Name - Shantanu Deshpande
Class - SYCSE
Roll no. - 7
Batch - S1
Sub. - D.S

Practical No.2

Aim: Implementation of evaluation of expressions

Theory:

A postfix expression is a collection of operators and operands in which the operator is placed after the operands.

Ex : the infix expression 2+3*4 is 234*+ in postfix notation and 231*+9- is postfix of 2+3*1-9

The postfix notation is used to represent algebraic expressions.
The expressions written in postfix form are evaluated faster compared to infix notation as parenthesis are not required in postfix.

Evaluation of postfix expression :

A postfix expression can be evaluated using the stack data structure. To evaluate a postfix expression using stack data structure we can use the following steps --

1. Read all the symbols one by one from left to right in the given postfix expression
2. If the reading symbol is operand, then push it on to the stack.
3. If the reading symbol is operator (+, -, *, / etc.) then perform TWO pop operations and store the two popped operands in two different variables (operand1 and operand2). Then perform reading symbol operation using operand1 and operand2 and push result back on the stack.

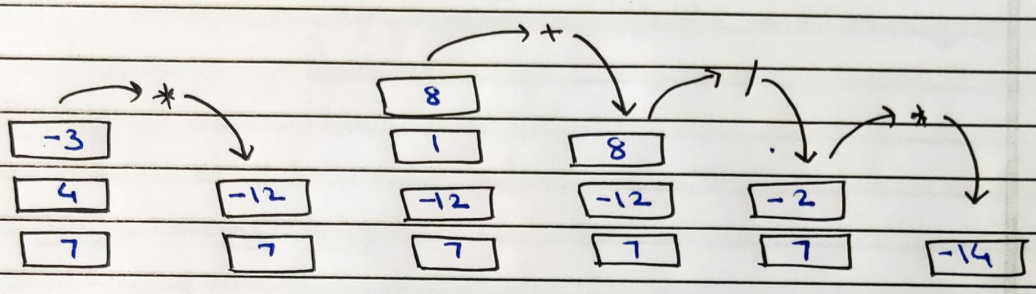
Finally! perform a pop operation and display the popped value as final result.

Date : / / 20

Eg :

Evaluating Postfin Enpressions

* Enpression = 7 4 -3 * 1 5 + / *



Conclusion :-

Thus we have implemented Evaluation of expressions.

Source Code=

```
#include<stdio.h>
int stack[20];
int top = -1;
void push(int x)
{
stack[++top] = x;
}
int pop()
{
return stack[top--];
}
int main()
{
char exp[20];
char *e;
int n1,n2,n3,num;
printf("Enter the expression :: ");
scanf("%s",exp);
e = exp;
while(*e != '\0')
{
if(isdigit(*e))
{
num = *e - 48;
push(num);
}
else
{
n1 = pop();
n2 = pop();
switch(*e)
{
case '+':
{
n3 = n1 + n2;
break;
}
case '-':
{
n3 = n2 - n1;
break;
}
case '*':
{
n3 = n1 * n2;
break;
}
case '/':
{
n3 = n2 / n1;
break;
}
}
```

```
push(n3);
}
e++;
}
printf("\nThe result of expression %s = %d\n\n",exp,pop());
return 0;
}
```

Output-

onlinegdb.com

Run | Debug | Stop | Share | Save | { } Beautify | Language C

main.c

```c

#include<stdio.h>
int stack[20];
int top = -1;
void push(int x)
{
stack[++top] = x;
}
int pop()
{
return stack[top--];
```

input

```
main.c:23:4: warning: implicit declaration of function ‘isdigit’ [-Wimplicit-function-declaration]
   23 | if(isdigit(*e))
      |    ^~~~~~~
Enter the expression :: 80-2*12+*

The result of expression 80-2*12+* = 48



...Program finished with exit code 0
Press ENTER to exit console.
```