**CS323 Documentation**

About 2 pages

1. **Problem Statement**

* *Write a simple lexer or lexical analyzer that returns the appropriate token when asked for*
* *Complete necessary Finite-State Machines (FSM)s*
* *Return the output of the token and the value of the token*
* *Print out both tokens and lexemes*
* *Read .txt files containing source code of RAT23S to create tokens and grab the results of the file*

1. **How to use your program**

Locate executable files in the submission folder.

Make sure to have the .txt files and project1.exe in the same file.

**Step1)** Open a terminal where the project folder is

**Step2)** Assemble the program by typing ./project.sh

**Step3)** Open the generated output file ./a.out

**Step4)** Specify input text file (test case file) “test\_case1.txt”

**OR**

**Step1)** Double click on project1.exe

**Step2)** specify the input file (test case file)

**Step3)** specify the output file

1. **Design of your program**
2. ->Program uses C++ 17

->Main files are Lex.cpp main.cpp Lex.h project.sh

->Lex.cpp provides the definitions of the following 16 functions:

-3 Functions for checking separators, operators, ch to column #

-1 Function char to col to do the conversion

-3 Functions for 3 Finite-State Machines

-2 Strings for token and lexeme

-1 Char for checking each char

-1 Function for getting tokens from the lexer

-1 Function to print the output

-4 Functions for mutators and accessors

->Lex.h holds the declarations of these functions along with the private variables

->Build a Lexer

->Find the Regular Expression RE for identifier, integer and real.

->Apply the same procedure for each Finite-State Machine

->Start with Integer FSM

->Convert from Regular Expression to Non-Deterministic using Thompson Method

->Create Construction of Deterministic DFSM Table from Non-Deterministic Finite State Machine using Subset Method

1. **Any Limitation**

*None*

1. **Any shortcomings**

*None*