Lab Program - 2

Date ___/___/_____

WAP to convert a given valid parenthesized infix ~~WA~~ arithmetic expression to postfix expression. The expression consists of single character operands and the binary operators + (plus), -(minus), *(multiply) and (/divide)

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

int F (char symbol)
{
switch (symbol)
{
case '+' :
 Case '-' : return 2;
case '*' :
Case '/' : return4;
Case '^' :
case '$' : return5;
case '(' : return 0;
case '#' : return-1;
default : return 8;
}
int G (char symbol)
{
 switch( symbol)
 {
  case '+' :
  Case '-' : return1;
  case '*' :
  Case '/' : return 3;
 case '^' :
 Case '$' : return 6;
```

Date __/__/__

```
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] = '#';
j = 0;
for (i=0; i<strlen(infix); i++)
{
~~fo~~ symbol = infix[i];
while (F(s[top]) > G(symbol))
{
postfix[j] = s[top--];
j++;
}
if (F(s[top]) != G(symbol))
  s[++top] = symbol;
else
top--;
while (s[top] != '#')
{
postfix[j++] = s[top--];
}
postfix[j] = '\0';
}
```

Scanned with CamScanner

Date ___ / ___ / ______

```
void main ()
{
 char infix [20];
 char postfix [20];
printf (" enter the valid infix expression \n");
Scanf ("%s", infix);
infix_postfix ( infix, postfix);
printf (" the positive expression is \n");
printf ("%s \n", postfix);
}
```