COURSE OBJECTIVE

- To provide basic principles C programming language.
- To provide design & develop of C programming skills.
- To provide practical exposures like designing flowcharts, algorithms, how to debug programs etc**.**

COURSE OUTCOME

- Gaining the Knowledge on various parts of a computer.
- Able to draw flowcharts and write algorithms
- Able to design and development of C problem solving skills.
- Able to design and develop modular programming skills.
- Able to trace and debug a program

COMPUTER LAB DO'S AND DON'T

DO'S

1. Know the location of the fire extinguisher and the first aid box and how to use them in case of an emergency.
2. Read and understand how to carry out an activity thoroughly before coming to the laboratory.
3. Report fires or accidents to your lecturer/laboratory technician immediately.
4. Report any broken plugs or exposed electrical wires to your lecturer/laboratory technician immediately.

DON'TS

1. Do not eat or drink in the laboratory.
2. Avoid stepping on electrical wires or any other computer cables.
3. Do not open the system unit casing or monitor casing particularly when the power is turned on. Some internal components hold electric voltages of up to 30000 volts, which can be fatal.
4. Do not insert metal objects such as clips, pins and needles into the computer casings. They may cause fire.
5. Do not remove anything from the computer laboratory without permission.
6. Do not touch, connect or disconnect any plug or cable without your lecturer/laboratory technician's permission.
7. Do not misbehave in the computer laboratory.

LIST OF EXPERIMENTS

Experiment No.1 Date:

ROOTS OF QUADRATIC EQUATION

AIM: To find the roots of the quadratic equation ($ax^2+bx+c=0$) with different possible input values for a, b and c.

ALGORITHM :

ALGM: Quadratic Equation [This algorithm takes three coefficients as input and computes the roots]

Steps:

1. [Initialize] Start

2. [Input coefficients of quadratic equation]
read *a ,b, c*

3. [Check for valid coefficients]
If *a =0* and *b= 0* **then**
print "Roots cannot be determined"
[Check for linear equation]
else *a=0***then**
root1 ← (-c/b)
print "Linear equation",root1
goto step 5

4. [Compute **discriminate** value]
*disc ← b*b-4*a*c*

5. [Based on **discriminate** value, classify and calculate all possible roots and print them]

5.1 [If discriminate value is *0*, roots are real & equal.]
if *disc=0* **then**
*root1← root2 ← (-b/2*a)*
print "Real & equal roots", root1, root2

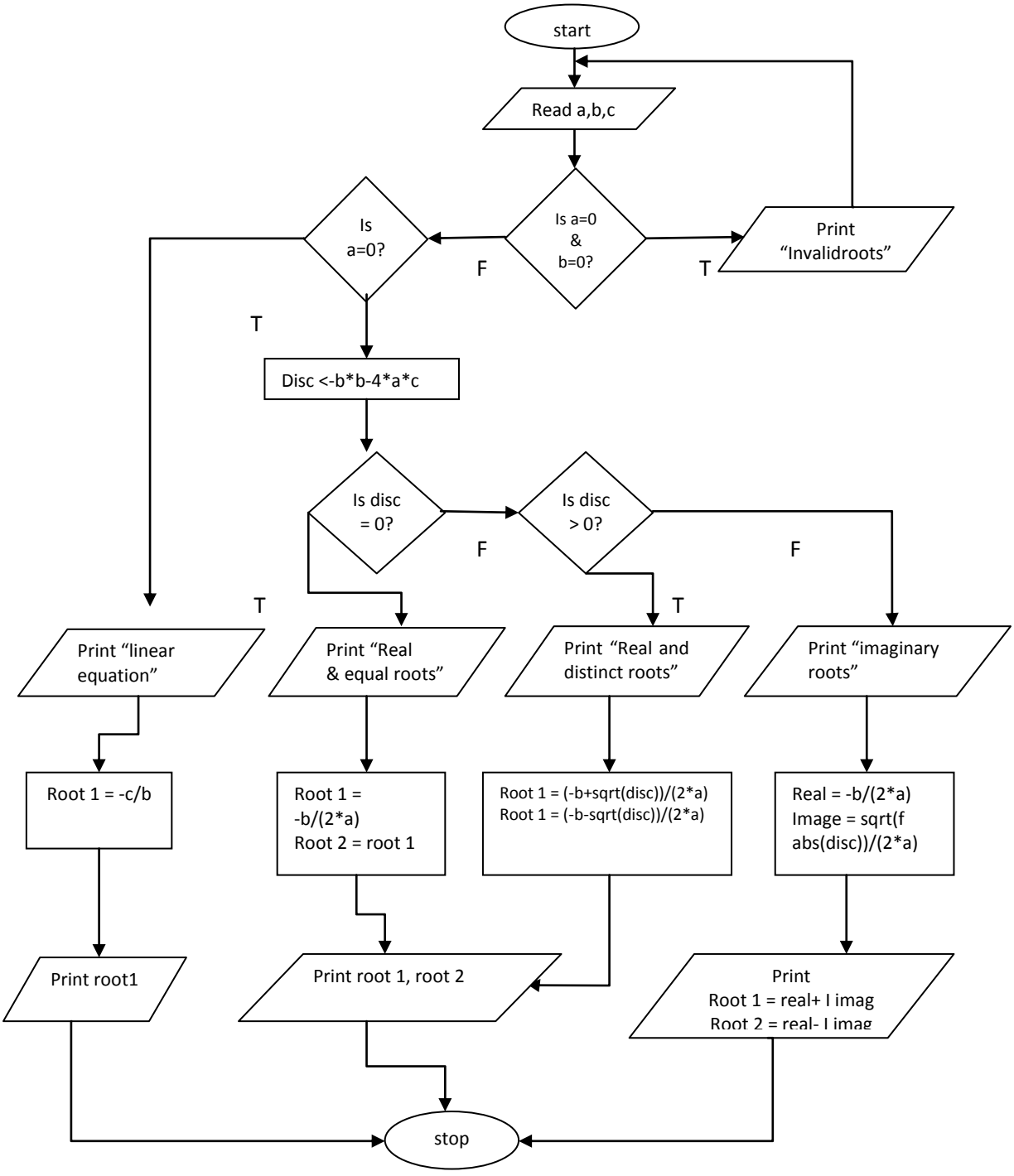
`

5.2 [If discriminate value is *>0*, roots are real & distinct.]
else if *disc>0***then**
*root1← (-b+√disc)/(2*a)*
*root2 ← (-b-√disc)/(2*a)*
print "Real & distinct roots", root1, root2

5.3 [If discriminate value is *<0*, roots are imaginary.]
else
*real ← -b/(2*a)*
*imag ← √(fabs(disc))/(2*a)*
root1 ← (real) + i (imag)
root2 ← (real) - i (imag)
print "Imaginary roots", root1, root2
endif
endif

6. [Finished] End

FLOWCHART:


start
Read a,b,c
Is a=0 & b=0?
F
T
Print "Invalidroots"
Is a=0?
T
Disc <-b*b-4*a*c
Is disc = 0?
F
Is disc > 0?
F
T
T
Print "linear equation"
Print "Real & equal roots"
Print "Real and distinct roots"
Print "imaginary roots"
Root 1 = -c/b
Root 1 = -b/(2*a) Root 2 = root 1
Root 1 = (-b+sqrt(disc))/(2*a) Root 1 = (-b-sqrt(disc))/(2*a)
Real = -b/(2*a) Image = sqrt(f abs(disc))/(2*a)
Print root1
Print root 1, root 2
Print Root 1 = real+ I imag Root 2 = real- I imag
stop

PROGRAM :

```
/* Program to calculate all possible roots of a quadratic equation */
#include<stdio.h>
#include<conio.h>
#include<math.h>
int main()
{
        float a, b, c, disc;
        float root1,root2,real,imag;
        clrscr();
        printf("Enter a,b,c values\n");
        scanf("%f%f%f",&a,&b,&c);
        if( (a == 0) && (b == 0) &&(c==0))
        {
                printf("Invalid coefficients\n");
                printf(" Try Again with valid inputs !!!!\n");
                getch();
        }

                disc = b*b - 4*a*c;
                if(disc == 0)
                {
                        printf("The roots are real and equal\n");
                        root1 = root2 = -b/(2*a);
                        printf("Root1 = %.3f \nRoot2 = %.3f", root1,root2);
                }
                else if(disc>0)
                {
                        printf("The roots are Real and Distinct\n");
                        root1 = (-b+sqrt(disc)) / (2*a);
                        root2 = (-b-sqrt(disc)) / (2*a);
                        printf("Root1 = %.3f \nRoot2 = %.3f",root1,root2);
                }
                else
                {
                        printf("The roots are Real and Imaginary\n");
                        real = -b / (2*a);
                        imag = sqrt(fabs(disc)) / (2*a);//fabs() returns only numberignoring sign
                        printf("Root1 = %.3f + i %.3f \n",real,imag);
                        printf("Root2 = %.3f - i %.3f",real,imag);
                }

        getch();
}
```

EXPECTED OUTPUT:

1. Enter a, b, c values
   0 0 1
   Invalid coefficients
   Try Again with valid inputs!!!!

2. Enter a, b, c values
   1 2 3
   The roots are Real and Imaginary
   Root1 = -1.000 + I 1.414
   Root2 = -1.000 – I 1.414

3. Enter a, b, c values
   1 2 1
   The roots are real and equal
   Root1 = -1.000
   Root2 = -1.000

4. Enter a, b, c values
   1 5 3
   The roots are Real and Distinct
   Root1 = -0.697
   Root2 = -4.303

5. Enter a, b, c values
   0 1 2
   Linear equation
   Root = -2.000

Experiment No.2 Date:

PALINDROME

AIM: To check whether the given integer is PALINDROME or NOT.

ALGORITHM :

ALGM: Palindrome [This algorithm takes an integer number as input and output the reverse of the same. Also checks the number is palindrome or not]

Steps:

1. [Initialize] Start

2. [Input the original number]
read *num*

3. [Set number *num* to a variable *n*]
n ← *num*

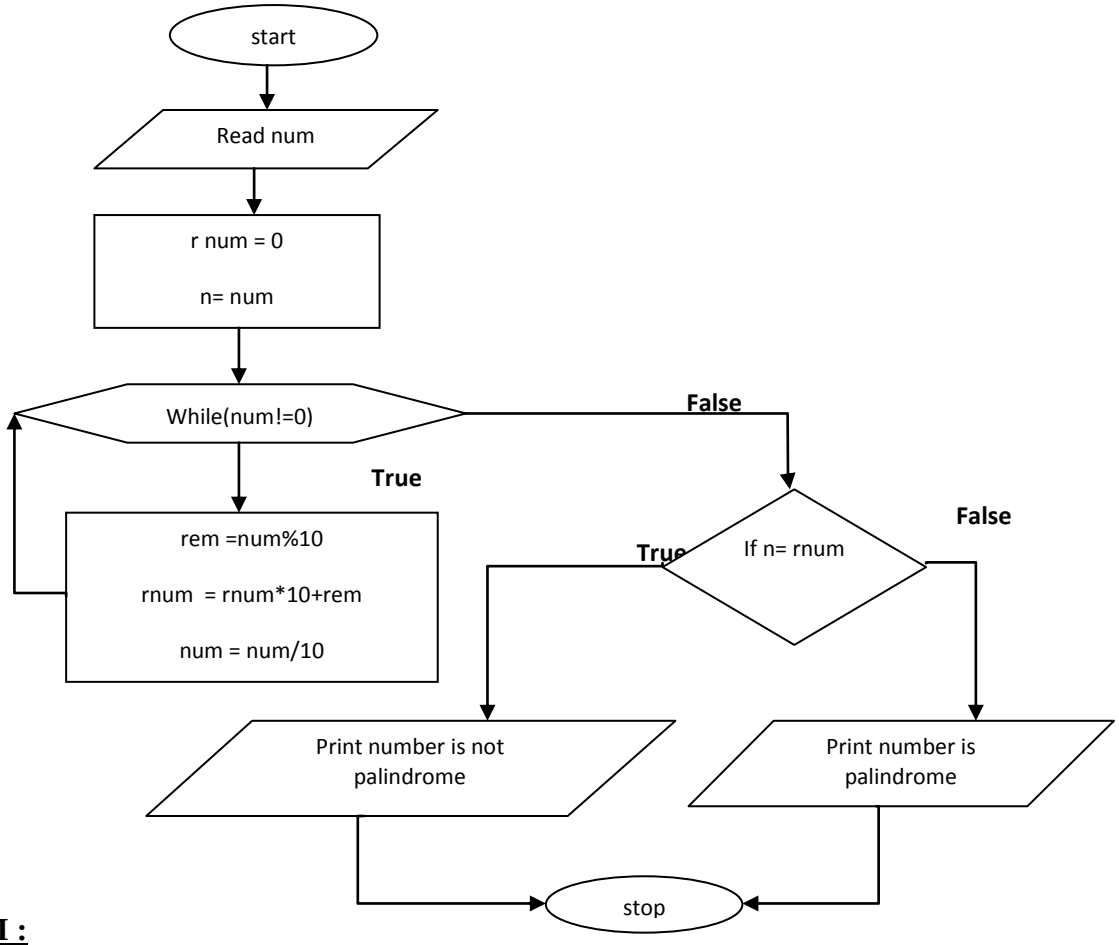
4. [Iterate until *num* is not equal to *0*.
If *num* value becomes 0, control comes out of the loop. So *num*'s original value is lost. So, *num* value is stored in other variable *n* in step 3.
In step 4, reverse of the number is calculated.]
while (*num != 0*) **do**
remainder ← *num mod 10*
num ← *num/10*
rev ← *rev * 10 +remainder*

5. [Print reverse number]
print *rev*

6. [Check if original number & reverse number are same. If it is, number is palindrome. Otherwise, not palindrome]
if (*rev = n)* **then**
print palindrome
else
print not a palindrome
endif

7. [Finished]
End

FLOWCHART:



PROGRAM :

```
/* Program to calculate whether a given number is palindrome or not */
#include<stdio.h>
#include<conio.h>
int main()
{
        int temp,rev=0,num,remainder ;
        clrscr();
        printf("Enter the number\n");
        scanf("%d",&num);
        temp=num;
        while(num!=0)                   //Reversing the number
        {
                remainder = num%10;
                num = num/10;
                rev = rev*10+ remainder;
        }
        printf("The reverse number is %d",rev);
        if(rev == temp)
```

```
                printf("\n%d is a palindrome",temp);
        else
                printf("\n%d is not a palindrome", temp);
getch();
}
```

EXPECTED OUTPUT:

1. Enter the number
   1001
   The reverse number is 1001
   1001 is a palindrome

2. Enter the number
   123
   The reverse number is 123
   123 is not a palindrome

Experiment No. 3(A) Date:

SQUARE ROOT

AIM: To find Square Root of a given Number.

ALGORITHM:

SQUARE ROOT [this algorithm takes an integer number as an input and prints the square root of the number]

Input: the integer number

Output: print the square root number

Step1: Initialize

Step2: Enter the number

Step3: if(n>=0)

then

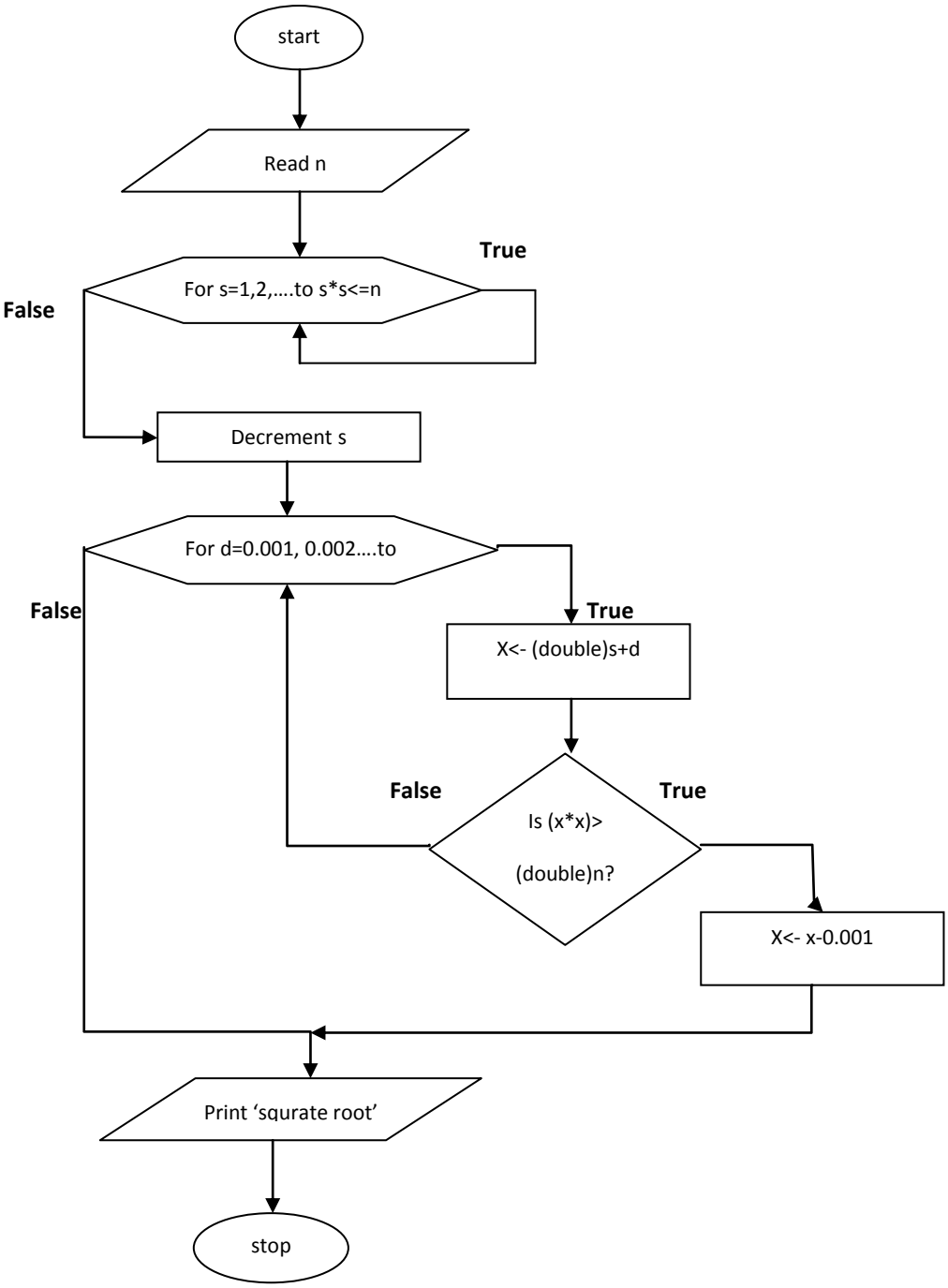
Print the square root result

Else

Print it is not a valid number

Step4: stop

FLOWCHART:


start
Read n
True
For s=1,2,….to s*s<=n
False
Decrement s
For d=0.001, 0.002….to
False
True
X<- (double)s+d
False
True
Is (x*x)>
(double)n?
X<- x-0.001
Print 'squrate root'
stop

PROGRAM :

```
/* Program to calculate square root of the given number without using built in function */
#include<stdio.h>
#include<conio.h>
#include<math.h>
int  main()                    // Uses Brute-Force Method
{
        int s;
        double x,d,n;
        clrscr();
        printf("Enter the number\n");
        scanf("%lf",&n);
        if(n>=0)
        {
                for(s=1;s*s<=n;s++);                    //calculating decimal part of the square root
                        s--;
                for(d = 0.001;d < 1.0;d += 0.001)               // calculating the fractional part
                {
                        x = (double)s + d;
                        if((x*x > (double)n))
                        {
                                x = x - 0.001;
                                break;
                        }
                }
                printf("The square root of the given number is %.3lf\n", x);
                printf("The square root as per built in function sqrt()is %.2lf", sqrt(n));
        }
        else
        {
                printf( "No square root to a negative number");
        }
        getch();
}
```

EXPECTED OUTPUT:

1. Enter the number
   16
   The square root of the given number is 4.000
   The square root as per built in function sqrt() is 4.00

2. Enter the number
   27
   The square root of the given number is 5.196
   The square root as per built in function sqrt() is 5.20

   3. Enter the number -4

   No square root to a negative number

Experiment No. 3(B) Date:

LEAP YEAR

AIM: To check whether the given year is leap year or not**.**

ALGORITHM:

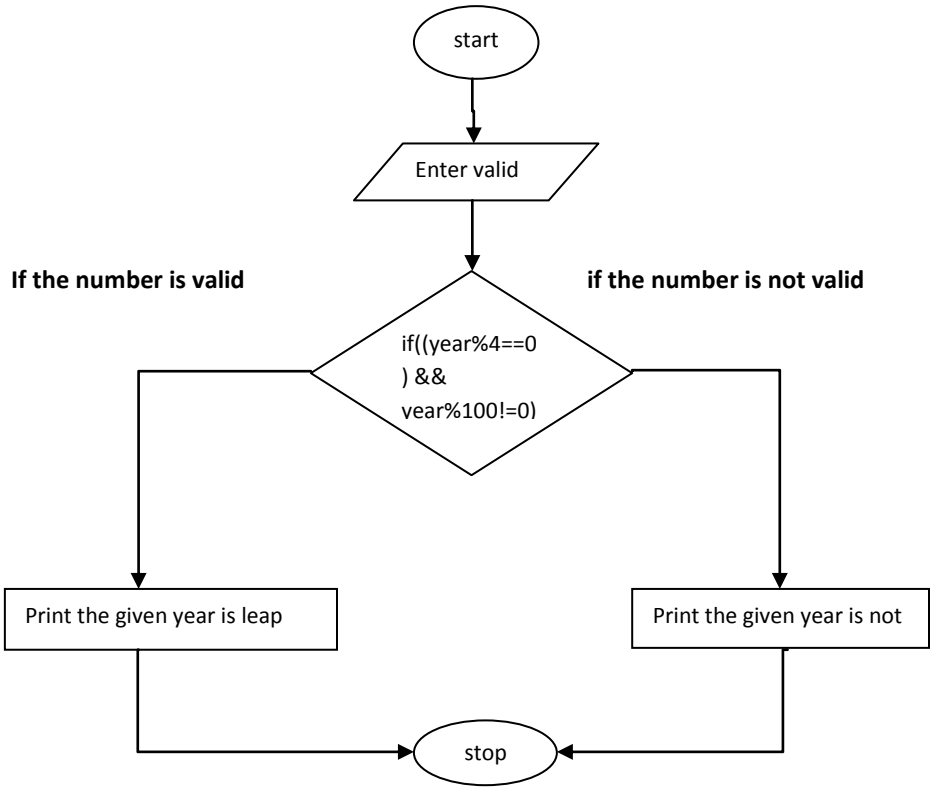
ALGM: Leap Year

Input An year
Output Leap year or not
complexity O(1).
leapyear(year)

Steps

Start
1 If (year %4==0 and year%100 !=0)
2 print "it is a leap year";
3 else
4 if(year%400==0)
5 print "it is a leap year";
6 else
7 print "it is not a leap year";
End

FLOWCHART:



PROGRAM :

```
/* Program to find whether given year is leap year or not */
#include<stdio.h>
#include<conio.h>
int main()
{
        int year;
        clrscr();
        printf("Enter valid year\n");
        scanf("%d",&year);
        if((year%4==0) && (year%100!=0) || (year%400 ==0)) //check for leap year
        {
                printf("%d is leap year", year);
        }
        else
        {
                printf("%d is not a leap year",year);
        }
getch();
}
```

EXPECTED OUTPUT:

1. Enter valid year
   2012
   2012 is leap year

2. Enter valid year
   1900
   1900 is not a leap year

Experiment No.4 Date:

Evaluation of Polynomial

AIM: To find the value of the polynomial Design and develop an algorithm for evaluating the polynomial $f(x) = a_4\ x4 + a_3\ x3 + a_2\ x2 + a_1\ x + a_0$, for a given value of x and its coefficients using Horner's method.

ALGORITHM :

ALGM: EVAL_POLYNOMIAL *f(x) = a4x4+ a3x3+ a2x2+ a1x+ a0*using Horner's method

Steps:

1. [Initialize] Start

2. [Input the number of coefficients *n*]
read *n*

3. Set *sum* to *0*

4. [Read all coefficients *a1, a2, a3, a4*and constant*a0*]
For each value i in array a(i)**do**
read n+1 coefficients
endfor

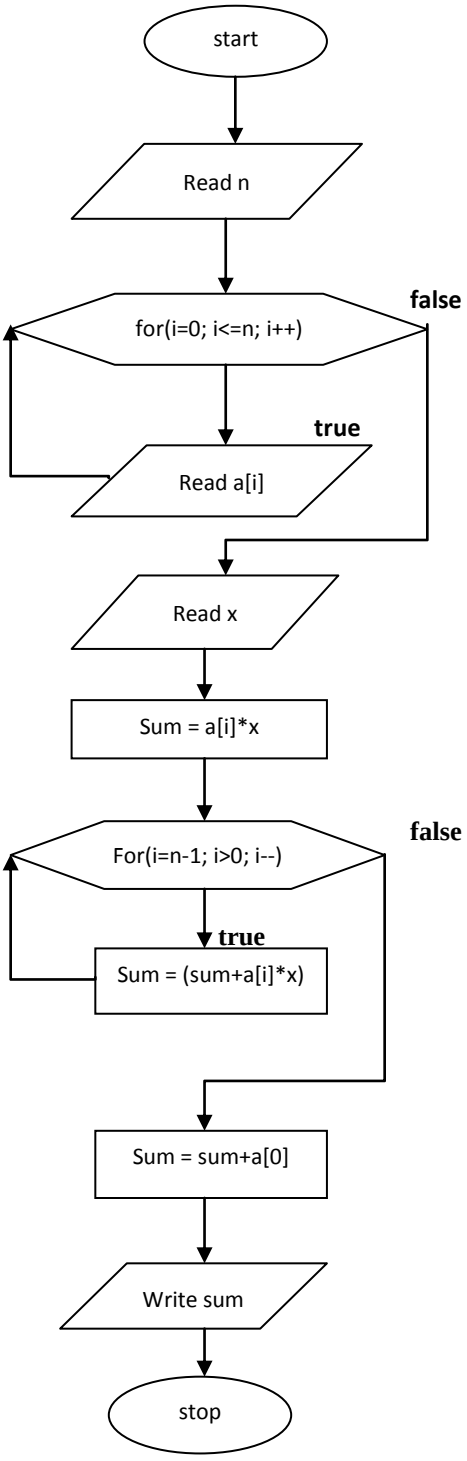
5. [Input variable x]
read *x*

6. [Iterate from n to *0*. Calculate each term *a4x4, a3x ,a2x2,a1x, a0* .]
For each value iin array a(i) **do**
*sum ← sum * x + a[i]*
endfor

7. Print *sum*

8. [Finished]
End

FLOWCHART:


start
Read n
for(i=0; i<=n; i++)
false
true
Read a[i]
Read x
Sum = a[i]*x
For(i=n-1; i>0; i--)
false
true
Sum = (sum+a[i]*x)
Sum = sum+a[0]
Write sum
stop

PROGRAM :

```
/* Evaluating the polynomial f(x) = a4x4+ a3x3+ a2x2+ a1x+ a0 using Horner's method (n, i, sum, a[], x) */
#include<stdio.h>
#include<conio.h>
int main()
{
        int n,i,sum=0,a[10],x;
        clrscr();
        printf("enter the number of co-efficients n>=0\n");
        scanf("%d",&n);
        printf("enter the n+1 co-efficients\n");
        for(i=n;i>=0;i--)
        {
                printf("Enter a[%d] th coefficient : ",i);
                scanf("%d",&a[i]);
        }
        printf("enter value of x\n");
        scanf("%d",&x);
        for(i=n;i>0;i--)
        {
                sum=sum*x+a[i];
        }
        sum=sum+a[0];
        printf("the value of sum is %d",sum);
getch();
}
```

EXPECTED OUTPUT:

1. enter the number of co-efficients n>=0
   4
   enter the n+1 co-efficients
   Enter a[4] th coefficient : 1
   Enter a[3] th coefficient : 2
   Enter a[2] th coefficient : 3
   Enter a[1] th coefficient : 4
   Enter a[0] th coefficient : 5
   enter value of x
   1
   the value of sum is 15

2. enter the number of co-efficients n>=0
   0
   enter the n+1 co-efficients
   Enter a[0] th coefficient : 25
   enter value of x
   25

Experiment No.5 Date:

SINE SERIES

AIM: To compute Sin(x) using Taylor series approximation given by Sin(x)= $\frac{x}{1!} - \frac{x^3}{3!} + \frac{x^5}{5!} -$.

ALGORITHM:

ALGORITHM: SINE SERIES[this algorithm takes degree as an input to print the sine function value]

Input : Degree of polynomial

Output : Sine function value

Step 1: start

Step 2: enter the degree

Step 3: convert degree into radian

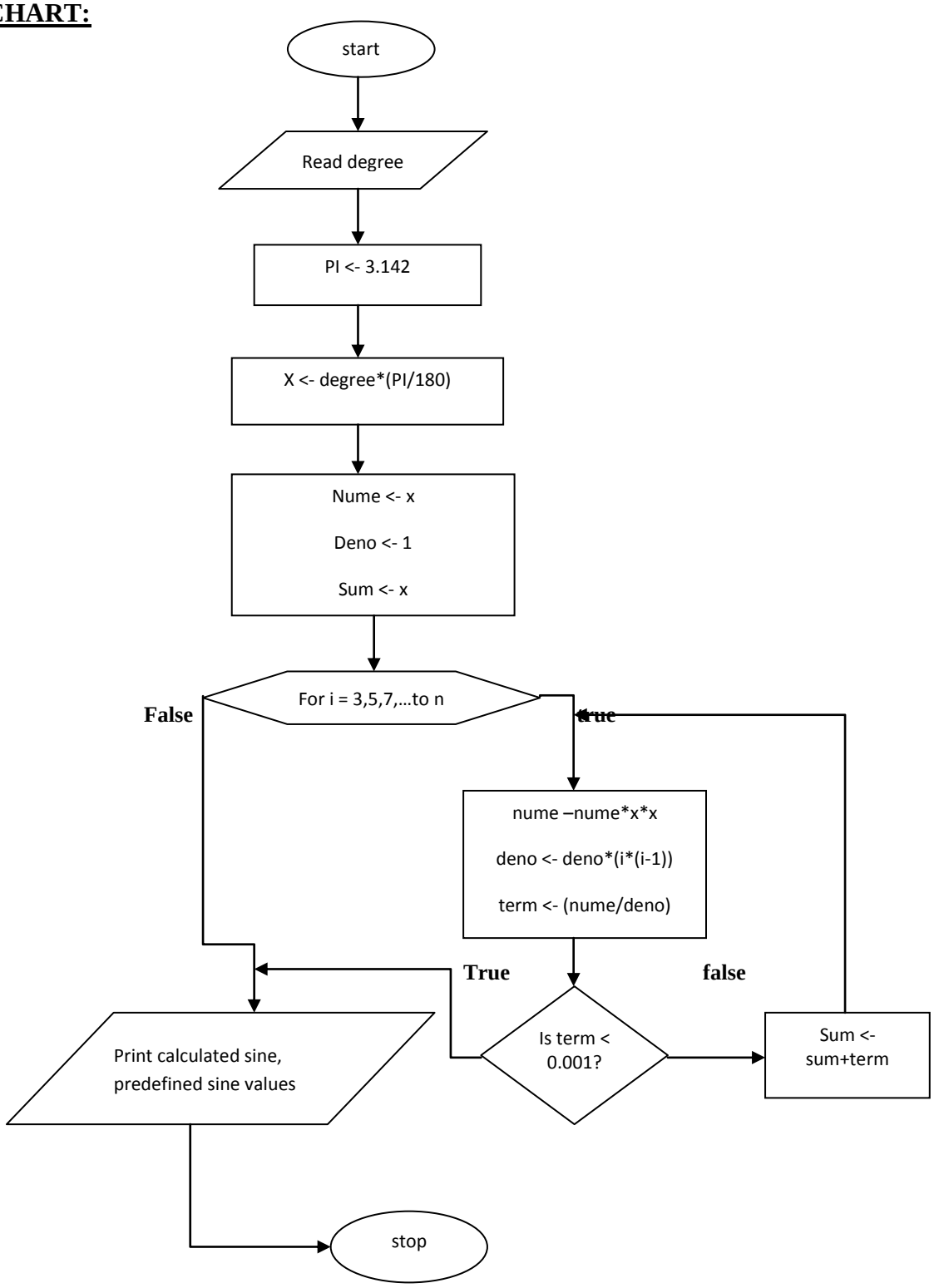
X <- degree*(PI/180)

Step 4: check the given number is odd/not

```
If(it is a odd number)
Then
{
term = nume/deno;
nume = -nume*x*x;
deno = deno*i*(i+1);
}
else
{
If(term<0.001)
{
Print thr sine value
}
else
{
Check the number
}
}
```

Step 5: stop

FLOWCHART:


start
Read degree
PI <- 3.142
X <- degree*(PI/180)
Nume <- x
Deno <- 1
Sum <- x
For i = 3,5,7,...to n
False
True
nume –nume*x*x
deno <- deno*(i*(i-1))
term <- (nume/deno)
True
false
Is term < 0.001?
Sum <- sum+term
Print calculated sine, predefined sine values
stop

PROGRAM:

```
/* Program to calculate sine value of given angle */
#include<stdio.h>
#include<conio.h>
#include<math.h>
#define PI 3.142
int main()
{
        int i, degree;
        float x, sum=0,term,nume,deno;
        clrscr();
        printf("Enter the value of degree");
        scanf("%d",&degree);
        x = degree * (PI/180);                  //converting degree into radian
        nume = x;
        deno = 1;
        i=2;
        do
        {                                       //calculating the sine value.
                term = nume/deno;
                nume = -nume*x*x;
                deno = deno*i*(i+1);
                sum=sum+term;
                i=i+2;
        } while (fabs(term) >= 0.00001);        // Accurate to 4 digits
        printf("The sine of %d is %.3f\n", degree, sum);
        printf("The sine function of %d is %.3f", degree, sin(x));
        getch();
}
```

EXPECTED OUTPUT:

1. Enter the value of degree
   0
   The sine of 0 is 0.000
   The sine function of 0 is 0.000

2. Enter the value of degree
   45
   The sine of 45 is 0.707
   The sine function of 45 is 0.707

3. Enter the value of degree
   90
   The sine of 90 is 1.000

   The sine function of 90 is 1.000

Experiment No.6 Date:

BUBBLE SORT

AIM: To arrange given N integers in ascending order using Bubble Sort.

ALGORITHM :

ALGM: Bubble Sort [This algorithm takes a list of unordered numbers and arrange them in ascending order using Bubble Sort method]

Steps: 1. [Initialize] Start

2. [Input number of elements]
read *n*

3. [Input unsorted elements in array]
read elements in array *a*[ ]

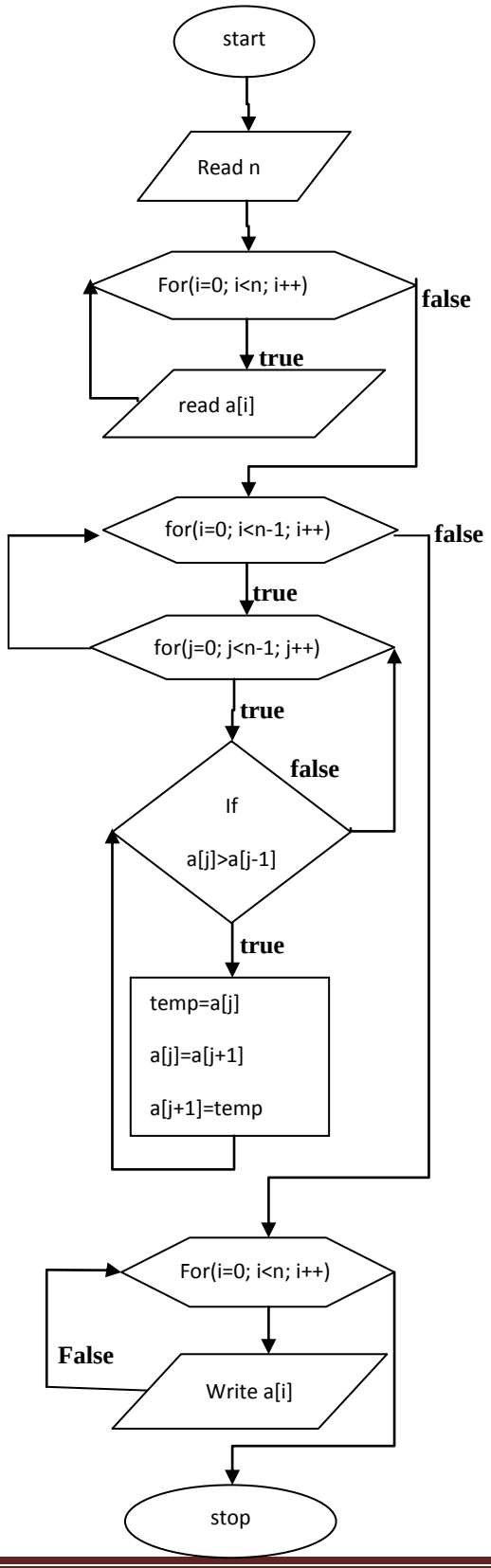
4. print elements of array *a*[ ]

5. [Iterate array *a*[ ] in two loops. Outer loop gives number of passes. Inner loop does swap task.In each pass, compare each pair of adjacent items. If former element is greater than latter one, swap them.]
[Iterate array a[ ] with
for each value *i* in array *a[i] to n* **do**
for each value *j* in array *a[j] to n-1* **do**
[Compare each pair of adjacent elements]
if (*a[j]* > a[j+1])**then**
[Swap these elements using *temp* variable]
temp ← *a[j]*
a[j] ← *a[j+1]*
a[j+1] ← *temp*
endif
endfor
endfor

6. Print array with sorted elements

7. [Finished] End.

FLOWCHART:


start
Read n
For(i=0; i<n; i++)
false
true
read a[i]
for(i=0; i<n-1; i++)
false
true
for(j=0; j<n-1; j++)
true
false
If
a[j]>a[j-1]
true
temp=a[j]
a[j]=a[j+1]
a[j+1]=temp
For(i=0; i<n; i++)
False
Write a[i]
stop

PROGRAM:

```
#include<stdio.h>
#include<conio.h>
int main()
{
        int n,i,j,a[10],temp;
        clrscr();
        printf("Enter the no. of elements : \n");
        scanf("%d",&n);
        printf("Enter the array elements \n");
        for(i = 0 ; i < n ; i++)
        scanf("%d",&a[i]);
        printf("The original elements are \n");
        for(i = 0 ; i < n ; i++)
                printf("%d ",a[i]);
        for(i= 0 ; i < n-1 ; i++)                       // Number of Passes
        {
                for(j= 0 ; j< n-i+1; j++)                       // Comparisons
                        if(a[j] > a[j+1])
                        {
                                temp = a[j];
                                a[j] = a[j+1];
                                a[j+1] = temp;
                        }
        }
        printf("\n The Sorted elements are \n");
        for(i = 0 ; i < n ; i++)
                printf("%d ",a[i]);
getch();
}
```

Output:

1. Enter the no. of elements :
   5
   Enter the array elements
   30 10 50 20 40
   The original elements are
   30 10 50 20 40
   The Sorted elements are
   10 20 30 40 50

2. Enter the no. of elements : 6
   Enter the array elements
   6 5 4 3 2 1
   The original elements are
   6 5 4 3 2 1
   The Sorted elements are
   1 2 3 4 5 6

Experiment No.7 Date:

MATRIX MULTIPLICATION

AIM: To read two matrices A(m x n)and B(p x q) and Compute the product A and B.

ALGORITHM:

ALGM: matrix multiplication

Step 1: GET THE MATRIX SIZE OF a
Input the size of matrix a and read the values of m and n.
Step 2: GET THE MATRIX VALUE OF a
For i=0 to n-1
For j=0 to n-1
Read a[i][j]
End For
End For
Step 3: GET THE MATRIX SIZE OF b
Input the size of matrix b and read the values of p and q.
Step 4: GET THE MATRIX VALUE OF b
For i=0 to p-1
For j=0 to q-1
Read b[i][j]
End For
End For
STEP :5 MULTIPLICATION OF MATRICES NOT POSSIBLE
If(n!=p)
Print("multiplication is not possible")
Stop
End if
STEP 6: MULTIPLICATION OF MATRICES IS POSSIBLE
For i=0 to m-1
For j=0 to q-1
Sum=0
For k=0 to n-1
Sum=Sum+a[i][k]*b[k][j]
End for
End for
Step 7: DISPLAY RESULT
For i=0 to m-1
For j=0 to q-1
Print c[i][j]
End for
End for
Step 8: STOP
stop

PROGRAM :

```
#include<stdio.h>
#include<conio.h>
int main()
{
        int a[5][5],b[5][5],c[5][5],m,n,p,q,i,j,k;
        clrscr();
        printf("Enter the size of first matrix\n");
        scanf("%d %d",&m,&n);
        printf("Enter the size of second matrix\n");
        scanf("%d %d",&p,&q);
        if(n!=p)
        printf("Matrix multiplication is not possible");
        else
        {
                printf("Enter the elements of first matrix\n");
                for(i=0;i<m;i++)
                for(j=0;j<n;j++)
                scanf("%d",&a[i][j]);
                printf("Enter the elements of the second matrix\n");
                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<q;j++)
                {
                        c[i][j]=0;
                        for(k=0;k<n;k++)
                        c[i][j]=c[i][j]+a[i][k]*b[k][j];
                }
                printf("\n A- matrix is\n");
                for(i=0;i<m;i++)
                {
                        for(j=0;j<n;j++)
                        printf("%d\t",a[i][j]);
                        printf("\n");
                }
                printf("\n B- matrix is\n");
                for(i=0;i<p;i++)
                {
                        for(j=0;j<q;j++)
                        printf("%d\t",b[i][j]);
                        printf("\n");
                }
                printf("The product of two matrix is\n");
                for(i=0;i<m;i++)
                {
                        for(j=0;j<q;j++)
                        printf("%d\t",c[i][j]);
```

```
                    printf("\n");
                }

        }
        getch();
}
```

EXPECTED OUTPUT:

1. Enter the size of first matrix
   2 3
   Enter the size of second matrix
   3 2
   Enter the elements of first matrix
   1 2 3 4 5 6
   Enter the elements of the second matrix
   1 2 3 4 5 6
   A- matrix is
   1 2 3
   4 5 6
   B- matrix is
   1 2
   3 4
   5 6
   The product of two matrix is
   22 28
   49 64

Experiment No.8 Date:

BINARY SEARCHING TECHNIQUE

AIM: To search a name in list of names using Binary Searching Technique

ALGORITHM:

ALGM: Binary search
OPTIMAL-BST(p, q, n)

```
1 for i = 1 to n + 1
2 do e[i, i - 1] = qi-1
3                 w[i, i - 1] = qi-1
4       for l =1 to n
5     do for i = 1 to n - l + 1
6     do j = i + l - 1
7     e[i, j] = â^ž
8      w[i, j] = w[i, j - 1] + pj + qj
9    for r = i to j
10   do t = e[i, r - 1] + e[r + 1, j] + w[i, j]
11  if t < e[i, j]
12   then e[i, j] = t
13    root[i, j] = r
14    return e and root
```

PROGRAM :

```
/* progrm to search an name using binary search */
#include<stdio.h>
#include<string.h>
#include<conio.h>
int main()
{
        char name[10][20], key[20];
        int n, i, low, high, mid, found=0;
        clrscr();
        printf("Enter the number of names to read, n=");
        scanf("%d", &n);
        printf("Enter the names in ascending order\n");
        for(i=0;i<n;i++)
                scanf("%s", name[i]);
        printf("Enter the name to be search:");
        scanf("%s", key);
        low=0;
        high=n-1;
        while(low<=high && !found)
```

```
{
        mid=(low + high)/2;
        if(strcmp(name[mid],key)==0)
                found=1;
else if(strcmp(name[mid],key)<0)
                low=mid+1;
else
                high=mid-1;
}
if(found == 1)
        printf("Name found in position : %d",mid+1);
else
        printf("Name not found");
getch();
}
```

EXPECTED OUTPUT:

1. Enter the number of names to read, n= 5
   Enter the names in ascending order
   Amar
   Chethan
   Girish
   Manoj
   Yadu
   Enter the name to be search:
   Chethan
   Name found in position :2

2. Enter the number of names to read, n= 5
   Enter the names in ascending order
   Girish
   Manoj
   Yadu
   Amar
   Chethan
   Enter the name to be search:
   Kiran
   Name not found

Experiment No: 9 (A) Date:

STRING COPY

AIM: TO execute a C program that Implements string copy operation *STRCOPY(str1,str2)* that copies a string *str1* to another string *str2* without using library function.

ALGORITHM:

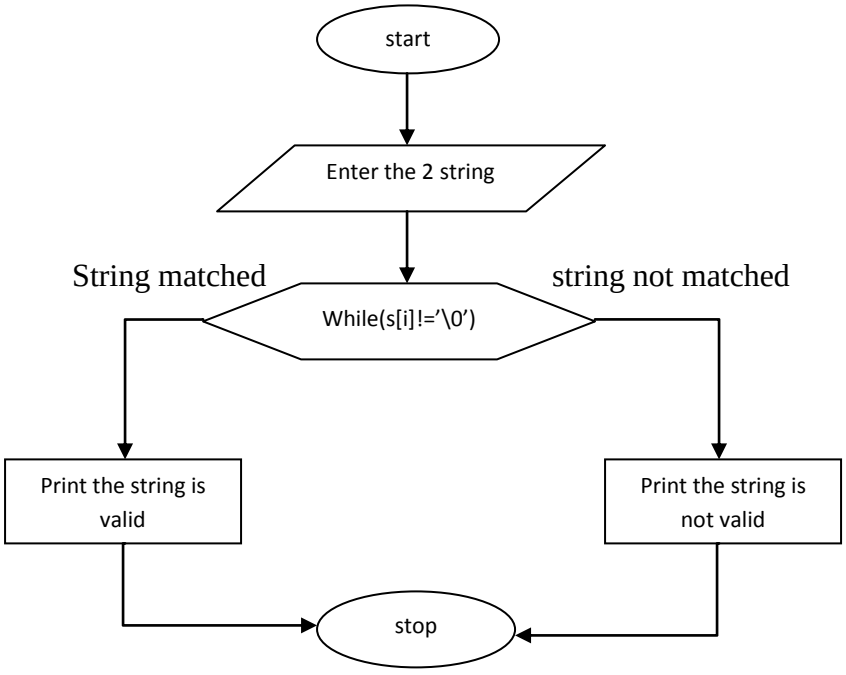
ALGM: EVAL_POLYNOMIAL [To copy string i/p to its o/p, replacing each string of one or more blanks by a single blank].

Inputs: str1 and str2.

Output: str1 -> str2.

1.Start
2. Read the text in Array c
3. FOR i← 0 to c [i] <> '\0' in steps of 1
IF c [i] = ' '
Print ' '
End If
Until c [i] = '\0'
Increment i
End Until
Print the character
End For
4. Stop

FLOWCHART:



Program :

```
/* program to copy a string without using library function */
#include<stdio.h>
#include<conio.h>
// function to copy a string
void strcopy(char s1[50], char s2[50])
{
        int i=0;
        while(s[1i]!='\0')
        {
                s2[i]=s1[i];
                i++;
        }
        s2[i]='\0';
}
int main()
{
        char str1[50],str2[50];
        clrscr();
        printf("Enter the source string\n");
        gets(str1);
        strcopy(str1,str2);
        printf("Destination string is\n");
        puts(str2);
        getch();
}
```

OUTPUT:

1. Enter the source string
   Drsmce
   Destination string is
   Drsmce

2. Enter the source string
   1234
   Destination string is
   1234

Experiment No. 9(B) Date:

VOWELS AND CONSONANTS COUNT

AIM: To Read a sentence and prints frequency of each of the vowels and total count of consonants.

ALGORITHM:

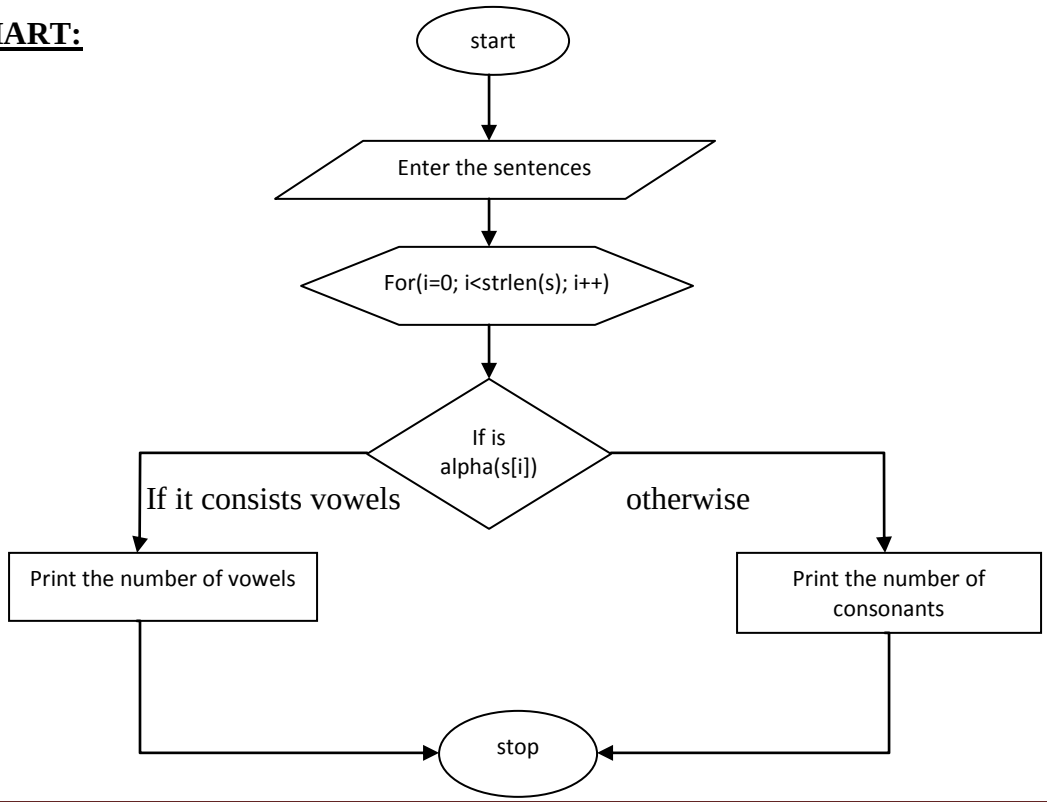
ALGM: VOWELS AND CONSONANTS COUNT[To read a sentence and prints frequency of each of the vowels and total count of consonants.**]**

Input: Sentence.

Output: Print vowels(a,e,i,o,u) ,consonants(other than vowels) and print their count.

STEP 1.Set countCo:=0[C is the counter variable for consonants]
STEP 2.Set countVo:=0[counter for vowels]
STEP 3.[find length of STR] Set L:=LENGTH(STR).
STEP 4:TOLOWER(STR,L) [change all characters in string to lower case]
STEP 5. Repeat for K:=0 to L-1:
If STR[K]='a' OR STR[K]='e' OR STR[K]='i' OR STR[K]='o' OR STR[K]='u'.Then:
Set countVo:=countVo+1
Else
Set countCo:=countCo+1
[End of If-Else]
[End of for loop]
STEP 6.PRINT countVo,countCo [display results]
STEP 7.EXIT/END

FLOWCHART:



PROGRAM :

/* program to prints frequency of vowels and total count of consonants */

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
char s[100],ch;
int i,vc=0,cc=0;
clrscr();
printf("Enter the sentence\n");
gets(s);
for(i=0;i<strlen(s);i++)
{
if(isalpha(s[i]))
{
ch=tolower(s[i]);
if(ch=='a'||ch=='e'||ch=='i'||ch=='o'||ch=='u')
vc++;
else
cc++;
}
}
printf("No of vowels=%d\n",vc);
printf("No of consonants=%d",cc);
getch();
}
```

OUTPUT:

1. Enter the sentence
Welcome to drsmce
No of vowels=5
No of consonants=10

2. Enter the sentence
qwerty@#$
No of vowels=1
No of consonants=5

Experiment No. 10(a) Date:

RIGHT ROTATE AND *ISPRIME* OR NOT

10. i)Design and develop a C function RightShift(x ,n) that takes two integers x and n as input and
returns value of the integer x rotated to the right by n positions. Assume the integers are unsigned.
Write a C program that invokes this function with different values for x and n and tabulate the
results with suitable headings.
ii) Design and develop a C function isprime(num) that accepts an integer argument and returns 1
if the argument isprime, a 0 otherwise. Write a C program that invokes this function to generate
prime numbers between the given ranges

10.1 Right Rotate by n Positions

Design and develop a C function Right Shift(x ,n) that takes two integers x and n as input and returns value of the integer X rotated to the right by n positions.

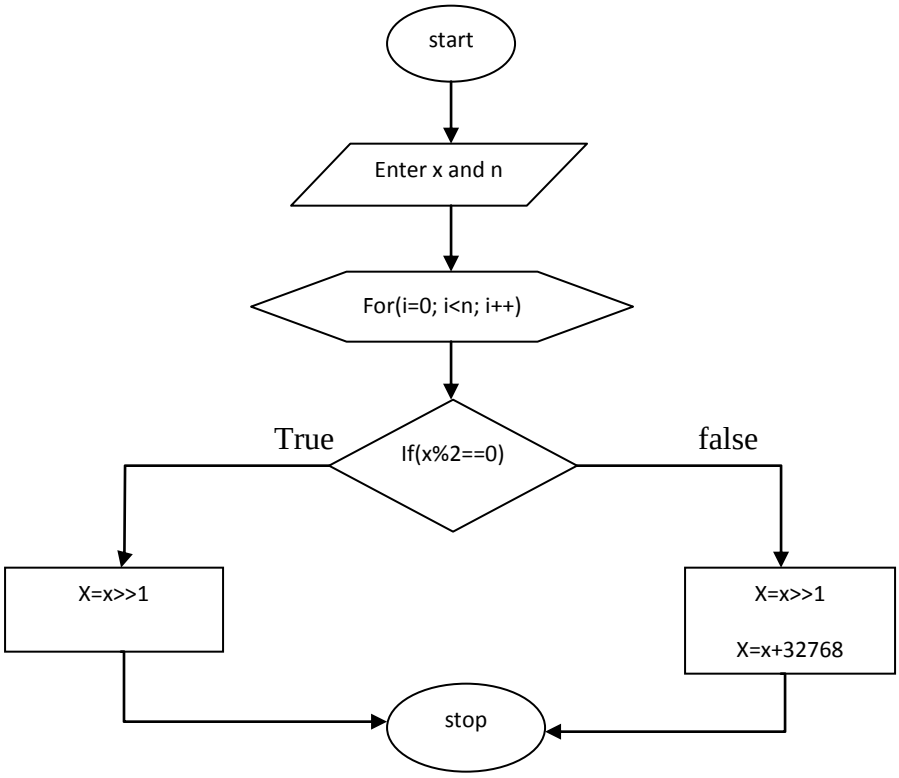
Algorithm:

ALGM: RIGHT ROTATE BY N POSITIONS [A 'C' function Right Shift(x ,n) that takes two integers x
and n as input and returns value of the integer X rotated to the right by n positions.]

Input: Integers x and n.
Output: Integer X rotated to the right by n positions.

1.Start
2.Read x and n
3.Callfunction RR(x,n)
4.Print the result
5.Stop
Algorithm to rotate value by bits
1.IF n==0
Return x
ELSE
Return((x>>n)|(x<<(32-n)))
2. Return

Flowchart:



Program :

```
/* program to rotate right */
#include<stdio.h>
#include<conio.h>
#include<math.h>
//function to right rotate
unsignedint RightShift(unsignedint x,unsignedint n)
{
int i;
for(i=0;i<n;i++)
{
if(x%2==0)
x=x>>1;
else
{
x=x>>1;
x=x+32768;
}
}
return(x);
}
void main()
{
unsigned int x,y,n,i,value;
char input;
clrscr();
```

```
        do
        {
        printf("\nEnter the number and the no of bits to be" "rotated\n");
        scanf("%u %u",&x,&n);
       y=x;
       value=RightShift(x,n);
       printf("\nCALULATED VALUE rightrot of %u,%u=%u",y,n,value);
       printf("\nTocontinue press 'y' else press 'n'\n");
       input=getche();
       }while(input!='n');
       getch();
}
```

Output:

1. Enter the number and the no of bits to be rotated
   4 1
   CALULATED VALUE rightrot of 4,1=2
   To continue press 'y' else press 'n'
   y
   Enter the number and the no of bits to be rotated
   5 2
   CALULATED VALUE rightrot of 5,2=16385
   To continue press 'y' else press 'n'

Experiment No: 10(b) Date:

TO CHECK PRIME OR NOT

10 ii). Design and develop a function isprime (x) that accepts an integer argument and returns 1 if the argument is prime and 0 otherwise. The function must use plain division checking approach to determine if a given number isprime. Invoke this function from the main with different values obtained from the user and print appropriate messages

Algorithm:

ALGM: CHECK PRIME OR NOT [A function isprime (x) that accepts an integer argument and returns 1 if the argument is prime and 0 otherwise.]

Input : Any integer number (up to 32767)
Output: Is it a prime number or Not

Complexity O(n)

Prime(num)

1 Set i=2

2 while i<=num/2

3 if num mod i = 0

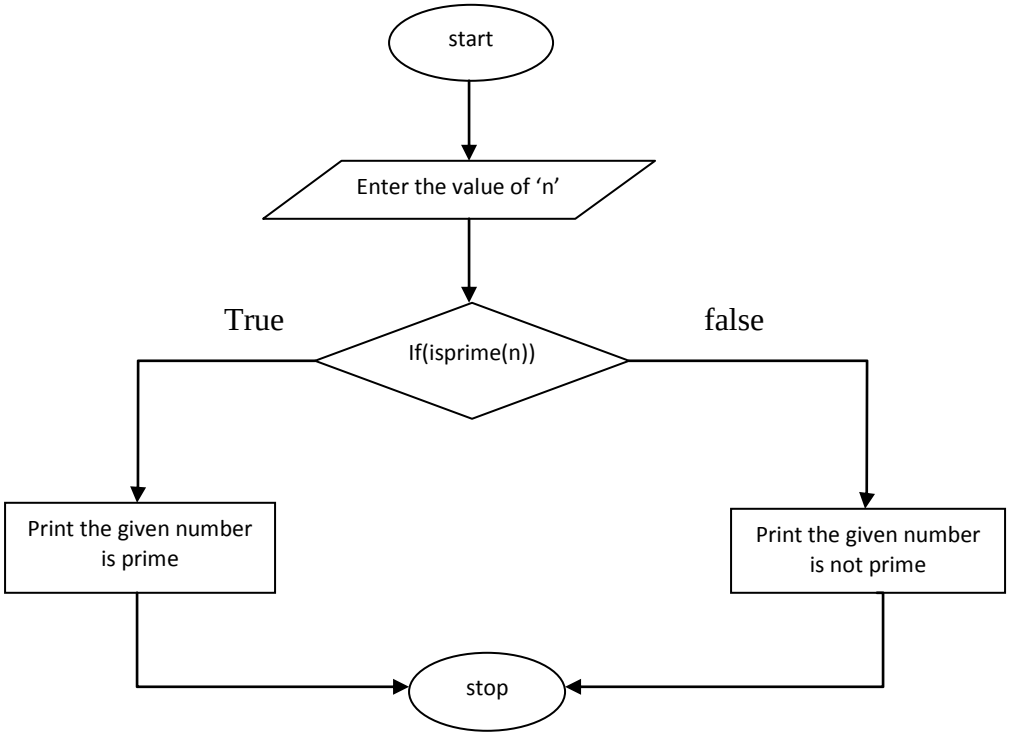
4 print "Not a Prime number" and exit;

5 i=i+1

6 if (i==(num/2)+1)

7 Print "Prime number"

Flowchart:



Program:

```
//Function to check the given number is prime or not
#include<stdio.h>
#include<conio.h>
void main()
{
int n;
clrscr();
printf("Enter value of n\n");
scanf("%d",&n);
if(isprime(n));
printf("%d is prime number\n");
        else
printf("%d is not a prime number\n") ;
getch();
}


int isprime(int num)
{
int i;
for(i=2;i<= m/2;i++)
{
if(m%i==0)
{
```

```
return 0;
                }
        }
        return 1;
}
```

Output:

1. Enter value of n

3

3 is prime number

2. Enter value of n

4

4 is not a prime number

Experiment No: 11 Date:

FACTORIAL OF NUMBER USING RECURSIVE FUNCTION

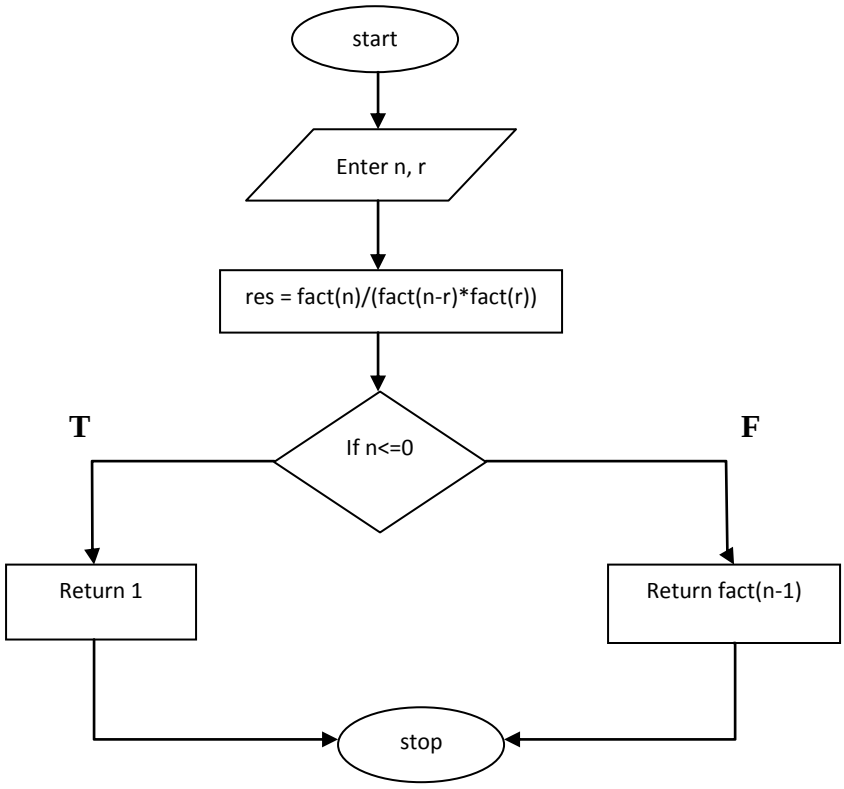
Draw the flow chart and write a Recursive C function to find the factorial of number n! defined by fact(n)=1, if n=0, otherwise fact(n)=n*fact(n-1),using this function write a c program to compute the binomial co-efficient nCr. Tabulate the results for different values of n and r using suitable messages.

Algorithm:

ALGM: Factorial of number [A recursive C function to find the factorial of number n! defined by fact(n)=1, if n=0, otherwise fact(n)=n*fact(n-1)]

```
FUNCTION FACTORIAL (N: INTEGER): INTEGER
(* RECURSIVE COMPUTATION OF N FACTORIAL *)
BEGIN
 (* TEST FOR STOPPING STATE *)
 IF N <= 0 THEN
   FACTORIAL := 1
 ELSE
   FACTORIAL := N * FACTORIAL(N - 1)
END; (* FACTORIAL *)
```

Flowchart



Program :

```
#include<stdio.h>
#include<conio.h>
int fact(int n)
{
        if(n==0)
        {
                return 1;
        }
        return (n*fact(n-1));
}
int main()
{
        int n,r,res;
        clrscr();
        printf("Enter the value of n and r\n");
        scanf("%d%d",&n,&r);
        res=fact(n)/(fact(n-r)*fact(r));
        printf("The NCR is = %d",res);
        getch();
}
```

Output:

1. Enter the value of n and r

4 2

The NCR is =6

2. Enter the value of n and

7 3

The NCR is =6

Experiment No: 12 Date:

COPY THE CONTENTS OF FILE

Given two university information files "studentname.txt" and "usn.txt" that contains students names and USN respectively. Write a C program to a new file called "output.txt" and copy the content of files "studentname.txt" and "usn.txt" into output file in the sequence shows below. Display the content of output file "output.txt" on to the screen.

Note : Students are required to create two files "Studname.txt" and "Studusn.txt" with the Contents

studname.txt	**studusn.txt**
Amar	1RN10CS005
Suresh	1RN10CS012
Kiran	1RN10CS036
Shashi	1RN10CS072

ALGORITHM:

Input : Create the file fp1,fp2,fp3

Output : Print whether the files are found or not

step1:start

Step2: create the files (fp1,fp2, & fp3)

Step3: while(!feof((fp1)&&!feof(fp2))

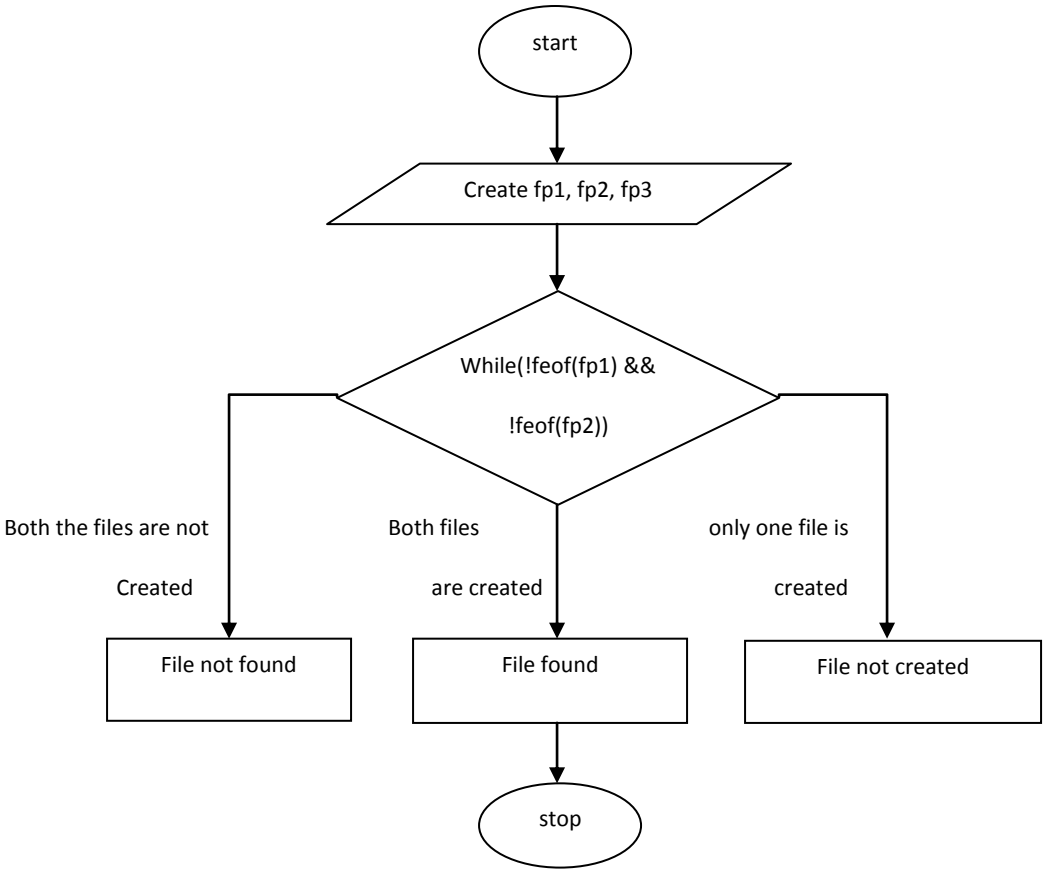
Step4:Both files the not created the print the file not found

a). if only one file is created then print files not found.

b).if both files are created the print files are found.

Step5: stop

FLOWCHART:



Program

```
/* Program on merge two files */
#include <stdio.h>
#incude<conio.h>
int main()
{
        FILE *fp1,*fp2,*fp3;
        char usn[20], name[20];
        clrscr();
        fp1=fopen("studname.txt", "r");
        if(fp1 == NULL)
        printf(" File not found");
        fp2=fopen("studusn.txt", "r");
        if(fp2 == NULL)
        printf(" File not found");
        fp3=fopen("output.txt","w");
        while( !feof(fp1) && !feof(fp2) )
        {
                fscanf(fp1,"%s",name);
                fscanf(fp2,"%s",usn);
                fprintf(fp3,"%15s %10s\n", name, usn);
        }
```

```
        fclose(fp1);
        fclose(fp2);
        fclose(fp3);
        fp3=fopen("output.txt","r");
        printf("\n----------------------------\n");
        printf(" Name USN \n");
        printf("----------------------------\n");
        while(!feof(fp3))
        {
                fscanf(fp3,"%s",name);
                fscanf(fp3,"%s \n",usn);
                printf("%-15s %10s \n", name,usn);

        }
        fclose(fp3);
        getch();
}
```

Output:

Name	USN
Amar	1RN10CS005
Suresh	1RN10CS001
Kiran	1RN10CS036
Shashi	1RN10CS072

Experiment No: 13 Date:

STUDENT RECORD USING ARRAY OF STRUCTURES

Write a C program to maintain a record of "n" student details sing an array of structures with four fields (Roll number, Name, marks, and Grade). Each field is of an appropriate data type. Print the marks of the student given name as input.

ALGORITHM:

Input: Enter the number of students
Output: print the marks of the student.

Step1: start

Step2: enter the num of students

Step3: if(strcmp(s[i].name, sname==0)

Then

{

Print the marks of students name & USN

}

Else

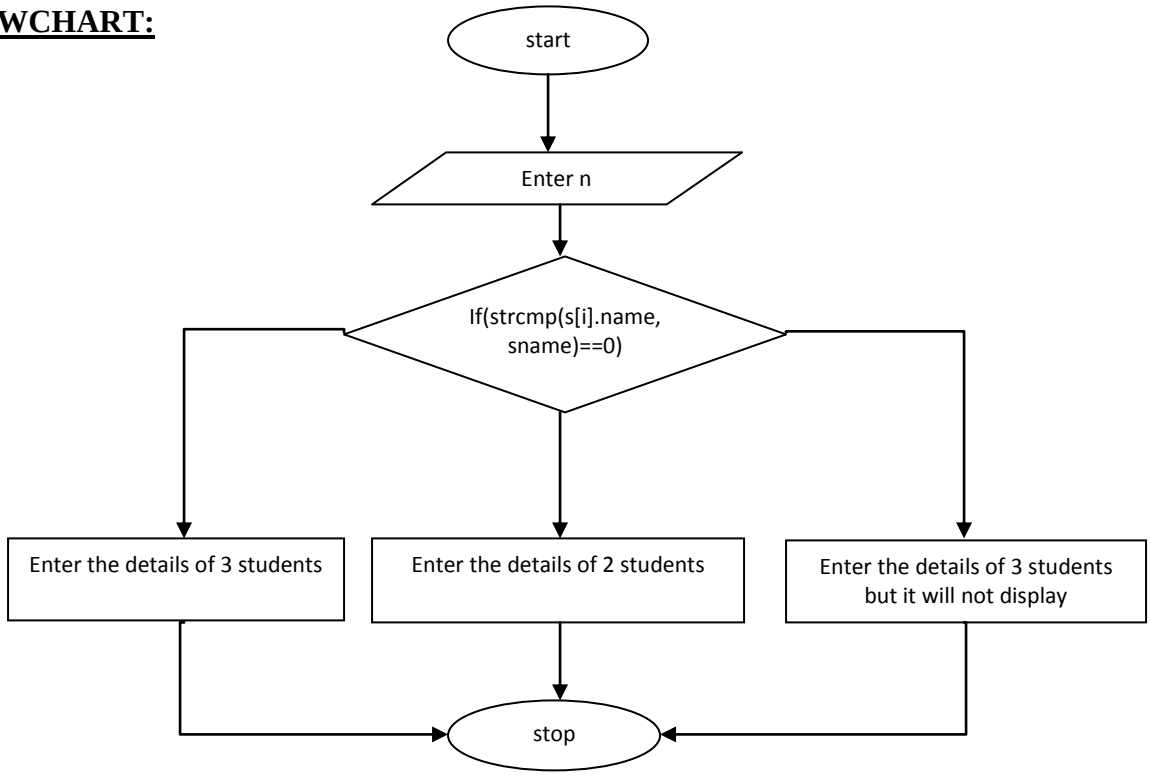
{

Print the given date is invaid

}

Step4: stop

FLOWCHART:



Program:

```
/* program to maintain a record of student using structrue */
#include<stdio.h>
#include<conio.h>
struct student
{
        int rollno, marks;
        char name[20], grade;
};
Void  main()
{
        int i,n,found=0;
        struct student s[10];
        char sname[20];
        clrscr();
        printf("Enter the number of student details n=");
        scanf("%d",&n);
        for(i=0;i<n;i++)
        {
                printf("\nenter the %d student details \n",i+1);
                printf("enter the roll number:");
                scanf("%d",&s[i].rollno);
```

```
                printf("enter the student name without white spaces:");
                scanf("%s", s[i].name);
                printf("enter the marks : ");
                scanf("%d", &s[i].marks);
                printf("enter the grade : ");
                fflush(stdin);
                scanf("%c",&s[i].grade);
        }
        printf("\nStudent details are \n");
        printf("\nRollno\tName\t\t\tMarks\tGrade\n");
        for(i=0;i<n;i++)
                printf("%d\t%s\t\t%d\t%c\n", s[i].rollno, s[i].name, s[i].marks, s[i].grade);
        printf("\nEnter the student name to print the marks:");
        scanf("%s", sname);
        for(i=0;i<n;i++)
        {
                if(strcmp(s[i].name,sname)==0)
                {
                        Printf("\Marks of the student is : %d",s[i].marks);
                        found=1;
                }
        }
        if(found ==0)
                printf(" Given student name not found\n");
                getch();
}
```

Output:

Enter the number of student details n=3

enter the 1 student details
enter the roll number: 100
enter the student name without white spaces:vishwa
enter the marks : 90
enter the grade : A

enter the 2 student details
enter the roll number: 101
enter the student name without white spaces:madhu
enter the marks : 50
enter the grade : B

enter the 3 student details
enter the roll number: 102
enter the student name without white spaces:kiran
enter the marks : 45
enter the grade : C

Roll	Name	Marks	Grad
100	vishwa	90	A
101	madhu	50	B
102	kiran	45	C

Enter the student name to print the marks: madhu

Marks of the student is : 50

Experiment No: 14 Date:

POINTERS

Write a C program using pointers to compute the sum, mean and standard deviation of all elements stored in an array of n real numbers.

Mean : is the **average** of the numbers. **i.e** the sum of a collection of numbers divided by the number of numbers in the collection

Standard deviation (**SD**): measures the amount of variation or dispersion from the average. For a finite set of numbers, the standard deviation is found by taking the square root of the average of the squared differences of the values from their average value.

ALGORITHM:

ALGM: COMPUTATION OF SUM, MEAN AND STANDARD DEVIATION [to compute the sum, mean and standard deviation of all elements stored in an array of n real numbers.]

Input: Enter the n values
Output: print the result for sum, mean, standard deviation

Step 1:start

Step2: Enter the value of n it may be integer or float

Step3: for(i=0; i<n;i++)

{

Sum=sum+*ptr;

Ptr++;

Mean=sum/n;

Ptr=a;

For(i=0;i<n;i++)

{

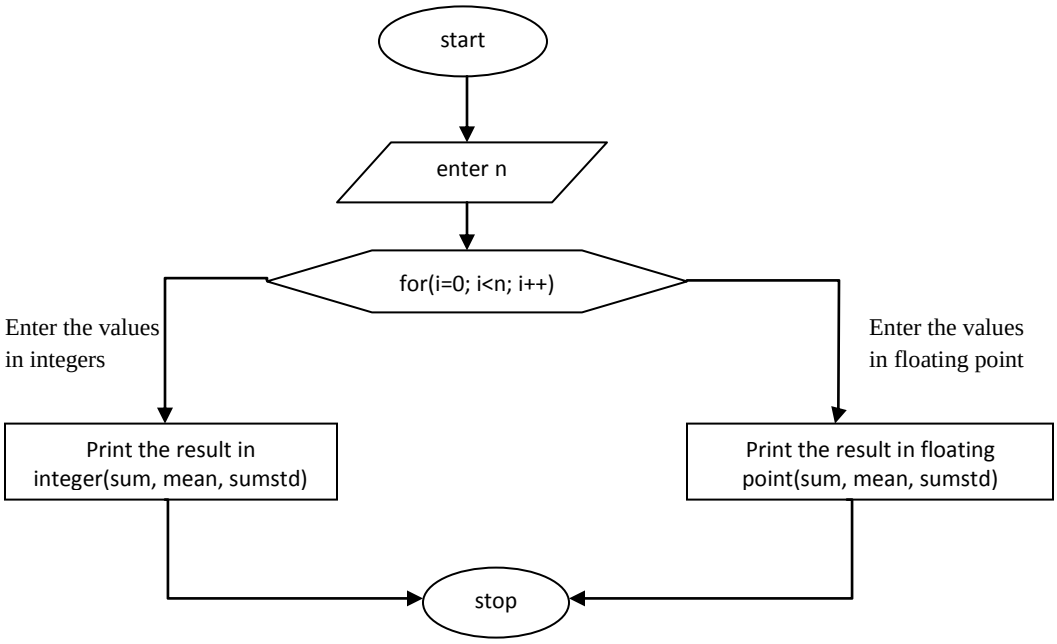
Sumstd=sumstd+pow((*ptr-mean),2);

Ptr++;

Step4:print the result for sum,mean,&stddev

Step5:stop

FLOWCHART:



Program:

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
int main()
{
        float a[10], *ptr, mean, std, sum=0, sumstd=0;
        int n,i;
        clrscr();
        printf("Enter the no of elements\n");
        scanf("%d",&n);
        printf("Enter the array elements\n");
        for(i=0;i<n;i++)
        {
                scanf("%f",&a[i]);
        }
        ptr=a;
        for(i=0;i<n;i++)
        {
                sum=sum+ *ptr;
                ptr++;
        }
        mean=sum/n;
        ptr=a;
        for(i=0;i<n;i++)
        {
```

```
                sumstd=sumstd + pow((*ptr - mean),2);
                ptr++;
        }
        std= sqrt(sumstd/n);
        printf("Sum=%.3f\t",sum);
        printf("Mean=%.3f\t",mean);
        printf("Standard deviation=%.3f\t",std);
        getch();
}
```

Output:

Enter the no of elements
5
Enter the array elements
5 3 9 8 7
Sum=32.000
Mean=6.400
Standard deviation=2.154