

**FACULTY OF INFORMATION TECHNOLOGY**

**BACHELOR OF SCIENCE IN INFORMATICS AND COMPUTER SCIENCE**

**ICS 4211: COMPILER CONSTRUCTION**

**LAB 2**

**OUTPUT DESCRIPTION**

**STUDENT NAME: Vikiru Allan Odindo**

**STUDENT NUMBER: 098587**

**CLASS: BICS 4A**

**Link to Project:** <https://github.com/AllanVikiru/98587_CCProjects/tree/lex>

In lab 1, a simple lexer was designed in C++ and Flex which takes in program code from a file, breaks it down into tokens and outputs the tokens and their respective roles.

A sample input code would be:

float c = a + (b \* 4);

the output would return:

float is a keyword

c is an identifier

= is an operator

a is an identifier

+ is an operator

( is a separator

b is an identifier

\* is an operator

4 is a number

) is a separator

; is a separator

The lexer works in that tokens have their various roles defined within the program for example int and void defined as keywords and + and \* as operators. Input code is then passed through a function containing a function containing a read-only iterator for string values (sregex\_iterator); which checks for a match between tokens found in the input and those defined within the various roles. The matches are then populated in an array then presented as output.

The following table represents the role definitions of tokens as used by the program:

|  |  |
| --- | --- |
| **Role** | **Tokens** |
| Keywords | "if","else","do","while","switch","case","break","continue",  "default","for","auto","const","int","float","double",  "string","void","char","char\*","bool","return",  "cin","cout","endl" |
| Operators | "+", "-", "\*", "/",">","<","=","++","==","--", "&&", "||", "!=" |
| Separators | " ", ";", "(", ")","{", "}", "<",">","[","]","&",":" |
| Identifiers | A sequence of alphabetic characters from a to z including capital letters |
| Numbers | A sequence of numeric characters from 0 to 9 |