**6.Write a lex program to implement a simple calculator.**

# Program:

**calc. l (flex source program)**

%{

#include <stdlib.h> void yyerror(char \*); #include "calc.tab.h"

%}

%%

/\* variables \*/ [a-z] {

yylval = \*yytext - 'a'; return VARIABLE;

}

/\* integers \*/ [0-9]+ {

yylval = atoi(yytext); return INTEGER;

}

/\* operators

[-+()=/\*\n] { return \*yytext; }

/\* skip whitespace \*/ [ \t] ;

/\* anything else is an error \*/

. yyerror("invalid character");

%%

int yywrap(void) { return 1;

}

# calc.y (bison source program)

%token INTEGER VARIABLE

%left '+' '-'

%left '\*' '/'

%{

#include <stdio.h> void yyerror(char \*); int yylex(void);

int sym[26];

%}

%%

program:

statement:

program statement '\n'

|

;

expr:

%%

expr { printf("%d\n", $1); }

| VARIABLE '=' expr { sym[$1] = $3; }

;

INTEGER

| VARIABLE { $$ = sym[$1]; }

| expr '+' expr { $$ = $1 + $3; }

| expr '-' expr { $$ = $1 - $3; }

| expr '\*' expr { $$ = $1 \* $3; }

| expr '/' expr { $$ = $1 / $3; }

| '(' expr ')' { $$ = $2; }

;

void yyerror(char \*s) { fprintf(stderr, "%s\n", s);

}

int main(void) { yyparse(); return 1;

}

# Sample Output:

G:\lex>flex calc.l G:\lex>bison calc.y

G:\lex>gcc calc.tab.c lex.yy.c -o calc.exe

G:\lex>calc.exe 3\*(5+6)

33

x = 3 \* (4 + 5)

y = 5 x

27

y 5

x + 2\*y 37

G:\lex>