

```

#include<stdio.h>
#include<ctype.h>
#define n 50
char stack[n];
int top=-1;
push(char elem)
{
    stack[++top]=elem;
}
char pop()
{
    return (stack[top--]);
}
int pr(char symbol)
{
    if(symbol=='^')
    {
        return(3);
    }
    else if(symbol=='*' || symbol=='/')
    {
        return(2);
    }
    else if(symbol=='+' || symbol=='-')
    {
        return(1);
    }
    else{
        return(0);
    }
}
void main()
{
    char infix[50],postfix[50],ch,elem;
    int i=0,k=0;
    printf("Enter the infix expression\n");
    scanf("%s",infix);
    push('#');
    while( (ch=infix[i++]) !='\0')
    {
        if(ch=='(')push(ch);
        else
            if(isalnum(ch))postfix[k++]=ch;
        else
            if(ch==')')

```

```

    {
        while(stack[top]!='(')
            postfix[k++]=pop();
        elem=pop();
    }
    else
    {
        while(pr(stack[top])>=pr(ch))
            postfix[k++]=pop();
        push(ch);
    }
}
while(stack[top]!='#')
    postfix[k++]=pop();
postfix[k]='\0';
printf("Postfix is %s\n",postfix);
}

```

The screenshot shows a Visual Studio Code editor window with a C++ file named 'lab2.c'. The code implements an infix-to-postfix conversion algorithm using a stack. The terminal at the bottom shows the execution of the program, which prompts for an infix expression and prints the resulting postfix expression.

Source Code (lab2.c):

```

14 int pr(char symbol)
20 else if(symbol=="*")||symbol=="/"
22     return(2);
23 }
24 else if(symbol=="+"||symbol=="-")
25 {
26     return(1);
27 }
28 else{
29     return(0);
30 }
31 }
32 void main()
33 {

```

Terminal Output:

```

PS C:\Users\BMSCECESE\Desktop\1BF24CS243> cd 'c:\Users\BMSCECESE\Desktop\1BF24CS243\output'
PS C:\Users\BMSCECESE\Desktop\1BF24CS243\output> & .\lab2.exe
Enter the infix expression
(1+2)*3
Postfix is 12+3*
PS C:\Users\BMSCECESE\Desktop\1BF24CS243\output> cd 'c:\Users\BMSCECESE\Desktop\1BF24CS243\output'
PS C:\Users\BMSCECESE\Desktop\1BF24CS243\output> & .\lab2.exe
Enter the infix expression
(2+5)*3
Postfix is 25+3*
PS C:\Users\BMSCECESE\Desktop\1BF24CS243\output> cd 'c:\Users\BMSCECESE\Desktop\1BF24CS243\output'
PS C:\Users\BMSCECESE\Desktop\1BF24CS243\output> & .\lab2.exe
Enter the infix expression
a+b*c
Postfix is abc*+
PS C:\Users\BMSCECESE\Desktop\1BF24CS243\output>

```

```

C:\Users> BMSCECSE > Desktop > 18F24CS243 > C lab2c > main()
14 int pr(char symbol)
15 {
16     if(symbol=='*')
17     {
18         return(3);
19     }
20     else if(symbol=='/' || symbol=='/')
21     {
22         return(2);
23     }
24     else if(symbol=='+' || symbol=='-')
25     {
26         return(1);
27     }
28     else{
29         return(0);
30     }
31 }
32 void main()
33 {
34     char infix[50], postfix[50], ch, elem;
35     int i=0, k=0;
36     printf("Enter the infix expression\n");
37     scanf("%s", infix);
38     push('&');
39     while( (ch=infix[i++]) !='\0')
40     {
41         if(ch=='(') push(ch);
42         else
43             if(isalnum(ch)) postfix[k++]=ch;
44         else
45             if(ch==' ')
46             {
47                 while(stack[top]!='(')
48                     postfix[k++]=pop();
49                 elem=pop();
50             }
51             else
52             {
53                 while(pr(stack[top])>pr(ch))
54                     postfix[k++]=pop();
55                 push(ch);
56             }
57     }
58     while(stack[top]!='&')
59         postfix[k++]=pop();
60     postfix[k]='\0';
61     printf("Postfix is %s\n", postfix);
62 }

```

```

C:\Users> BMSCECSE > Desktop > 18F24CS243 > C lab2c > main()
1 #include<stdio.h>
2 #include<ctype.h>
3 #define n 50
4 char stack[n];
5 int top=-1;
6 push(char elem)
7 {
8     stack[++top]=elem;
9 }
10 char pop()
11 {
12     return (stack[top--]);
13 }
14 int pr(char symbol)
15 {
16     if(symbol=='*')
17     {
18         return(3);
19     }
20     else if(symbol=='/' || symbol=='/')
21     {
22         return(2);
23     }
24     else if(symbol=='+' || symbol=='-')
25     {
26         return(1);
27     }
28     else{
29         return(0);
30     }
31 }
32 void main()
33 {
34     char infix[50], postfix[50], ch, elem;
35     int i=0, k=0;
36     printf("Enter the infix expression\n");
37     scanf("%s", infix);
38     push('&');
39     while( (ch=infix[i++]) !='\0')
40     {
41         if(ch=='(') push(ch);
42         else
43             if(isalnum(ch)) postfix[k++]=ch;
44         else
45             if(ch==' ')
46             {
47                 while(stack[top]!='(')
48                     postfix[k++]=pop();
49                 elem=pop();
50             }
51             else

```