

Lab Assignment 3- Lexical Analysis (LEX/Flex)

Installing Flex: `sudo apt-get update`
`sudo apt-get install flex`

Editor for writing Lex/Flex program: Use any text editor

How to compile/execute: Check the lecture notes shared related to LEX/Flex tool.

SECTION 1

Q 1.1 Write a LEX/Flex program that recognizes binary strings containing odd number of 0's.

Q 1.2 [Optional] Write a LEX/Flex program that recognizes binary strings containing odd number of 0's and even number of 1's.

Q 1.3 Write a LEX/Flex program that recognizes binary strings whose integer equivalent is divisible by 3.

SECTION 2

Q2. We had discussed about the lexical analyzer generator Lex/ Flex. Consider the example grammar for branching statements discussed in the class given below:

```
stmt  →  if expr then stmt
        |  if expr then stmt else stmt
        |  ε
expr   →  term relop term
        |  term
term   →  id
        |  number
```

The patterns for the tokens in the language are described below (that we have also considered in the previous assignment):

```
digit  →  [0-9]
digits →  digit+
number →  digits ( . digits)? ( E [+ -]? digits )?
letter →  [A-Za-z]
id      →  letter ( letter | digit )*
if      →  if
then    →  then
else    →  else
relop   →  < | > | <= | >= | = | <>
```

Q 2.1. Write a Lex/Flex program to describe the tokens of the above grammar, and generate a lexical analyzer using the Lex/Flex tool.

Q 2.2. Test the lexical analyzer with some input strings (You should show and explain the output of the lexical analyzer for the considered examples).

SECTION 3

Q 3. Construct a lexical analyzer for the following simple “C” like language using the Lex/Flex tool.

1. **Data Type** : integer (INT/int), floating point (FLOAT/float)
2. **Condition constructs**: if
3. **Loop Constructs**: for, while
4. **Input / Output Constructs**:
 - a. read(x) - Read into variable x
 - b. print(x) - Write variable x to output
5. Relational operators, assignment and arithmetic operators
6. Only function is **main()**, there is no other function.

You may test it using the below example:

Example Input:

```
main ()
{
    INT i=0;
    INT sum=0;
    INT count;

    read(count);
    for (i=0;i<10;i++)
    {
        read(x);
        sum+=x;
    }

    print(sum);
}
```
