CS 152 Project Phase 3 Lab Report

Fall 2018 - Apollo Truong 861168740, Sidney Son 861214496

Motivation

In this phase we are creating an intermediate code generator which takes the “MINI-L” programming language and generates “MIL” code which can be executed. We do this by using the lex files created in the first phase of lab and combining it with the bison file created in the second phase. This is a one-pass code generation which is created by the bison file and creates an executable .mil file. This is then executed using an interpreted called mil_run.

Approach

We broke our code down into three sections. The first being the function name itself which only states the “func name” and “end func”. The second part of our code is the parameters and local variables that are declared before any logic is done. These are contained in BEGINPARAMS - ENDPARAMS, and BEGINLOCALS - ENDLOCALS For example, in test.min the local variables were n, i, j, k, and t. These variables could only be of type integer or an array of integers. Finally we have the statements which are contained in BEGINBODY to ENDBODY. These statements contained most of the logic of the code.

In our approach we separated the functions, variables, and statements in separate string vectors. By the end of the code, we iterate through each vector and print out the strings contained at each index of the vector. In order to fill the vectors, we used the grammar given and instead of printing out the syntax tree (like in phase 2), we contain the information a string which is then pushed into its appropriate vector.

Roles

Both partners contributed equally to the project. Sidney emphasized rebuilding the grammar in order to get the MIL code format. He also created most of the code for printing out the declaration portion of the MIL code. Apollo created most of the code for the statement portion of the MIL code. Both partners worked on debugging problems that cause the code to not compile.

Problems

In the beginning of the project, we had problems understanding the MIL code syntax. The largest problem we encountered was getting the code to compile. Most of this was due to small bugs like externally declaring FILE * yyin or not including certain headers. Our code still contains many bugs which we were unable to fix. The main bug is the output syntax and when printing multiple variables contained in a single line. We believe the variable errors are coming from when the grammar is being parsed and strings begin concatenating with each other when entering a loop. We were also unable to finish completing the do while and while loops. This is mainly due to problems placing labels and getting information from a recursive call.