Maria Kalikas
Compiler Design
April 4, 2017

Homework 3 Paper

The goal of the classes defined in this segment of the compiler is to evaluate the source code of a program through the use of a lexer and then generate an abstract syntax tree from the parser. The classes defined take a string of tokens from the lexer and evaluate them using the parser class, semantics class, and the expr class. The parser class includes functions that are set to match each production defined by the grammar. The goal of this class is to create an abstract syntax tree given the sequence of tokens lexed by the lexer class. The semantics class will integrate with the type check system and the parser to determine that the types of the expressions result as intended and to match the sequence of tokens with a production.

This homework contains the implementation of a lexer. I created a lexer class which includes a function to move through input by evaluating each character. Each character that is being evaluated is a token defined by the language. The LookAhead function function uses a pointer and increments it to look at the next character until the end of input reaches an end. A switch statement is used to evaluate each token. When each token is evaluated, the consume member function is called to increment the character pointer to the next token. This is intended to move the character to the end of the line and input file, as well as to evaluate the cases where multiple tokens are used together in an operation. This idea is used in the cases where a not equal, less than or equal, negative number, and other operations that involve multiple tokens inputted in a particular sequence to relate to the given operation.

I also created a token class which includes the kind of token defined by the grammar. The different types of tokens included in this language are punctuation and operation tokens. These include the symbols used for relational expressions, equality, logical expressions, and conditional expressions.