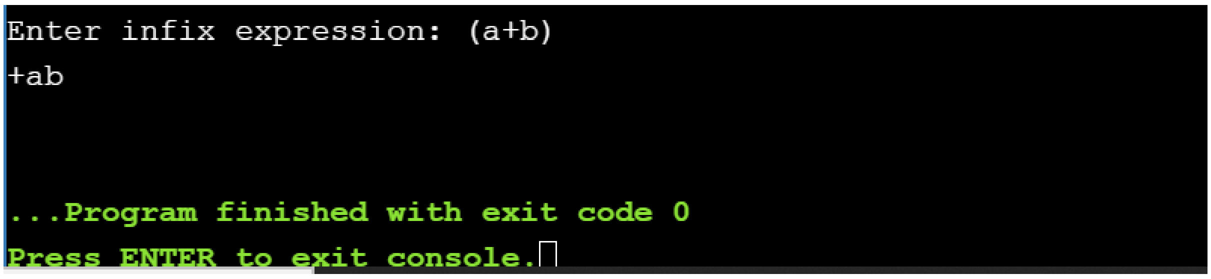
LAB 3

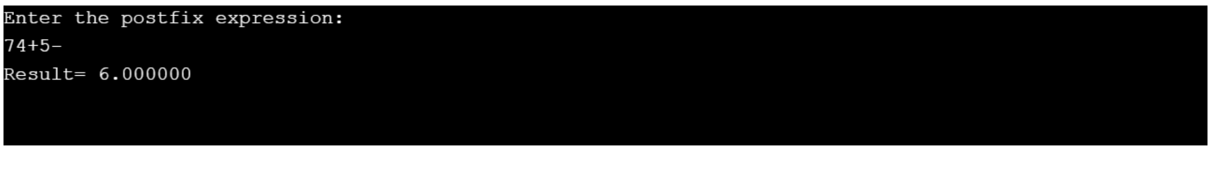
1)

```
# include <stdio.h>
# include <string.h>
# define MAX 20
void infixtoprefix(char infix[20], char prefix[20]);
void reverse(char array[30]);
char pop();
void push(char symbol);
int isOperator(char symbol);
int prcd(char symbol);
int top = -1;
char stack[MAX];

main() {
char infix[20], prefix[20], temp;
printf("Enter infix operation: ");
gets(infix);
infixtoprefix(infix, prefix);
reverse(prefix);
puts((prefix));
}
void infixtoprefix(char infix[20], char prefix[20]) {
int i, j = 0;
char symbol;
stack[++top] = '#';
reverse(infix);
for (i = 0; i < strlen(infix); i++) {
symbol = infix[i];
if (isOperator(symbol) == 0) {
  prefix[j] = symbol;
  j++;
} else {
  if (symbol == ')') {
    push(symbol);
  } else if (symbol == '(') {
    while (stack[top] != ')') {
      prefix[j] = pop();
      j++;
    }
    pop();
  } else {
    if (prcd(stack[top]) <= prcd(symbol)) {
      push(symbol);
    } else {
      while (prcd(stack[top]) >= prcd(symbol)) {
        prefix[j] = pop();
      j++;
     }
     push(symbol);
   }

    }
  }

  }

 while (stack[top] != '#') {
   prefix[j] = pop();
   j++;
  }
 prefix[j] = '\0';
}

void reverse(char array[30]) {

int i, j;
char temp[100];
for (i = strlen(array) - 1, j = 0; i + 1 != 0; --i, ++j) {
 temp[j] = array[i];
}
temp[j] = '\0';
strcpy(array, temp);//copying temp array to array

}

 char pop() {
 char a;
 a = stack[top];
 top--;
 return a;
 }

void push(char symbol) {
top++;
stack[top] = symbol;
}

int prcd(char symbol) {

switch (symbol) {
 case '+':
  case '-':
   return 2;
   break;
  case '*':
   case '/':
    return 4;
    break;
   case '$':
   case '^':
    return 6;
    break;
   case '#':
   case '(':
   case ')':
    return 1;
    break;
  }
}

int isOperator(char symbol) {
switch (symbol) {
case '+':
case '-':
case '*':
case '/':
case '^':
case '$':
case '&':
case '(':
case ')':
  return 1;
  break;
default:
  return 0;

  }
 }
```


```
Enter infix expression: (a+b)
+ab


...Program finished with exit code 0
Press ENTER to exit console.
```

2)

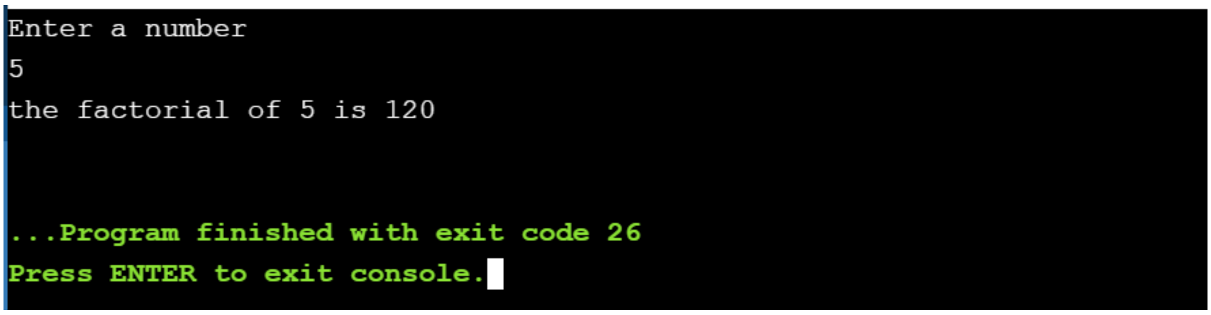
```
#include<stdio.h>
#include<math.h>
#include<string.h>
double compute(char symbol,double op1,double op2)
{
   switch(symbol)
   {
      case '+':return op1+op2;
      case '-':return op1-op2;
      case '*':return op1*op2;
      case '/':return op1/op2;
      case '$':
      case '^':return pow(op1,op2);
   }
}
void main()
{
   double s[20];
   double res;
   double op1,op2;
   int top,i;
   char postfix[20],symbol;
   printf("Enter the postfix expression:\n");
   scanf("%s",postfix);
   top=-1;
   for(i=0;i<strlen(postfix);i++)
   {
      symbol=postfix[i];
      if(isdigit(symbol))
         s[++top]=symbol-'0';
      else{
         op2=s[top--];
         op1=s[top--];
         res=compute(symbol,op1,op2);
         s[++top]=res;
      }
   }
   res=s[top--];
   printf("Result= %f\n",res);
}
```



```
Enter the postfix expression:
74+5-
Result= 6.000000
```

3)

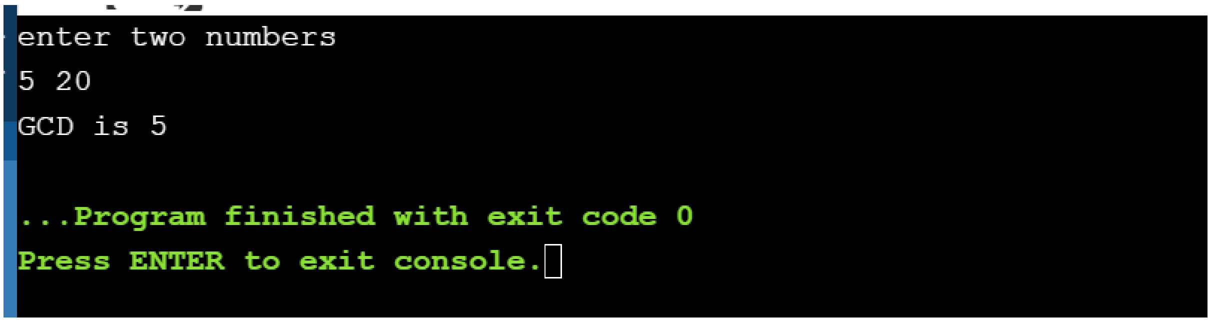
```
#include <stdio.h>
int fact(int n)
{
   if(n == 0) return 1;
   return n*fact(n-1);
}
void main(){
   int n;
   printf("Enter a number\n");
   scanf("%d",&n);
   printf("the factorial of %d is %d\n", n,fact(n));
}
```



```
Enter a number
5
the factorial of 5 is 120


...Program finished with exit code 26
Press ENTER to exit console.
```

4)

```
#include <stdio.h>
int compute(int x, int y){
   int i,great,gcd;
   if(x>y){
      great =x;
   }
   else if(y>x){
      great =y;
   }
   for(i=1; i<great; i++){
      if(x%i == 0 && y%i == 0){
         gcd = i;
      }
   }
   printf("GCD is %d", gcd);
}
int main()
{
   int x,y;
   printf("enter two numbers\n");
   scanf("%d %d", &x, &y);
   compute(x, y);
}
```



```
enter two numbers
5 20
GCD is 5

...Program finished with exit code 0
Press ENTER to exit console.
```