1BM19CS006

9/10/20 Lab 2 - Infix to Postfix

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#define SIZE 50
char stack[SIZE];
int top=-1;

void push (char ch)
{
  if (top==SIZE-1)
   printf ("Stack Overflow\n");
  else {
    top++;
    stack[top]=ch;
  }
}
char pop()
{
  char ele;
  {
    ele=stack[top];
     top--;
    return ele;
  }
}
```

IBM19CS006

```
int stackempty()
{
  if(top==-1) return 1;
   else return 0;
 }
char stacktop()
{
   return stack[top];
   }

int priority(char ch)
{
  switch(ch)
  {
     case '+':
     case '-': return (1);
     case '*':
     case '/': return (2);
      case '^': return (3);
     default : return (0);
   }
}
 int main()
{
  char infix[SIZE];
  int i, item, opbrac=0, clobrac=0, operands=0,
   operators=0;
printf("Enter the infix expression: ");
```

```
scanf ("%s", infix);
for (i=0; infix [i]!='\0'; i++)
 {
  if (infix[i]=='+' || infix[i]=='-' || infix[i]=='*'
     || infix[i]=='/' || infix[i]=='^')
    operators++;
  if (infix [i]>='a' && infix[i]<='z'  ||
     infix [i]>='A'  && infix[i]<='Z')
  operands++;

 if ( infix [i]=='(')
   ~~ch~~ opbrac++;
  if (infix [i]==')')
     clobrac++;
  }

if (operands!=(operators+1) || opbrac != clobrac)
 {
   printf ("Invalid expression ");
    exit(0);
 }
printf ("Expression given is : %s \n", infix);
printf ("Postfix :  ");
 i=0;
while ( infix[i] != '\0')
 {
   switch (infix[i])
 {
```

```
case '(': push(infix[i]);
        break;
case ')': while ((item=pop()) != '(')
        printf ("%c", item);
        break;
case '+':
case '-':
case '*':
case '/':
case '^':
     while (!stackempty() && priority(infix[i])<=
            priority (stackempty()))
  {
    item=pop();
    printf ("%c", item);
   }
  push (infix [i]);
  break;
 }
  i++;
 }
while (!stackempty())
  { char item;
    item=pop();
    printf ("%c", item);
   }
printf ("\n");
 return 0;
}
```