CSE420 (Compiler) Lab, Summer2017

Assignment 4

**Problem Description:**

In this assignment, you will need to do the same thing as what you have done in assignment 1 (Lexical Analyzer) using jFlex. But here it will scan a input.java file containing code of a valid java program and distinguish the token. Similar to assignment 1, you will identify all the numerical values, identifiers, keywords, math operators, logical operators and others [distinct]. See the example for more details. In this assignment, there may have a space or not after each keyword or token. You will need to write, in fact you must write a lex file in jFlex that will do this work. Basically, we have seen these in class about the working procedure of jFlex. For simplicity, we will consider only simple set of instructions. Also you will need to scan the input.java file sequentially and output the tokens sequentially in all categories. Everybody must know how to compile and run the lex file using jFlex. See the following:-

**Keywords:** int, float, double, if, else, for, return [we will consider 'include' and 'stdio.h' as exceptional keywords]

**Functions:** main, printf, scanf

**Identifiers:** letter|(letter|digit)\*

**Strings:** Everything surrounded by double quote [" "].

**Numeric values:** 25, 25.25

**Math Operators:** +, -, x, /, =, ++, --

**Logical Operators:** >,<, >=, <=, ==

**Brackets:** (, {,}, )

**Others:** seek only once [if more than once, print only once]

**Hints:** Read the manual.pdf file of jFlex carefully and explore java.lex file in "examples\java\" directory.

**Example:** input.c file will look like the following:

#include <stdio.h>

int main()

{

int ab1, ac;

double d = 5.5;

ab1 = 3;

ac = 2;

if(ab1>ac)

{

d = d+10.4;

}

else

{

d = d-2.5;

}

printf("Value of d = %f",d);

return 0;

}

**Output:**

**Keywords:** include, stdio.h, int, int, double, if, else, return

**Functions:** main, printf

**Identifiers:** ab1, ac, d

**Strings: "**Value of d = %f"

**Numeric Values:** 5.5, 3, 2, 10.4, 2.5, 0

**Math Operators:** =, =, =, =, +, =, -

**Logical Operators:** <, >, >

**Brackets:** (, ), {, (, ), {, }, {, }, }

**Others:** #, ,, ;