ASSIGNMENT 1:

LEXER

CPSC323 TuTh 4-5:15pm

Professor: Dr. Choi, PhD

Due: 3 March 2015

Program by:

Art Grichine

Zeed Jarrah

**CS323 Documentation**

**Problem Statement**

<write the problem statement here. You can mostly get it from the assignment itself>

Write a lexer (lexical analyzer). A final state machine must be implemented for identifier, integer, and real tokens (the rest can be written ad-hoc). A major component of this assignment is to write a procedure (Function) – lexer() – that returns a token when it is needed. The lexer() should return a record, one field for the token and another field for the lexeme, i.e. the instance of a token. The main program should test the lexer, i.e. the program should read a file containing the source code of Rat15S to generate tokens and write out the results to a file.

**How to use your program**

<write detailed steps how to execute your program>

The data.txt file, which contains the input script, must be in the same directory as the Lexer.py python script. To run the program enter the directory of the data file and python script and type: $ python3.4 Lexer.py

Once the program is opened, it will prompt the user for input:

‘Enter file you would like to open (type “quit” to exit): ’

You may enter the name of a file you would like to input such as: testcase1.txt

DO NOT use quotations around the filename.

If a file is not recognized the program will respond with:

‘Your file was not found!’

Then it will prompt you to enter another filename. To quit the program type: quit into the terminal and the program will exit.

Successful runs will output to the terminal as well as be written into a file with the name of the input (user defined) file and the extension “.RAT”.

Once the program is called, it will run on a loop until the user enters: quit

**Design of your program**

<write major components of your program. Also, data structures you are utilizing, particular algorithms you have chosen, etc.>

**Any Limitation**

<Limit your program due to resource limitations, such as Maximum number of lines in the source code, size of identifier, integer, etc. SAY ‘None’ if there is no limitation>

None

**Any shortcomings**

<Anything you could NOT implement although that is required by the assignment. SAY ‘None’ if there is no shortcoming>

None