Name: Sean Rettig
Date: 2015-03-01

Milestone 4 Report

Specification (what do you think the purpose of this milestone is)

The purpose of this milestone was to continue learning the process of designing and implementing translators. In this milestone, we used the output of our parser from the previous milestone to create a parse tree that we can traverse in an order that allows us to produce suitable gforth code, and also translate the tokens of the tree into their corresponding gforth operations.

Processing (how did you go about solving the problem)

I started by attempting to create a generalized parse tree like the ones we used near the start of class. The output of my parser from the previous milestone produced a tree, but it was created directly from the grammar and as such contained many nonterminal nodes, making it unsuitable for direct traversal. I modified the output of the parser to allow it to produce a tree with only the terminal symbols, and from there created a simple script that removed the parentheses and finally produced a classic parse tree like the ones we worked with near the start of this class, where each operator has its operands as its children. I then simply performed a post-order traversal of this tree to produce the proper postfix format required for gforth, and replaced the tokens at each node with valid gforth symbols (e.g. replacing “%” tokens with “mod”) before outputting them.

Testing Requirement (how did you test for correctness)

I have 6 test files which are intended to pass, i.e. produce valid gforth code. Each file

Name: Sean Rettig
Date: 2015-03-01

tests different features, such as integer and real arithmetic, automatic int to real stack moving, comparisons, nested statements, multiple statements, trigonometric functions, boolean logic, and string concatenation. I feel that these tests are decently comprehensive. They all pass!

Retrospective (what did <u>you</u> learn in this milestone)

This milestone didn't involve too many new concepts, unlike the previous milestone. However, making sure the parse tree was in the correct format and that all of the tokens were interpreted properly was a bit frustrating until I managed to get everything working correctly.