Documentation for P4

Overall, in this project, I changed ast.java, Sym.java, test.crrt, SymTable.java and nameErrors.crrt. To invoke my project, first type “make”. Then type “make test”. If there are name errors in the program, you will see those errors on the screen. If not, the program will call unparsed, you will see the result in the corresponding file.

ast.java:

Sym nameAnalyze(Symtable table) for every expression node.

The table is the Symtable for this whole program. It for each declNode, it will do name analysis on current table, and sometimes add a sym or a scope to the table. Sometimes remove a scope. The table was passed to the children nodes. And because we need to deal with DotAcessNode, which can be recursively defined, I make the function return a sym, which is the sym this id points to. So that I can tell that if the LHS is a struct, and store information like linenum and charnum in it.

For IdNode, I add a link to its declaration sym if there is no error.

Void nameAnalyze(Symtable table) for every node other than expression node.

For these nodes, because they won’t be used after their name analysis, there is no need to return their information. In this function, I just check all errors.

Sym.java:

I created three subclass for Sym, varSym for varDecl, funcSym for funcDecl, and structDeclSym for structDecl. In funcDecl, I added a list to store information for formals. In structDeclSym, I added a SymTable for it to store all declarations in its body.

I also add many new functions to help name analysis

SymTable.java:

I added a field called isError. Because in specifications, it says if there is no error in name analysis, call unparsed. **So I presume that if there are errors, I shouldn’t call unparsed**. So I need a field to know if there is any error.

test.crrt:

A Carrot program that thoroughly tests every possible error and can pass the name analysis.

nameErrors:

A Carrot program that thoroughly tests every possible error. It will cause all kinds of errors.