DATE .....
PAGE NO. .....

WAP to convert a given valid parenthesized infix arithmetic expression to postfix expression. The expression consists of single character operands & the binary operators +, -, /, *.

```
#include <stdio.h>
#include <string.h>
#include <stdlib.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 0;
case ')' : return 0;
default : return 7;
    }
}

void infix_postfix(char infix[3, char postfix[])
{
int top, i, j;
char s[20], symbol;
top = -1;
s[++top] = '#';
j=0
for (i=0; i<strlen(infix); i++)
{
symbol(F(s[top] > G(symbol))
{
postfix[j] = s[top--];
j++
}
while (s[top] != '#')
{
postfix[j++] = s[top--];
}
postfix[j] = '\0';
}
Void main(){
char infix[20];
char postfix[20];
printf("Enter the valid infix expression: \t");
scanf("%s", infix);
infix_postfix(infix, postfix);
```

```
printf("\n The postfix expression is : \t");
printf("%s \n", postfix);
}
```