

```
3. #include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

```
int F (Sym char Symbol) {
    switch (Symbol) {
        case '+':
        case '-': return 2;
        case '*':
        case '/': return 4;
        case '^':
        case '$': return 5;
        case '#': return 0;
        case '@': return -1;
        default: return 8;
    }
}
```

```
int G (Char Symbol) {
    switch (Symbol) {
        case '+':
        case '-': return 1;
        case '*':
        case '/': return 3;
        case '^':
        case '$': return 6;
        case '(': return 9;
        case ')': return 0;
        default: return 7;
    }
}
```



```

void infix_postfix(char infix[]) {
    int top, j, i;
    char s[30], postfix[30];
    char symbol;
    top = -1;
    s[++top] = '#';
    j = 0;
    for (i = 0; i < strlen(infix); i++) {
        symbol = infix[i];
        while (F(s[top]) > G(symbol)) {
            postfix[j] = s[top--];
            j++;
        }
        if (F(s[top]) != '#') {
            postfix[j++] = s[top];
            s[++top] = symbol;
        }
        else
            top--;
    }
    while (s[top] != '#') {
        postfix[j++] = s[top--];
    }
    postfix[j] = '\0';
    puts(postfix);
}

```



```
int main()
```

```
{
```

```
    char exp[30];
```

```
    printf("Enter an expression: \n");
```

```
    gets(exp);
```

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

```
    infix -> postfix(exp);
```

```
    return 0;
```

```
}
```



my.c

Saved



```
1  #include<stdio.h>
2  #include<stdlib.h>
3  #include<string.h>
4  int F(char symbol){
5      switch(symbol){
6          case '+' :
7          case '-' : return 2;
8          case '*' :
9          case '/' : return 4;
10         case '^' :
11         case '$' : return 5;
12         case '(' : return 0;
13         case '#' : return -1;
14         default : return 8;
15     }
16 }
17 int G(char symbol){
18     switch(symbol){
19         case '+' :
20         case '-' : return 1;
21         case '*' :
22         case '/' : return 3;
23         case '^' :
24         case '$' : return 6;
25         case '(' : return 9;
26         case ')' : return 0;
27         default : return 7;
28     }
29 }
30 void infix_postfix(char infix[]){
31     int top,j,i;
32     char s[30],postfix[30];
33     char symbol;
34     top=-1;
35     s[++top]='#';
36     j=0;
37     for(i=0;i<strlen(infix);i++){
38         symbol=infix[i];
39
40         while(F(s[top])>G(symbol)){
41             postfix[j]=s[top--];
42             j++;
43         }
44         if(F(s[top])!=G(symbol)){
45             s[++top]=symbol;
46         }
47         else
48             top--;
49     }
```

```
50     while(s[top]!='#'){
51         postfix[j++]=s[top--];
52     }
53     postfix[j]='\0';
54     printf("Postfix: ");
55     puts(postfix);
56 }
57 int main()
58 {
59     char exp[30];
60     printf("enter a expression:\n");
61     scanf("%s",exp);
62     infix_postfix(exp);
63     return 0;
64 }
```



Terminal



enter a expression:

$(a+(b-c)*d)$

Postfix: $abc-d*+$

Process finished.