**CPSC 323 Documentation**

**Assignment 1: Lexical Analyzer**

**Randy Le**

**Alex Ma**

# Problem Statement

The goal for this assignment was to create a lexer that returns a token when it is needed. The lexer returns a record that has tokens in one field and the values of that token (instance of a token) in the other. One of the main requirements of this assignment is that we had to construct a FSM. Our main reads in a file and outputs the tokens and values. The output file has Tokens in the first column and lexeme in the second column.

# How to use your program

The program can either be ran by the executable or by directly compiling in visual studio. The files provided are compatible on Visual Studio 2017 as tested in the lab. Since this is the first part of the compiler, only the lexer is functional. It is highly recommended to test the program using Visual Studio 2017. Make sure the source code that is being read is in the same directory as the Main.cpp. While running the program, it will ask you to input a file. While inputting a file, make sure include the file extension.

>> Enter a file name: file\_name.txt

# Design of your program

We created a class called LexerAnalyzer that helps us parse the input and determines if the given text is a comment, integer, float, keyword, separator, operator or identifier. We read in the input from the file and then output the data correctly with the tokens and lexeme. How it works is we call the lexer for a token then print the token and lexeme. This is done using a while loop. The while loop will keep running until it reaches the end of the file. Inside the while loop is a switch statement that guides the program to the right state. Each case in the switch statement represents a state. Each state has its own method and does their action when called. After finishing in a state method, it will then change the state depending on the character input read in the file and the current state it is in by using a 2D multidimensional array.

# Any Limitation

*Size of identifier is limited*

*Size of Operator is limited*

*Size of Separator is limited*

# Any shortcomings

*None*