Code:

lex.l (lex)

%{

/\* Definition section \*/

#include "y.tab.h"

%}

/\* Rule Section \*/

%%

[+-]?([0-9]\*[.])?[0-9]+ {

yylval = atoi(yytext);

return NUMBER;

}

[\t] ;

[\n] return 0;

[ ] {}

. return yytext[0];

%%

int yywrap()

{

return 1;

}  
  
  
-------------------------------------------------------------------------------------------------------------------------------

post.y (yacc)  
  
%{

/\* Definition section \*/

#include<stdio.h>

#include<stdlib.h>

int flag=0;

int yylex();

int yyerror(const char \*s);

%}

%token NUMBER

%left '+' '-'

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

%left '(' ')'

/\* Rule Section \*/

%%

ArithmeticExpression: E{

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

return 0;

};

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

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

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

|E'/'E {

if($3==0){

yyerror("Error:Division by zero");

$$=0;

}else{$$=$1/$3;}}

|E'%'E {$$=(int)$1%(int)$3;}

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

| NUMBER {$$=$1;}

;

%%

int main()

{

printf("\nEnter Any Arithmetic Expression which can have operations Addition, Subtraction,Multiplication, Division, Modulus and Round brackets\n");

yyparse();

if(flag==0)

printf("\nEntered arithmetic expression is Valid\n\n");

return 0;

}

int yyerror(const char \*s)

{

printf("\nEntered arithmetic expression is Invalid\n\n");

flag=1;

return 0;

}Output:

