Name: Yash Sarang. Roll no. : 47 Class: D6AD

Evaluate Postfix Expression using Stack ADT

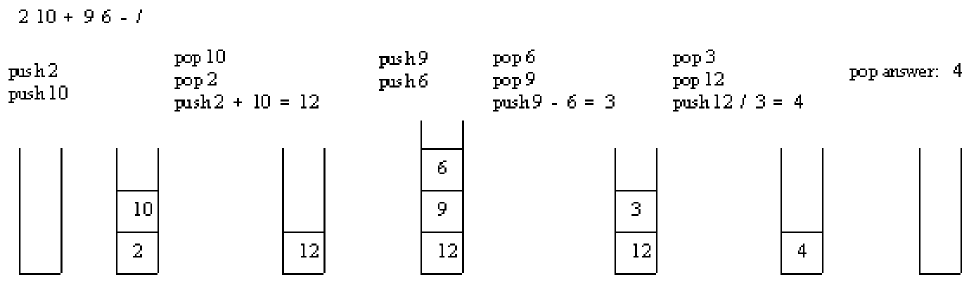
Aim: To Evaluate postfix expression using stack ADT

Theory : 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. We have discussed infix to postfix conversion.

Algorithm:

Create an empty stack and start scanning the postfix expression from left to right.

- If the element is an operand, push it into the stack.
- If the element is an operator O, pop twice and get A and B respectively. Calculate BOA and push it back to the stack.
- When the expression is ended, the value in the stack is the final answer.

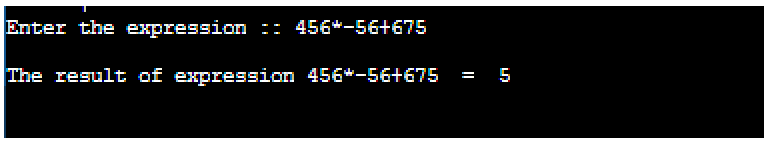


Program Code :

```
#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:



```
Enter the expression :: 456*-56+675

The result of expression 456*-56+675  =  5
```