

K. J. Somaiya College of Engineering, Mumbai
(A Constituent College of Somaiya Vidyavihar University)
Department of Computer Engineering

Batch: D2 Roll No.:16010122323

Experiment / assignment / tutorial No.

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of the Staff In-charge with date

Title: Implementation of Stack applications.

Objective: To implement applications of stack

Expected Outcome of Experiment:

CO	Outcome
1	Explain the different data structures used in problem solving

Books/ Journals/ Websites referred:

1. *Fundamentals Of Data Structures In C* – Ellis Horowitz, Satraj Sahni, Susan Anderson-Fred
2. *An Introduction to data structures with applications* – Jean Paul Tremblay, Paul G. Sorenson
3. *Data Structures A Pseudo Approach with C* – Richard F. Gilberg & Behrouz A. Forouzan
4. <https://www.cprogramming.com/tutorial/computersciencetheory/stack.html>
5. <https://www.geeksforgeeks.org/stack-data-structure-introduction-program/>
6. <https://www.thecrazyprogrammer.com/2013/12/c-program-for-array-representation-of-stack-push-pop-display.html>

Assigned Stack application:
INFIX TO POSTFIX OPERATION

Algorithm:

Input: Infix expression as a string

Output: Postfix expression as a string

Step 1: Initialize the necessary data structures

Create an empty stack for operators (operator stack).

Create an empty list or queue to store the output (postfix expression).

Step 2: Process each token in the infix expression Split the infix expression into tokens (e.g., numbers, operators, parentheses) using a lexer or regular expressions.:

Step 3: If the token is an operand (number)

Add it to the output (postfix expression).

Step 4: If the token is an operator

Step 5: If the token is an open parenthesis '('

Step 6: If the token is a close parenthesis ')'.:

Step 7: Repeat Steps 3-6 for all tokens in the infix expression

Step 8: After processing all tokens, pop any remaining operators from the operator stack and add them to the output.

Step 9: The postfix expression is the final output.

K. J. Somaiya College of Engineering, Mumbai
(A Constituent College of Somaiya Vidyavihar University)
Department of Computer Engineering

Example:

$A + B * C - D - E * F + G$		
Input char	opstack	Output
A		A
+	+	AB
B	+	AB
*	+*	ABC
C	+*	ABC*
-	-	ABC*+
D	-	ABC*+D
-	-	ABC*+D-
E	-	ABC*+D-E
*	-*	ABC*+D-E*
F	-*	ABC*+D-EF
+	+	ABC*+D-EF*
G	+	ABC*+D-EF*+
NULL	Empty stack	ABC*+D-EF*+G

Sourcecode:

```
#include<stdio.h>
```

```
#include<ctype.h>
```

```
char stack[100];
```

```
int top = -1;
```

```
void push(char x)
```

```
{
```

```
    stack[++top] = x;
```

```
}
```

```
char pop()
{
    if(top == -1)
        return -1;
    else
        return stack[top--];
}
```

```
int priority(char x)
{
    if(x == '(')
        return 0;
    if(x == '+' || x == '-')
        return 1;
    if(x == '*' || x == '/')
        return 2;
    return 0;
}
```

```
int main()
{
    char exp[100];
    char *e, x;
    printf("Enter the expression : ");
```

```
scanf("%s",exp);

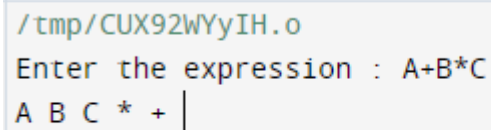
printf("\n");

e = exp;


while(*e != '\0')
{
    if(isalnum(*e))
        printf("%c ",*e);
    else if(*e == '(')
        push(*e);
    else if(*e == ')')
    {
        while((x = pop()) != '(')
            printf("%c ", x);
    }
    else
    {
        while(priority(stack[top]) >= priority(*e))
            printf("%c ",pop());
        push(*e);
    }
    e++;
}
```

```
while(top != -1)
{
    printf("%c ",pop());
}return 0;
}
```

Output Screenshots:



```
/tmp/CUX92WYyIH.o
Enter the expression : A+B*C
A B C * + |
```

Conclusion:

HENCE, WE ARE ABLE TO IMPLEMENT INFIX TO POSTFIX OPERATION USING LINKED LIST