Needs["SymbolicC`"]

|  |  |
| --- | --- |
| SymbolicC Simplified | mbool x,y; |
| enum[data,  val1 = -1,  val2  ] enum[data,{}] | enum data{  val1 = -1,  val2  }; |
| cast[CPointerType[double], x] | (double\*) x; |
| struct[point,  {double, x},  {double, y}  ]  Struct[point,{ {double, x} }] | struct point  {  double x;  double y;  } |
| union[data, {  {int, ival},  {double, dval} }] | union data{  int ival;  double dval;  } |
| CTypedef[{const, int}, {myint, test2}] | typedef const int myint, test2; |
| CSizeOf | sizeof[test] |
| CPointerMember | test->test2 |
| CAddress | (value)& |
| CDereference | $(x) |
| CPointerMember[obj, length] Rule[t\_, p\_] -> CPointerMember[t,p]; |  |
| CArray {} | {} |
| CString CParentheses | String[“test”] |
| CArary, CInclude, CConstant, CParentheses, CDefine, CError, Cline, CPragma, CUndef | {} $include |

{HoldPattern[a\_ <> b\_] -> test[a, b]}