```c
#include<stdio.h>
#include<string.h>
int F(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[],char
postfix[]){
   int top,i,j;
   char s[30],symbol;
   top=-1;
   s[++top]='#';
```

```
33      char s[30],symbol;
34      top=-1;
35      s[++top]='#';
36      j=0;
37      for(i=0;i<strlen(infix);i++){
38          symbol=infix[i];
39          while(F(s[top])>G(symbol)){
40              postfix[j]=s[top--];
41              j++;
42          }
43              if(F(s[top])!=G(symbol)){
44                  s[++top]=symbol;
45              }
46                  else
47                  top--;
48              }
49              while(s[top]!='#'){
50                  postfix[j++]=s[top--];
51              }postfix[j]='\0';
52          }
53  int main()
54  {
55      char infix[20],postfix[20];
56      printf("Enter the valid infix
    expression\n");
57      scanf("%s",infix);
58      infix_postfix(infix,postfix);
59      printf("The postfix expression is \n");
60      printf("%s",postfix);
61  }
```

```
Enter the valid infix expression
((a+b)*c-(d-e))^(f+g)
The postfix expression is
ab+c*de--fg+^
[Program finished]
```

```
Enter the valid infix expression
((a+(b-c)*d)^e+f)
The postfix expression is
abc-d*+e^f+
[Program finished]
```

```
Enter the valid infix expression
a^b*c-d+e/f/(g+h)

[Program finished]The postfix expression is
ab^c*d-ef/gh+/+
```

LAB-2

WAP to convert infix expression to postfix expression.

```
# include <stdio.h>
# include <string.h>
# include <process.h>
int F(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 in-postfix(char infix[], char postfix[])
{
int top, i, j;
char s[30], symbol;
top = -1;
s[++top] = '#';
j = 0;
for(i=0; i<strlen(infix); i++)
{
Symbol = infix[i];
while (f(s[top]) > (symbol))
{
postfix[j] = s[top--];
j++;
}
if (f(s[top]) != (symbol))
  s[++top] = symbol;
else
  top--;
}
while(s[top] != '#')
{
postfix[j++] = s[top--];
}
postfix[j] = '\0';
}
Void main()
{
char infix[20], postfix[20];
printf("Enter valid infix expression\n");
scanf("%s", infix);
```

```
in_postfix (infix, postfix);
printf ("The postfix expression is : \n");
printf ("%s \n", postfix);
return 0;
}
```