LAB PROGRAMS
Data Structure and Applications

4. Design, Develop and Implement a Program in C for converting an Infix Expression to Postfix Expression. Program should support for both parenthesized and free parenthesized expressions with the operators: +, -, *, /, %(Remainder), ^(Power) and alphanumeric operands.

```
#include<stdio.h>

void   infix_to_postfix();
void   push(char);
char   pop();
int   priority(char);
char   infix[30],postfix[30],stack[30];
int  top=-1;

void main()
{
printf(" Enter a valid infix epression\n");
scanf("%s",infix);
infix_to_postfix();
printf("\nInfix expression is: %s",infix);
printf("\n postfix expression is: %s", postfix);

}
void push(char item)
{

   stack[++top]=item;
}
char pop()
{
```

```
   return stack[top--];
}
int priority(char symbol)
{
   int p;
   switch(symbol)
         {
         case '+':
         case '-': p=1;
                  break;

          case '/':
          case '*':
          case '%':p=2;
                  break;
          case '^':p=3;
                  break;

          default:p=0;

         }
         return p;


       }
void infix_to_postfix()
{

   int i=0,j=0;
   char symbol,temp;
   //push("#");
   for(i=0;infix[i]!='\0';i++)
   {
      symbol=infix[i];
```

```
        switch(symbol)
        {
           case  '(':push(symbol);
                    break;
           case ')':temp=pop();
                    while(temp!='(')
                    {
                       postfix[j++]=temp;
                       temp=pop();
                    }
                    break;
           case '+':
           case '-':
           case '*':
           case '/':
           case '%':
           case '^':while(priority(stack[top])>=priority(symbol))
                          {

                            temp=pop();
                            postfix[j++]=temp;
                          }
                          push(symbol);
                          break;
           default:postfix[j++]=symbol;
                    break;
     }
}
while(top>=0)
{
   temp=pop();
   postfix[j++]=temp;
}
postfix[j]='\0';
}
```