Ex No: 4
Date:

DESIGN A DESK CALCULATOR USING LEX TOOL

AIM:

To create a calculator that performs addition, subtraction, multiplication and division using lex tool.

ALGORITHM:

- In the headers section declare the variables that is used in the program including header files if necessary.
- In the definitions section assign symbols to the function/computations we use along with REGEX expressions.
- In the rules section assign dig() function to the dig variable declared.
- In the definition section increment the values accordingly to the arithmetic functions
respectively.
- In the user defined section convert the string into a number using atof() function.
- Define switch case for different computations.
- Define the main () and yywrap() function.

PROGRAM:

```
%{
int op = 0,i;
float a, b;
%}
dig [0-9]+|([0-9]*)"."([0-9]+)
add "+"
sub "-"
mul "*"
div "/"
pow "^"
ln \n
%%
{dig} {digi();}
{add} {op=1;}
{sub} {op=2;}
{mul} {op=3;}
{div} {op=4;}
{pow} {op=5;}
{ln} {printf("\n The Answer :%f\n\n",a);}
%%
digi(){
if(op==0)
a=atof(yytext);
else{
b=atof(yytext);
```

```
 switch(op){
 case 1:a=a+b;
 break;
 case 2:a=a-b;
 break;
 case 3:a=a*b;
 break;
 case 4:a=a/b;
 break;
 case 5:for(i=a;b>1;b--)
 a=a*i;
 break;
 }
 op=0; } }
 main(int argv,char *argc[])
 {
 yylex();}
 yywrap()
 {
 return 1;
}
```

OUTPUT:

```
[student@localhost ~]$ lex calci306.l
[student@localhost ~]$ cc lex.yy.c
calci306.l: In function ‘yylex’:
calci306.l:15:2: warning: implicit declaration of function ‘digi’; did you mean ‘div’?
 {dig} {digi();}
  ^~~~
  div
calci306.l: At top level:
calci306.l:24:1: warning: return type defaults to ‘int’ [-Wimplicit-int]
 {
 ^
calci306.l:55:1: warning: return type defaults to ‘int’ [-Wimplicit-int]
 {
 ^
calci306.l:60:1: warning: return type defaults to ‘int’ [-Wimplicit-int]
 {
 ^
[student@localhost ~]$ ./a.out
45/5

 The Answer :9.000000
```

RESULT: