

**Name-Jeevan R**

**Sec- I**

**DSA in C**

**Lab Program 2: program to convert a given valid arithmetic expression to Postfix expression**

**Code:**

```
#include <stdio.h>

char stack[100];

int top = -1;

void push(char c) {
    stack[++top] = c;
}

char pop() {
    if (top == -1) return '\0';
    return stack[top--];
}

int precedence(char c) {
    if (c == '+' || c == '-') return 1;
    if (c == '*' || c == '/') return 2;
    return 0;
}

int main() {
    char in[100], out[100];

    int k = 0;

    printf("Enter Infix: ");

    scanf("%s", in);

    for (int i = 0; in[i] != '\0'; i++) {
        char c = in[i];

        if ((c >= 'A' && c <= 'Z') || (c >= 'a' && c <= 'z')) {
```

```

        out[k++] = c;
    }
    else if (c == '(') {
        push(c);
    }
    else if (c == ')') {

        while (top != -1 && stack[top] != '(') {
            out[k++] = pop();
        }
        if (top != -1 && stack[top] == '(') {
            pop();
        }
    }
    else {
        while (top != -1 && precedence(stack[top]) >= precedence(c)) {
            out[k++] = pop();
        }
        push(c);
    }
}

while (top != -1) {
    out[k++] = pop();
}

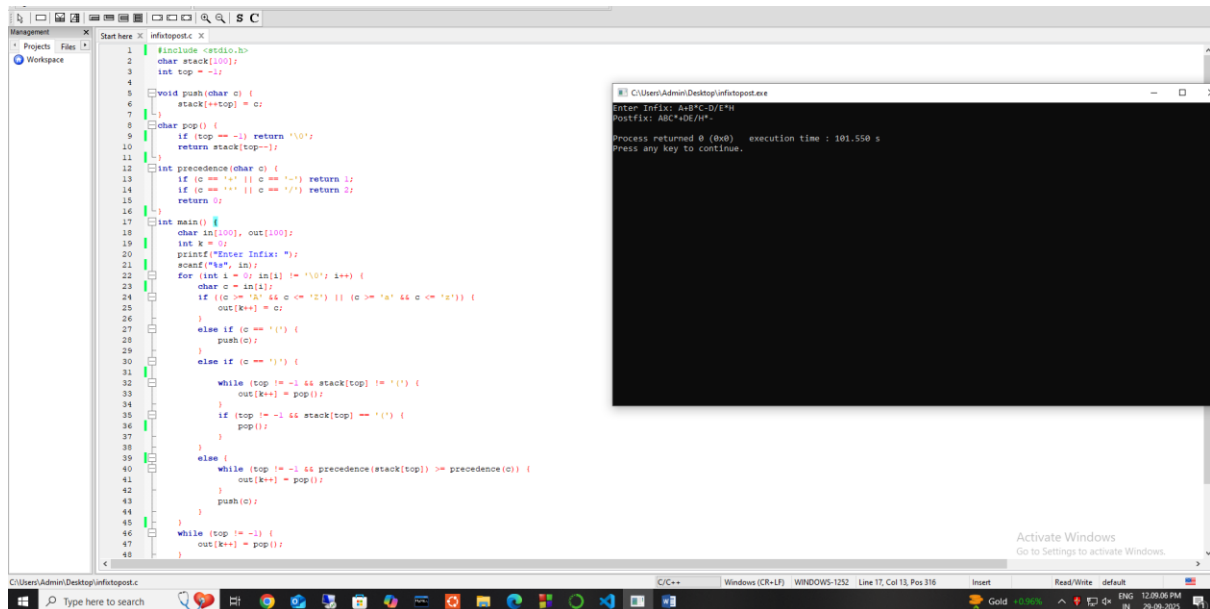
out[k] = '\0';
printf("Postfix: %s\n", out);
return 0;
}

```

## Expected Output:

Enter Infix: A+B\*C-D/E\*H

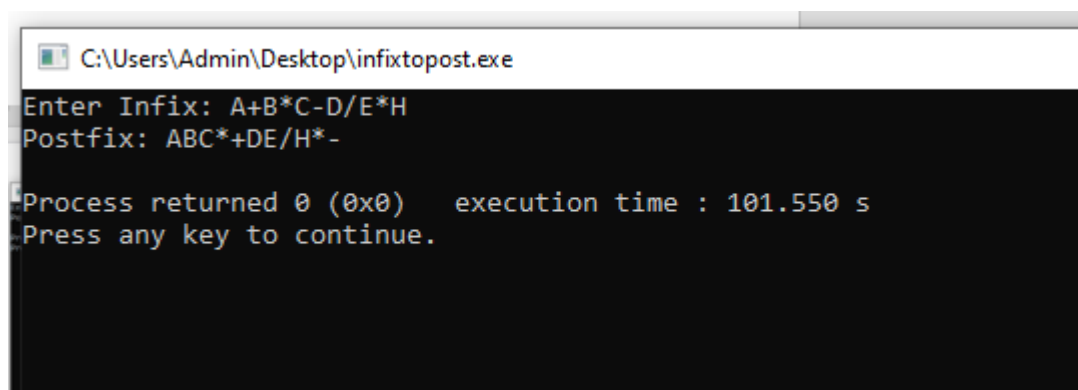
Postfix: ABC\*+DE/H\*-



The screenshot shows a C++ IDE with the source code for an infix to postfix converter. The code includes a stack, precedence function, and main function that reads an infix expression and outputs the postfix expression. The execution output window shows the input 'A+B\*C-D/E\*H' and the output 'ABC\*+DE/H\*-', along with the execution time and a return message.

```
1 #include <stdio.h>
2 char stack[100];
3 int top = -1;
4
5 void push(char c) {
6     stack[++top] = c;
7 }
8
9 char pop() {
10    if (top == -1) return '\0';
11    return stack[top--];
12 }
13
14 int precedence(char c) {
15    if (c == '+' || c == '-') return 1;
16    if (c == '*' || c == '/') return 2;
17    return 0;
18 }
19
20 int main() {
21    char in[100], out[100];
22    int k = 0;
23    printf("Enter Infix: ");
24    scanf("%s", in);
25    for (int i = 0; in[i] != '\0'; i++) {
26        char c = in[i];
27        if ((c == '(' || c == '<math>^*</math>' || c == '/' || c == '-') && c != '(') {
28            out[k++] = c;
29        }
30        else if (c == '(') {
31            push(c);
32        }
33        else if (c == ')') {
34            while (top != -1 && stack[top] != '(') {
35                out[k++] = pop();
36            }
37            if (top != -1 && stack[top] == '(') {
38                pop();
39            }
40        }
41        else {
42            while (top != -1 && precedence(stack[top]) >= precedence(c)) {
43                out[k++] = pop();
44            }
45            push(c);
46        }
47    }
48    while (top != -1) {
49        out[k++] = pop();
50    }
51    out[k] = '\0';
52    printf("Postfix: %s\n", out);
53    return 0;
54 }
```

Enter Infix: A+B\*C-D/E\*H  
Postfix: ABC\*+DE/H\*-  
Process returned 0 (0x0) execution time : 101.550 s  
Press any key to continue.



The screenshot shows a command prompt window with the title 'C:\Users\Admin\Desktop\infixtopost.exe'. It displays the same input and output as the previous screenshot, confirming the correct conversion of the infix expression to postfix.

```
C:\Users\Admin\Desktop\infixtopost.exe
Enter Infix: A+B*C-D/E*H
Postfix: ABC*+DE/H*-
Process returned 0 (0x0) execution time : 101.550 s
Press any key to continue.
```