

**S. B. JAIN INSTITUTE OF TECHNOLOGY, MANAGEMENT & RESEARCH, NAGPUR.**

Practical No. 02

**Aim:** Implementation of LEX (Compiler Writing Tool) using flex window to Identify -

1. LEX Program to find out upper case and lower case of any string.
2. LEX program to find out keyword, identifier, number, special character from given string.
3. LEX program to find out Vowels and Consonants from given string.

# Name of Student: Md. Nawaid Sheikh

# Roll No: CS21040

**Semester/Year:** 6th Semester/3rd Year

**Academic Session:** 2023-24 **Date of Performance:**

**Date of Submission:**

**AIM:** Implementation of LEX (Compiler Writing Tool) using flex window to Identify -

1. LEX Program to find out upper case and lower case of any string.
2. LEX program to find out keyword, identifier, number, special character from given string. **(f)**

LEX program to find out Vowels and Consonants from given string.

## OBJECTIVE / EXPECTED LEARNING OUTCOME:

The objectives and expected learning outcome of this practical are:

* + To illustrate the use of Compiler writing Tool (LEX)
  + To understand, how to demonstrate the LEX program.

## HARDWARE AND SOFTWARE REQUIRMENTS:

### Hardware Requirement:

* + Processor: Dual Core
  + RAM: 1GB
  + Hard Disk Drive: > 80 GB **Software Requirement:**
  + FLEX Software

## THEORY:

**LEX:**

* + It stands for Lexical Analyzer Generator.
  + LEX is a tool for generating lexical analyzer or scanners.
  + Scanners are programs that recognize lexical patterns in text. These lexical patterns are defined in a particular syntax called regular expression.

**LEX Skeleton:** LEX skeleton is given below:

%{

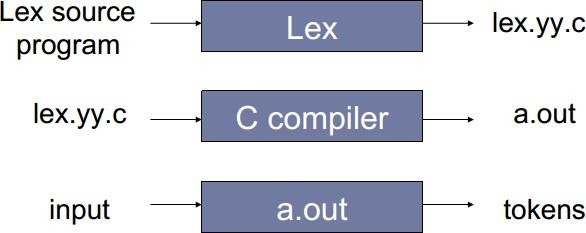
Declaration Section

}%

%%

Rule Section

%%



User Code (C/C++ Language)

### Procedure to Execute LEX Program:

**Diagrammatic Representation of LEX:**

**LEX Functions:** yylex( ): It is the starting point of lex from which scanning of source program starts. yywrap( ): It is called when EOF is encounter, indicate end of parsing by lexical analyzer. yymore( ): It append next string match to current content of yytext. yyless ( ): It removes from yytext first n char.

### LEX Variables:

yytext: Text match most recently is stored.

yyleng: Number of char in text most recently match. yylval: Associated val of current token.

yyin: This points to current file parsed by lexer. yyout: This point to location where output of lexer will be written.

### Procedure to Execute LEX Program:

These are the following sequence of commands to be executed to run a LEX programs:

* vi f1.lex
* lex f1.lex
* cc lex.yy.c
* ./a.out

**AIM: (a)** LEX Program to find out Upper Case and Lower Case of any string.

**CODE:** %{



#include<stdio.h> int up=0; int low=0;

%}

%%

[A-Z]{printf("Uppercase : \t ");up++;}

[A-Z]{printf("Lowercase : \t ");low++;}

%%

int yywrap()

{ return 1; }

main() {

printf("Enter the string : "); yylex();

printf("uppercase = %d and Lowercase = %d ",up,low); }

## OUTPUT:

**AIM: (b)** LEX Program to find out Keyword, Identifier, Number, Special character from given string.

**CODE:** %{

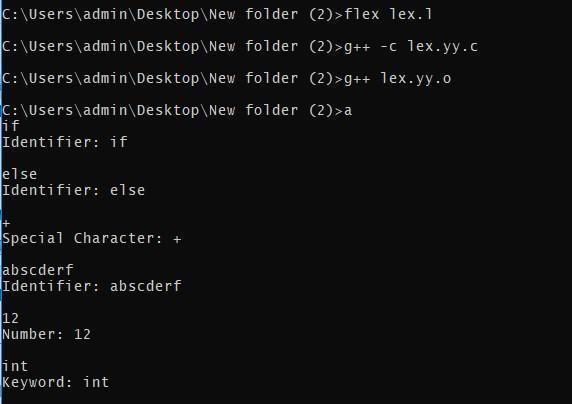
#include <stdio.h>

%}

%%

int|float|char { printf("Keyword: %s\n", yytext); }

[a-zA-Z\_][a-zA-Z0-9\_]\* { printf("Identifier: %s\n", yytext); } [0-9]+ { printf("Number: %s\n", yytext); }



. { printf("Special Character: %c\n", yytext[0]); }

%%

int yywrap() { return 1;

} int main() {

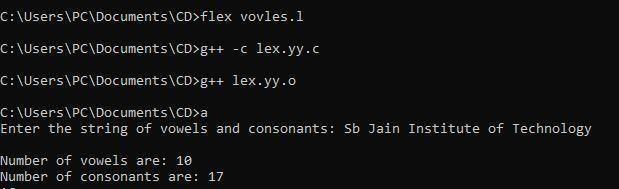
yylex(); return 0;

}

## OUTPUT:

**AIM: (c)** LEX Program to find out Vowels and Consonants from given string.

**CODE:** %{



int vow\_count=0; int const\_count =0;

%}

%%

[aeiouAEIOU] {vow\_count++;} [a-zA-Z] {const\_count++;}

%%

int yywrap(){} int main() {

printf("Enter the string of vowels and consonants:"); yylex();

printf("Number of vowels are: %d\n", vow\_count); printf("Number of consonants are: %d\n", const\_count); return 0;

}

# OUTPUT:

**CONCLUSION:** In this implementation, LEX (Flex) has been used to create lexical analyzers for different tasks. Each program uses regular expressions to define patterns and associated actions to be taken when these patterns are identified in the input string. The flexibility and expressiveness of LEX allow for the easy definition of rules for recognizing uppercase and lowercase letters, keywords, identifiers, numbers, special characters, vowels, and consonants. These programs serve as basic building blocks for more advanced lexical analysis in the context of compiler construction or text processing. The combination of regular expressions and corresponding actions in LEX makes it a powerful tool for recognizing and processing patterns in input text.

## DISCUSSION AND VIVA VOCE:

1. What is the difference between token and lexeme?
2. What is lexical analyzer?
3. Which compiler is used for lexical analyzer?
4. What is the output of Lexical analyzer?
5. What is LEX source Program?

## REFERENCE:

* + Lab Manual of Compiler Design (Institute of Aeronautical Engineering, Dundigal, Hyderabad)
  + https://en.wikipedia.org/wiki/Lex\_(software)