Exercise 2:

1) 2's complement of a number is obtained by scanning it from right to left and complementing all the bits after the first appearance of a 1. Thus 2's complement of 11100 is 00100.

Algorithm:

Step 1: start

Step 2**:** input binary number (i.e. a)

Step 3: for i=0 to '\0' if a[i]! ='\0' True goto step 4

False goto step 6

Step 4: if (a[i]! ='0' && a[i]!='1') True step 5

False step 7

Step 5: print the no is invalid number

Step 6: complement (a)

Step 7: stop

Algorithm for complement ()

Step 1: declare c=0

Step 2: l=strlen(a)

Step 3: put i=(l-1) and if i>=0 True goto step 4

False goto step 5

Step 4: if a[i]=='0' True it prints b[i]='1'

False it prints b[1]='0'

Step 5:put i=l-1 if i>=0 True increment the value of I by one and goto step 6

False goto step 7

Step 6: if (b[i]=='0') goto step if True b[i]='1' will be printed

False b[i]='0' will be printed and assign the value to c=1

Step 7: if(c==1 && b[i]=='0') if True b[i]='1'and assigns the value to c=0

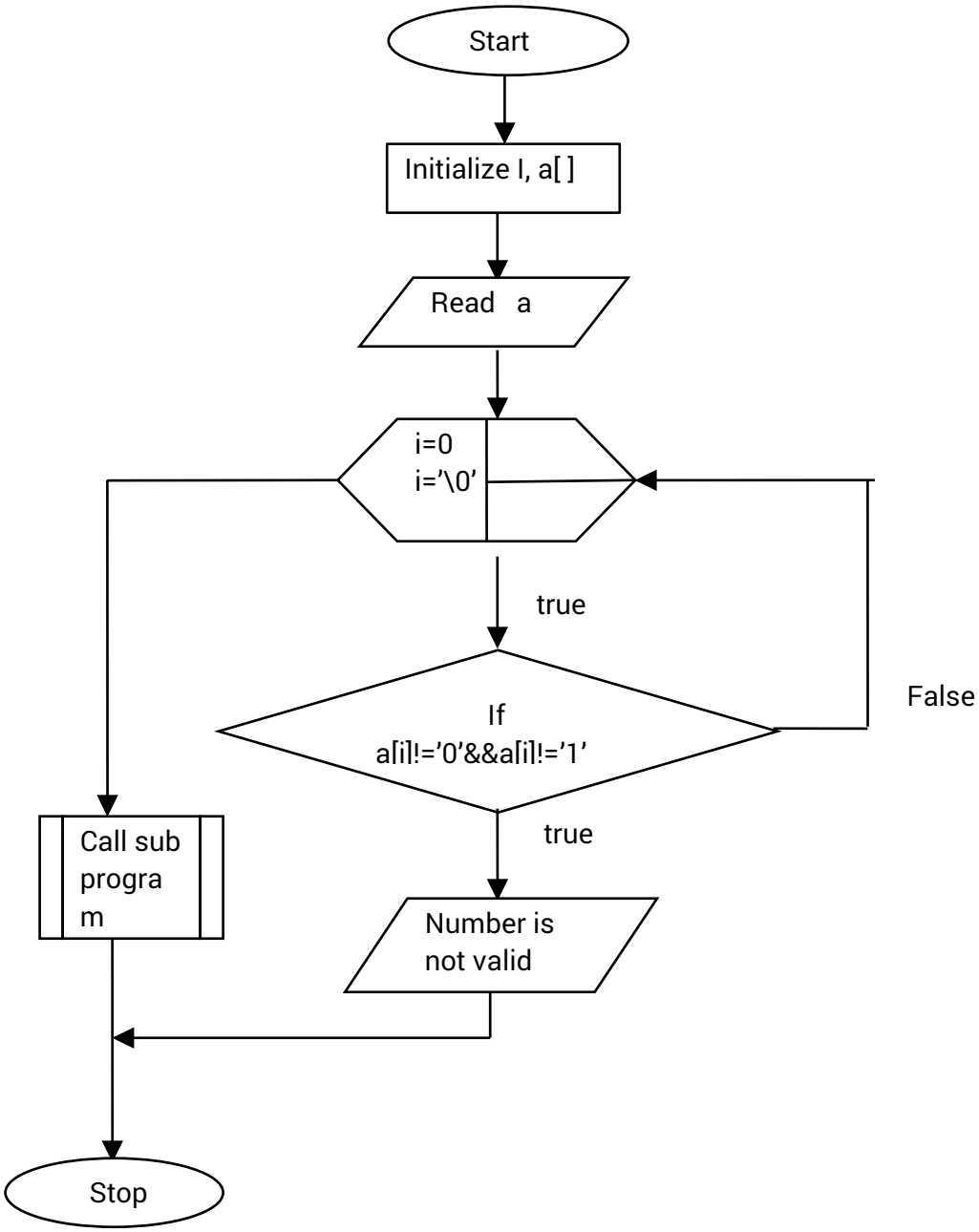
False goto step 8

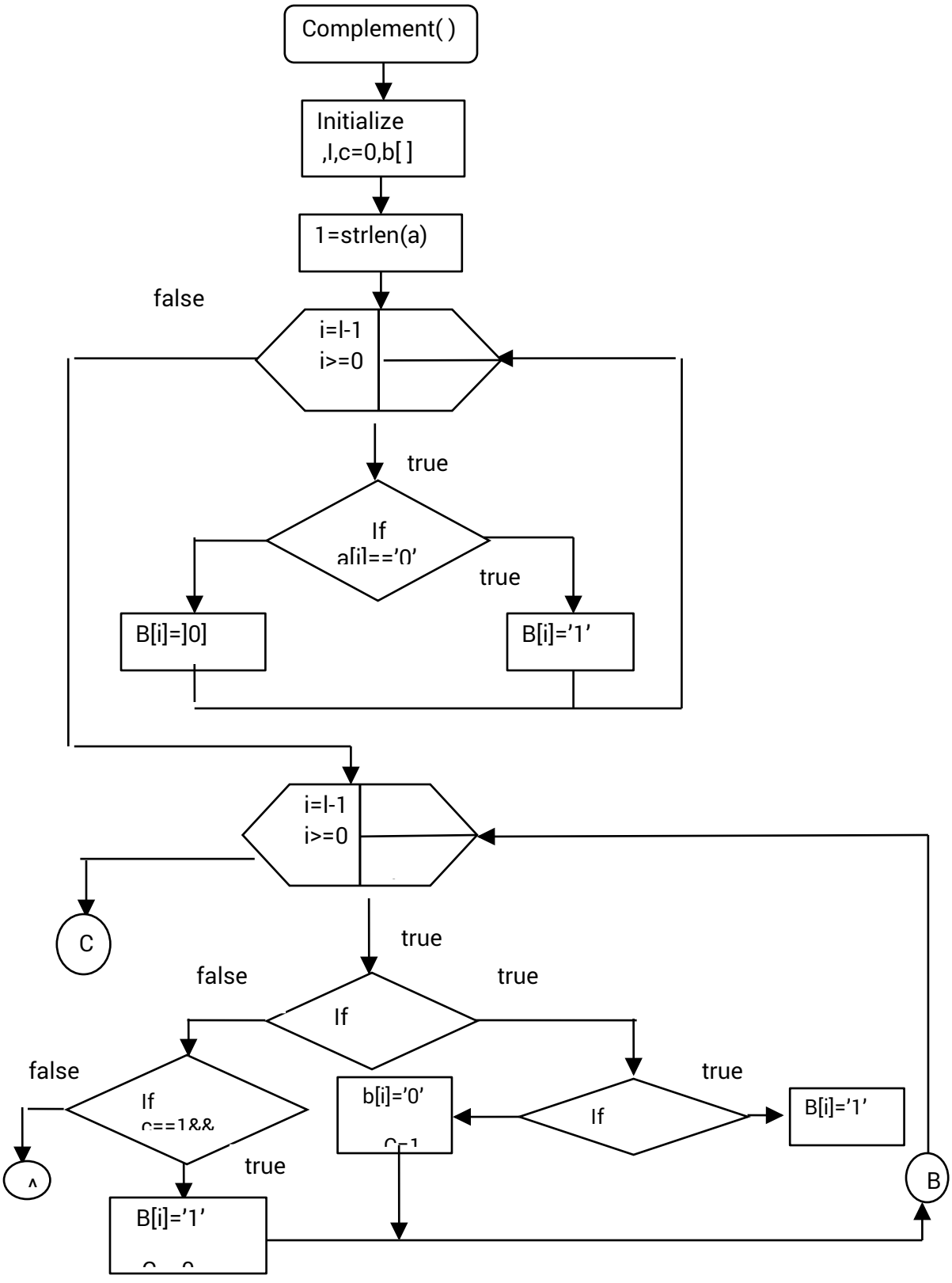
Step 8: if (c==1 && b[i]=='1') if True b[i]='1' and assigns the value to c=1

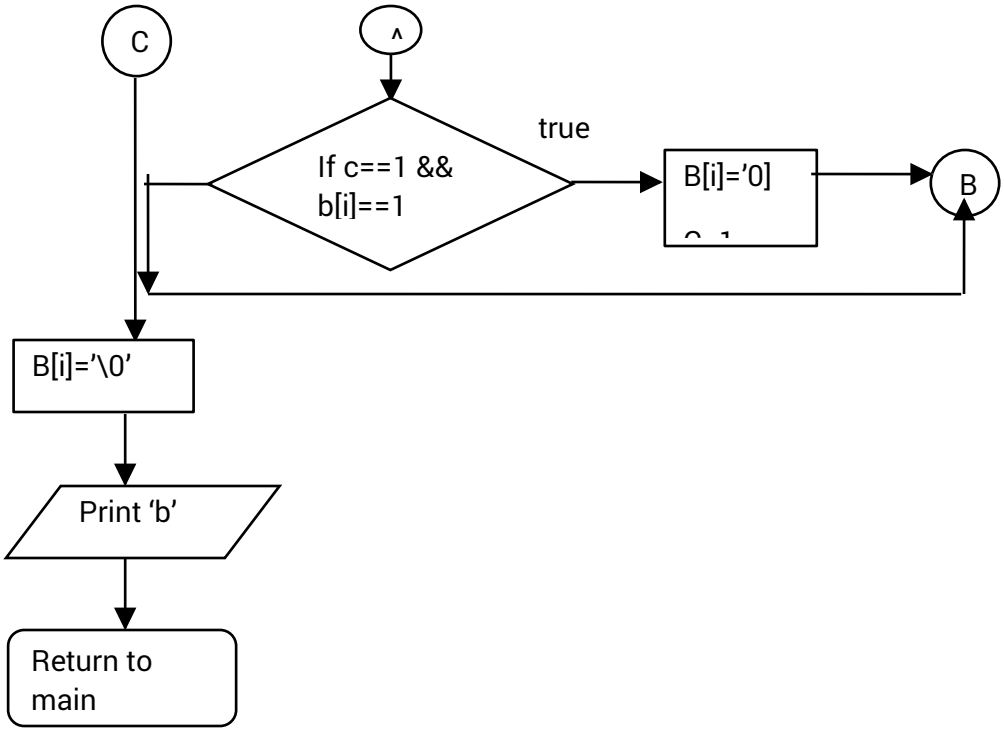
Step 9: b[i]='\0' and prints the 2's complement

Step 10: stop

Flow chart:


Start
Initialize I, a[ ]
Read a
i=0
i='\0'
true
If
a[i]!='0'&&a[i]!='1'
False
true
Call sub
progra
m
Number is
not valid
Stop

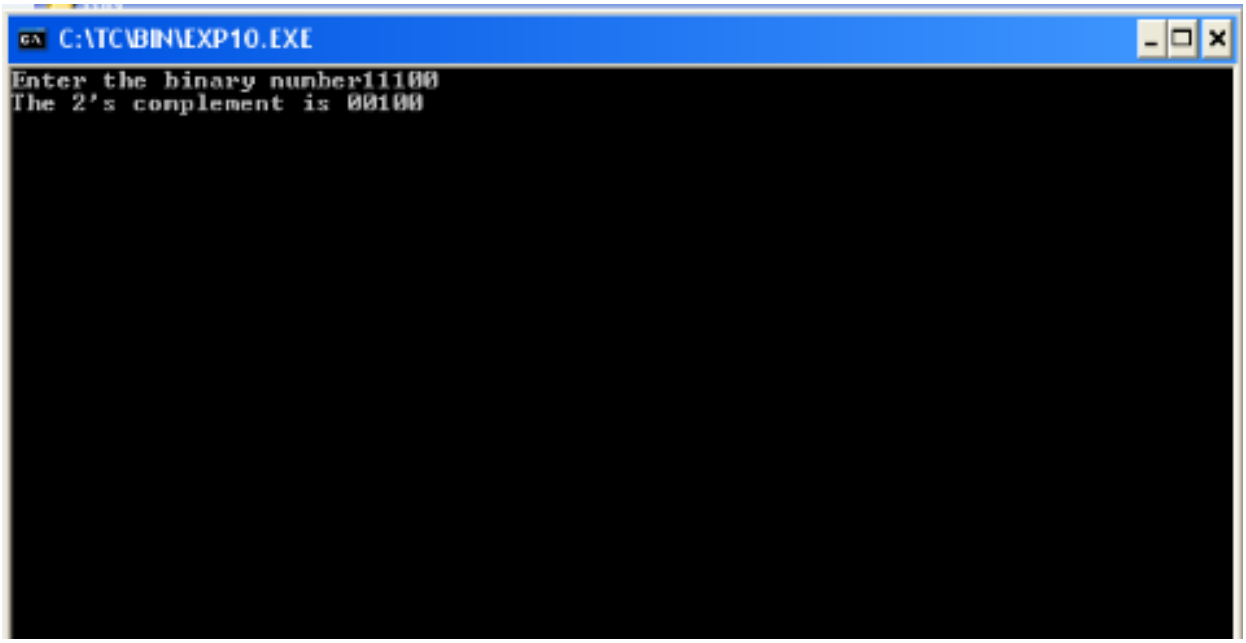

Complement( )
Initialize
,l,c=0,b[ ]
1=strlen(a)
false
i=l-1
i>=0
true
If
a[i]=='0'
true
B[i]=]0]
B[i]='1'
i=l-1
i>=0
C
true
false
true
If
false
If
c==1&&
b[i]='0'
If
true
B[i]='1'
^
true
B[i]='1'
B


C
^
If c==1 &&
b[i]==1
true
B[i]='0]
B
B[i]='\0'
Print 'b'
Return to
main

```
#include <stdio.h>
#include<conio.h>
void complement (char *a);
void main()
{
char a[16];
int i;
clrscr();
printf("Enter the binary number");
gets(a);
for(i=0;a[i]!='\0'; i++)
{
if (a[i]!='0' && a[i]! ='1')
{
printf("The number entered is not a binary number. Enter the correct number");
exit(0);
}
}
complement(a);
getch();
}
void complement (char *a)
{
int l, i, c=0;
char b[16];
l=strlen(a);
for (i=l-1; i>=0; i--)
{
if (a[i]=='0')
b[i]='1';
else
b[i]='0';
}
for(i=l-1; i>=0; i--)
{
if(i==l-1)
{
if (b[i]=='0')
b[i]='1';
else
{
b[i]='0';
c=1;
}
}
else
{
```

```
if(c==1 && b[i]=='0')
{
b[i]='1';
c=0;
}
else if (c==1 && b[i]=='1')
{
b[i]='0';
c=1;
}
}
}
}
b[l]='\0';
printf("The 2's complement is %s", b);
}
```

Output:



2.b) . Write a ‘C’ Program to find the roots of a quadratic equation.

Algorithm:

Step 1: start

Step 2: declare root1,root2

Step 3: input a,b,c

Step 4: calculate d=b*b-(4*(a*c))

Step 5: d<0 if True goto step 6

False goto step 7

Step 6: display complex numbers by using

i)-b/(2*a)

ii) sqrt(-d)/(2*a), -sqrt(-d)/(2*a)

Step 7: d==0 if True goto step 8

False goto step 9

Step 8: display roots are equal by calculating r1

i) r1=-b/(2*a)

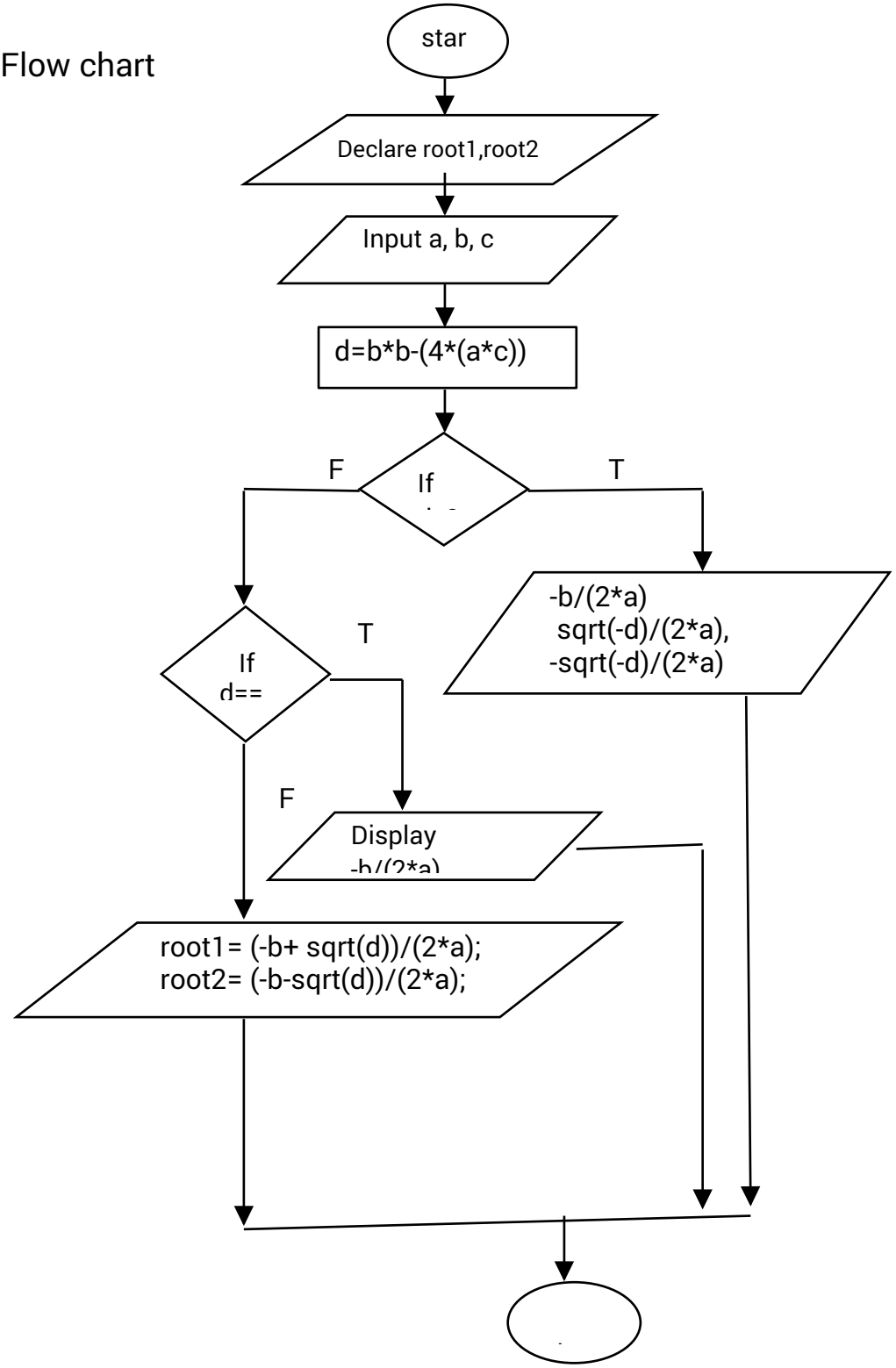
Step 9: display roots are real as root1, root2

root1= (-b+ sqrt(d))/(2*a);

root2= (-b-sqrt(d))/(2*a);

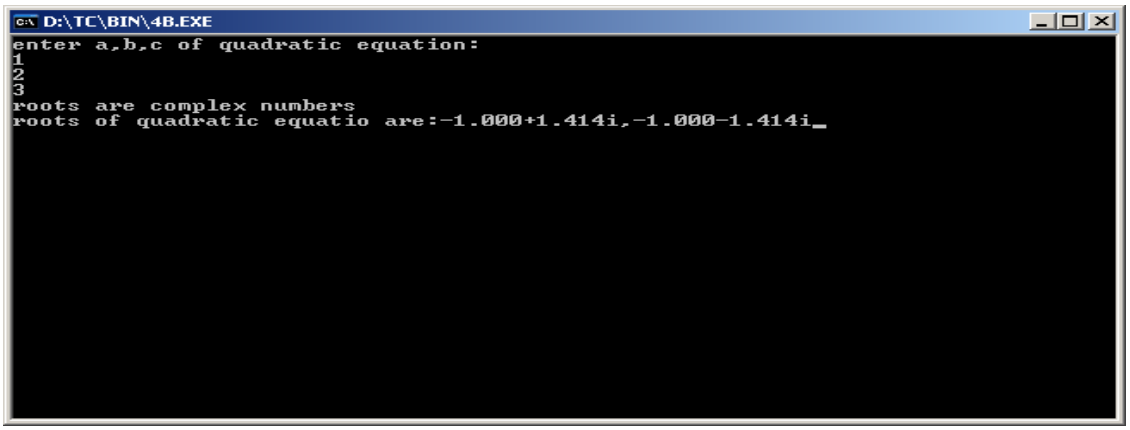
step 10:stop

Flow chart


star
Declare root1,root2
Input a, b, c
d=b*b-(4*(a*c))
F
If
T
-b/(2*a)
sqrt(-d)/(2*a),
-sqrt(-d)/(2*a)
If
d==
T
F
Display
root1= (-b+ sqrt(d))/(2*a);
root2= (-b-sqrt(d))/(2*a);

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main()
{
float a,b,c ;
float d,root1,root2;
clrscr();
printf("enter a,b,c of quadratic equation:\n");
scanf("%f%f%f",&a,&b,&c);
d=(b*b)-(4*(a*c));
if(d<0)
{
printf("roots are complex numbers\n");
printf("roots of quadratic equatio are:");
printf("%.3f%+.3fi",-b/(2*a),sqrt(-d)/(2*a));
printf(",%.3f%+.3fi",-b/(2*a),-sqrt(-d)/(2*a));
}
else if(d==0)
{
printf("both roots are equal.\n");
root1=-b/(2*a);
printf("root of quqdratic equation is: %.3f",root1);
}
else
{
printf("roots are real numbers.\n");
root1=(-b+sqrt(d))/(2*a);
root2=(-b-sqrt(d))/(2*a);
printf("roots of quadratic equation are:%.3f,%.3f",root1,root2);
}
getch();
}
```

Output:



2.c) Write a 'C' Program, which takes two integers operands and one operator from the user, performs the operation and prints the result. (Consider the operators +,-,*,/,% and use Switch Statement)

Algorithm:

Step 1: start

Step 2: Enter a, b values.

Step 3 :Print 'MENU'.

(i) Print '+ Addition'.
(ii) Print '- Subtraction'.
(iii) Print '* Multiplication'.
(iv) Print '/ Division'.
(v) Print '% Remainder'.
(vi) Print 'E Exit'.

Step 4: Print 'Enter your choice'.

Step 5: If op=='E' then goto step 8 otherwise follow the below steps

Step 6: Switch(op)

a. case +:

   i. Print 'Addition'.
   ii. c=a+b.
   iii. Print 'Sum='c.
   iv. break

b. case -:

   v. Print 'Subtraction'.
   vi. c=a-b.
   vii. Print 'Difference='c.
   viii. break

c. case *:

ix. Print ‘Multiplication’.
x. c=a*b.
xi. Print ‘Product=’c.
xii. break

d. case /:

xiii. Print ‘Division’.
xiv. c=a/b.
xv. Print ‘Quotient=’c.
xvi. break

e. case %:

xvii. Print ‘Remainder’.
xviii. c=a%b.
xix. Print ‘Remainder=’c.
xx. Break

f. case e:
exit(1)

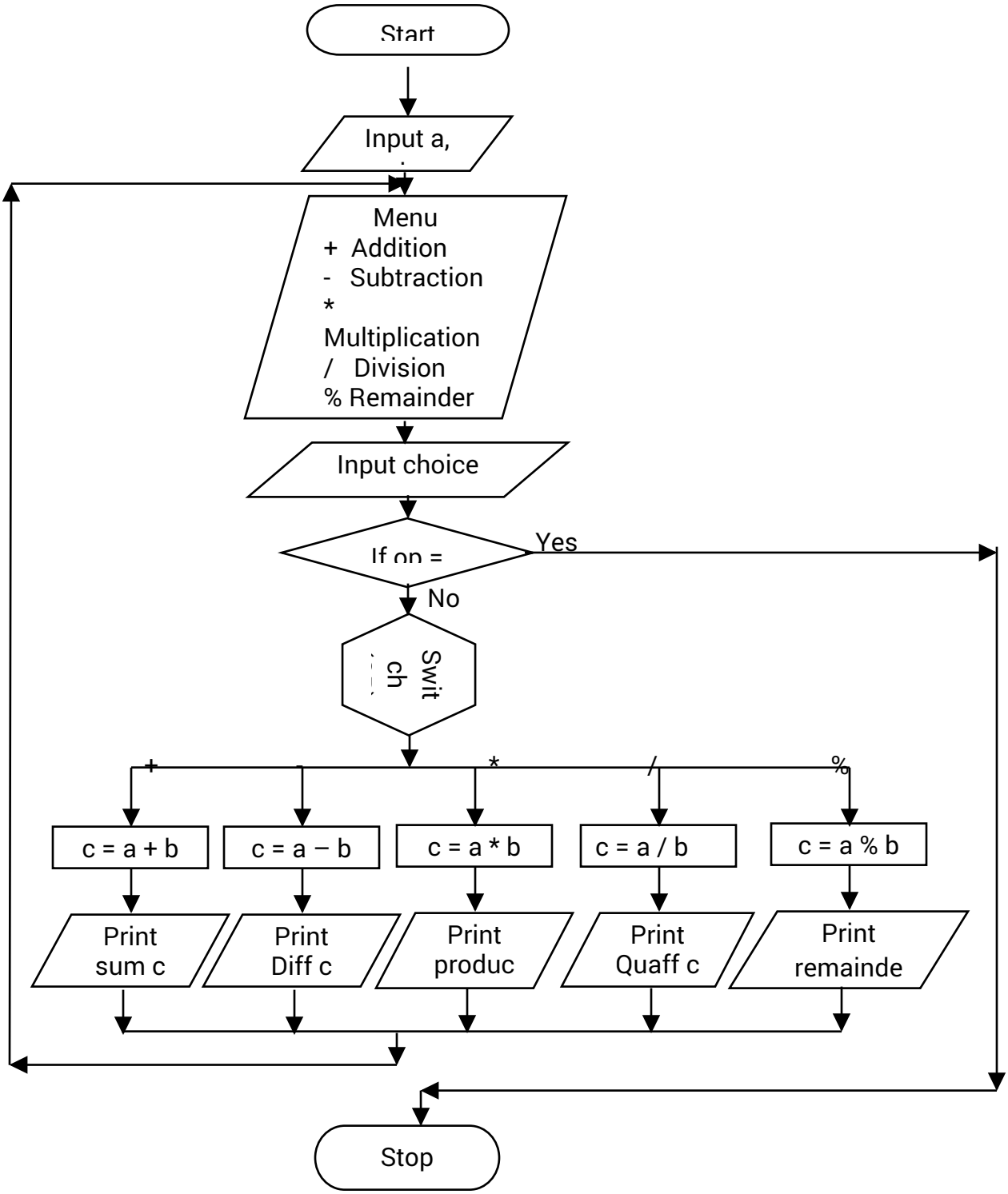
g. default:

xxi. Print ‘Invalid Option’.
xxii. Break

Step 7: while(1) then goto step 3.

Step 8: Stop.

Flowchart:


Start
Input a,
Menu
+ Addition
- Subtraction
*
Multiplication
/ Division
% Remainder
Input choice
If op =
Yes
No
Swit
ch
+
-
*
/
%
c = a + b
c = a – b
c = a * b
c = a / b
c = a % b
Print
sum c
Print
Diff c
Print
produc
Print
Quaff c
Print
remainde
Stop

Program:

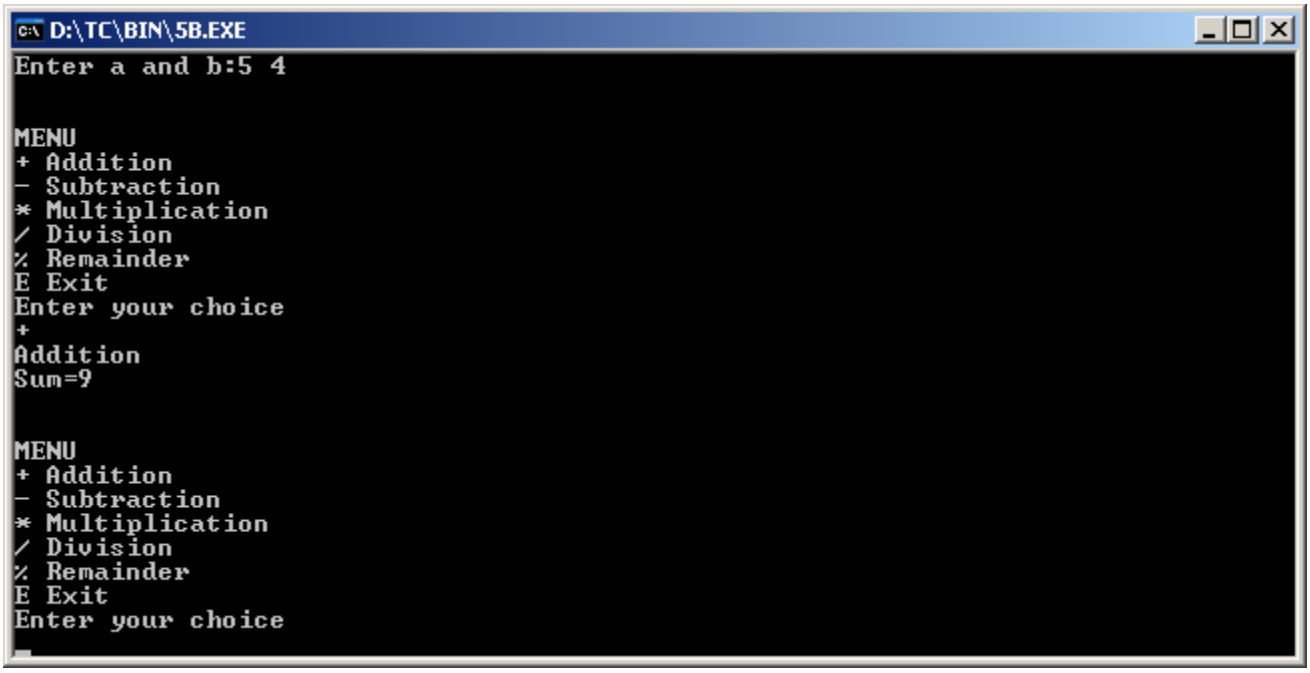
```
#include<stdio.h>
#include<conio.h>
#include<math.h>
main()
{
        int a, b, c;
        char op;
        clrscr();
        printf("Enter a and b:");
        scanf("%d%d",&a,&b);
        do{
                printf("\n\nMENU\n");
                printf("+ Addition\n");
                printf("- Subtraction\n");
                printf("* Multiplication\n");
                printf("/ Division\n");
                printf("% Remainder\n");
                printf("E Exit\n");
                printf("Enter your choice");
                getchar();
                op=getchar();
                if(op=='E'||op=='e')
                        exit(1);
                switch(op)
                {
                        case '+':
                                printf("Addition\n");
                                c=a+b;
                                printf("Sum=%d\n",c);
                                break;
                        case '-':
                                printf("Subtraction\n");
                                c=a-b;
                                printf("Difference=%d\n",c);
                                break;
                        case '*':
                                printf("Multiplication\n");
                                c=a*b;
                                printf("Product=%d\n",c);
                                break;

                         case '/':
                                printf("Division\n");
                                c=a/b;
```

```
                        printf("Quotient=%d\n",c);
                                                break; case '%':
                        printf("Remainder\n");
                        c=a%b;
                        printf("Remainder=%d\n",c);
                        break;
                default:
                        printf("Invalid Option\n");
                        break;
        }       /*end of switch statement*/
    }
    while(1);       /*End of while*/

}/*End of main function*/
```

Output:



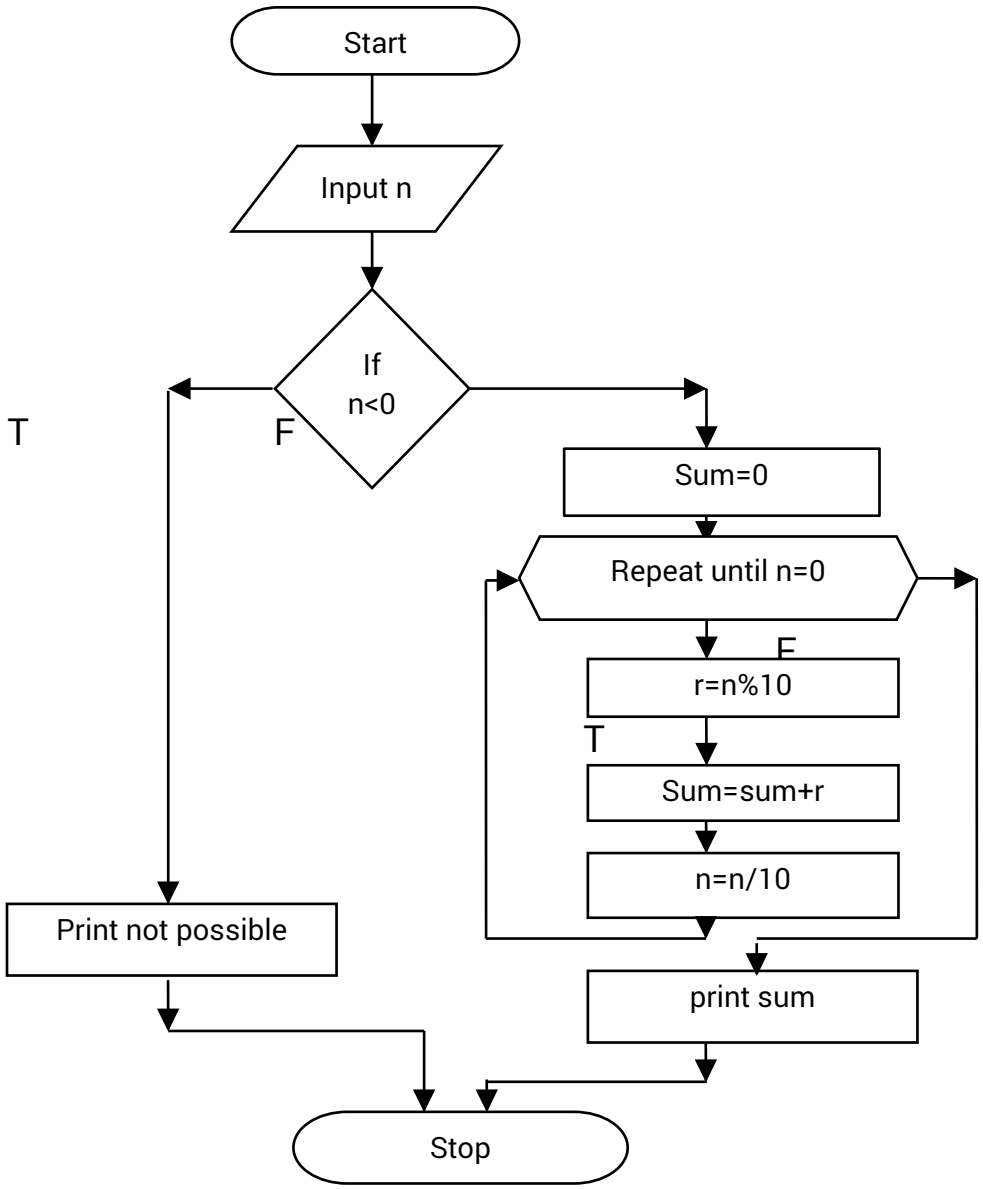
Exercise 3

a. Write a Program in C to find the sum of individuals Digits of a positive Integer.

Algorithm:

1) Input a number n.
2) Check n is +ve, if not goto step 6.
3) S=0,m=n.
4) Repeat the following until n=0.
   i. r=n%10.
   ii. sum=sum+r.
   iii. n=n/10.
5) print sum.
6) Stop

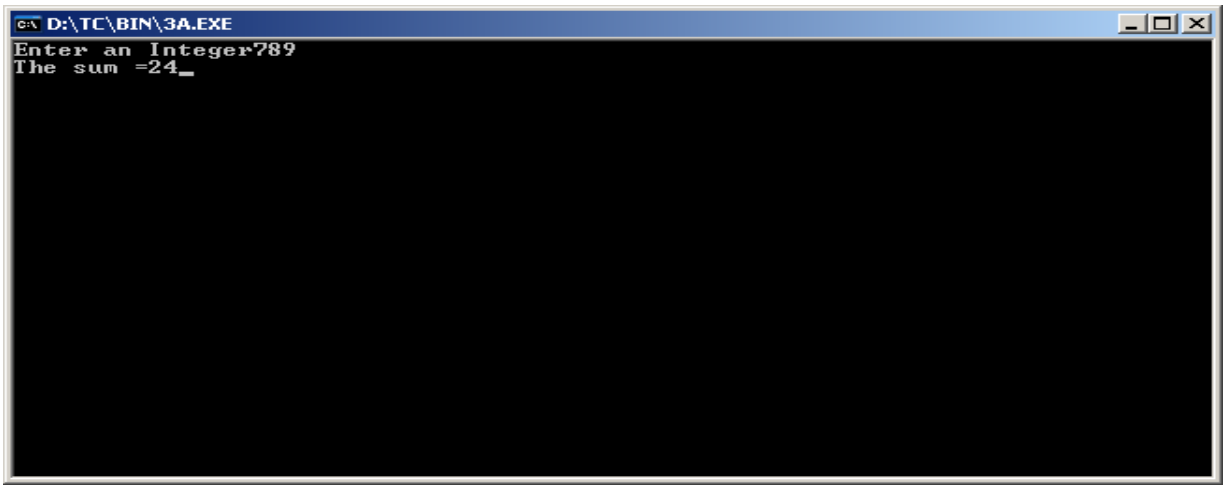
Flowchart



Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int r,n,sum;
        clrscr();
        printf("Enter an Integer");
        scanf("%d",&n);
        if(n<0)
        printf("The given number is not +ve Integer");
        else
        {
                sum=0;
                while(n!=0)
                {
                        r=n%10;
                        sum=sum+r;
                        n=n/10;
                }
                printf("The sum =%d",sum);
}
        getch();
}
```

Output:



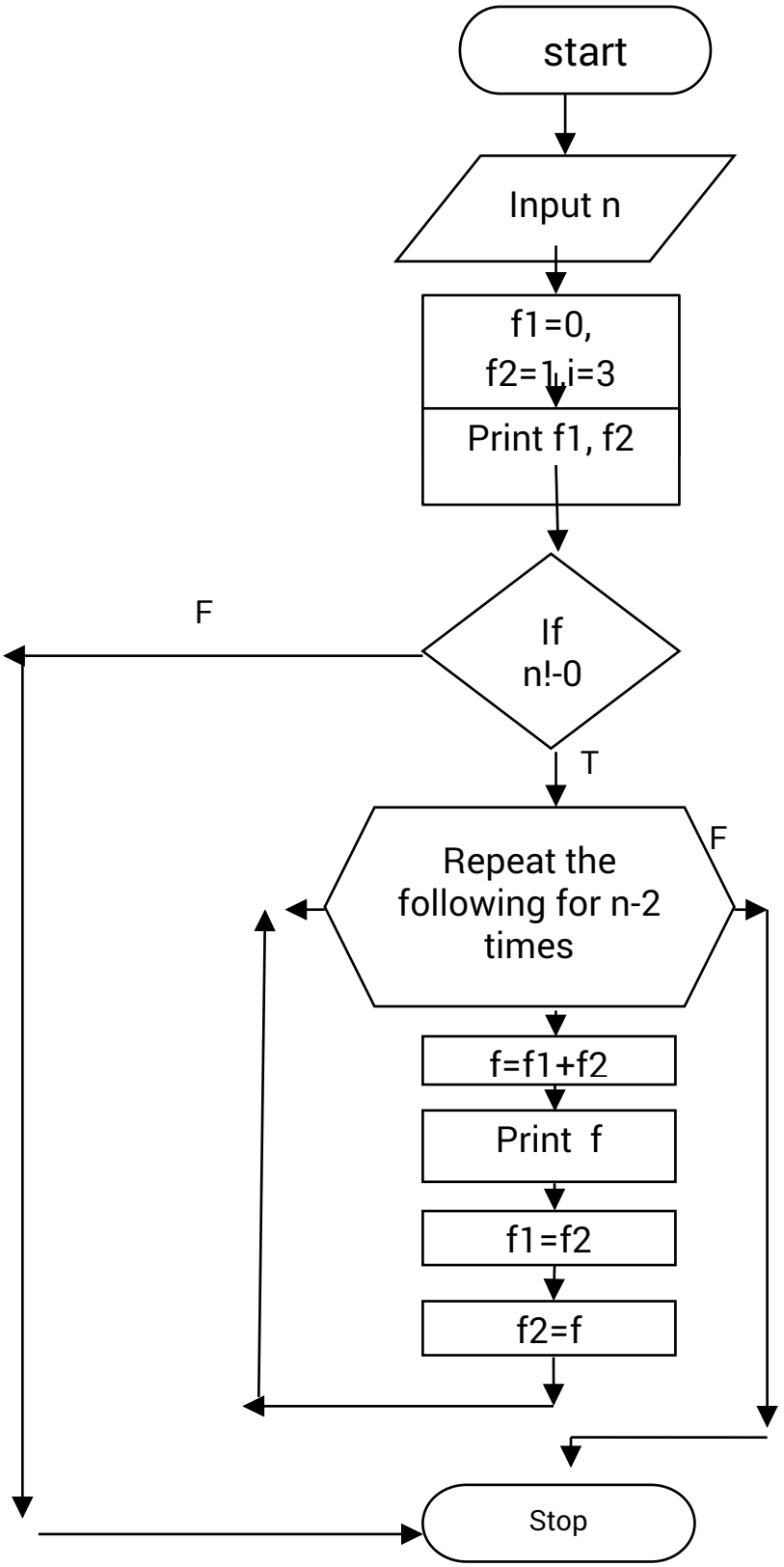
b. Write a program a Fibonacci sequence is defined as follows.

The first and second terms in the sequence are 0 and 1. Subsequent terms are found by adding the precedes two preceding two terms in the sequence, Write a C Program to Generate the first 'n' terms of the sequence.

Algorithm

1. Enter the no of Fibonacci, n to be generated.
2. f1=0
3. f2=1
4. print f1,f2
5. repeat the following for n-2 times.
    a. f=f1+f2
    b. print f
    c. f1=f2
    d. f2=f
6. stop.

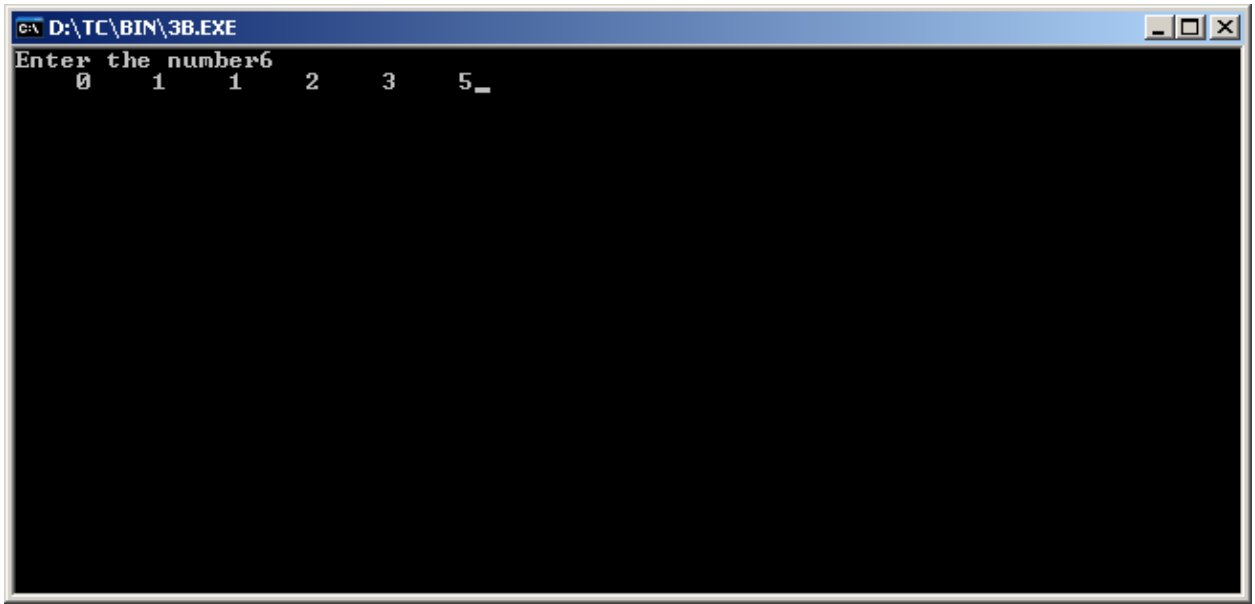
Flowchart :


start
Input n
f1=0,
f2=1, i=3
Print f1, f2
If
n!-0
F
T
Repeat the following for n-2 times
F
f=f1+f2
Print f
f1=f2
f2=f
Stop

Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
        int f1,f2,f;
        int i,n;
        clrscr();
        f1=0;
        f2=1;
        printf("Enter the number");
        scanf("%d",&n);
        if(n!=0)
        {
          printf("%5d%5d",f1,f2);
                i=3;
                while(i<=n)
                {
               f=f1+f2;
              printf("%5d",f);
                        f1=f2;
                        f2=f;
                    i++;
                }
        }
        getch();
}
```

Output:

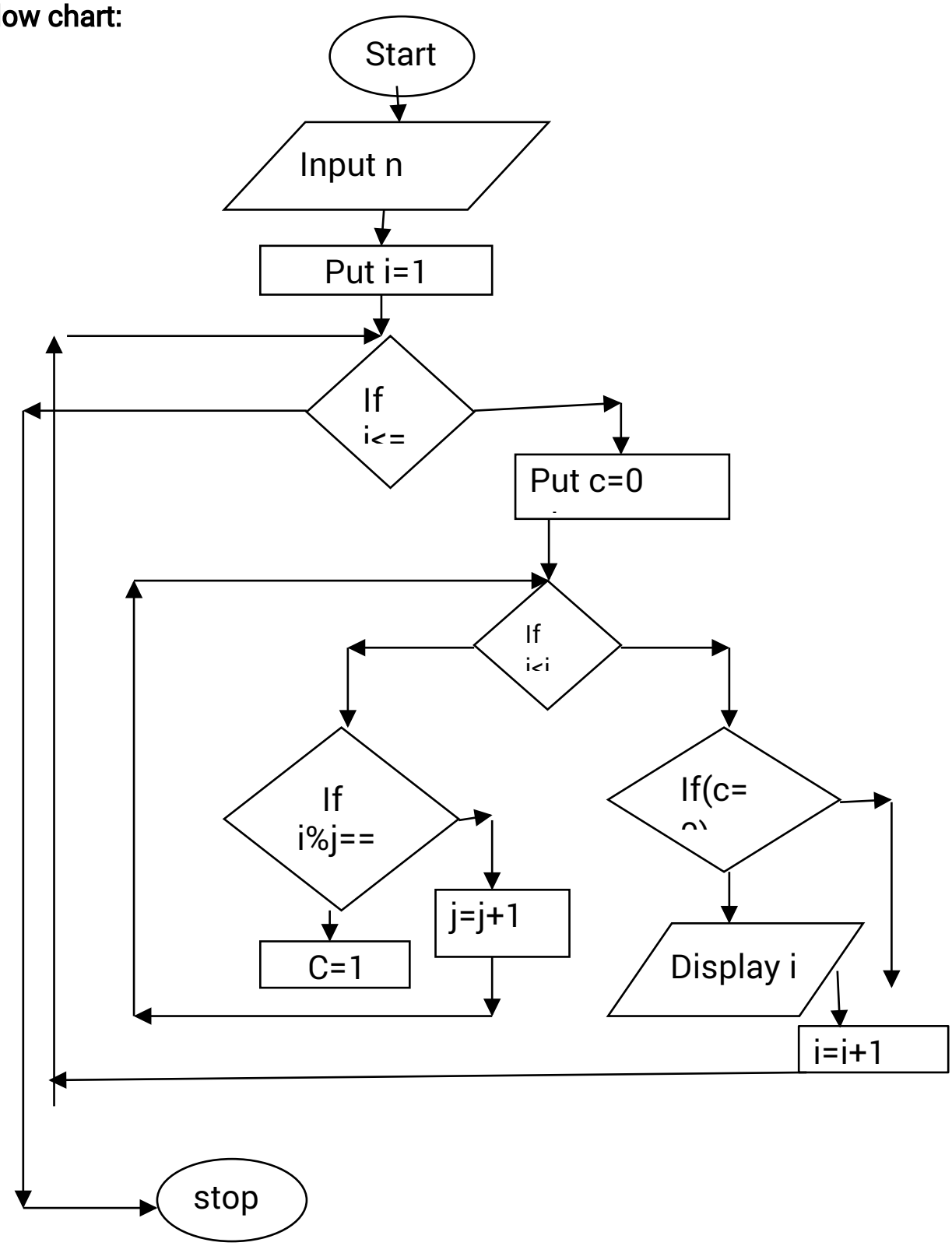


C. Write a C program to generate all the prime numbers between 1 and n, where n is a value supplied by the user.

Algorithm:

Step 1: start
Step 2: input n
Step 3 put i=1
Step 4: i<=n if True goto step 5
False goto step 9
Step 5: put c=0 and initialize j=2
Step 6: j<i if True goto step 7
False goto step 9
Step 7: (i%j==0) if True goto step 7
False goto step 6 by incrementing j
Step 8: assign c=1
Step 9: if c==0 display I otherwise increment i
Step 10: stop

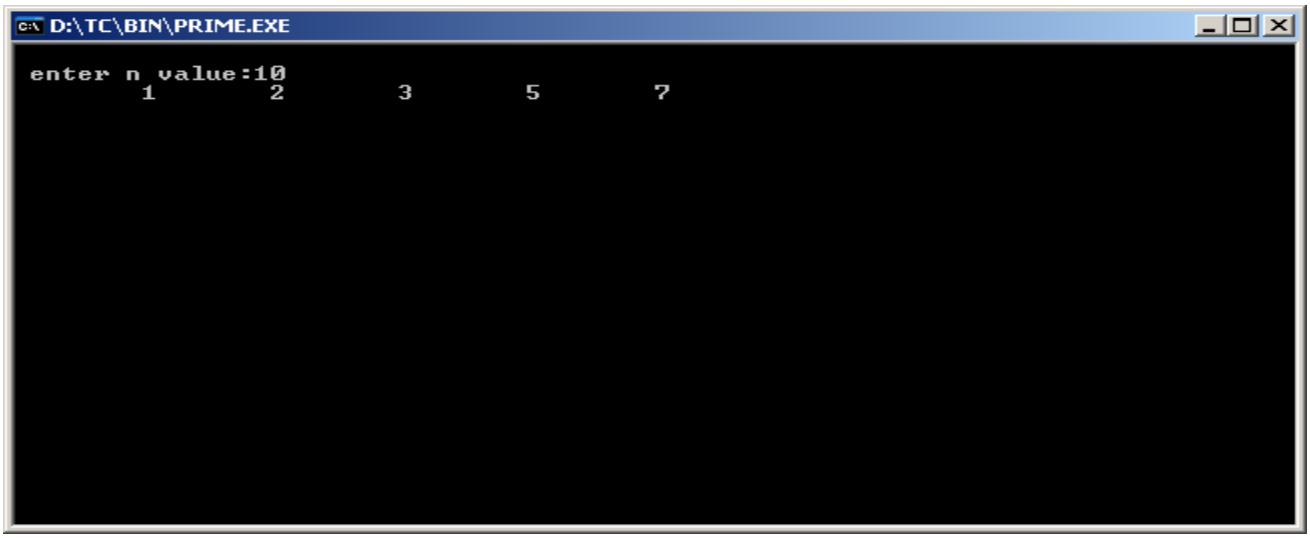
Flow chart:


Start
Input n
Put i=1
If
Put c=0
If
If
i%j==
If(c=
j=j+1
C=1
Display i
i=i+1
stop

Program:

```
#include<stdio.h>
#include<conio.h>
#include<stdio.h>
void main()
{
int n,i,c,j;
clrscr();
printf("\n enter n value:");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
c=0;
for(j=2;j<i;j++)
{
if(i%j==0)
{
c=1;
break;
}
}
if(c==0)
printf("\t%d",i);
}
getch();
}
```

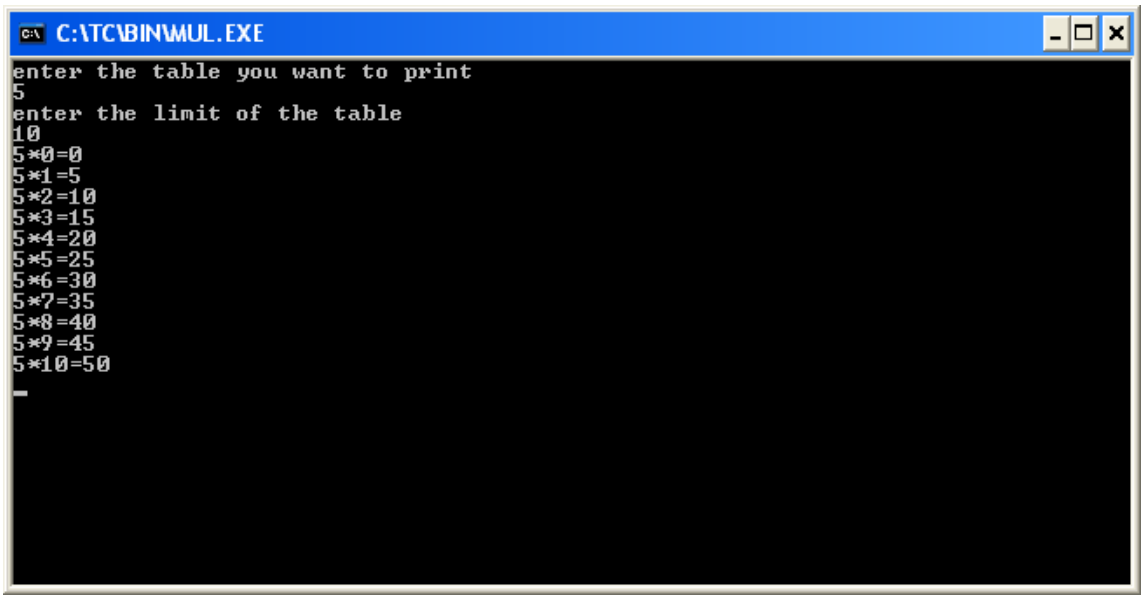
Output:



Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n,d,i;
clrscr();
printf("enter the table you want to print\n");
scanf("%d",&n);
printf("enter the limit of the table\n");
scanf("%d",&d);
for(i=0;i<=d;i++)
printf("%d*%d=%d\n",n,i,n*i);
getch();
}
```

Output:



b) Write a C Program to enter a decimal number , and calculate and display binary equivalent of that number.

Algorithm:

Step 1: start

Step 2: initialize array as static and declare the variables n,i

Step 3: read the value of decimal number

Step 4: for i=0 to 15 if n>0 True goto step 5

False goto step 6

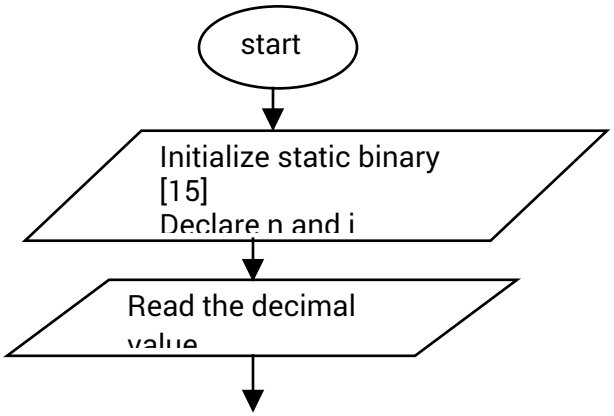
Step 5:set binary[i]=n%2

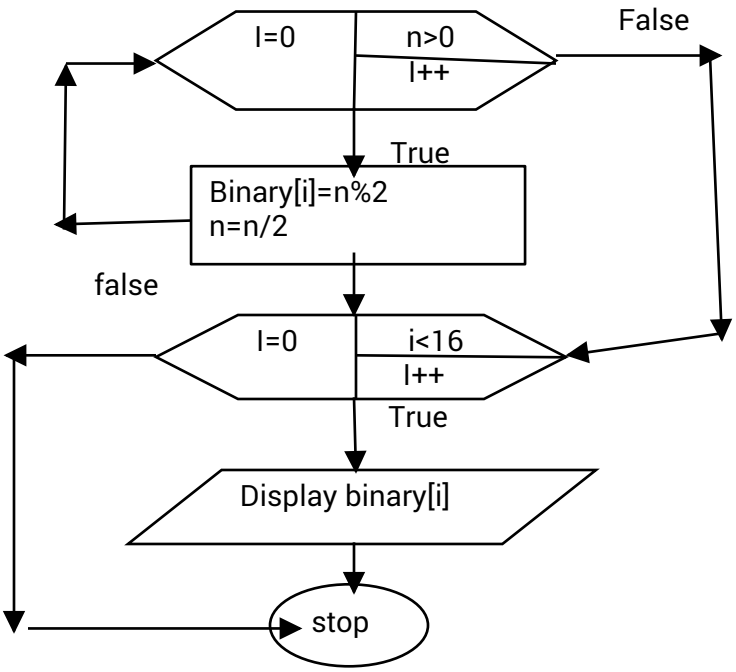
n=n/2

step 6:for i=0 to 16 display the value of binary [i].

step 7: stop

Flowchart:





Program

```
#include<stdio.h>
#include<stdio.h>
void main()
{
  int n,i;
  static int binary[16];
  clrscr();
  printf("\nEnter Decimal Number  ");
  scanf("%d",&n);
  for(i=15;n>0;i--)
  {
     binary[i]=n%2;
     n=n/2;
  }
  printf("\n The equivalent binary number is  ");
   for(i=0;i<16;i++)
   {
     printf("%d",binary[i]);
   }
getch();
}
```

Output:

c)Write a C program to check whether the given number is Armstrong number or not

Algorithm: Step 1: start

Step 2: initialize arm=0 and declare the variables temp,n,s

Step 3: read the value of n

Step 4:assign n value to temp variable

Step 5: repeat these steps until n!=0

s =n%10

Arm=arm+(s*s*s)

n=n\10

step 6:if temp==n True goto step 8

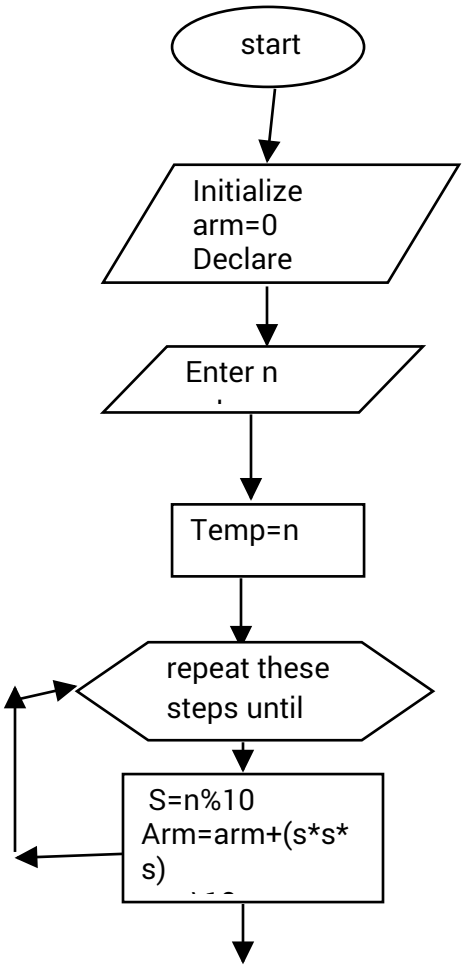
False goto step 9

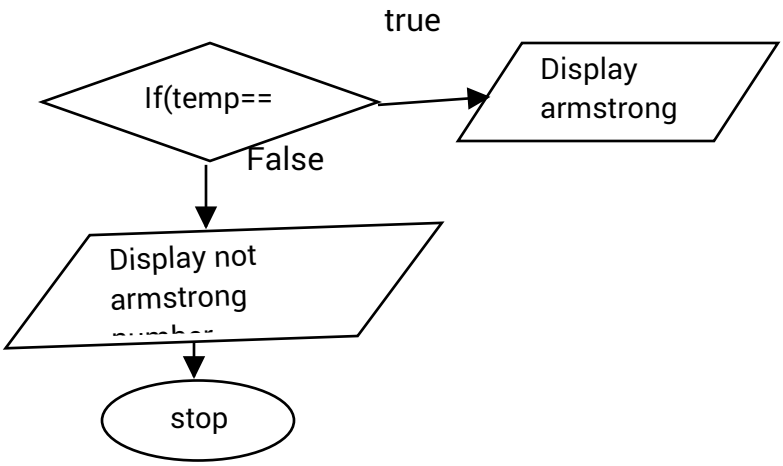
Step 7: display Armstrong number

Step 8: display not an Armstrong number

Step 9;stop.

Flow chart:

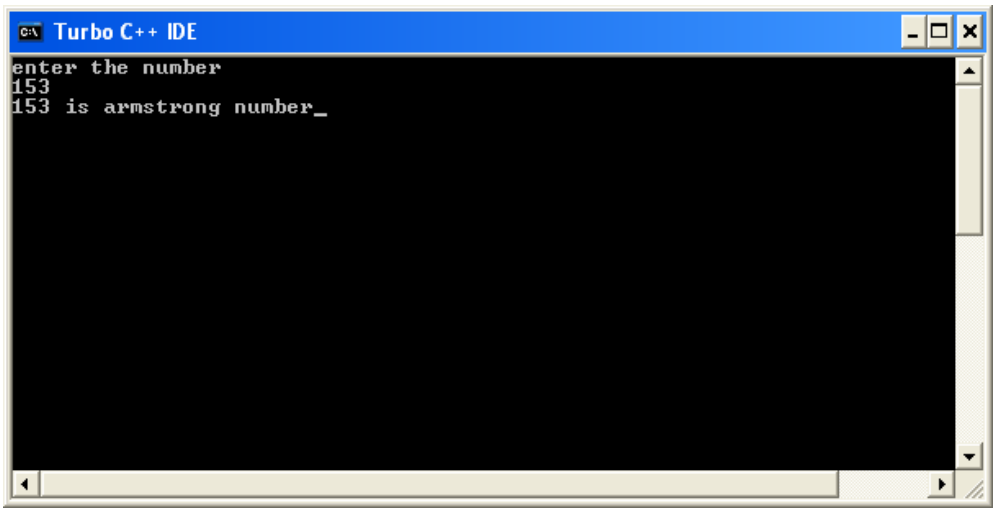

start
Initialize
arm=0
Declare
Enter n
Temp=n
repeat these
steps until
S=n%10
Arm=arm+(s*s*
s)



Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int arm=0,temp,n,s;
clrscr();
printf("enter the number\n");
scanf("%d",&n);
temp=n;
while(n!=0)
{
s=n%10;
arm=arm+(s*s*s);
n=n/10;
}
if(temp==arm)
{
printf("%d is armstrong number",temp);
}
else
printf("%d is not armstrong number",temp);
getch();
}
```

Output:



Exercise 5

b) write a C program to implement a linear search.

Algorithm:

Step 1: start

Step 2: declare the variables i,n,s and array a[10]

Step 3: read the value of n

Step 4: for i=0 to n read the values into array until i<n

Step 5: read the search element

Step 6:for i=0 to n if i<n true goto step 7

False goto step 9

Step 7:if (s==a[i]) true goto step 8

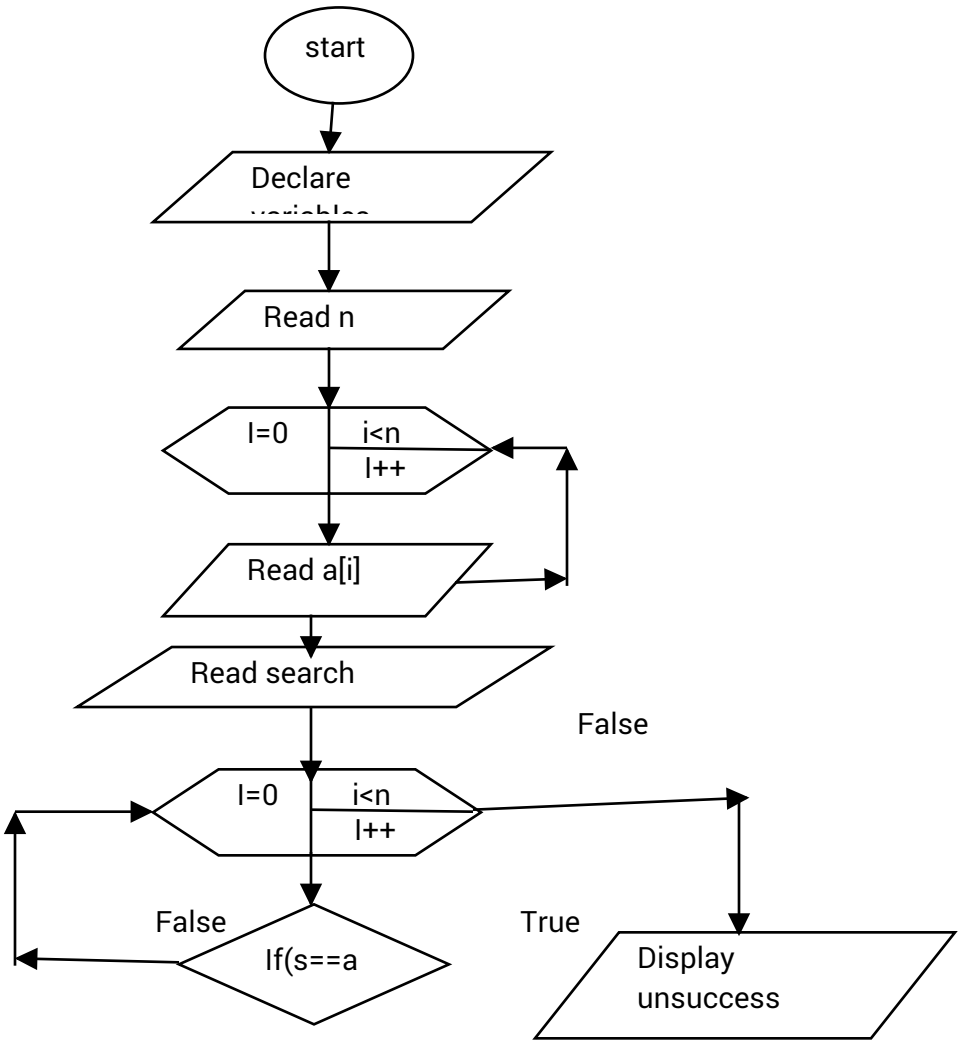
False increment i value

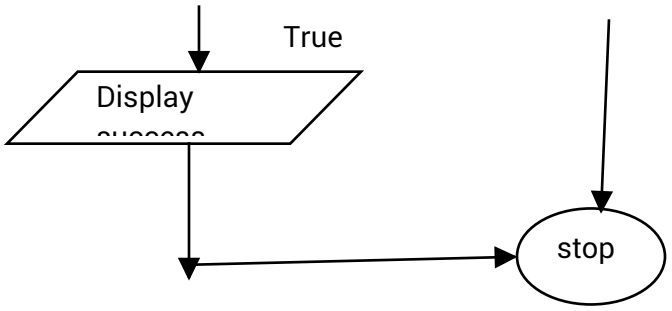
Step 8:display success

Step 9: display unsucess

Step 10:stop

Flow chart:


start
Declare
Read n
I=0
i<n
I++
Read a[i]
Read search
False
I=0
i<n
I++
False
If(s==a
True
Display
unsuccess

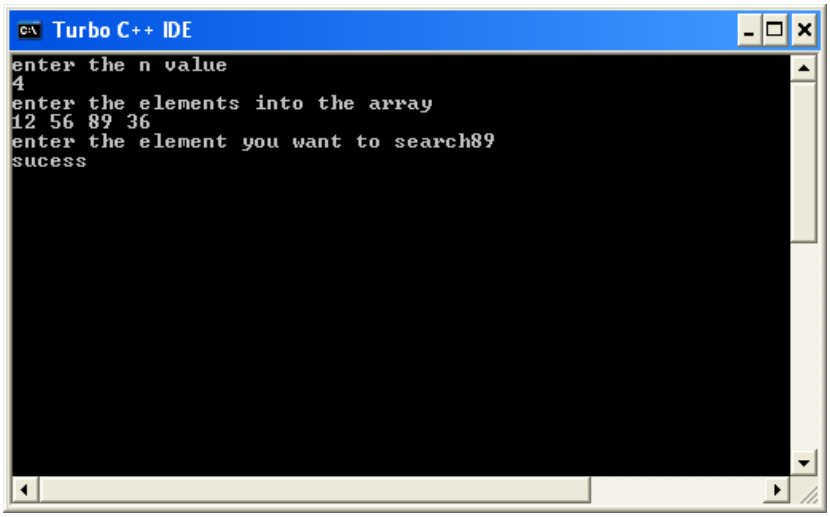


Program:

```
#include<stdio.h>
#include<conio.h>
#include<process.h>
void main()
{
int i,n,s,a[10];
clrscr();
printf("enter the n value\n");
scanf("%d",&n);
printf("enter the elements into the array\n");
for(i=0;i<n;i++)
scanf("%d",&a[i]);
printf("enter the element you want to search");
scanf("%d",&s);
for(i=0;i<n;i++)
{
  if(s==a[i])
  {
  printf("sucess");
  getch();
  exit(0);
  }
}
```

```
printf("unsucess");
getch();
}
```

Output:



Exercise 6

b) write a c program to input two m*n matrices ,check the compatibility and perform addition and multiplication of them

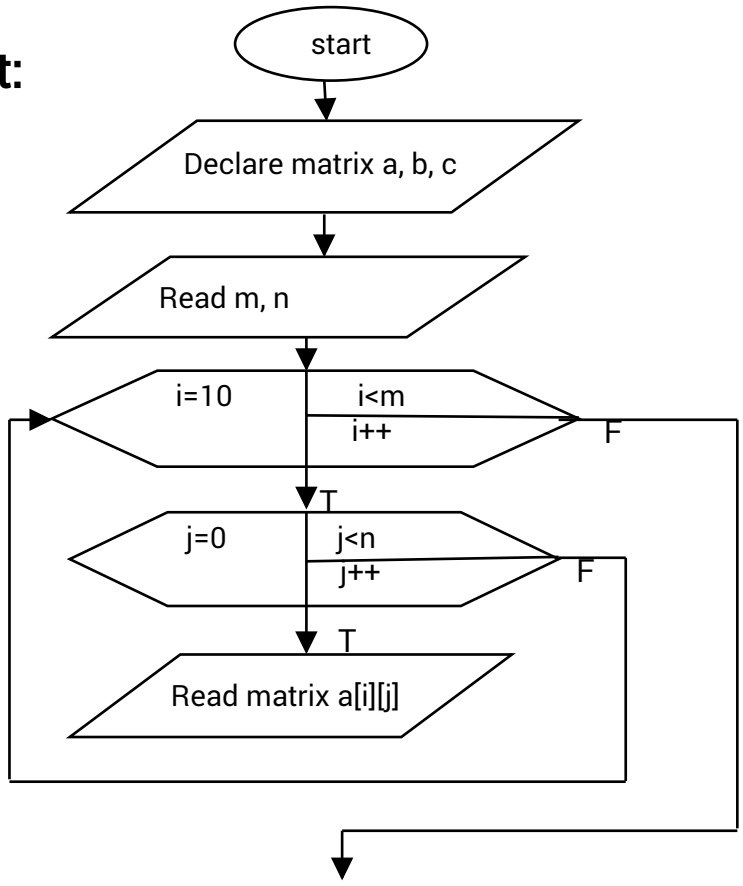
a) Addition of Two Matrices

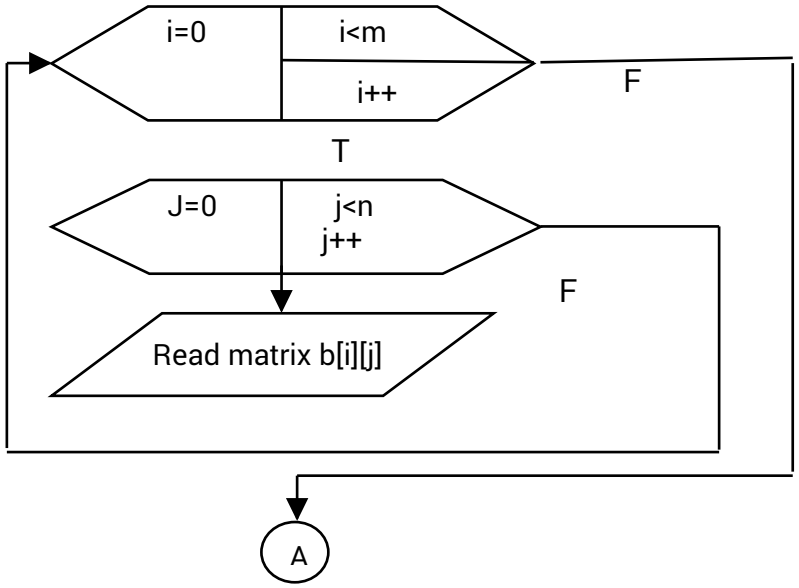
Algorithm:

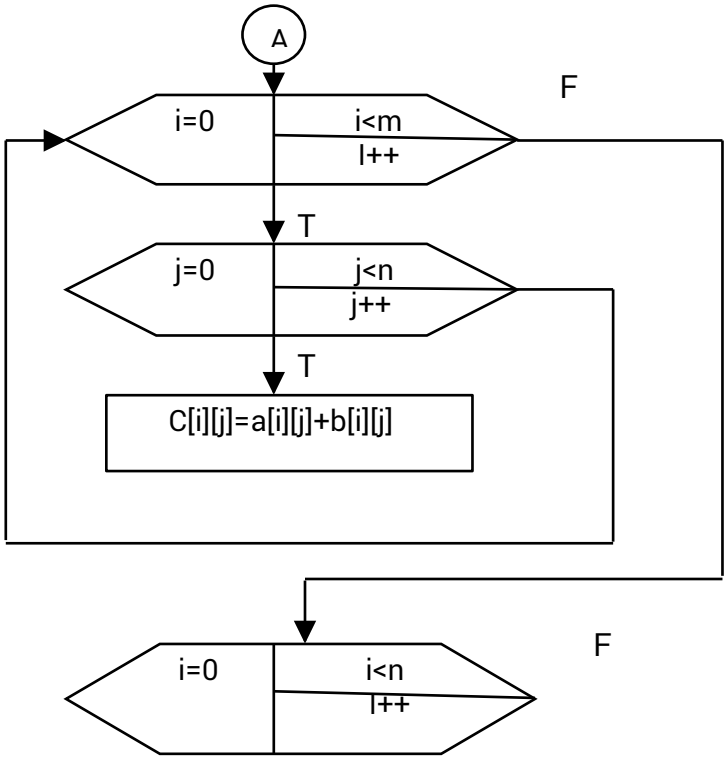
Step 1: start
Step 2: declare a[10][10],b[10][10],c[10][10].
Step 3: enter rows and columns for a & b Matrices (i.e. m& n).
Step 4: for i=0 to m for rows[a] if i<m True goto step 5
False goto step 7
Step 5: for j=0 to n for columns[a] if j<n True goto step 6
False goto step 3 by incrementing i value by one
Step 6: read elements into matrix (i.e. a[i][j]) and increment j value by one
Step 7: for i=0 to m for rows[b] if i<m True goto step 8
False goto step 10
Step 8: for j=0 to n for columns[b] if j<n True goto step 9
False goto step 7 by incrementing i value by one
Step 9: read elements into matrix (i.e. b[i][j]) and increment j value by one
Step 10: for i=0 to n for rows[c] if i<m True goto step 11
False goto step 13
Step 11: for j=0 to n for columns[c] if j<n True goto step 12
False goto step 10 by incrementing i value by

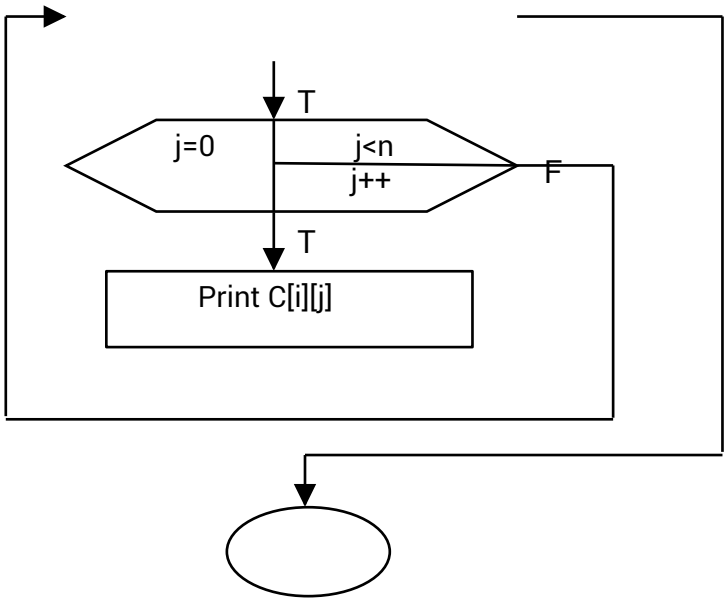
one
Step 12: calculating sum of a & b (i.e. c[i][j]=a[i][j]+b[i][j]) and increment j value by one
Step 13: for i=0 to m for rows[c] if i<m True goto step 14
False goto step 16
Step 14: for j=0 to n for columns[c] if j<n True goto step 15
False goto step 13 by incrementing i value by one
Step 15: display the elements of matrix c (i.e. c[i][j])
Step 16: stop

Flow chart:




i=0
i<m
i++
F
T
J=0
j<n
j++
F
Read matrix b[i][j]
A

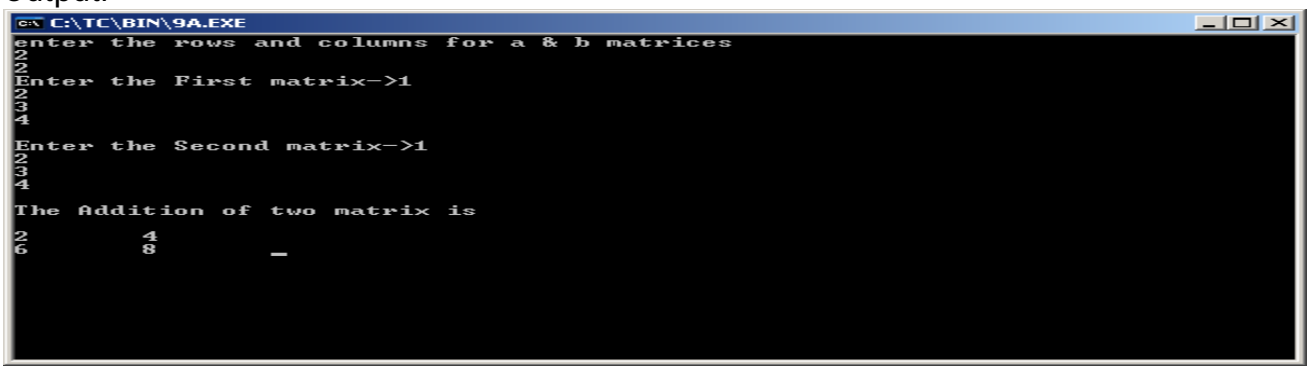

A
F
i=0
i<m
I++
T
j=0
j<n
j++
T
C[i][j]=a[i][j]+b[i][j]
F
i=0
i<n
I++




```
//program
#include<stdio.h>
#include<conio.h>
void main()
{
 int a[10][10],b[10][10],c[10][10],i,j,m,n;
 printf("enter the rows and columns for a & b matrices\n");
 scanf("%d%d",&m,&n);
 printf("Enter the First matrix->");
 for(i=0;i<m;i++)
 {
  for(j=0;j<n;j++)
  {
  scanf("%d",&a[i][j]);
  }
 }
 printf("\nEnter the Second matrix->");
 for(i=0;i<m;i++)
 {
  for(j=0;j<n;j++)
  {
    scanf("%d",&b[i][j]);
  }
 }
  for(i=0;i<m;i++)
```

```
  {
     for(j=0;j<n;j++)
     {
     c[i][j]=a[i][j]+b[i][j];
     }
  }
  printf("\nThe Addition of two matrix is\n");
  for(i=0;i<m;i++)
  {
     printf("\n");
     for(j=0;j<n;j++)
     printf("%d\t",c[i][j]);
  }
  getch();
}
```

Output:



Matrix multiplication by checking compatibility

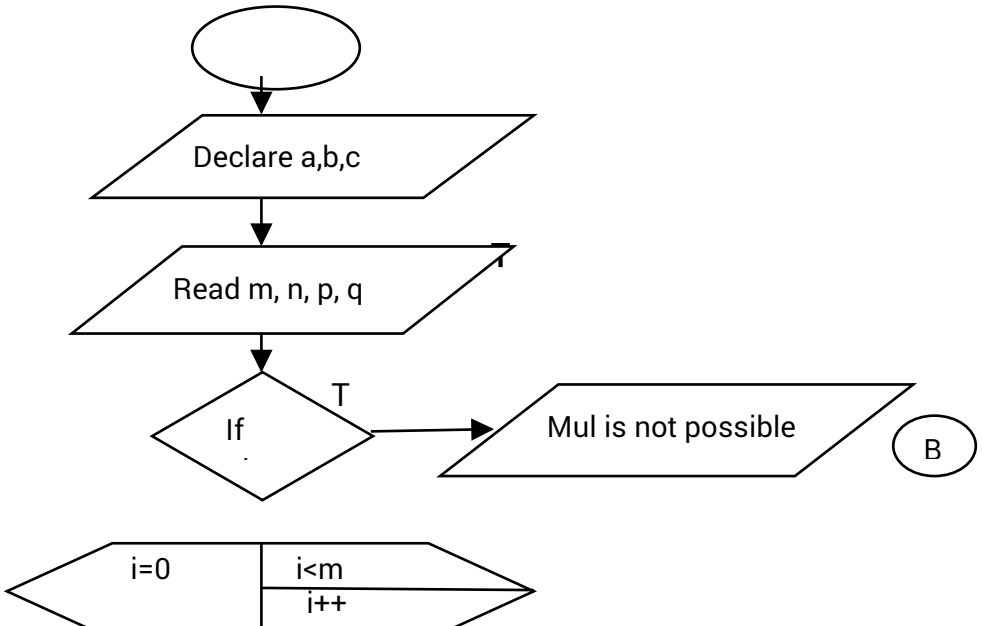
Algorithm:

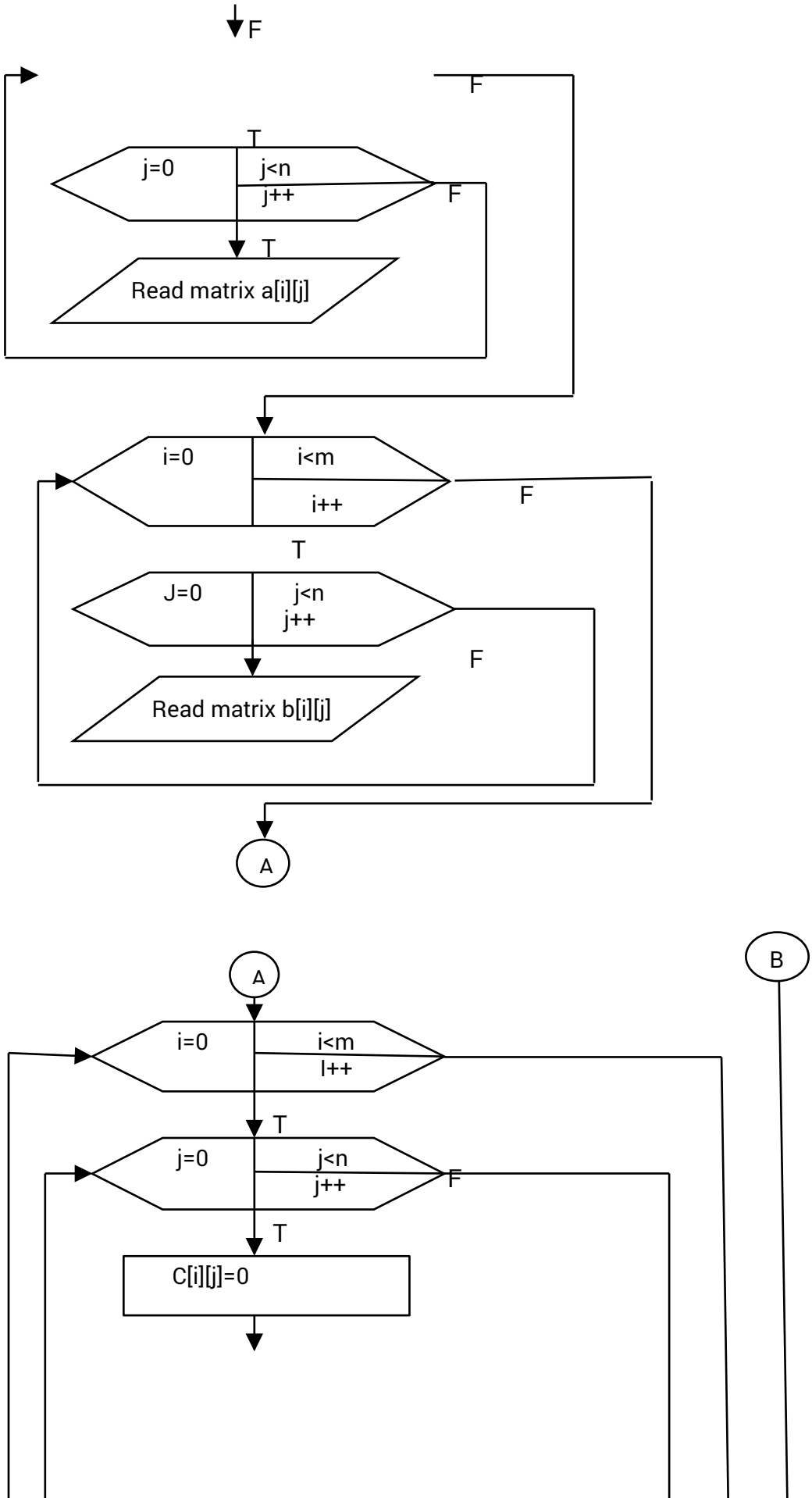
Step 1: start
Step 2: declare a[10][10],b[10][10],c[10][10].
Step 3: enter rows and columns for a Matrix (i.e. m & n).
Step 4: enter rows and columns for b Matrix (i.e. p & q)
Step 5: n! =p if True goto step 6
False goto step 7
Step 6: matrix multiplication is not possible
Step 7: for i=0 to m for rows[a] if i<m True goto step 8
False goto step 10
Step 8: for j=0 to n for columns[a] if j<n True goto step 9
False goto step 3 by incrementing i value by one
Step 9: read elements into matrix (i.e. a[i][j]) and increment j value by one
Step 10: for i=0 to m for rows[b] if i<m True goto step 11
False goto step 13
Step 11: for j=0 to n for columns[b] if j<n True goto step 12

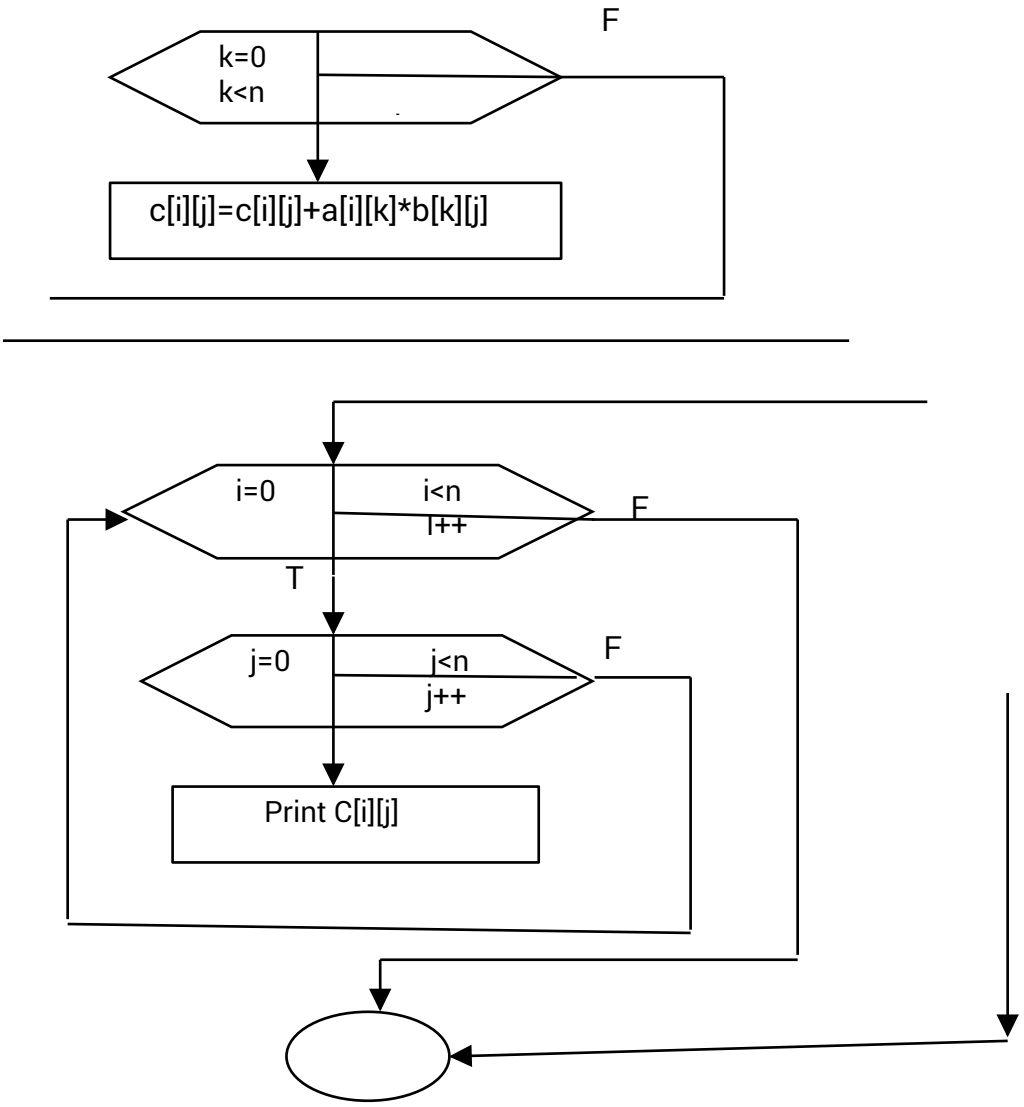
False goto step 10 by incrementing i value by one
Step 12: read elements into matrix (i.e. b[i][j]) and increment j value by one
Step 13: for i=0 to n for rows[c] if i<m True goto step 13
False goto step 18
Step 14: for j=0 to n for columns[c] if j<n True goto step 15
False goto step 13 by incrementing i value by one
Step 15: assign c[i][j]=0
Step 16: for k=0 to n if k<n True goto step 17
False goto step 14
Step 17: calculate product by c[i][j]=c[i][j]+a[i][k]*b[k][j]

Step 18: for i=0 to n for rows[c] if i<m True goto step 19
False goto step 21
Step 19: for j=0 to n for columns[c] if j<n True goto step 15
False goto step 18by incrementing i value by one
Step 20: print c[i][j]
Step 21 : stop.

Flow chart:




F
F
T
j=0
j<n
j++
F
T
Read matrix a[i][j]
i=0
i<m
i++
F
T
J=0
j<n
j++
F
Read matrix b[i][j]
A
A
B
i=0
i<m
I++
T
j=0
j<n
j++
F
T
C[i][j]=0

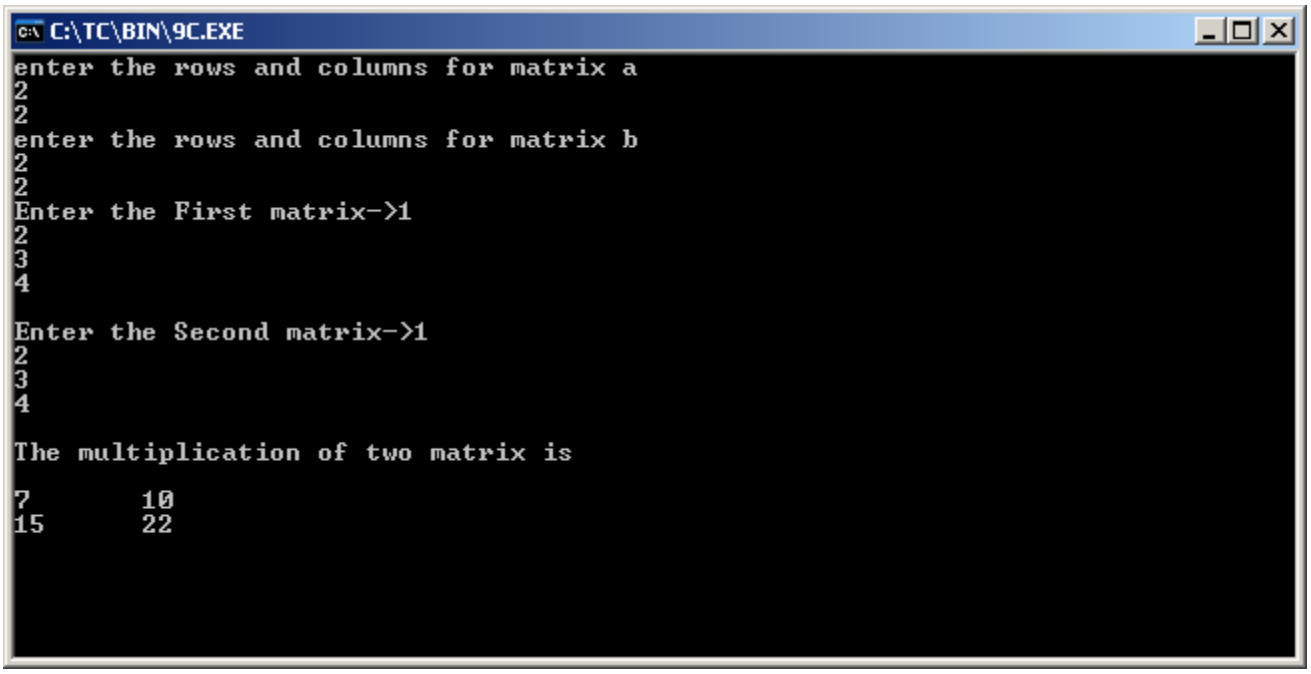


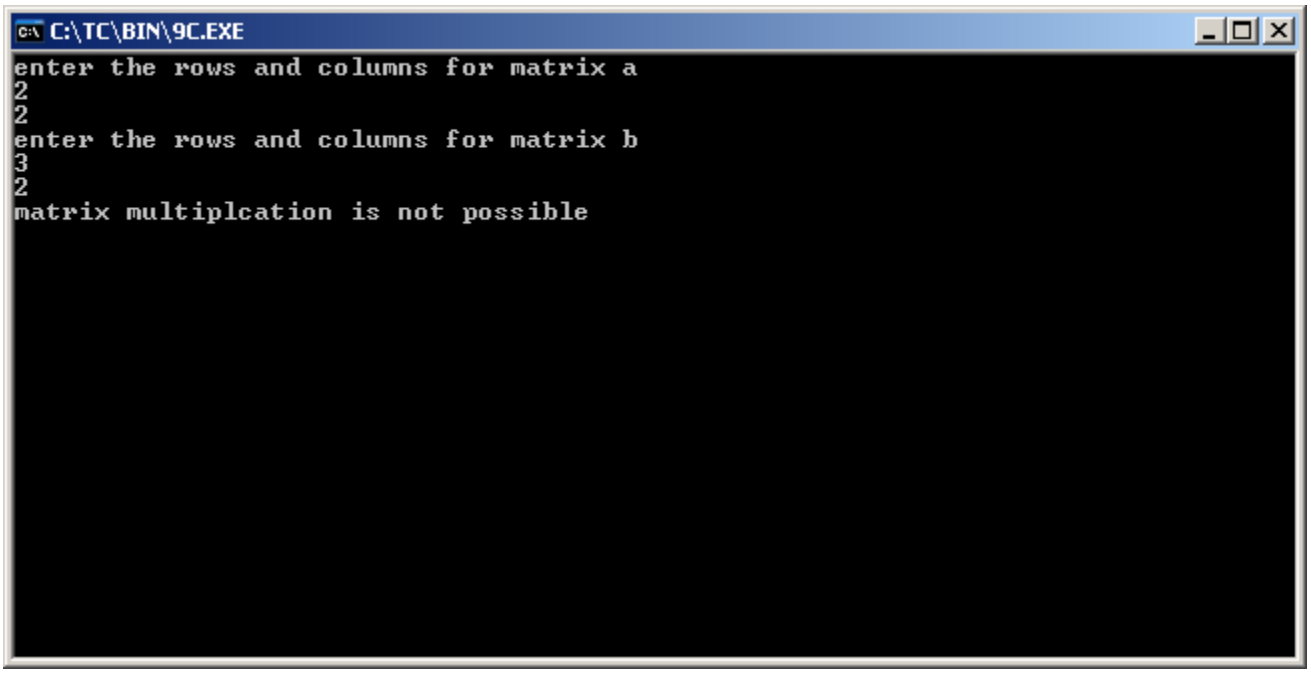

```
//program
#include<stdio.h>
#include<conio.h>
void main()
{
  int a[10][10],b[10][10],c[10][10],i,j,m,n,p,q,k;
  printf("enter the rows and columns for matrix a\n");
  scanf("%d%d",&m,&n);
  printf("enter the rows and columns for matrix b\n");
  scanf("%d%d",&p,&q);
  if(n!=p)
  {
  printf("matrix multiplcation is not possible");
  }
  else
  {
  printf("Enter the First matrix->");
```

```
 for(i=0;i<m;i++)
 {
  for(j=0;j<n;j++)
  {
  scanf("%d",&a[i][j]);
  }
 }
 printf("\nEnter the Second matrix->");
 for(i=0;i<p;i++)
 {
  for(j=0;j<q;j++)
  {
    scanf("%d",&b[i][j]);
  }
 }
  for(i=0;i<m;i++)
  {
     for(j=0;j<n;j++)
     {
     c[i][j]=0;
     for(k=0;k<n;k++)

     {
     c[i][j]=c[i][j]+a[i][k]*b[k][j];
     }
     }
  }
  printf("\nThe multiplication of two matrix is\n");
  for(i=0;i<n;i++)
  {
     printf("\n");
     for(j=0;j<n;j++)
     printf("%d\t",c[i][j]);
  }
  }
  getch();
}
```

Output:





Exercise 7

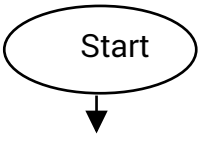
Write a C program that uses functions to perform the following operations:

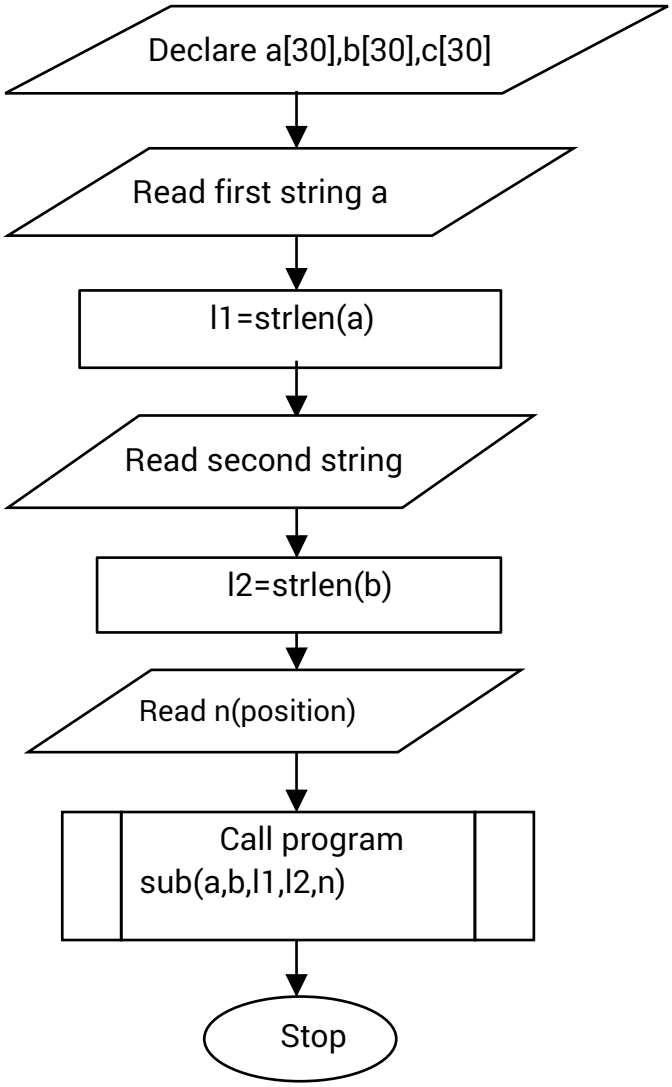
i. To insert a sub-string in to given main string from a given position.

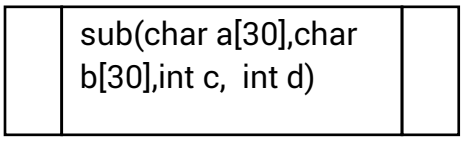
Algorithm:

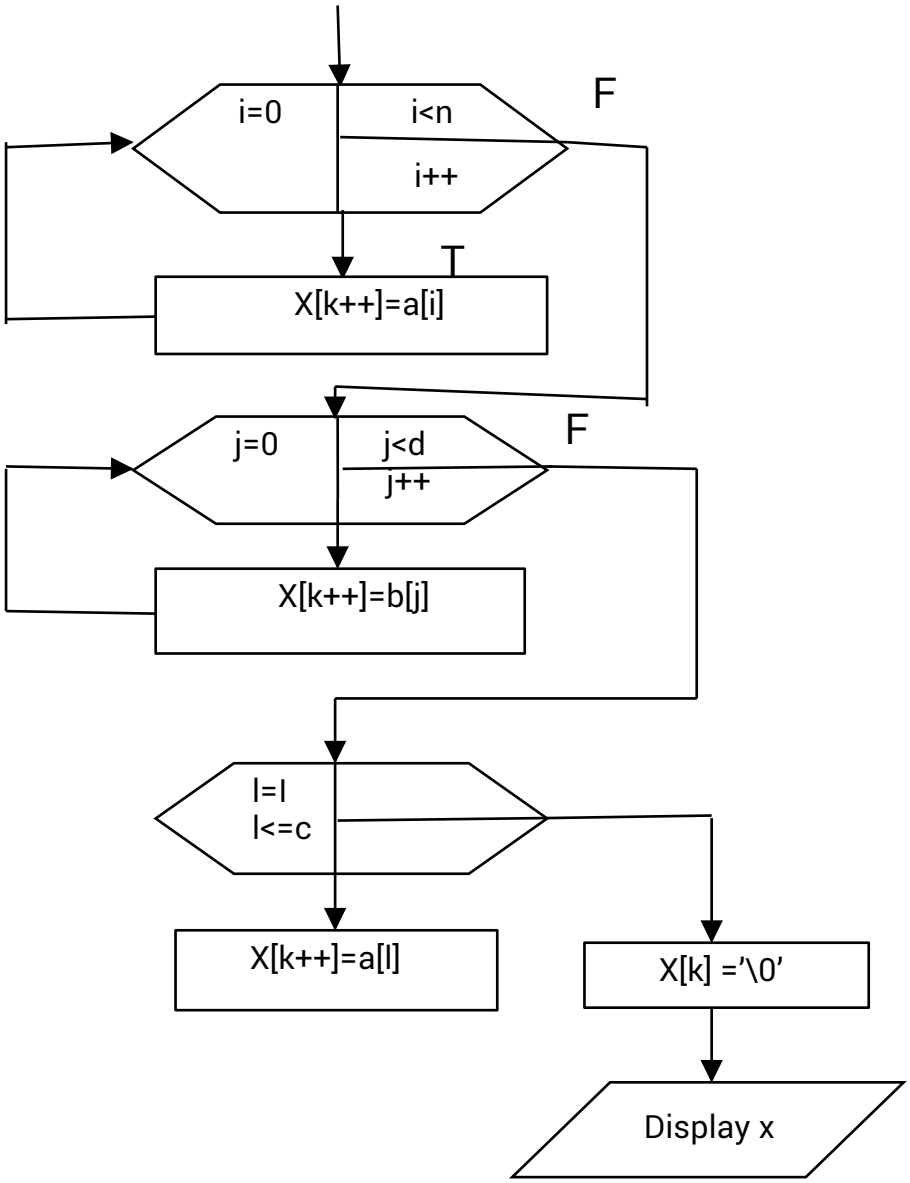
Step 1: declare a[30],b[30],c[30]
Step 2: read first string (i.e. a)
Step 3: find length of the string(i.e. l1)
Step 4: read second string(i.e. b)
Step 5: find length of second string(i.e. l2)
Step 6: read the position (i.e. n)
Step 7: call sub program sub(a,b,l1,l2,n)
Step 8: stop
Algorithm for subprogram
Sub (a, b, l1, l2, n)
Step 1: declare x [20] and initialize k=0
Step 2: for i=0 to n (upto position) if i<n if True goto step 3
False goto step 4
Step 3: assign a[i] value to x[k] and increment the value of k and goto step 2 by incrementing value of i by one.
Step 4: for j=0 to d(length of second string) if j<d if True goto step 5
False goto step 6
Step 5: assign b[j] value to x[k] and increment the value of k and goto step 4 by incrementing value of j by one
Step 6:for l=i to c if i<=c if True goto step 7
False goto step 8
Step 7: i) assign a[l] value to x[k] and increment value of k and goto step 6 by incrementing value of l by one
Step 8 : x[k]='\0'
Step 9: display x.

Flow chart:




Declare a[30],b[30],c[30]
Read first string a
l1=strlen(a)
Read second string
l2=strlen(b)
Read n(position)
Call program
sub(a,b,l1,l2,n)
Stop


sub(char a[30],char
b[30],int c, int d)

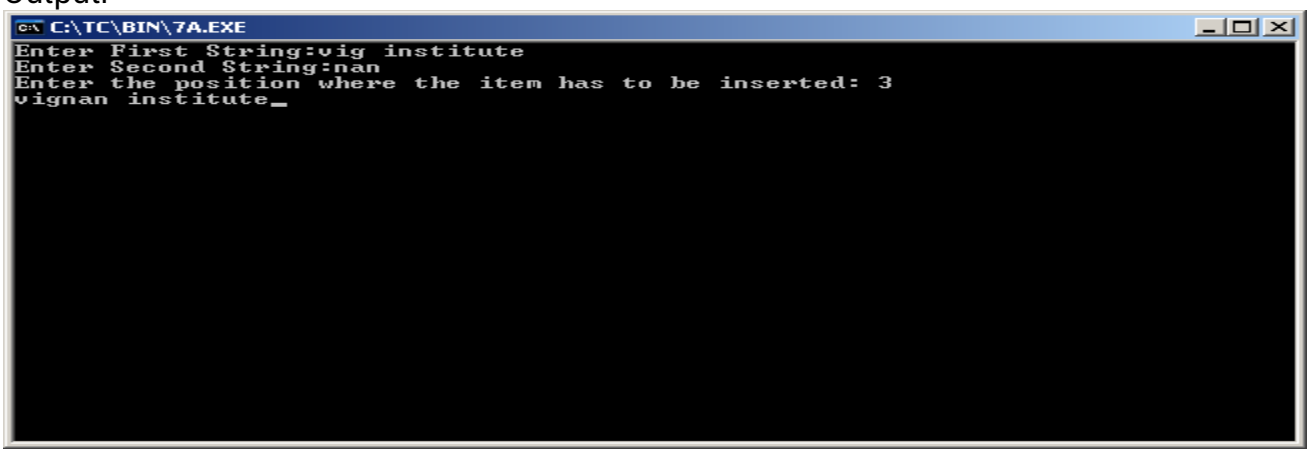


//program

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

```
void main()
{
char a[10],b[10],c[10];
int i,j,k,n,l1,l2;
void sub(char a[],char b[],int,int,int);
clrscr();
printf("Enter First String:");
gets(a);
l1=strlen(a);
printf("Enter Second String:");
gets(b);
l2=strlen(b);
printf("Enter the position where the item has to be inserted: ");
scanf("%d",&n);
sub(a,b,l1,l2,n);
getch();
}
void sub(char a[30],char b[30],int c,int d,int n)
{
char x[20];
int i,l,j,k=0;
for(i=0;i<n;i++)
x[k++]=a[i];
for(j=0;j<d;j++)
x[k++]=b[j];
for(l=i;l<=c;l++)
x[k++]=a[l];
x[k]='\0';
printf("%s",x);
}
```

Output:



ii. To delete n Characters from a given position in a given string.

Algorithm:

Step 1: declare a[30]
Step 2: read first string (i.e. a)

Step 3: find length of the string(i.e. l1)

Step 4: read the position (i.e. n)

Step 5: read no of characters (i.e.n1)

Step 6 : call sub program sub(a,l1,n,n1)

Step 7: stop

Algorithm for subprogram

Sub (a, l1, n, n1)

Step 1: declare x [20] and initialize k=0

Step 2: for i=0 to n (upto position) if i<n if True goto step 3

False goto step 4

Step 3: assign a[i] value to x[k] and increment the value of k and goto step 2 by incrementing value of i by one.

Step 4: for l=n+n1 to c(length of second string) if l<=c if True goto step 5

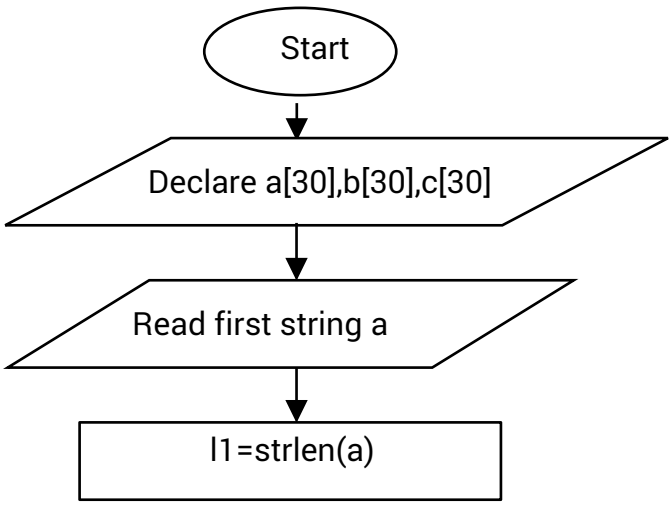
False goto step 6

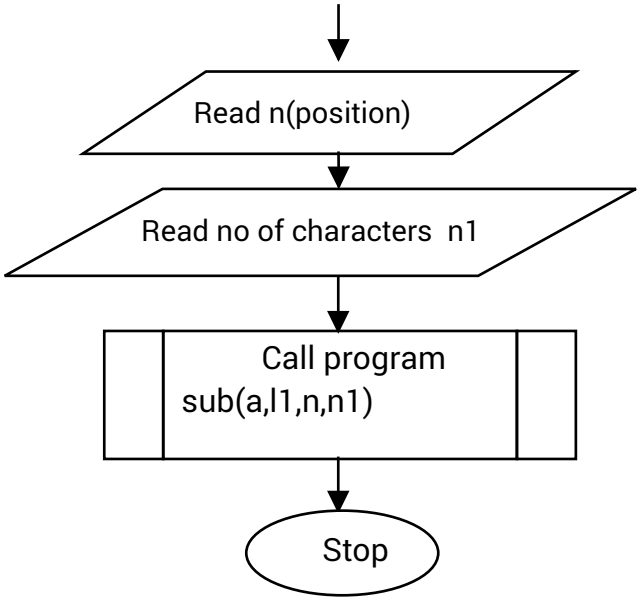
Step 5: assign a[l] value to x[k] and increment the value of k and goto step 4 by incrementing value of l by one

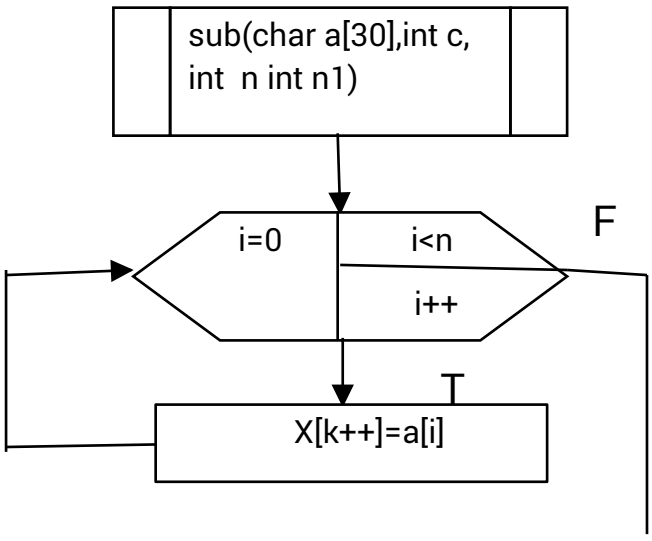
Step 6: x[k]='\0'

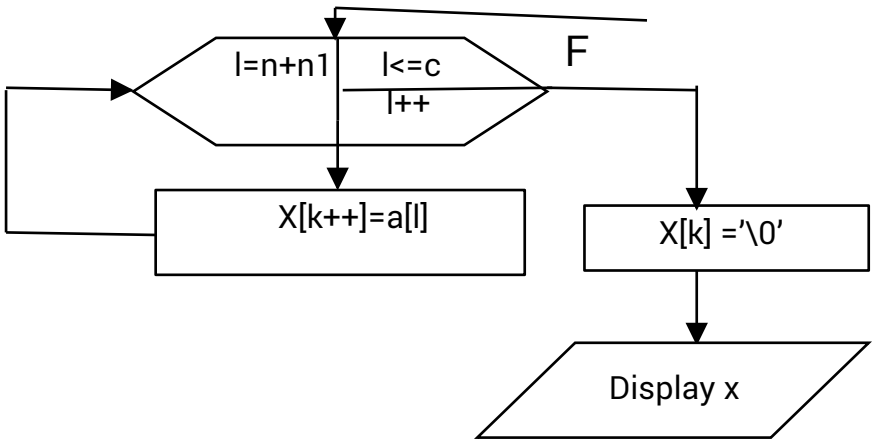
Step 7: display x.

Flow chart:




Read n(position)
Read no of characters n1
Call program
sub(a,l1,n,n1)
Stop


sub(char a[30],int c,
int n int n1)
i=0
i<n
i++
F
T
X[k++]=a[i]

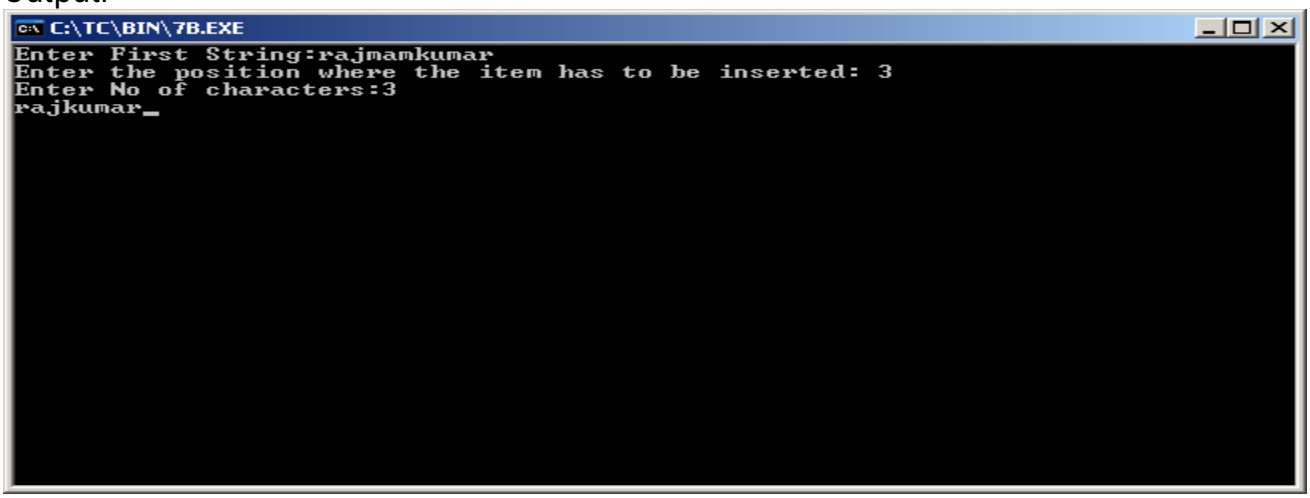



```
//program
#include <stdio.h>
#include <conio.h>
#include <string.h>

void main()
{
char a[10];
int i,j,k,n,l1,n1;
void sub(char a[],int,int,int);
clrscr();
printf("Enter First String:");
gets(a);
l1=strlen(a);
printf("Enter the position where the item has to be inserted: ");
scanf("%d",&n);
printf("Enter No of characters:");
scanf("%d",&n1);
```

```
sub(a,l1,n,n1);
getch();
}
void sub(char a[30],int c,int n,int n1)
{
char x[20];
int i,l,j,k=0;
for(i=0;i<n;i++)
x[k++]=a[i];
for(l=n+n1;l<=c;l++)
x[k++]=a[l];
x[k]='\0';
printf("%s",x);
}
```

Output:



iii. To replace a character of string either from beginning or ending or at a specified location

Algorithm:

Step 1: declare a[30],b[30],c[30]

Step 2: read first string (i.e. a)

Step 3: find length of the string(i.e. l1)

Step 4: read second string(i.e. b)

Step 5: find length of second string(i.e. l2)

Step 6: call sub program sub(a,b,l1,l2)

Step 7: stop

Algorithm for subprogram

Sub (a, b, l1, l2)

Step 1: declare x [20] and initialize k=0

Step 2: read c1(choice for beginning goto step 3

Ending goto step 5

Specified location goto step 8)

Step 3: for i=0 to d if i<d if True goto step 4

False goto step 4

Step 4: assign b[i] value to a[k] and increment the value of k and goto step 3 by incrementing value of i by one.

Step 5: display a

Step 6: for i=c-d to c if i<c if true goto step 7

False goto step 8

Step 7: assign x[k] value to a[i]and increment the value of k and goto step 6 by incrementing value of i by one

Step 8: read position(i.e.p)

Step 9: for i=p to b[k]='\0' if true goto step 10

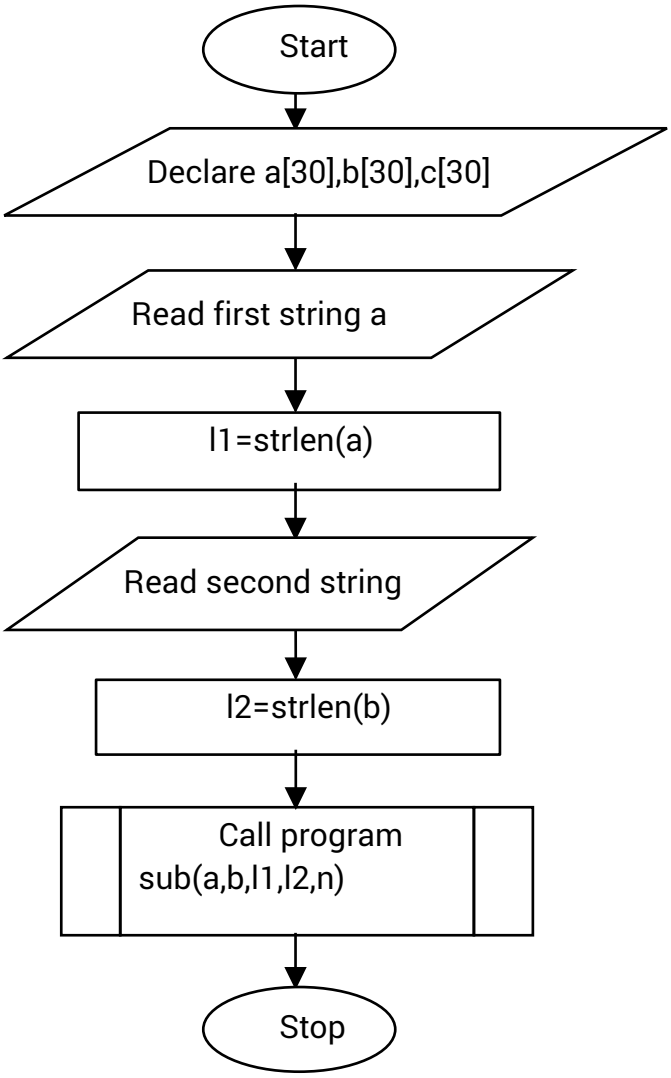
False goto step 11

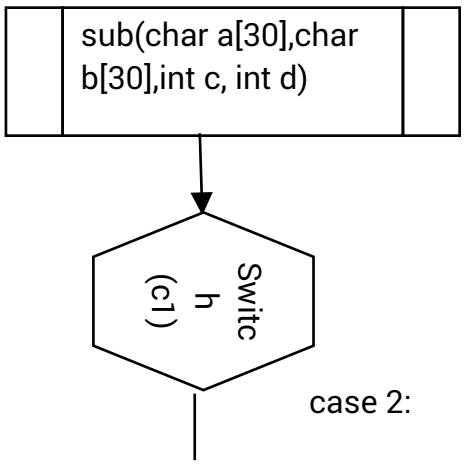
Step 9: i) assign b[k] value to a[i] and increment value of k and goto step 9 by incrementing value of i by one

Step 10 :x[k]='\0'

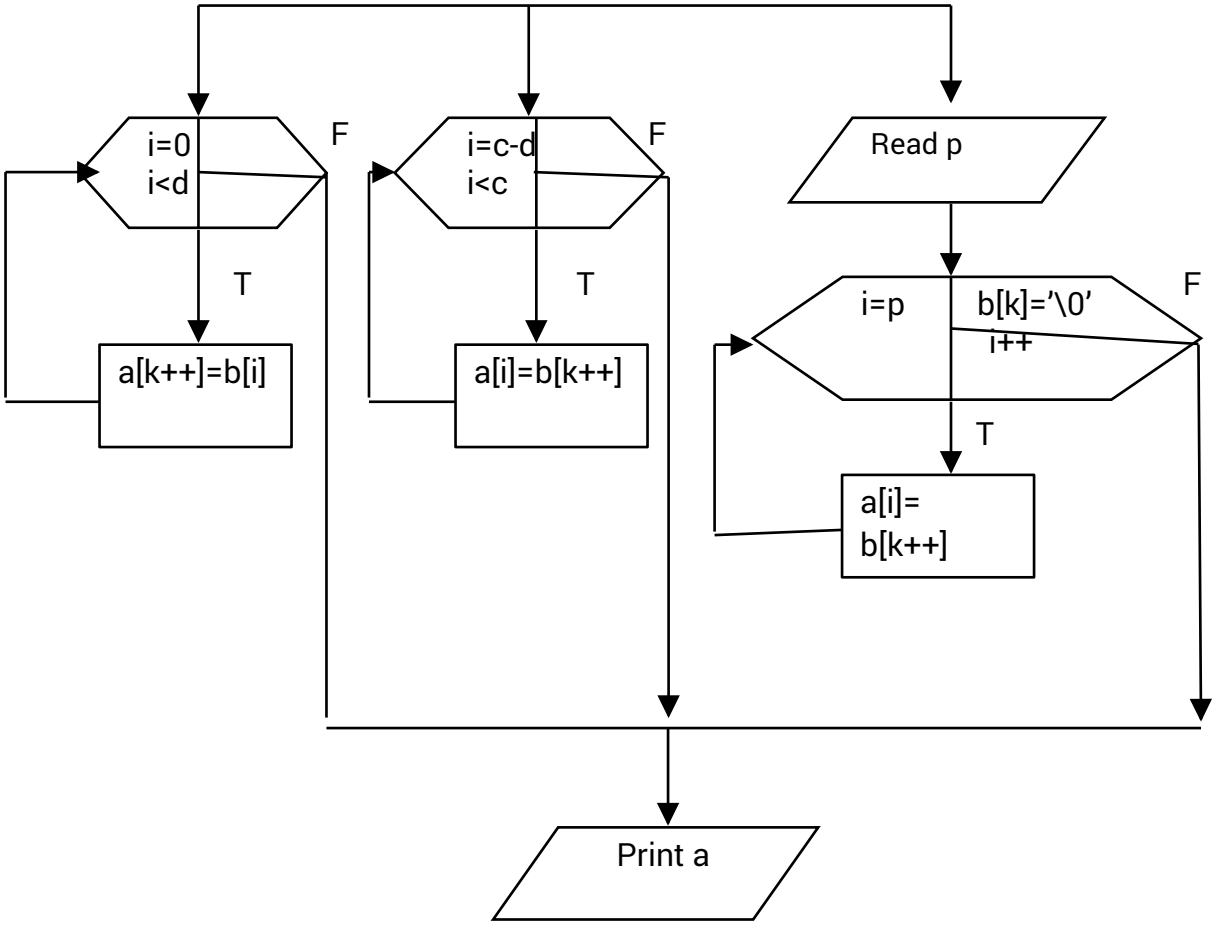
Step 11: display x.

Flow chart:





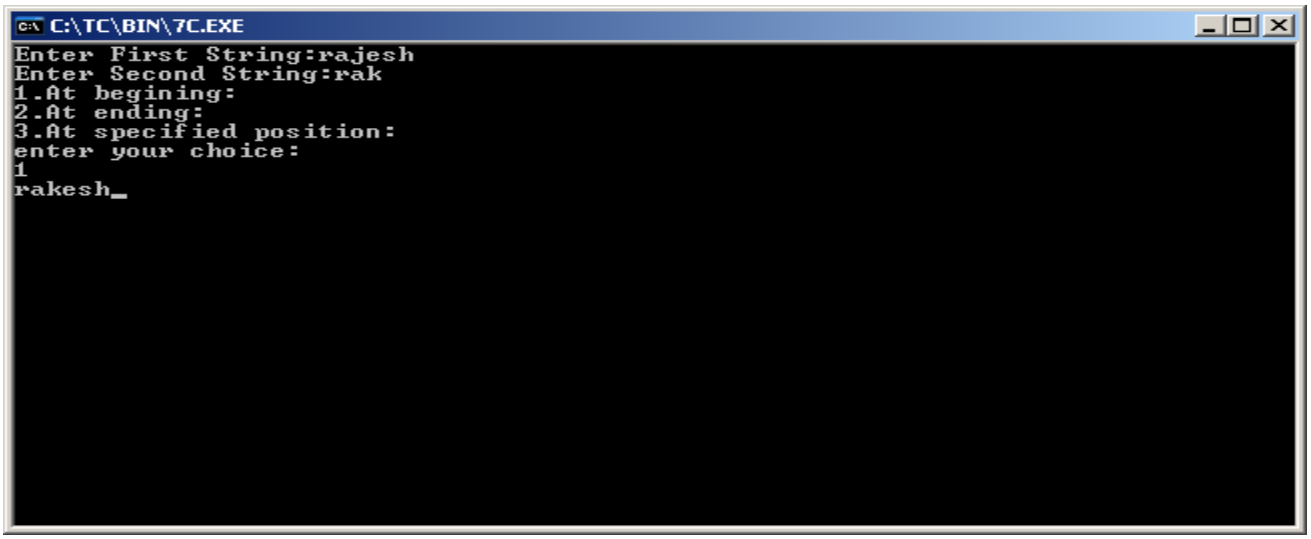
Case 1: case 2: case 3:




```
//program:
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{
char a[30],b[30],c[60];
int n,l1,l2;
void sub(char a[],char b[],int,int);
clrscr();
```

```
printf("Enter First String:");
gets(a);
l1=strlen(a);
printf("Enter Second String:");
gets(b);
l2=strlen(b);
sub(a,b,l1,l2);
getch();
}
void sub(char a[30],char b[30],int c,int d)
{
char x[20];
int p,c1,i,l,j,k=0;
printf("1.At begining:\n");
printf("2.At ending:\n");
printf("3.At specified position:\n");
printf("enter your choice:\n");
scanf("%d",&c1);
switch(c1)
{
case 1:
for(i=0;i<d;i++)
a[k++]=b[i];
printf("%s",a);
break;
case 2:
for(i=c-d;i<c;i++)
a[i]=b[k++];
printf("%s",a);
break;
case 3:
printf("enter position:\n");
scanf("%d",&p);
for(i=p;b[k]!='\0';i++)
a[i]=b[k++];
printf("%s",x);
break;
}
}
```

Output:



Exercise 8

Write a C program that uses functions to perform the following operations using Structure:

i) Reading a complex number ii) Writing a complex number

iii) Addition of two complex numbers iv) Multiplication of two complex numbers

Algorithm:

Step 1: Start

Step 2: declare structure for complex numbers

Step 3: read the complex number

Step 4: read choice

Step 5: if choice=1 then addition operation will perform and it contains following steps

i) w.realpart = w1.realpart+w2.realpart;

ii) w.imgpart = w1.imgpart+w2.imgpart; goto step 4

Step 6: if choice=2 then multiplication operation will perform and it contains following steps

i) w.realpart=(w1.realpart*w2.realpart)-(w1.imgpart*w2.imgpart);

ii) w.imgpart=(w1.realpart*w2.imgpart)+(w1.imgpart*w2.realpart); goto step 4

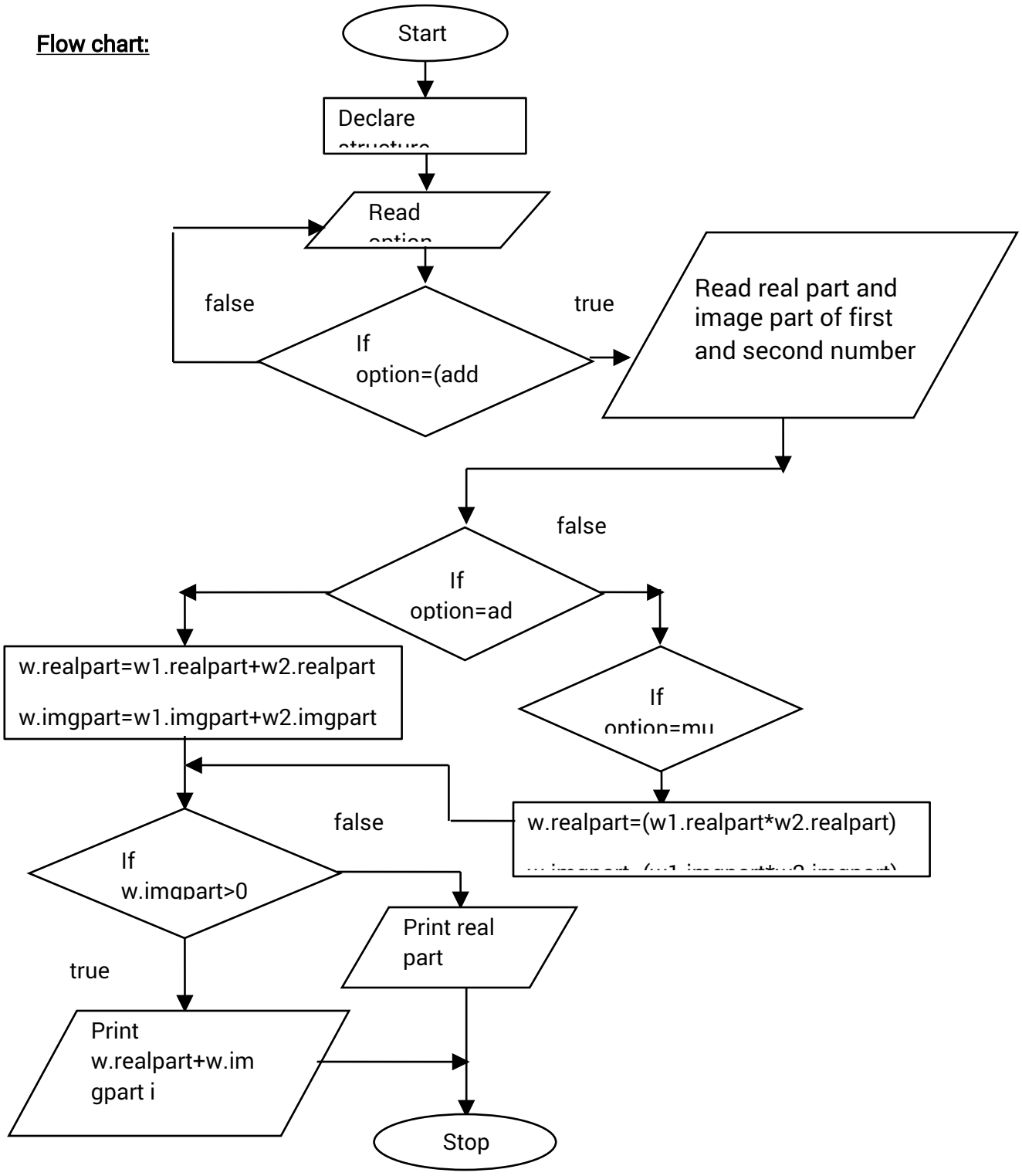
Step 7: if choice=0 then exit operation will perform

Step 8:if w.imgpart>0 then print realpart+imgpart else

Print realpart.

Step 9: Stop

Flow chart:


Start
Declare
Read
false
true
If
option=(add
Read real part and
image part of first
and second number
false
If
option=ad
w.realpart=w1.realpart+w2.realpart
w.imgpart=w1.imgpart+w2.imgpart
If
option=mu
w.realpart=(w1.realpart*w2.realpart)
false
If
w.imgpart>0
Print real
part
true
Print
w.realpart+w.im
gpart i
Stop

//program

```
#include<stdio.h>

#include<math.h>
void arithmetic(int opern);
struct comp
        {
         double realpart;
         double imgpart;
        };
        void main()
        {
         int opern;
         clrscr();
         printf("\n\n \t\t\t***** MAIN MENU *****");
         printf("\n\n Select your option: \n 1 : ADD\n 2 : MULTIPLY\n 0 : EXIT \n\n\t\t
        Enter your Option [  ]\b\b");
         scanf("%d",&opern);
         if(opern>2)
         {
                 printf("invalid option");
         }
         else
                 {
                         switch(opern)
                         {
          case 0:
                      exit(0);
          case 1:
          case 2:
          arithmetic(opern);
                         default:
                          main();
         }
         }
         getch();
        }
        void arithmetic(int opern)
        {
         struct comp w1, w2, w;
         printf("\n Enter two Complex Numbers  (x+iy):\n Real Part of First Number:");
         scanf("%lf",&w1.realpart);
         printf("\n Imaginary Part of First Number:");
         scanf("%lf",&w1.imgpart);
```

```
        printf("\n Real Part of Second Number:");
        scanf("%lf",&w2.realpart);
        printf("\n Imaginary Part of Second Number:");
        scanf("%lf",&w2.imgpart);
        switch(opern)
        {
                /*addition of complex number*/
                case 1:
                w.realpart = w1.realpart+w2.realpart;
         w.imgpart = w1.imgpart+w2.imgpart;
         break;
                /*multiplication of complex number*/
         case 2:
                 w.realpart=(w1.realpart*w2.realpart)-(w1.imgpart*w2.imgpart);
                w.imgpart=(w1.realpart*w2.imgpart)+(w1.imgpart*w2.realpart);
                 break;
        }
                if (w.imgpart>0)
                printf("\n Answer = %lf+%lfi",w.realpart,w.imgpart);
        else
        printf("\n Answer = %lf%lfi",w.realpart,w.imgpart);
                getch();
       // main();
       }
```

output: