**Experiment Number: 2a**

**Title**: Evaluation of Postfix Expression Using C.

**Problem statement**: Write a C program to evaluate a Postfix Expression, using strack.

**Algorithm:**

1) Create a stack to store operands (or values).

2) Scan the given expression and do following for every scanned element.

a) If the element is a number, push it into the stack

b) If the element is an operator, pop operands for the operator from stack. Evaluate the operator and push the result back to the stack

3) When the expression is ended, the number in the stack is the final answer.

**Program:**

#include<stdio.h>

#include<math.h>

#include<ctype.h>

#include<string.h>

#define Op 100

#define exp 100

int top=-1;

float STACK[Op];

void push(float);

float pop(void);

float evaluate(char \*ep);

int main()

{

char Exp[exp];

float result;

printf("\n Enter a valid postfix expression");

gets(Exp);

result=evaluate(Exp);

printf("\n The Result of the Expression is %0.3f ",result);

return 0;

}

void push(float a)

{

if(top>=Op-1)

{

printf("\n Sorry Stack Overflow!!!");

}

else

{

top++;

STACK[top]=a;

}

}

float pop(void)

{

float item;

if(top==-1)

{

printf("\n Sorry Stack underflow !!!");

}

else

{

item=STACK[top];

top--;

return(item);

}

}

float evaluate(char \*ep)

{

int op1,op2,i;

char ch;

float res;

for(i=0;ep[i]!='\0';i++)

{

ch=ep[i];

if(isdigit(ch))

{

push(ch-'0');

}

if(ch=='+'||ch=='-'||ch=='\*'||ch=='/'||ch=='^')

{

op2=pop();

op1=pop();

switch(ch)

{

case '+': res=op1+op2;

break;

case '-': res=op1-op2;

break;

case '\*': res=op1\*op2;

break;

case '/': res=(float)op1/op2;

break;

case '^': res=pow(op1, op2);

break;

}

push(res);

}

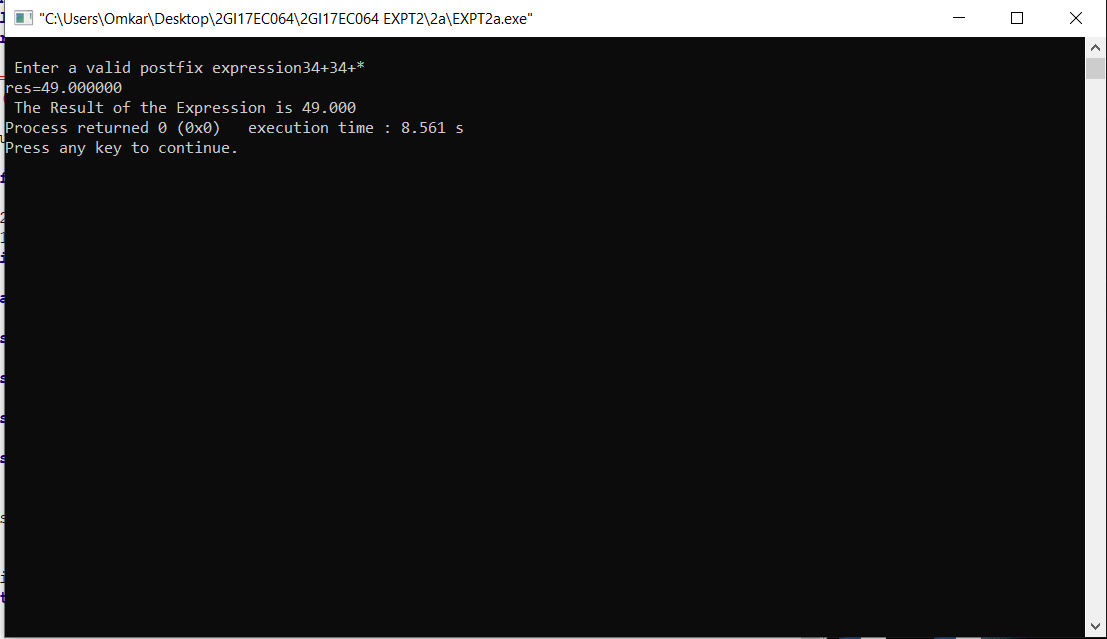
}

printf("res=%f",res);

return(pop());

}

**Output:**



**Analysis/Limitations:**

There are following limitations of above implementation.

1) It supports only 4 binary operators ‘+’, ‘\*’, ‘-‘and ‘/’. It can be extended for more operators by adding more switch cases.

2) The allowed operands are only single digit operands. The program can be extended for multiple digits by adding a separator like space between all elements (operators and operands) of given expression.