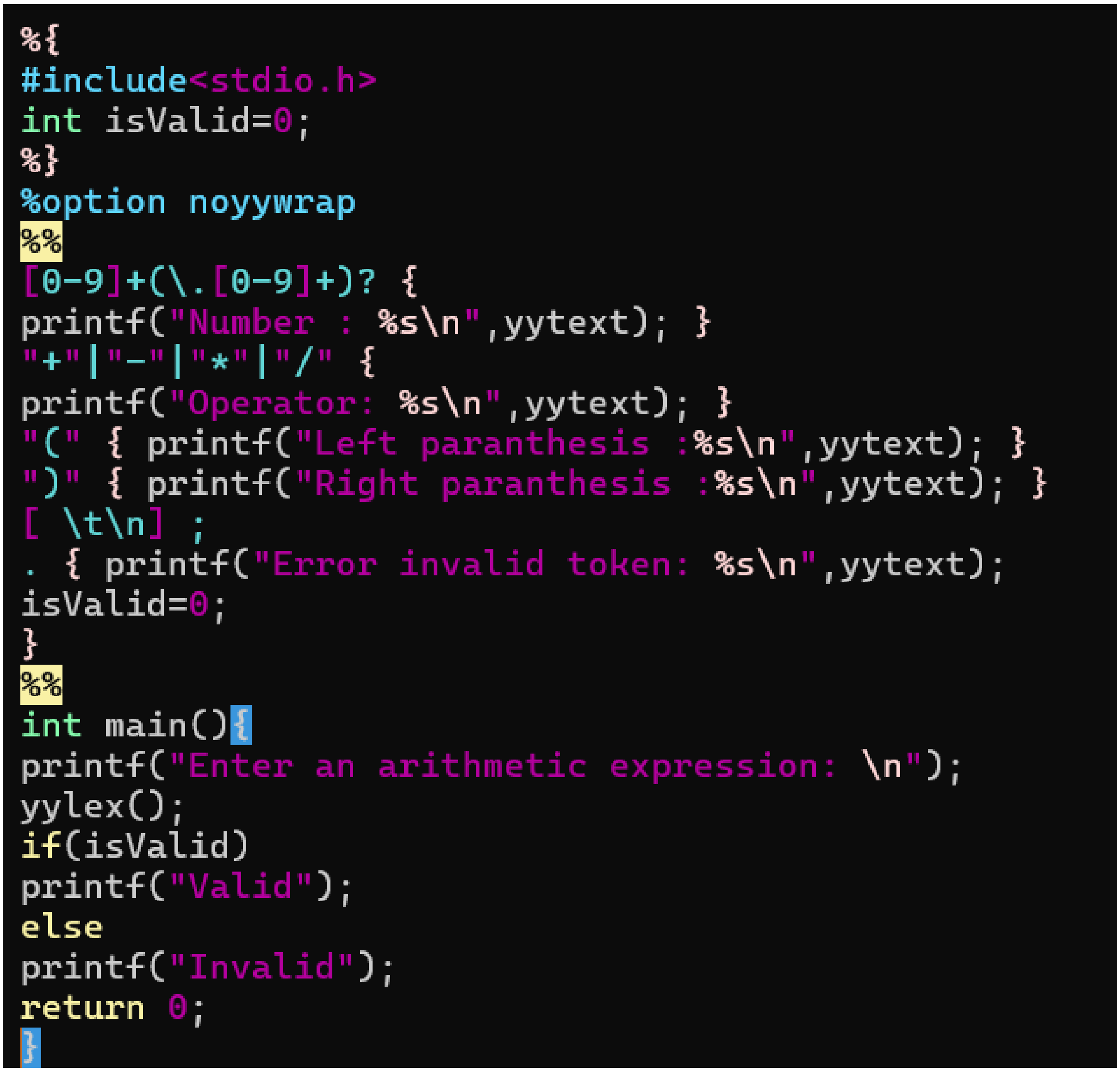
EXP NO-4

DESIGN AND IMPLEMENT A DESK CALCULATOR USING THE LEX TOOL

AIM:

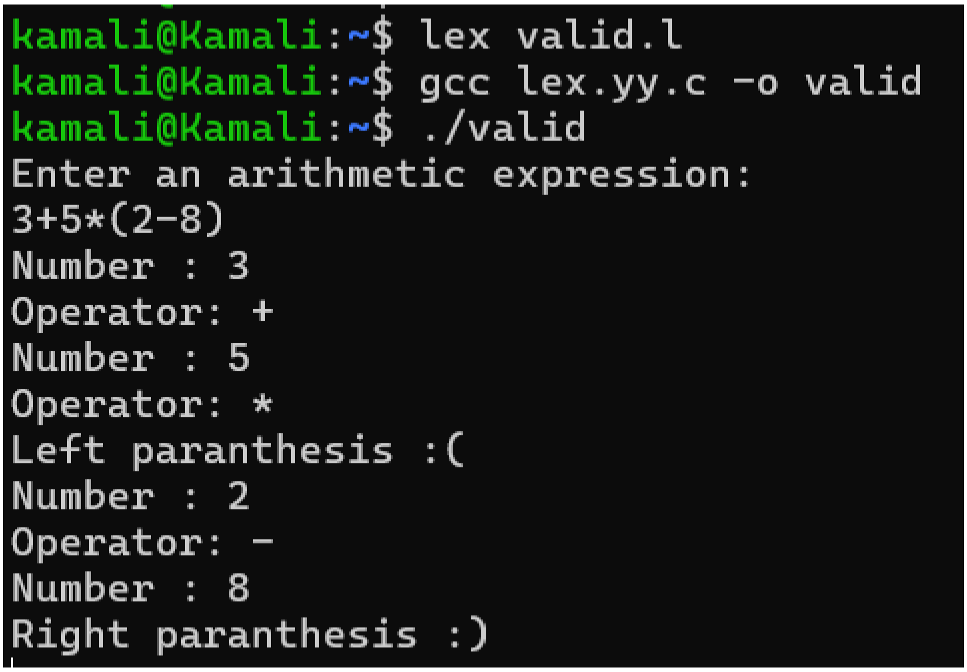
To design and implement a Desk Calculator using the LEX tool, which validates arithmetic expressions containing +, -, *, /, numbers, and parentheses. The program ensures that the expression follows correct arithmetic syntax rules.

PROGRAM: VALID.L



```
%{
#include<stdio.h>
int isValid=0;
%}
%option noyywrap
%%
[0-9]+(\.[0-9]+)? {
printf("Number : %s\n",yytext); }
"+"|"-"|"*"|"/" {
printf("Operator: %s\n",yytext); }
"(" { printf("Left paranthesis :%s\n",yytext); }
")" { printf("Right paranthesis :%s\n",yytext); }
[ \t\n] ;
. { printf("Error invalid token: %s\n",yytext);
isValid=0;
}
%%
int main(){
printf("Enter an arithmetic expression: \n");
yylex();
if(isValid)
printf("Valid");
else
printf("Invalid");
return 0;
}
```

OUTPUT



```
kamali@Kamali:~$ lex valid.l
kamali@Kamali:~$ gcc lex.yy.c -o valid
kamali@Kamali:~$ ./valid
Enter an arithmetic expression:
3+5*(2-8)
Number : 3
Operator: +
Number : 5
Operator: *
Left paranthesis :(
Number : 2
Operator: -
Number : 8
Right paranthesis :)
```

RESULT:

Thus the above program reads an arithmetic expression, tokenizes it using LEX rules, and validates the syntax by recognizing numbers, operators (+, -, *, /), and parentheses. If the expression is valid, it prints "Valid arithmetic expression." Otherwise, it detects and reports invalid tokens.

KAMALI K A -220701118