

**Ex No: 7**

**Date:**

## **EVALUATE EXPRESSION THAT TAKES DIGITS, \*, + USING LEX AND YACC**

**AIM:**

To perform arithmetic operations that takes digits, \*, + using lex and yacc.

**ALGORITHM:**

- Using the flex tool, create lex and yacc files.
- In the definition section of the lex file, declare the required header files along with an external integer variable yylval.
- In the rule section, if the regex pertains to digit convert it into integer and store yylval. Return the number.
- In the user definition section, define the function yywrap()
- In the definition section of the yacc file, declare the required header files along with the flag variables set to zero. Then define a token as number along with left as '+', '-',  
, 'or', '\*', '/', '%' or '(' )'
- In the rules section, create an arithmetic expression as E. Print the result and return zero.
- Define the following:
  - E: E '+' E (add)
  - E: E '-' E (sub)
  - E: E '\*' E (mul)
  - E: E '/' E (div)

If it is a single number, return the number.

- In driver code, get the input through yyparse(); which is also called as main function.
- Declare yyerror() to handle invalid expressions and exceptions.
- Build lex and yacc files and compile.

**PROGRAM:****evaluate.l:**

```
% {
#include<stdio.h>
#include "y.tab.h"
extern int yylval;
% }

%%
[0-9]+ {
    yylval=atoi(yytext);
    return NUMBER;
}
[\t] ;
[\n] return 0;
. return yytext[0];
%%
int yywrap()
{
return 1;
}
```

**evaluate.y:**

```
% {
    #include<stdio.h>
    int flag=0;
% }
%token NUMBER
%left '+' '-'
%left '*' '/' '%'
%left '(' ')'
%%

ArithmeticExpression: E{
    printf("\nResult=%d\n",$$);
    return 0;
}
E:E+'E' {$$=$1+$3;}
|E-'E' {$$=$1-$3;}
|E'*E' {$$=$1*$3;}
|E'/E' {$$=$1/$3;}
```

```
|E'% 'E { $$=$1%$3;}  
|'(E)' { $$=$2;}  
| NUMBER { $$=$1;}  
;  
%%
```

```
void main()  
{  
    printf("\nEnter Any Arithmetic Expression which can have operations  
Addition,      Subtraction, Multiplication, Divison, Modulus and Round  
brackets:\n");  
    yyparse();  
    if(flag==0)  
        printf("\nEnter arithmetic expression is Valid\n\n");  
  
}  
  
void yyerror()  
{  
    printf("\nEnter arithmetic expression is Invalid\n\n");  
    flag=1;  
}
```

**OUTPUT:**

```
[user@localhost ~]$ vi 281.l
[user@localhost ~]$ vi 281.y
[user@localhost ~]$ lex 281.l
[user@localhost ~]$ yacc -d 281.y
[user@localhost ~]$ cc lex.yy.c y.tab.h
[user@localhost ~]$ ./a.out

Enter Any Arithmetic Expression which can have operations Addition, Subtraction,
Multiplication, Division, Modulus and Round brackets:
5 + (3 * 2)

Result=11
Entered arithmetic expression is Valid
[user@localhost ~]$
```

**RESULT :**