- pderivative
 - GiNaC::function, [578](#)
- permutation_sign
 - GiNaC, [251](#)
- permute_free_index_to_front
 - GiNaC, [97](#)
- Pi
 - GiNaC, [256](#)
- PIEvalf
 - GiNaC, [219](#)
- pivot
 - GiNaC::matrix, [735](#)
- point
 - GiNaC::pseries, [926](#)
- pointer
 - GiNaC::const_iterator, [386](#)
 - GiNaC::const_postorder_iterator, [391](#)
 - GiNaC::const_preorder_iterator, [394](#)
- pointer_to_map_function
 - GiNaC::pointer_to_map_function, [852](#)
- pointer_to_map_function_1arg
 - GiNaC::pointer_to_map_function_1arg< T1 >, [854](#)
- pointer_to_map_function_2args
 - GiNaC::pointer_to_map_function_2args< T1, T2 >, [855](#)
- pointer_to_map_function_3args
 - GiNaC::pointer_to_map_function_3args< T1, T2, T3 >, [857](#)
- pointer_to_member_to_map_function
 - GiNaC::pointer_to_member_to_map_function< C >, [859](#)
- pointer_to_member_to_map_function_1arg
 - GiNaC::pointer_to_member_to_map_function_1arg< power_funcp_10 C, T1 >, [860](#)
- pointer_to_member_to_map_function_2args
 - GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 >, [862](#)
- pointer_to_member_to_map_function_3args
 - GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >, [864](#)
- pole_error
 - GiNaC::pole_error, [866](#)
- poly
 - factor.cpp, [1133](#)
- polylog
 - GiNaC::Gt_detail::pathintegral_term, [849](#)
- polynomial
 - GiNaC::factor_options, [546](#)
 - GiNaC::info_flags, [672](#)
- pop
 - GiNaC::Gt_detail::pathintegral_term_list, [851](#)
- pos_sig
 - GiNaC::minkmetric, [744](#)
 - GiNaC::tensepsilon, [1061](#)
- posint
 - GiNaC::info_flags, [672](#)
- positive
 - GiNaC::domain, [450](#)
 - GiNaC::info_flags, [672](#)
- possymbol
 - GiNaC::possymbol, [871](#)
- postorder_begin
 - GiNaC::ex, [494](#)
- postorder_end
 - GiNaC::ex, [494](#)
- pow
 - GiNaC, [220](#), [234](#)
 - GiNaC::matrix, [728](#)
- power
 - GiNaC::add, [295](#)
 - GiNaC::function, [579](#)
 - GiNaC::Gt_detail::pathintegral_term, [849](#)
 - GiNaC::mul, [770](#)
 - GiNaC::ncmul, [814](#)
 - GiNaC::numeric, [830](#)
 - GiNaC::power, [876](#)
- power.cpp, [1198](#)
- power.h, [1198](#)
- power_const
 - GiNaC::pseries, [924](#)
- power_dyn
 - GiNaC::numeric, [832](#)
- power_f
 - GiNaC::function_options, [617](#)
- power_func
 - GiNaC::function_options, [603–605](#), [610](#)
- power_funcp
 - GiNaC, [60](#)
- power_funcp_1
 - GiNaC, [62](#)
- power_funcp_10
 - GiNaC, [73](#)
- power_funcp_11
 - GiNaC, [75](#)
- power_funcp_12
 - GiNaC, [76](#)

- power_funcp_13
 - GiNaC, [78](#)
- power_funcp_14
 - GiNaC, [79](#)
- power_funcp_2
 - GiNaC, [63](#)
- power_funcp_3
 - GiNaC, [64](#)
- power_funcp_4
 - GiNaC, [65](#)
- power_funcp_5
 - GiNaC, [67](#)
- power_funcp_6
 - GiNaC, [68](#)
- power_funcp_7
 - GiNaC, [69](#)
- power_funcp_8
 - GiNaC, [71](#)
- power_funcp_9
 - GiNaC, [72](#)
- power_funcp_exvector
 - GiNaC, [80](#)
- power_use_exvector_args
 - GiNaC::function_options, [620](#)
- pp
 - factor.cpp, [1134](#)
- precedence
 - GiNaC::add, [287](#)
 - GiNaC::basic, [320](#)
 - GiNaC::clifford, [360](#)
 - GiNaC::container< C >, [412](#)
 - GiNaC::expairseq, [531](#)
 - GiNaC::function, [573](#)
 - GiNaC::Gt, [629](#)
 - GiNaC::indexed, [664](#)
 - GiNaC::integral, [677](#)
 - GiNaC::mul, [759](#)
 - GiNaC::ncmul, [809](#)
 - GiNaC::numeric, [823](#)
 - GiNaC::power, [876](#)
 - GiNaC::pseries, [916](#)
 - GiNaC::relational, [948](#)
 - GiNaC::structure< T, ComparisonPolicy >, [990](#)
- prefactor
 - GiNaC::Gt_detail::pathintegral_term, [849](#)
- prem
 - GiNaC, [192](#)
- preorder_begin
 - GiNaC::ex, [493](#)
- preorder_end
 - GiNaC::ex, [493](#)
- prepend
 - GiNaC::container< C >, [417](#)
- prime
 - GiNaC::info_flags, [672](#)
- primitive_dirichlet_character
 - GiNaC, [176](#)
- primpart
 - GiNaC::ex, [509](#)
- print
 - GiNaC::numeric_digits, [279](#)
 - GiNaC::basic, [319](#)
 - GiNaC::ex, [495](#)
 - GiNaC::expair, [522](#)
 - GiNaC::fderivative, [558](#)
 - GiNaC::function, [573](#)
 - GiNaC::structure< T, ComparisonPolicy >, [990](#)
- print.cpp, [1199](#)
- print.h, [1200](#)
 - GINAC_DECLARE_PRINT_CONTEXT, [1201](#)
 - GINAC_DECLARE_PRINT_CONTEXT_BASE, [1201](#)
 - GINAC_DECLARE_PRINT_CONTEXT_COMMON, [1201](#)
 - GINAC_IMPLEMENT_PRINT_CONTEXT, [1202](#)
- print_add
 - GiNaC::add, [294](#)
- print_cache
 - GiNaC::Gt_detail::TransformExpressionWithCache< Type >, [1074](#)
- print_context
 - GiNaC::print_context, [888](#)
- print_context_class_info
 - GiNaC, [82](#)
- print_context_options
 - GiNaC::print_context_options, [889](#)
- print_csrc
 - GiNaC::print_csrc, [891](#)
- print_csrc_cl_N
 - GiNaC::print_csrc_cl_N, [893](#)
- print_csrc_double
 - GiNaC::print_csrc_double, [894](#)
- print_csrc_float
 - GiNaC::print_csrc_float, [895](#)
- print_dflt
 - GiNaC::print_dflt, [896](#)
- print_dispatch
 - GiNaC::basic, [331](#), [332](#)
- print_dispatch_table
 - GiNaC::function_options, [618](#)
 - GiNaC::registered_class_options, [942](#)
- print_elements
 - GiNaC::matrix, [736](#)
- print_func
 - GiNaC::function_options, [611–613](#)
 - GiNaC::registered_class_options, [941](#)
- print_func< print_context >
 - GiNaC, [117](#)
- print_func< print_dflt >
 - GiNaC, [85](#), [96](#), [245](#)
- print_funcp
 - GiNaC, [61](#)
- print_funcp_1
 - GiNaC, [62](#)
- print_funcp_10
 - GiNaC, [73](#)

- print_funcp_11
 - GiNaC, [75](#)
- print_funcp_12
 - GiNaC, [76](#)
- print_funcp_13
 - GiNaC, [78](#)
- print_funcp_14
 - GiNaC, [79](#)
- print_funcp_2
 - GiNaC, [63](#)
- print_funcp_3
 - GiNaC, [64](#)
- print_funcp_4
 - GiNaC, [65](#)
- print_funcp_5
 - GiNaC, [67](#)
- print_funcp_6
 - GiNaC, [68](#)
- print_funcp_7
 - GiNaC, [69](#)
- print_funcp_8
 - GiNaC, [71](#)
- print_funcp_9
 - GiNaC, [72](#)
- print_funcp_exvector
 - GiNaC, [81](#)
- print_functor
 - GiNaC::print_functor, [897](#)
- print_index
 - GiNaC::idx, [651](#)
- print_index_dimensions
 - GiNaC::print_options, [903](#)
- print_indexed
 - GiNaC::indexed, [669](#)
- print_integer_csrc
 - GiNaC, [203](#)
- print_latex
 - GiNaC::print_latex, [900](#)
- print_memfun_handler
 - GiNaC::print_memfun_handler< T, C >, [901](#)
- print_numeric
 - GiNaC::numeric, [840](#)
- print_operator
 - GiNaC, [237](#)
- print_overall_coeff
 - GiNaC::mul, [768](#)
- print_power
 - GiNaC::power, [884](#)
- print_ptrfun_handler
 - GiNaC::print_ptrfun_handler< T, C >, [904](#)
- print_python
 - GiNaC::print_python, [905](#)
- print_python_repr
 - GiNaC::print_python_repr, [906](#)
- print_real_cl_N
 - GiNaC, [205](#)
- print_real_csrc
 - GiNaC, [204](#)
- print_real_number
 - GiNaC, [203](#)
- print_series
 - GiNaC::pseries, [924](#)
- print_sym_pow
 - GiNaC, [233](#)
- print_tree
 - GiNaC::print_tree, [907](#)
- print_use_exvector_args
 - GiNaC::function_options, [620](#)
- printindices
 - GiNaC::indexed, [668](#)
- printpair
 - GiNaC::expairseq, [536](#)
- printraw
 - GiNaC::archive, [300](#)
 - GiNaC::archive_node, [310](#)
- printseq
 - GiNaC::container< C >, [416](#)
 - GiNaC::expairseq, [536](#)
- product_to_exvector
 - GiNaC, [121](#)
- property
 - GiNaC::archive_node::property, [909](#)
- property_info
 - GiNaC::archive_node::property_info, [910](#)
- property_type
 - GiNaC::archive_node, [305](#)
- propinfovector
 - GiNaC::archive_node, [304](#)
- props
 - GiNaC::archive_node, [310](#)
- pseries
 - GiNaC::pseries, [915](#), [916](#)
- pseries.cpp, [1202](#)
- pseries.h, [1203](#)
- psi
 - GiNaC, [142](#), [212](#)
- psi1_deriv
 - GiNaC, [150](#)
- psi1_eval
 - GiNaC, [149](#)
- psi1_evalf
 - GiNaC, [149](#)
- psi1_series
 - GiNaC, [150](#)
- psi2_deriv
 - GiNaC, [150](#)
- psi2_eval
 - GiNaC, [150](#)
- psi2_evalf
 - GiNaC, [150](#)
- psi2_series
 - GiNaC, [151](#)
- ptr
 - GiNaC::pointer_to_map_function, [853](#)
 - GiNaC::pointer_to_map_function_1arg< T1 >, [854](#)

- GiNaC::pointer_to_map_function_2args< T1, T2 >, [856](#)
 - GiNaC::pointer_to_map_function_3args< T1, T2, T3 >, [857](#)
 - GiNaC::pointer_to_member_to_map_function< C >, [859](#)
 - GiNaC::pointer_to_member_to_map_function_1arg< C, T1 >, [861](#)
 - GiNaC::pointer_to_member_to_map_function_2args< C, T1, T2 >, [863](#)
 - GiNaC::pointer_to_member_to_map_function_3args< C, T1, T2, T3 >, [865](#)
 - GiNaC::ptr< T >, [929](#)
- ptr.h, [1204](#)
- PTYPE_BOOL
 - GiNaC::archive_node, [305](#)
- PTYPE_NODE
 - GiNaC::archive_node, [305](#)
- PTYPE_STRING
 - GiNaC::archive_node, [305](#)
- PTYPE_UNSIGNED
 - GiNaC::archive_node, [305](#)
- purely_indefinite
 - GiNaC::status_flags, [985](#)
- python
 - GiNaC, [232](#)
- python_repr
 - GiNaC, [232](#)
- q_expansion_modular_form
 - GiNaC::Eisenstein_h_kernel, [465](#)
 - GiNaC::Eisenstein_kernel, [473](#)
 - GiNaC::modular_form_kernel, [750](#)
- qbar
 - integration_kernel.cpp, [1176](#)
- qExpand
 - GiNaC::Gt, [632](#)
- qExpand_minOrder
 - GiNaC::Gt, [637](#)
- qExpand_stepSize
 - GiNaC::Gt, [637](#)
- quo
 - GiNaC, [190](#)
- R
 - factor.cpp, [1134](#)
- r
 - factor.cpp, [1129](#)
 - GiNaC::Eisenstein_h_kernel, [466](#)
- rank
 - GiNaC, [185](#)
 - GiNaC::matrix, [732](#), [733](#)
- rational
 - GiNaC::info_flags, [672](#)
- rational_function
 - GiNaC::info_flags, [672](#)
- rational_polynomial
 - GiNaC::info_flags, [672](#)
- rbegin
 - GiNaC::container< C >, [418](#)
- read_archive
 - GiNaC::basic, [332](#)
 - GiNaC::clifford, [360](#)
 - GiNaC::color, [377](#)
 - GiNaC::constant, [401](#)
 - GiNaC::container< C >, [414](#)
 - GiNaC::expairseq, [534](#)
 - GiNaC::fderivative, [559](#)
 - GiNaC::function, [576](#)
 - GiNaC::Gt, [631](#)
 - GiNaC::idx, [648](#)
 - GiNaC::indexed, [666](#)
 - GiNaC::integral, [680](#)
 - GiNaC::matrix, [725](#)
 - GiNaC::minkmetric, [743](#)
 - GiNaC::numeric, [828](#)
 - GiNaC::power, [882](#)
 - GiNaC::pseries, [920](#)
 - GiNaC::relational, [949](#)
 - GiNaC::spinidx, [974](#)
 - GiNaC::symbol, [1033](#)
 - GiNaC::symmetry, [1044](#)
 - GiNaC::tensepsilon, [1060](#)
 - GiNaC::varidx, [1090](#)
 - GiNaC::wildcard, [1097](#)
- read_real_float
 - GiNaC, [202](#)
- read_unsigned
 - GiNaC, [83](#)
- real
 - GiNaC, [223](#)
 - GiNaC::domain, [450](#)
 - GiNaC::info_flags, [672](#)
 - GiNaC::numeric, [839](#)
- real_part
 - GiNaC, [103](#)
 - GiNaC::add, [291](#)
 - GiNaC::basic, [330](#)
 - GiNaC::constant, [401](#)
 - GiNaC::container< C >, [415](#)
 - GiNaC::ex, [499](#)
 - GiNaC::function, [576](#)
 - GiNaC::indexed, [665](#)
 - GiNaC::matrix, [725](#)
 - GiNaC::mul, [762](#)
 - GiNaC::ncmul, [812](#)
 - GiNaC::numeric, [828](#)
 - GiNaC::power, [882](#)
 - GiNaC::pseries, [919](#)
 - GiNaC::realsymbol, [937](#)
 - GiNaC::symbol, [1033](#)
- real_part_conjugate
 - GiNaC, [126](#)
- real_part_eval
 - GiNaC, [126](#)
- real_part_evalf
 - GiNaC, [126](#)

- real_part_expl_derivative
 - GiNaC, [127](#)
- real_part_f
 - GiNaC::function_options, [617](#)
- real_part_func
 - GiNaC::function_options, [593–595](#), [610](#)
- real_part_funcp
 - GiNaC, [60](#)
- real_part_funcp_1
 - GiNaC, [61](#)
- real_part_funcp_10
 - GiNaC, [73](#)
- real_part_funcp_11
 - GiNaC, [74](#)
- real_part_funcp_12
 - GiNaC, [75](#)
- real_part_funcp_13
 - GiNaC, [77](#)
- real_part_funcp_14
 - GiNaC, [78](#)
- real_part_funcp_2
 - GiNaC, [62](#)
- real_part_funcp_3
 - GiNaC, [64](#)
- real_part_funcp_4
 - GiNaC, [65](#)
- real_part_funcp_5
 - GiNaC, [66](#)
- real_part_funcp_6
 - GiNaC, [67](#)
- real_part_funcp_7
 - GiNaC, [69](#)
- real_part_funcp_8
 - GiNaC, [70](#)
- real_part_funcp_9
 - GiNaC, [71](#)
- real_part_funcp_exvector
 - GiNaC, [80](#)
- real_part_imag_part
 - GiNaC, [127](#)
- real_part_print_latex
 - GiNaC, [126](#)
- real_part_real_part
 - GiNaC, [127](#)
- real_part_use_exvector_args
 - GiNaC::function_options, [619](#)
- really_subs_idx
 - GiNaC::subs_options, [1021](#)
- realsymbol
 - GiNaC::realsymbol, [936](#)
- recombine_pair_to_ex
 - GiNaC::add, [294](#)
 - GiNaC::expairseq, [537](#)
 - GiNaC::mul, [766](#)
- reduced_matrix
 - GiNaC, [183](#)
- reeval_ncmul
 - GiNaC, [187](#)
- GiNaC::ncmul, [814](#)
- refcount
 - GiNaC::refcounted, [939](#)
- refcounted
 - GiNaC::refcounted, [939](#)
- reference
 - GiNaC::const_iterator, [386](#)
 - GiNaC::const_postorder_iterator, [391](#)
 - GiNaC::const_preorder_iterator, [394](#)
- REGISTER_FUNCTION
 - function.h, [1150](#)
 - GiNaC, [126–128](#), [130](#), [131](#), [133–139](#), [144](#), [145](#), [147–149](#), [153–155](#), [158](#), [160–162](#), [164–174](#)
- register_new
 - GiNaC::function, [579](#)
- registered_class_info
 - GiNaC, [82](#)
- registered_class_options
 - GiNaC::registered_class_options, [940](#)
- registered_functions
 - GiNaC::function, [579](#)
- registrar.cpp, [1204](#)
- registrar.h, [1205](#)
 - GINAC_DECLARE_REGISTERED_CLASS, [1206](#)
 - GINAC_DECLARE_REGISTERED_CLASS_COMMON, [1206](#)
 - GINAC_DECLARE_REGISTERED_CLASS_NO_CTORS, [1206](#)
 - GINAC_IMPLEMENT_REGISTERED_CLASS, [1207](#)
 - GINAC_IMPLEMENT_REGISTERED_CLASS_OPT, [1207](#)
 - GINAC_IMPLEMENT_REGISTERED_CLASS_OPT_T, [1207](#)
- regularise
 - GiNaC::Gt, [631](#)
- relation
 - GiNaC::info_flags, [672](#)
- relation_equal
 - GiNaC::info_flags, [672](#)
- relation_greater
 - GiNaC::info_flags, [672](#)
- relation_greater_or_equal
 - GiNaC::info_flags, [672](#)
- relation_less
 - GiNaC::info_flags, [672](#)
- relation_less_or_equal
 - GiNaC::info_flags, [672](#)
- relation_not_equal
 - GiNaC::info_flags, [672](#)
- relational
 - GiNaC::relational, [948](#)
- relational.cpp, [1208](#)
- relational.h, [1208](#)
- relative_integration_error
 - GiNaC::integral, [681](#)
- rem
 - GiNaC, [191](#)

- remember
 - GiNaC::function_options, 614
- remember.cpp, 1209
- remember.h, 1209
- remember_assoc_size
 - GiNaC::function_options, 619
- remember_size
 - GiNaC::function_options, 619
- remember_strategy
 - GiNaC::function_options, 619
 - GiNaC::remember_table, 956
 - GiNaC::remember_table_list, 960
- remember_table
 - GiNaC::remember_table, 955
- remember_table_entry
 - GiNaC::function, 580
 - GiNaC::remember_table_entry, 957
- remember_table_list
 - GiNaC::remember_table_list, 960
- remember_tables
 - GiNaC::remember_table, 956
- remove_all
 - GiNaC::container< C >, 417
- remove_dirac_ONE
 - GiNaC, 91
- remove_first
 - GiNaC::container< C >, 417
- remove_last
 - GiNaC::container< C >, 417
- remove_reference
 - GiNaC::refcounted, 939
- rename_dummy_indices
 - GiNaC, 120
- rename_dummy_indices_uniquely
 - GiNaC, 123
- rend
 - GiNaC::container< C >, 418
- replace_contr_index
 - GiNaC::tensor, 1071
- replace_dim
 - GiNaC::idx, 651
- replace_with_symbol
 - GiNaC, 200
- reposition_dummy_indices
 - GiNaC, 121
 - GiNaC::indexed, 670
- representation_label
 - GiNaC::clifford, 364
 - GiNaC::color, 379
- reserve
 - GiNaC::container_storage< C >, 422
- rest
 - GiNaC::expair, 523
- result
 - GiNaC::remember_table_entry, 958
- result_type
 - GiNaC::map_function, 716
- resultant
 - GiNaC, 202
- return_type
 - GiNaC::add, 292
 - GiNaC::basic, 330
 - GiNaC::clifford, 361
 - GiNaC::color, 379
 - GiNaC::ex, 514
 - GiNaC::expairseq, 534
 - GiNaC::fail, 549
 - GiNaC::function, 578
 - GiNaC::function_options, 618
 - GiNaC::indexed, 667
 - GiNaC::integral, 679
 - GiNaC::matrix, 726
 - GiNaC::minkmetric, 743
 - GiNaC::mul, 765
 - GiNaC::ncmul, 813
 - GiNaC::power, 883
 - GiNaC::relational, 950
 - GiNaC::structure< T, ComparisonPolicy >, 1000
 - GiNaC::su3d, 1006
 - GiNaC::su3f, 1011
 - GiNaC::tensdelta, 1055
 - GiNaC::tensepsilon, 1061
 - GiNaC::tensmetric, 1066
 - GiNaC::tensor, 1071
- return_type_tinfo
 - GiNaC::add, 292
 - GiNaC::basic, 330
 - GiNaC::clifford, 361
 - GiNaC::color, 379
 - GiNaC::ex, 514
 - GiNaC::function, 578
 - GiNaC::function_options, 618
 - GiNaC::indexed, 667
 - GiNaC::integral, 679
 - GiNaC::mul, 765
 - GiNaC::ncmul, 813
 - GiNaC::power, 883
 - GiNaC::relational, 951
 - GiNaC::structure< T, ComparisonPolicy >, 1000
- rh
 - GiNaC::relational, 953
- rhs
 - GiNaC, 109
 - GiNaC::ex, 498
 - GiNaC::relational, 952
- rl
 - GiNaC::return_type_t, 962
- root
 - GiNaC::archive::archived_ex, 313
- rotate_left
 - GiNaC, 250
- row
 - GiNaC::matrix, 737
- rows
 - GiNaC, 184
 - GiNaC::matrix, 726

- s
 - GiNaC::const_postorder_iterator, [393](#)
 - GiNaC::const_preorder_iterator, [395](#)
 - GiNaC::derivative_map_function, [426](#)
 - GiNaC::Eisenstein_h_kernel, [466](#)
 - GiNaC::print_context, [888](#)
 - GiNaC::symbolset, [1038](#)
- S_deriv
 - GiNaC, [153](#)
- S_eval
 - GiNaC, [153](#)
- S_evalf
 - GiNaC, [153](#)
- S_print_latex
 - GiNaC, [154](#)
- S_series
 - GiNaC, [153](#)
- safe_bool
 - GiNaC::relational, [947](#)
- same_metric
 - GiNaC::clifford, [362](#)
- scalar_mul_indexed
 - GiNaC::basic, [329](#)
 - GiNaC::matrix, [724](#)
 - GiNaC::structure< T, ComparisonPolicy >, [999](#)
- seq
 - GiNaC::container_storage< C >, [423](#)
 - GiNaC::expairseq, [542](#)
 - GiNaC::pseries, [925](#)
 - GiNaC::remember_table_entry, [958](#)
- serial
 - GiNaC::constant, [404](#)
 - GiNaC::function, [580](#)
 - GiNaC::G2_SERIAL, [621](#)
 - GiNaC::G3_SERIAL, [622](#)
 - GiNaC::iterated_integral2_SERIAL, [692](#)
 - GiNaC::iterated_integral3_SERIAL, [693](#)
 - GiNaC::psi1_SERIAL, [927](#)
 - GiNaC::psi2_SERIAL, [927](#)
 - GiNaC::symbol, [1036](#)
 - GiNaC::zeta1_SERIAL, [1100](#)
 - GiNaC::zeta2_SERIAL, [1100](#)
- series
 - GiNaC, [107](#)
 - GiNaC::add, [289](#)
 - GiNaC::basic, [325](#)
 - GiNaC::Eisenstein_h_kernel, [463](#)
 - GiNaC::Eisenstein_kernel, [471](#)
 - GiNaC::ex, [504](#)
 - GiNaC::fderivative, [558](#)
 - GiNaC::function, [575](#)
 - GiNaC::integral, [680](#)
 - GiNaC::integration_kernel, [687](#)
 - GiNaC::modular_form_kernel, [749](#)
 - GiNaC::mul, [762](#)
 - GiNaC::power, [880](#)
 - GiNaC::pseries, [918](#)
 - GiNaC::structure< T, ComparisonPolicy >, [996](#)
 - GiNaC::symbol, [1031](#)
- series_coeff
 - GiNaC::integration_kernel, [689](#)
- series_coeff_impl
 - GiNaC::basic_log_kernel, [342](#)
 - GiNaC::Ebar_kernel, [456](#)
 - GiNaC::ELi_kernel, [481](#)
 - GiNaC::integration_kernel, [688](#)
 - GiNaC::Kronecker_dtau_kernel, [701](#)
 - GiNaC::Kronecker_dz_kernel, [708](#)
 - GiNaC::multiple_polylog_kernel, [802](#)
- series_f
 - GiNaC::function_options, [618](#)
- series_func
 - GiNaC::function_options, [605–607](#), [610](#)
- series_funcp
 - GiNaC, [61](#)
- series_funcp_1
 - GiNaC, [62](#)
- series_funcp_10
 - GiNaC, [73](#)
- series_funcp_11
 - GiNaC, [75](#)
- series_funcp_12
 - GiNaC, [76](#)
- series_funcp_13
 - GiNaC, [78](#)
- series_funcp_14
 - GiNaC, [79](#)
- series_funcp_2
 - GiNaC, [63](#)
- series_funcp_3
 - GiNaC, [64](#)
- series_funcp_4
 - GiNaC, [65](#)
- series_funcp_5
 - GiNaC, [67](#)
- series_funcp_6
 - GiNaC, [68](#)
- series_funcp_7
 - GiNaC, [69](#)
- series_funcp_8
 - GiNaC, [71](#)
- series_funcp_9
 - GiNaC, [72](#)
- series_funcp_exvector
 - GiNaC, [80](#)
- series_to_poly
 - GiNaC, [235](#)
- series_use_exvector_args
 - GiNaC::function_options, [620](#)
- series_vec
 - GiNaC::integration_kernel, [690](#)
- set
 - GiNaC::matrix, [729](#)
- set_cache_step
 - GiNaC::integration_kernel, [689](#)
- set_name

- GiNaC::function_options, 587
 - GiNaC::symbol, 1035
- set_print_context
 - GiNaC, 231
- set_print_func
 - GiNaC, 236
 - GiNaC::function_options, 615
 - GiNaC::registered_class_options, 942
- set_print_options
 - GiNaC, 231
- set_refcount
 - GiNaC::refcounted, 939
- set_return_type
 - GiNaC::function_options, 613
- set_symmetry
 - GiNaC::function_options, 614
- set_TeX_name
 - GiNaC::symbol, 1035
- set_type
 - GiNaC::symmetry, 1045
- setArgs
 - GiNaC::Gt, 628
- setflag
 - GiNaC::basic, 335
- shaker_sort
 - GiNaC, 251
- share
 - GiNaC::ex, 517
- shift_exponents
 - GiNaC::pseries, 924
- show_statistics
 - GiNaC::remember_table, 956
- simplify_indexed
 - GiNaC, 107, 122
 - GiNaC::ex, 511
 - GiNaC::indexed, 670
- simplify_indexed_product
 - GiNaC, 122
 - GiNaC::indexed, 670
- sin
 - GiNaC, 206
- sin_conjugate
 - GiNaC, 161
- sin_deriv
 - GiNaC, 160
- sin_eval
 - GiNaC, 160
- sin_evalf
 - GiNaC, 160
- sin_imag_part
 - GiNaC, 161
- sin_real_part
 - GiNaC, 161
- sinh
 - GiNaC, 208
- sinh_conjugate
 - GiNaC, 169
- sinh_deriv
 - GiNaC, 168
- sinh_eval
 - GiNaC, 168
- sinh_evalf
 - GiNaC, 168
- sinh_imag_part
 - GiNaC, 168
- sinh_real_part
 - GiNaC, 168
- size
 - GiNaC::basic_multi_iterator< T >, 345
 - GiNaC::Gt_detail::pathintegral_term_list, 850
- smod
 - GiNaC, 216
 - GiNaC::add, 290
 - GiNaC::basic, 327
 - GiNaC::ex, 510
 - GiNaC::mul, 763
 - GiNaC::numeric, 827
 - GiNaC::structure< T, ComparisonPolicy >, 997
- solve
 - GiNaC::matrix, 732
- sort
 - GiNaC::container< C >, 418
- sort_
 - GiNaC::container< C >, 416
- spinidx
 - GiNaC::spinidx, 973
- spinor_metric
 - GiNaC, 247
- split_ex_to_pair
 - GiNaC::add, 293
 - GiNaC::expairseq, 536
 - GiNaC::mul, 765
- spm
 - GiNaC::scalar_products, 965
- spmap
 - GiNaC, 81
- spmapkey
 - GiNaC::spmapkey, 983
- sprem
 - GiNaC, 192
- sqrfree
 - GiNaC, 199
- sqrfree_parfrac
 - GiNaC, 199
- sqrfree_yun
 - GiNaC, 198
- sqrt
 - GiNaC, 218, 234
- sr_gcd
 - GiNaC, 194
- STATISTICS
 - normal.cpp, 1187
- std, 274
 - swap, 275
- std::equal_to< GiNaC::ex >, 482
 - operator(), 483

- std::hash< GiNaC::ex >, 641
 - operator(), 642
- std::less< GiNaC::ptr< T > >, 711
 - operator(), 711
- std::less< ptr< T > >
 - GiNaC::ptr< T >, 931
- step
 - GiNaC, 220
 - GiNaC::numeric, 833
- step_conjugate
 - GiNaC, 131
- step_eval
 - GiNaC, 130
- step_evalf
 - GiNaC, 130
- step_imag_part
 - GiNaC, 131
- step_real_part
 - GiNaC, 131
- step_series
 - GiNaC, 131
- STLT
 - GiNaC::container< C >, 410
 - GiNaC::container_storage< C >, 421
- store_remember_table
 - GiNaC::function, 579
- struct_compare
 - GiNaC::compare_all_equal< T >, 380
 - GiNaC::compare_bitwise< T >, 381
 - GiNaC::compare_std_less< T >, 382
- struct_is_equal
 - GiNaC::compare_all_equal< T >, 380
 - GiNaC::compare_bitwise< T >, 381
 - GiNaC::compare_std_less< T >, 382
- structure
 - GiNaC::structure< T, ComparisonPolicy >, 989
- structure.h, 1210
- sub
 - GiNaC::matrix, 727
 - GiNaC::numeric, 829
- sub_dyn
 - GiNaC::numeric, 831
- sub_matrix
 - GiNaC, 183
- subs
 - GiNaC, 110
 - GiNaC::basic, 323
 - GiNaC::clifford, 363
 - GiNaC::container< C >, 413
 - GiNaC::ex, 500, 501
 - GiNaC::expairseq, 533
 - GiNaC::Gt, 630
 - GiNaC::idx, 647
 - GiNaC::matrix, 723
 - GiNaC::numeric, 825
 - GiNaC::power, 880
 - GiNaC::pseries, 918
 - GiNaC::relational, 949
 - GiNaC::structure< T, ComparisonPolicy >, 993
 - GiNaC::symbol, 1031
- subs_algebraic
 - GiNaC::subs_options, 1021
- subs_no_pattern
 - GiNaC::subs_options, 1021
- subs_one_level
 - GiNaC::basic, 333
- subschildren
 - GiNaC::container< C >, 419
 - GiNaC::expairseq, 542
- subsvalue
 - GiNaC, 174
- successful_hits
 - GiNaC::remember_table_entry, 959
- sufficiently_accurate
 - GiNaC::lanczos_coeffs, 710
- suppress_branchcut
 - GiNaC::series_options, 966
- swap
 - GiNaC, 109, 114
 - GiNaC::ex, 493
 - GiNaC::expair, 522
 - GiNaC::ptr< T >, 930
 - std, 275
- swapped
 - GiNaC::sy_swap, 1023
- sy_anti
 - GiNaC, 243
- sy_cycl
 - GiNaC, 243, 244
- sy_is_less
 - GiNaC::sy_is_less, 1022
 - GiNaC::symmetry, 1046
- sy_none
 - GiNaC, 241, 242
- sy_swap
 - GiNaC::sy_swap, 1023
 - GiNaC::symmetry, 1046
- sy_symm
 - GiNaC, 242, 243
- sym
 - GiNaC::sym_desc, 1025
- sym_desc
 - GiNaC::sym_desc, 1024
- sym_desc_vec
 - GiNaC, 81
- symbol
 - GiNaC::info_flags, 672
 - GiNaC::symbol, 1030
- symbol.cpp, 1211
- symbol.h, 1211
- symbolic_matrix
 - GiNaC, 182, 186
- symbolset
 - GiNaC::symbolset, 1038
- symm
 - GiNaC, 240

- GiNaC::terminfo, 1072
- symmetric
 - GiNaC::symmetry, 1043
- symmetric2
 - GiNaC, 239
- symmetric3
 - GiNaC, 239
- symmetric4
 - GiNaC, 239
- symmetrize
 - GiNaC, 108, 241, 244
 - GiNaC::ex, 513
- symmetrize_cyclic
 - GiNaC, 108, 241, 244
 - GiNaC::ex, 514
- symmetry
 - GiNaC::symmetry, 1043
- symmetry.cpp, 1212
- symmetry.h, 1213
- symmetry_type
 - GiNaC::symmetry, 1043
- symminfo
 - GiNaC::symminfo, 1048
- symmterm
 - GiNaC::symminfo, 1048
- syms
 - factor.cpp, 1135
- syms_wox
 - factor.cpp, 1134
- symtree
 - GiNaC::function_options, 621
 - GiNaC::indexed, 671
- synthesize_func
 - GiNaC, 58
- table_size
 - GiNaC::remember_table, 956
- tan
 - GiNaC, 206
- tan_conjugate
 - GiNaC, 164
- tan_deriv
 - GiNaC, 163
- tan_eval
 - GiNaC, 163
- tan_evalf
 - GiNaC, 163
- tan_imag_part
 - GiNaC, 163
- tan_real_part
 - GiNaC, 163
- tan_series
 - GiNaC, 163
- tanh
 - GiNaC, 209
- tanh_conjugate
 - GiNaC, 171
- tanh_deriv
 - GiNaC, 170
- tanh_eval
 - GiNaC, 170
- tanh_evalf
 - GiNaC, 170
- tanh_imag_part
 - GiNaC, 171
- tanh_real_part
 - GiNaC, 171
- tanh_series
 - GiNaC, 171
- tau
 - GiNaC::Gt, 638
 - GiNaC::Kronecker_dz_kernel, 708
- tauToFundamental
 - GiNaC::Gt, 632
- tcoeff
 - GiNaC::ex, 503
- tensepsilon
 - GiNaC::tensepsilon, 1059
- tensor
 - GiNaC, 256
- tensor.cpp, 1215
- tensor.h, 1216
- terminfo
 - GiNaC::terminfo, 1072
- terms
 - GiNaC::Gt_detail::pathintegral_term_list, 851
- test
 - GiNaC::has_distance< T >, 640
- test_and_set_nparams
 - GiNaC::function_options, 615
- TEST_PERMUTATION
 - color.cpp, 1114
- TeX_name
 - GiNaC::constant, 404
 - GiNaC::function_options, 616
 - GiNaC::symbol, 1037
- tgamma
 - GiNaC, 212
- tgamma_conjugate
 - GiNaC, 148
- tgamma_deriv
 - GiNaC, 148
- tgamma_eval
 - GiNaC, 147
- tgamma_evalf
 - GiNaC, 147
- tgamma_series
 - GiNaC, 148
- thiscontainer
 - GiNaC::clifford, 361
 - GiNaC::color, 378, 379
 - GiNaC::container< C >, 415, 416
 - GiNaC::fderivative, 559
 - GiNaC::function, 575
 - GiNaC::indexed, 666
 - GiNaC::ncmul, 812
- thisexpairseq

- GiNaC::add, [292](#), [293](#)
 - GiNaC::expairseq, [535](#)
 - GiNaC::mul, [765](#)
- tinfo
 - GiNaC::return_type_t, [962](#)
- tinfo_key
 - GiNaC::registered_class_options, [942](#)
- to_cl_N
 - GiNaC::numeric, [839](#)
- to_double
 - GiNaC, [223](#)
 - GiNaC::numeric, [839](#)
- to_int
 - GiNaC, [223](#)
 - GiNaC::numeric, [838](#)
- to_integer
 - GiNaC::Gt, [637](#)
- to_long
 - GiNaC, [223](#)
 - GiNaC::numeric, [838](#)
- to_numeric
 - GiNaC::Gt, [636](#)
- to_polynomial
 - GiNaC, [106](#)
 - GiNaC::basic, [327](#)
 - GiNaC::ex, [506](#)
 - GiNaC::expairseq, [532](#)
 - GiNaC::numeric, [826](#)
 - GiNaC::power, [881](#)
 - GiNaC::structure< T, ComparisonPolicy >, [997](#)
 - GiNaC::symbol, [1032](#)
- to_rational
 - GiNaC, [105](#)
 - GiNaC::basic, [326](#)
 - GiNaC::ex, [505](#)
 - GiNaC::expairseq, [532](#)
 - GiNaC::numeric, [826](#)
 - GiNaC::power, [881](#)
 - GiNaC::structure< T, ComparisonPolicy >, [997](#)
 - GiNaC::symbol, [1032](#)
- toggle_dot
 - GiNaC::spinidx, [975](#)
- toggle_variance
 - GiNaC::varidx, [1091](#)
- toggle_variance_dot
 - GiNaC::spinidx, [975](#)
- too_late
 - GiNaC::_numeric_digits, [280](#)
- trace
 - GiNaC, [184](#)
 - GiNaC::matrix, [730](#)
- trace_string
 - GiNaC, [89](#)
- TransformExpressionWithCache
 - GiNaC::Gt_detail::TransformExpressionWithCache< Type >, [1074](#)
- transpose
 - GiNaC, [184](#)
- GiNaC::matrix, [729](#)
- traverse
 - GiNaC::ex, [502](#)
- traverse_postorder
 - GiNaC::ex, [502](#)
- traverse_preorder
 - GiNaC::ex, [501](#)
- tree
 - GiNaC, [232](#)
- tree_node
 - GiNaC::class_info< OPT >::tree_node, [1076](#)
- trig_info
 - GiNaC, [161](#)
- trivial
 - GiNaC::composition_generator, [384](#)
- tryfactsubs
 - GiNaC, [186](#)
- type
 - GiNaC::archive_node::property, [909](#)
 - GiNaC::archive_node::property_info, [910](#)
 - GiNaC::symmetry, [1047](#)
- uintvector
 - GiNaC, [81](#)
- unarch_map
 - GiNaC::unarchive_table_t, [1078](#)
- unarch_table_instance
 - GiNaC, [255](#)
- unarchive
 - GiNaC::archive_node, [309](#)
- unarchive_ex
 - GiNaC::archive, [298](#), [299](#)
- unarchive_map_t
 - GiNaC, [58](#)
- unarchive_table_t
 - GiNaC::unarchive_table_t, [1077](#)
- unatomize
 - GiNaC::archive, [301](#)
- unique
 - GiNaC::container< C >, [418](#)
- unique_
 - GiNaC::container< C >, [417](#), [419](#)
- unit
 - factor.cpp, [1134](#)
 - GiNaC::ex, [507](#)
- unit_matrix
 - GiNaC, [182](#), [185](#)
- unitcontprim
 - GiNaC::ex, [510](#)
- unlikely
 - compiler.h, [1116](#)
- unlink_ex
 - GiNaC, [113](#)
- unsignedvector
 - GiNaC, [81](#)
- USE_REMEMBER
 - normal.cpp, [1186](#)
- use_remember
 - GiNaC::function_options, [619](#)

- use_return_type
 - GiNaC::function_options, 618
- USE_SAME_DEGREE_FACTOR
 - factor.cpp, 1129
- use_sr_gcd
 - GiNaC::gcd_options, 623
- USE_TRIAL_DIVISION
 - normal.cpp, 1187
- usecount
 - GiNaC::unarchive_table_t, 1078
- used_indices
 - GiNaC::make_flat_inserter, 714
- user_defined_kernel
 - GiNaC::user_defined_kernel, 1083
- uses_Laurent_series
 - GiNaC::Eisenstein_h_kernel, 464
 - GiNaC::Eisenstein_kernel, 473
 - GiNaC::integration_kernel, 688
 - GiNaC::modular_form_kernel, 750
 - GiNaC::user_defined_kernel, 1084
- utils.cpp, 1217
- utils.h, 1219
 - DEFAULT_COMPARE, 1221
 - DEFAULT_CTOR, 1221
 - DEFAULT_PRINT, 1221
 - DEFAULT_PRINT_LATEX, 1221
- utils_multi_iterator.h, 1221
- v
 - GiNaC::basic_multi_iterator< T >, 347
 - GiNaC::sy_is_less, 1022
 - GiNaC::sy_swap, 1023
- v1
 - GiNaC::spmapkey, 983
- v2
 - GiNaC::spmapkey, 983
- v_internal
 - GiNaC::multi_iterator_shuffle< T >, 793
- v_orig
 - GiNaC::multi_iterator_shuffle< T >, 793
- validate
 - GiNaC::indexed, 669
 - GiNaC::symmetry, 1045
- value
 - factor.cpp, 1129
 - GiNaC::archive_node::property, 909
 - GiNaC::composition_generator::coolmulti::element, 475
 - GiNaC::has_distance< T >, 640
 - GiNaC::idx, 652
 - GiNaC::numeric, 842
- value_type
 - GiNaC::const_iterator, 386
 - GiNaC::const_postorder_iterator, 390
 - GiNaC::const_preorder_iterator, 394
- var
 - GiNaC::pseries, 925
- varidx
 - GiNaC::varidx, 1089
- version.h, 1223
 - GINAC_LT_AGE, 1224
 - GINAC_LT_CURRENT, 1224
 - GINAC_LT_REVISION, 1224
 - GINACLIB_ARCHIVE_AGE, 1224
 - GINACLIB_ARCHIVE_VERSION, 1224
 - GINACLIB_MAJOR_VERSION, 1224
 - GINACLIB_MICRO_VERSION, 1224
 - GINACLIB_MINOR_VERSION, 1224
 - GINACLIB_STR, 1224
 - GINACLIB_STR_HELPER, 1224
 - GINACLIB_VERSION, 1224
- version_major
 - GiNaC, 258
- version_micro
 - GiNaC, 258
- version_minor
 - GiNaC, 258
- vn
 - factor.cpp, 1135
- vnlst
 - factor.cpp, 1135
- wild
 - GiNaC, 255
- wildcard
 - GiNaC::wildcard, 1097
- wildcard.cpp, 1225
- wildcard.h, 1225
- write_real_float
 - GiNaC, 203
- write_unsigned
 - GiNaC, 83
- x
 - factor.cpp, 1133
 - GiNaC::basic_partition_generator::mpartition2, 752
 - GiNaC::Ebar_kernel, 457
 - GiNaC::ELi_kernel, 482
 - GiNaC::integral, 681
 - GiNaC::user_defined_kernel, 1084
 - integration_kernel.cpp, 1176
- y
 - GiNaC::Ebar_kernel, 457
 - GiNaC::ELi_kernel, 482
- yes_type
 - GiNaC::has_distance< T >, 640
- z
 - GiNaC::Gt, 638
 - GiNaC::Kronecker_dtau_kernel, 701
 - GiNaC::multiple_polylog_kernel, 802
- z_j
 - GiNaC::Kronecker_dz_kernel, 708
- zeta
 - GiNaC, 141, 211
- zeta1_deriv

- GiNaC, [156](#)
- zeta1_eval
 - GiNaC, [155](#)
- zeta1_evalf
 - GiNaC, [155](#)
- zeta1_print_latex
 - GiNaC, [156](#)
- zeta2_deriv
 - GiNaC, [156](#)
- zeta2_eval
 - GiNaC, [156](#)
- zeta2_evalf
 - GiNaC, [156](#)
- zeta2_print_latex
 - GiNaC, [156](#)
- zetaderiv_deriv
 - GiNaC, [136](#)
- zetaderiv_eval
 - GiNaC, [135](#)
- zi
 - GiNaC::Gt::kernel, [694](#)
- zisToFundamental
 - GiNaC::Gt, [631](#)