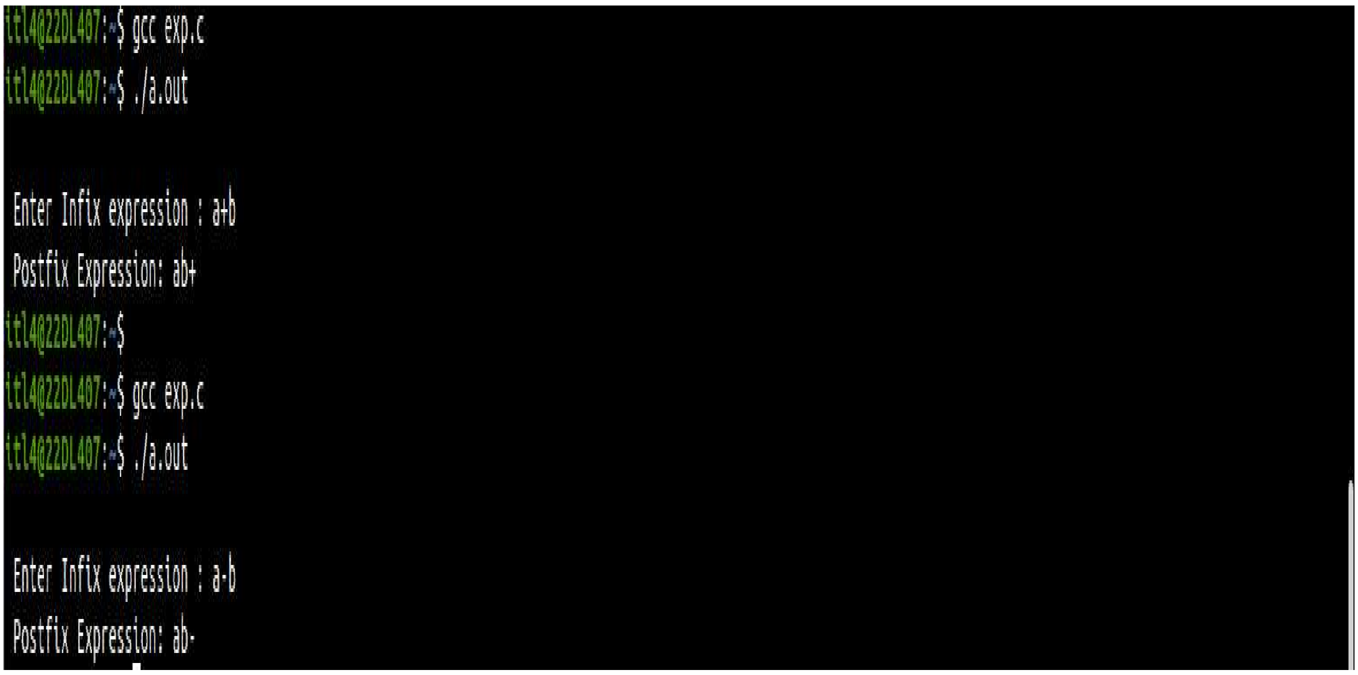
NAME:JAYKUMAR.P.GOR
ROLL NO.:16

PROBLEM STATEMENT:IMPLEMENTATION FROM INFIX TO POSTFIX.

CODE:
```
#include<stdio.h>
//#include<conio.h>
#include<stdlib.h>
#include<ctype.h>
#include<string.h>


#define size 100
char stack[size];
int top=-1;



void push(char item)
{
  if(top >= size-1)
   {
      printf("STACK IS FULL!!!\n");
   }
  else
   {
     top++;
     stack[top]=item;
   }
 }
 char pop()
 {
     char item;
    if(top==-1)
      {
       printf("STACK IS EMPTY!!\n");
      }
    else
    {
       item=stack[top];
       top--;
       return(item);
    }
 }
 int operator(char symbol)
 {
   if(symbol == '^' || symbol == '*' || symbol == '/' || symbol == '+' || symbol =='-')
     {
        return 1;
     }
   else
     {
       return 0;
     }
}
int precedence(char symbol)
{
  if(symbol == '^')
          {
                    return(3);
          }
          else if(symbol == '*' || symbol == '/')
          {
                    return(2);
          }
          else if(symbol == '+' || symbol == '-')
          {
                    return(1);
          }
          else
          if(top>0)
          {
                    printf("\nInvalid infix Expression.\n");
                    getchar();
                    exit(1);
          }

        {
                    return(0);
          }
}
void InfixToPostfix(char infix[],char postfix[])
{
  char item;
  int i,j;
  char x;

  push('(');
  strcat(infix,")");

  i=0;
  j=0;
  item=infix[i];

  while(item!='\0')
   {
      if(item=='(')
       {
         push(item);
       }
     else if( isdigit(item) || isalpha(item))
       {
              postfix[j]=item;
              j++;
       }
     else if (operator(item)==1)
      {
        x=pop();
        while(operator(x)==1 && precedence(x)>= precedence(item))
         {
          postfix[j]=x;
          j++;
          x=pop();
          }
   push(x);

   push(item);
   }
      else if(item == ')')
      {
        x=pop();
        while(x!='(')
        {
         postfix[j]=x;
         j++;
         x=pop();
         }
        }
       else
       {
       printf("INVALID INFIX EXPRESSION!!!");
       exit(1);
       }
       i++;


       item=infix[i];
     }
     if(top>0)
          {
                    printf("\nInvalid infix Expression.\n");
                    getchar();
                    exit(1);
          }
          }




int main()
{
          char infix[size], postfix[size];
        printf("\n Enter Infix expression : ");
        scanf("%s",infix);

        InfixToPostfix(infix,postfix);
        printf(" Postfix Expression: ");
        puts(postfix);

        return 0;
}
```

SCREENSHOT:



```
itl4@22DL407:~$ gcc exp.c
itl4@22DL407:~$ ./a.out


 Enter Infix expression : a+b
 Postfix Expression: ab+
itl4@22DL407:~$
itl4@22DL407:~$ gcc exp.c
itl4@22DL407:~$ ./a.out


 Enter Infix expression : a-b
 Postfix Expression: ab-
```