

Introduction

So far, we have written a scanner for our mad squirrel battle language, created a symbol table to store the variables and constants, and written a parser for constructing a parse tree from a language file. In this lab, we are going to use the parse tree to fill out type information in the symbol table.

Your Mission

This assignment has three parts:

A. Fix any remaining problems from Projects 1 and 2
(40 pts)

B. Write an "insertTypes" function.
(40 pts)

Currently, your scanner can insert type information only for constants. That is because we cannot tell the type of a variable from its name. This means that computing the correct type for a variable can only be done in the parser, which understands the relationship between the two parts of a declaration: the type name (e.g. TREE) and the identifier (e.g. myTree).

To implement this, write a function, "InsertTypes" that takes one parameter -- a parse node representing the root of the parse tree -- and descends the tree. Whenever it encounters an "INT", "TREE", or "DIR" declaration node, it should examine the argument node (which identifies the symbol table index of the variable being declared) and update it's "type" field.

HINT: It might help to add a "type" variable to your parse node class. Use the "type" field to determine which kind of node you are currently looking at as you descend the tree. If you encounter the right kind of node, you can cast the "parsenode pointer" to the correct type of pointer for that node.

D. Glitter
(20 pts)

Glitter points are assigned for extending the project assignment in a creative way.

Some suggestions for ways to earn glitter points:

1. Implement type checking -- write a function that descends the parse tree and makes sure that both arguments of an assignment, an "==" comparison, a "<" comparison, etc., are the same type. **(A correct implementation of type checking is worth the full 20 points!)**
2. Add internal documentation
Commenting each class and function in your code is a way to earn at least 15/20 glitter points.
3. Add external documentation
Writing a document that describes the mad squirrel grammar, gives a storyline, and also explains how to install, compile, and run your program is worth at least 10/20 glitter points.
4. Your own, original, idea.
Coming up with your own creative glitter idea is worth more than simply selecting one of these alternatives (except for type checking, obviously), as long as your idea makes a significant and creative contribution to how the project works, looks, or behaves.

As usual, submission is through the course interface at <http://narnia.homeunix.com/~robert/submit>