**Practical - 5**

**Roll No - 18BCE107**

**Aim - To implement a calculator in YACC.**

**Code: -**

**calculator.l file:**

%{

    /\* Definition Section\*/

    /\*Lex Definition for Calculator\*/

    #include <stdio.h>

    #include "calculator.tab.h"

    extern int yylval;

%}

/\*Rule Section\*/

%%

[0-9]+ {yylval = atoi(yytext);

        return NUMBER;}

[\t] ;

[\n] return 0;

. return yytext[0];

%%

int yywrap()

{

    return 1;

}

**calculator.y file:**

/\*Parser Definition for Calculator\*/

%{

    /\*Definition Section\*/

    #include <stdio.h>

    int flag=0;

%}

/\*Tokens and Operator Precedence\*/

%token NUMBER

%left '+' '-'

%left '\*' '/' '%'

%left '(' ')'

/\*Rule Section\*/

%%

/\*Starting Symbol - Expression\*/

Expression:E{

                printf("\nResult = %d\n",$$);

                return 0;

            }

/\*Context Free Grammar\*/

E:E'+'E {$$=$1+$3;}

 |E'-'E {$$=$1-$3;}

 |E'\*'E {$$=$1\*$3;}

 |E'/'E {$$=$1/$3;}

 |E'%'E {$$=$1%$3;}

 |'-'E {$$=-$2;}

 |'('E')' {$$=$2;}

 | NUMBER {$$=$1;}

 ;

%%

//Driver Code to Accept user input

void main()

{

    while(1)

    {

        printf("\nEnter Expression\n");

        yyparse();

        if(!flag)

            printf("\nExpression Valid\n");

    }

}

void yyerror()

{

    printf("\nExpression Invalid\n");

    flag = 1;

}

**Output: -**

