

COMPUTER PROGRAMMING LABORATORY (Effective from the academic year 2018 -2019) SEMESTER – I/II			
Subject Code	18CPL17/ 27	CIE Marks	40
Number of Contact Hours/Week	3	SEE Marks	60
Total Number of Lab Contact Hours	30	Exam Hours	3 Hrs
Credits – 1			
Descriptions (if any):			
<ul style="list-style-type: none">The laboratory should be preceded or followed by a tutorial to explain the approach or algorithm being implemented / implemented for the problems given.Note that experiment 1 is mandatory and written in the journal.Questions related with experiment 1, need to be asked during viva-voce for all experiments.Every experiment should have algorithm and flowchart be written before writing the program.Code should be traced using minimum two test cases which should be recorded.It is preferred to implement using Linux and GCC.			
Laboratory Programs:			
1.	Familiarization with computer hardware and programming environment, concept of naming the program files, storing, compilation, execution and debugging. Taking any simple C- code.		
PART A			
2.	Develop a program to solve simple computational problems using arithmetic expressions and use of each operator leading to simulation of a commercial calculator. (No built-in math function)		
3.	Develop a program to compute the roots of a quadratic equation by accepting the coefficients. Print appropriate messages.		
4.	Develop a program to find the reverse of a positive integer and check for palindrome or not. Display appropriate messages.		
5.	An electricity board charges the following rates for the use of electricity: for the first 200 units 80 paise per unit: for the next 100 units 90 paise per unit: beyond 300 units Rs 1 per unit. All users are charged a minimum of Rs. 100 as meter charge. If the total amount is more than Rs 400, then an additional surcharge of 15% of total amount is charged. Write a program to read the name of the user, number of units consumed and print out the charges.		
6.	Introduce 1D Array manipulation and implement Binary search.		
7.	Implement using functions to check whether the given number is prime and display appropriate messages. (No built-in math function)		
PART B			
8.	Develop a program to introduce 2D Array manipulation and implement Matrix multiplication and ensure the rules of multiplication are checked.		
9.	Develop a Program to compute Sin(x) using Taylor series approximation .Compare your result with the built- in Library function. Print both the results with appropriate messages.		
10.	Write functions to implement string operations such as compare, concatenate, string length. Convince the parameter passing techniques.		
11.	Develop a program to sort the given set of N numbers using Bubble sort.		
12.	Develop a program to find the square root of a given number N and execute for all possible inputs with appropriate messages. Note: Don't use library function sqrt(n).		
13.	Implement structures to read, write, compute average- marks and the students scoring above and below the average marks for a class of N students.		
14.	Develop a program using pointers to compute the sum, mean and standard deviation of all elements stored in an array of n real numbers.		
15.	Implement Recursive functions for Binary to Decimal Conversion.		
Laboratory Outcomes: The student should be able to:			

- Write algorithms, flowcharts and program for simple problems.
- Correct syntax and logical errors to execute a program.
- Write iterative and wherever possible recursive programs.
- Demonstrate use of functions, arrays, strings, structures and pointers in problem solving.

Conduct of Practical Examination:

- All laboratory experiments, excluding the first, are to be included for practical examination.
- Experiment distribution
 - For questions having only one part: Students are allowed to pick one experiment from the lot and are given equal opportunity.
 - For questions having part A and B: Students are allowed to pick one experiment from part A and one experiment from part B and are given equal opportunity.
- Strictly follow the instructions as printed on the cover page of answer script for breakup of marks
- Change of experiment is allowed only once and marks allotted for procedure part to be made zero.
- Marks Distribution (*Subjected to change in accordance with university regulations*)
 - a) For questions having only one part
Procedure + Execution + Viva-Voce: $15+70+15 = 100$ Marks
 - b) For questions having part A and B
 - i. Part A – Procedure + Execution + Viva = $4 + 21 + 5 = 30$ Marks
 - ii. Part B – Procedure + Execution + Viva = $10 + 49 + 11 = 70$ Marks

Algorithm:

- An algorithm is a basic tool which is used to express the solution for a given problem systematically in the form of instructions. This solution is called logic. It takes a set of input values and produces the desired output.
- An algorithm is defined as unambiguous, step by step procedure (instructions) to solve a given problem in finite number of steps by accepting a set of inputs and producing the desired output. After producing the result, the algorithm should terminate. Usually, Algorithm are written in simple English like statements along with simple mathematical expressions.

Characteristics of an Algorithm:

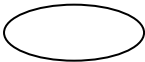
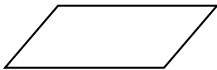
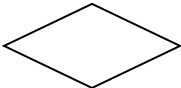

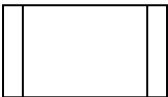
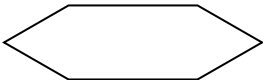
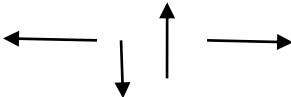
- **Input:** It may accept a zero or more inputs.
- **Output:** It should produce at least one output (result).
- **Definiteness:** Each instruction must be clear, well defined and precise. There should not be any ambiguity.
- **Finiteness:** It should be a sequence of finite instructions. That is, it should end after a fixed time. It should not enter into an infinite loop.
- **Effectiveness:** This means that operations must be simple and are carried out in a finite time at one or more levels of complexity. It should be effective whenever traced manually for the results.

Algorithmic Notations:

- Name of the Algorithm
- Step number
- Explanatory Comment
- Termination

Flowchart:

- A flow chart is a pictorial representation of an algorithm. That is flowchart consists of sequence of instructions that are carried out in an algorithm. All the steps are drawn form of different shapes of boxes, circle and connecting arrows. Flowcharts are mainly used to help programmer to understand the logic of the program.
- The various types of geometric shapes, arrows and symbols used while drawing the flowchart are called flowchart symbols. The symbols used and the meaning associated with each symbol are shown below.

<u>Symbols used</u>	<u>Meaning associated with symbols</u>
	Start or end of the flowchart (program)
	Input or output operation
	Decision making and branching
	Connector
	Predefined computation or process (used with functions are used)
	Repetition or a loop which is normally used to execute a group of instructions for a specifies number of times
	Flow of control

1. Familiarization with programming environment, concept of naming the program files, storing, compilation, execution and debugging. Taking any simple C- code.

Algorithm	Flowchart
<p>Step 1. [Initialize] Start</p> <p>Step 2. [Input the length and breadth] Read l, b</p> <p>Step 3. [Calculate area] $\text{area} = l * b$</p> <p>Step 4. [Calculate perimeter] $\text{peri} = 2(l+b)$</p> <p>Step 5. [Display area and perimeter] Print area, peri</p> <p>Step 6. [Finished] Stop.</p>	<pre> graph TD Start([Start]) --> Print1[/Print "Enter Length and Breadth"/] Print1 --> Read[/Read l, b/] Read --> Process[area = l * b peri = 2(l+b)] Process --> Print2[/Print area, peri/] Print2 --> Stop([Stop]) </pre>
Program	Output
<pre> #include<stdio.h> { int l, b, area, peri; printf("Enter length and breadth\n"); scanf("%d%d", &l, &b); area = l * b; peri = 2 * (l + b); printf("The area is %d\n", area); printf("The perimeter is %d\n", peri); } </pre>	<p>Enter length and breadth</p> <p>2</p> <p>4</p> <p>The area is 8</p> <p>The perimeter is 12</p>

Linux machine, you have to follow the steps mentioned below:

- Go to the Linux command prompt (# or \$).
- Type **vi program1.c** for example vi program1.c
- The vi editor will open.
- Press **Insert** key or **I** key to start to type. and Type the program in the editor.
- After typing the program, Press **Esc, Shift and then :**
- Type **w and q** followed by **filename** for example program1.c where **w** is used to **Save a file**, where as **q** to **quit**.
- To **compile** a program, At the command prompt type **cc program1.c**
- Type **./a.out** to run the program.

2. Develop a Program to solve simple computational problems using arithmetic expressions and use of each operator leading to simulation of a Commercial calculator (no Built in math function).

Algorithm: COMMERCIAL CALCULATOR

Step 1: Start

Step 2: Read the values of a and b.

Step 3: Read Ch

Step4: Verify ch value by using switch statement if ch=1 then goto step 5 or if ch=2 then goto step 6 or if read ch is 3 then goto step 7 or if read ch is 4 then goto step 8 or if read ch is 5 then goto step 9 or otherwise goto step 10.

Step 5: if ch=1 then compute $res=a+b$ and goto step 11.

Step 6: if ch=2 then compute $res=a-b$ and goto step 11.

Step 7: if ch=3 then compute $res=a*b$ and goto step 11.

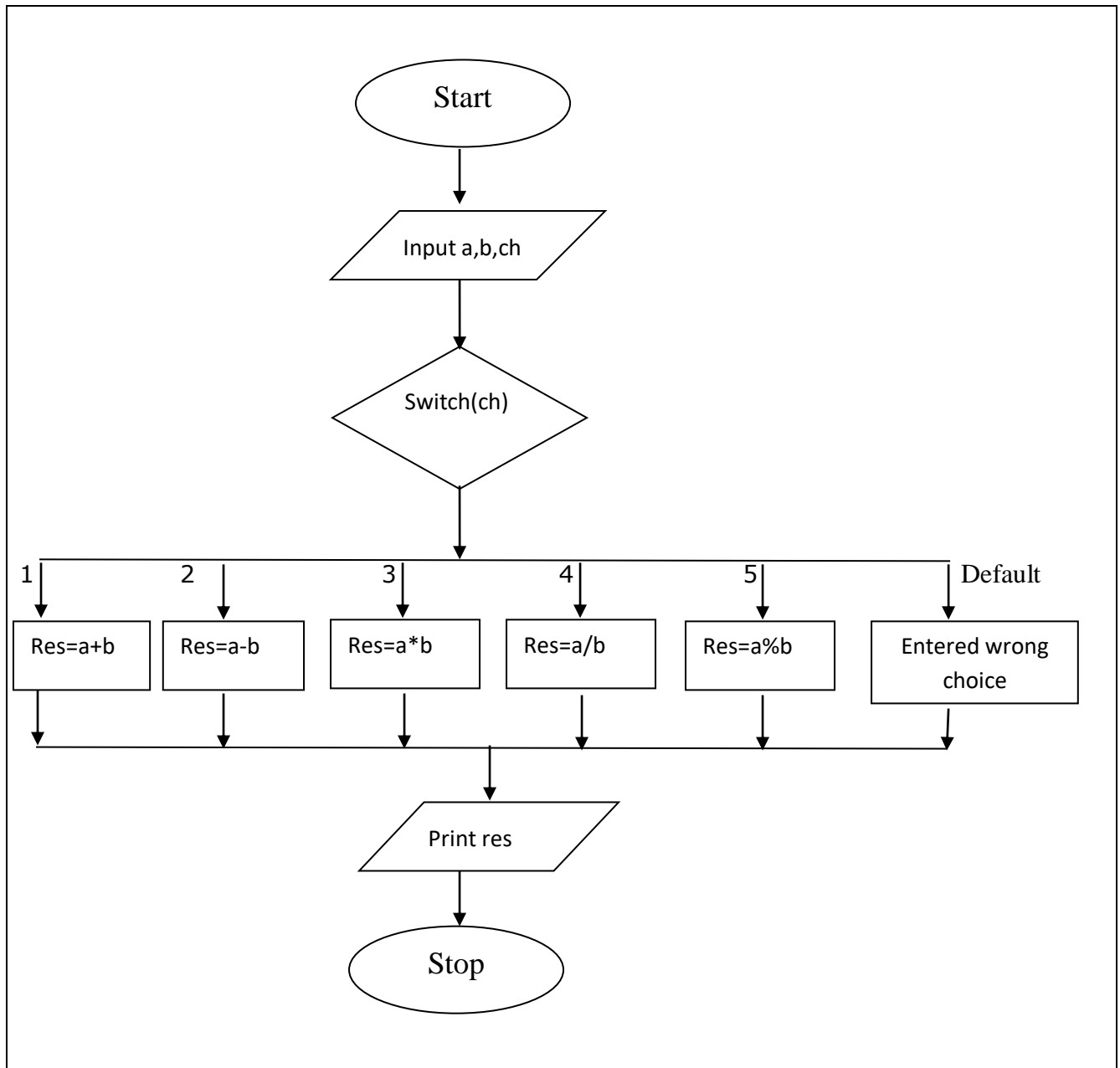
Step 8: if ch=4 then compute $res=a/b$ and goto step 11.

Step 9: if ch=5 then compute $res=a\%b$ and goto step 11.

Step 10: if ch value is other than above options then print wrong choice entered then goto step 12.

Step 11: Print res and goto step 12

Step 12: stop.

Flowchart**Viva Questions:**

- What is switch statement.
- What is a case in a switch statement?
- Is Default necessary in switch case?
- How many cases can you have in a switch statement?

Program 2

```
#include<stdio.h>
#include<stdlib.h>
void main()
{
    int ch,a,b;
    float res;
    printf("Enter two numbers:\n");
    scanf("%d%d",&a,&b);
    printf("1=Add, 2=Sub, 3=Mul, 4=Div, 5=Rem\n");
    printf("Enter your choice:\n");
    scanf("%d",&ch);
    switch(ch)
    {
        case 1: res=a+b;
                break;
        case 2: res=a-b;
                break;
        case 3: res=a*b;
                break;
        case 4: res=(float)a/b;
                break;
        case 5: res=a%b;
                break;
        default: printf("Entered Wrong choice\n");
    }
    printf("Result=%d\n",res);
}
```

Output

```
Enter two numbers: 10      20
1=Add, 2=Sub, 3=Mul, 4=Div, 5=Rem
Enter your choice:1
Result=30
```


3) Develop a program to compute the roots of a quadratic equation by accepting the coefficients. Print the appropriate messages.

ALGORITHM: ROOTS OF A QUADRATIC EQUATION

Step 1: Start

Step 2: Read the values of non-zero coefficients a,b and c.

Step 3: If a|b|c is zero goto Step 9. Otherwise Compute the value of discriminant (disc) which is Equal to $(b*b)-(4*a*c)$.

Step 4: Check if disc is equal to 0. If true, then go to Step 5. Otherwise, goto Step 6

Step 5: Compute the roots.

$$\text{root1} = (-b)/(2*a)$$

$$\text{root2} = \text{root1}$$

Output the values of roots, root1 and root2. Go to Step 9

Step 6: Check if disc is greater than zero or not. If true, then go to Step 7. Otherwise, goto Step 8.

Step 7: Compute the real and distinct roots,

$$\text{root1} = (-b + \sqrt{\text{disc}})/(2*a)$$

$$\text{root2} = (-b - \sqrt{\text{disc}})/(2*a)$$

Output the values of roots, root1 and root2. Go to Step 9.

Step 8: Compute the complex and distinct roots.

$$\text{Compute the real part, } r_part = (-b)/(2*a)$$

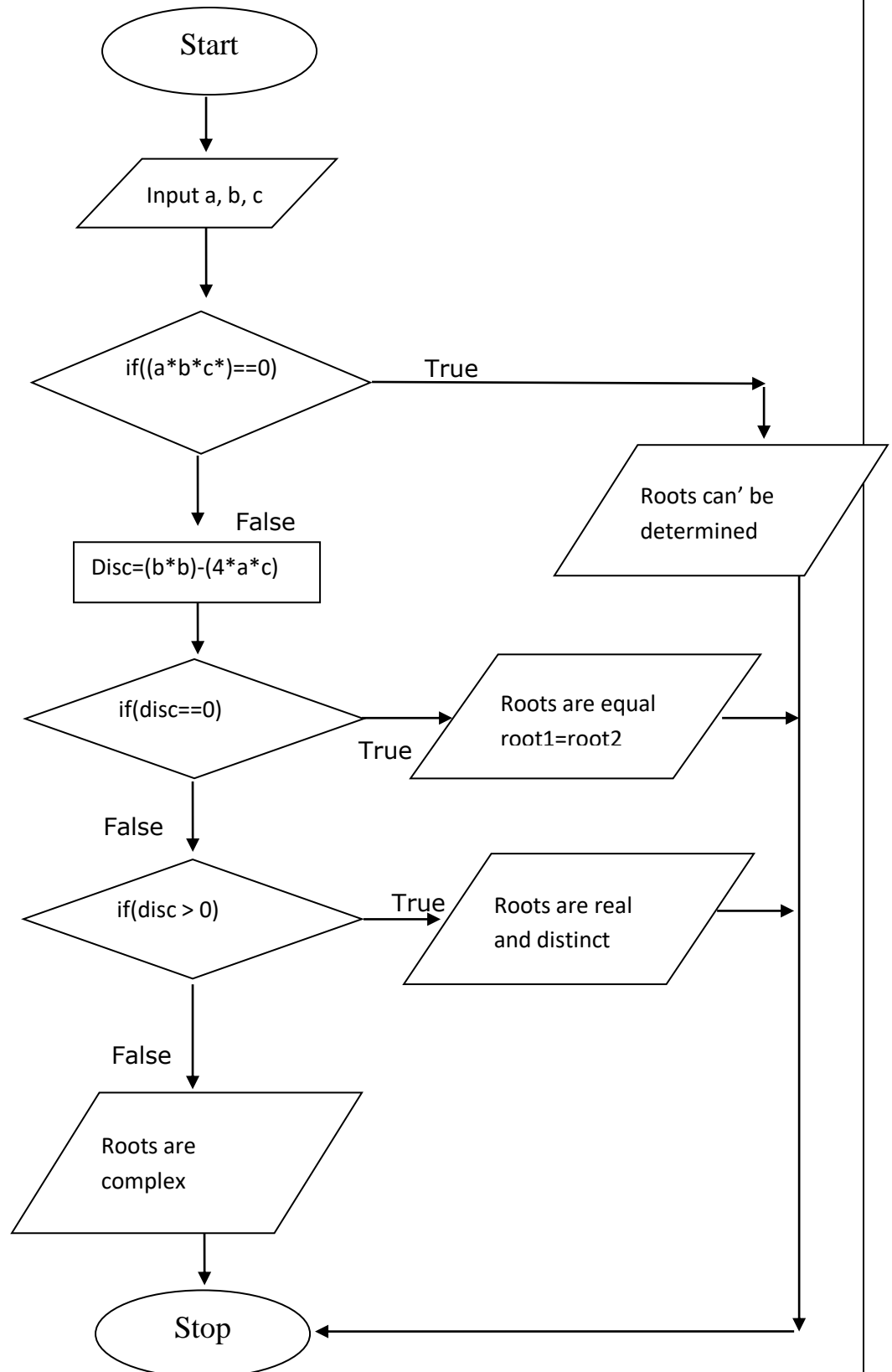
$$\text{Compute the imaginary part, } i_part = \sqrt{-\text{disc}}/(2*a)$$

Output roots as

$$\text{root1} = r_part + i_part$$

$$\text{root2} = r_part - i_part$$

Step 9: Stop.

Flow chart

Program 3

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
int main( )
{
    float a, b, c, root1, root2, rpart, ipart, disc;
    printf("Enter the 3 coefficients:\n");
    scanf("%f%f%f", &a, &b, &c);
    if((a*b*c) == 0)
    {
        printf("Roots cannot be Determined:\n");
        exit(0);
    }
    disc = (b*b) - (4*a*c);
    if(disc == 0)
    {
        printf("Roots are equal\n");
        root1=root2= -b / (2*a);
        printf("root1=root2=%f",root1);
    }
    else if(disc>0)
    {
        printf("Roots are real and distinct\n");
        root1= (-b + sqrt(disc)) / (2*a);
        root2= (-b - sqrt(disc)) / (2*a);
        printf ("root1 = %f \n root2 = %f\n", root1, root2);
    }
    else
    {
        printf("Roots are complex\n");
        rpart = -b / (2*a);
        ipart = (sqrt (-disc)) / (2*a);
        printf("root1 = %f + i %f \n", rpart, ipart);
        printf("root2 = %f - i %f \n", rpart, ipart);
    }
}
```

Output**First Run:**

Enter the 3 coefficients:

1

2

1

Roots are equal

root1=root2=-1.000000

Second Run:

Enter the 3 coefficients:

2

3

2

Roots are real and equal

root1=-0.500000

root2=-2.000000

Third Run:

Enter the 3 coefficients:

2

2

2

Roots are complex

root1=-0.500000+i 0.866025

root2=-0.500000- i 0.866025

Viva Questions:

- What is quadratic Equation?
- What is math.h ? why it is used in C program?
- What are decision making statements in C language?
- Write the syntax of “ if ”statement?
- Difference between if and switch statements?

4) Develop a program to find the *reverse* of a positive integer and check for PALINDROME or NOT. Display appropriate messages.

ALGORITHM: TO CHECK WHETHER PALINDROME OR NOT.

Step 1: Start

Step 2: Read the integer number, digitno.

Step 3: Keep digitno in a temporary variable to verify with it's reverse later and decide whether it is a palindrome or not. $num = digitno$

Step 4: Initialize reverse to zero.

Step 5: Output the four digit number, digitno.

Step 6: Check if digitno is not equal to zero. If true, repeat Step 7 to Step 9. Otherwise (i.e if digitno is equal to zero), goto Step 10.

Step 7: Compute remainder = $digitno \% 10$.

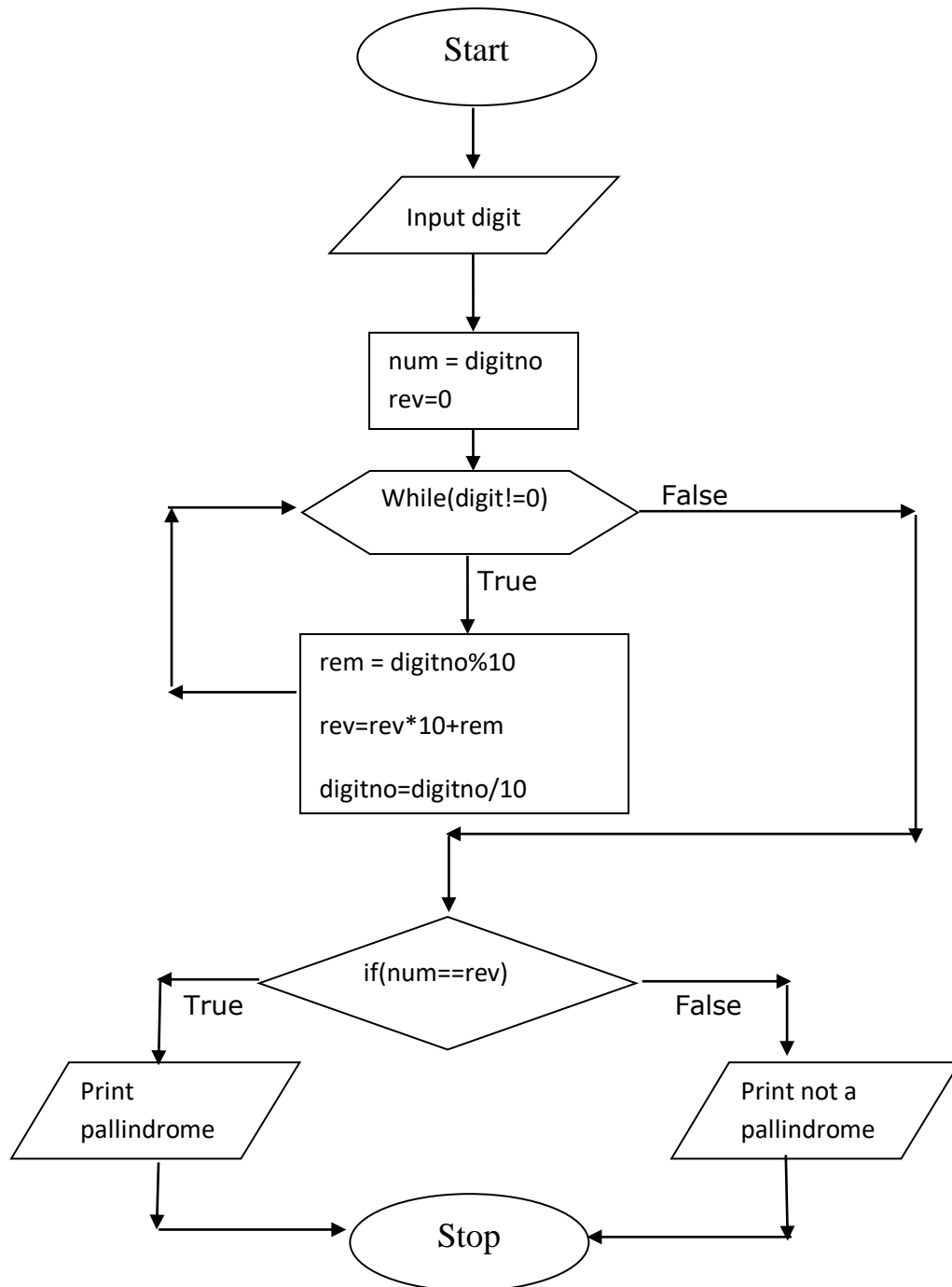
Step 8: Compute reverse = $reverse * 10 + remainder$.

Step 9: Compute $digitno = digitno / 10$. Goto Step 6.

Step 10: Output reverse.

Step 11: Check if reverse is equal to num. If true, output the message "The number is a palindrome". Otherwise, output the message "The number is not a palindrome".

Step 12: Stop.

Flow chart

Program 4

```
#include<stdio.h>

int main()
{
    int digitno, num, rev, rem;

    printf("Enter the integer number:\n");

    scanf ("%d", &digitno);

    num = digitno;

    rev=0;

    printf("The given integer number = %d\n",num);

    while(digitno != 0)
    {
        rem = digitno % 10;

        rev = rev * 10 + rem;

        digitno = digitno / 10;

    }

    printf("The reversed number is %d\n", rev);

    if(num == rev)

    printf("The given number %d is a palindrome\n", num);

    else

    printf("The given number %d is not a palindrome\n", num);

}
```

Output:**First Run:**

Enter the integer number

2442

The given integer number = 2442

The reversed number 2442

The given number 2442 is a palindrome

Second Run:

Enter the integer number

1234

The given integer number = 1234

The reversed number 4321

The given number 1234 is not a palindrome

Viva Questions:

- a. What are Looping control statements?
- b. Explain palindrome.
- c. What is the difference between while and for loops?
- d. What are the Entry controlled and Exit controlled loops in C?
- e. Write the flowchart for „while“ loop.
- f. What is the difference between while loop & do-while loop?

5) An electricity board charges the following rates for the use of electricity: for the first 200 units 80 paise per unit: for the next 100 units 90 paise per unit: beyond 300 units rupees 1 per unit. All users are charged a minimum of rupees 100 as meter charge. If the total amount is more than Rs 400, then an additional Surcharge of 15% of total amount is charged. Write a program to read the name of the user, number of units consumed and print out the charges.

Algorithm:

Input: Customer name and unit consumed

Output: Customer name, units consumed and total amount to be paid

Step 1: Start

Step 2: Read the name of customer and the unit consumed by the customer.

Step 3: Check if the unit consumed is greater than 1 and less than 200, if true goto step 4 else goto step 5.

Step 4: Compute: $\text{amt} = 100 + (0.8 * \text{units})$.

Step 5: if unit is greater than 200 and less than 300, if true goto step 6 else goto step 7

Step 6: Compute $\text{amt} = 100 + (200 * 0.8) + ((\text{units} - 200) * 0.9)$

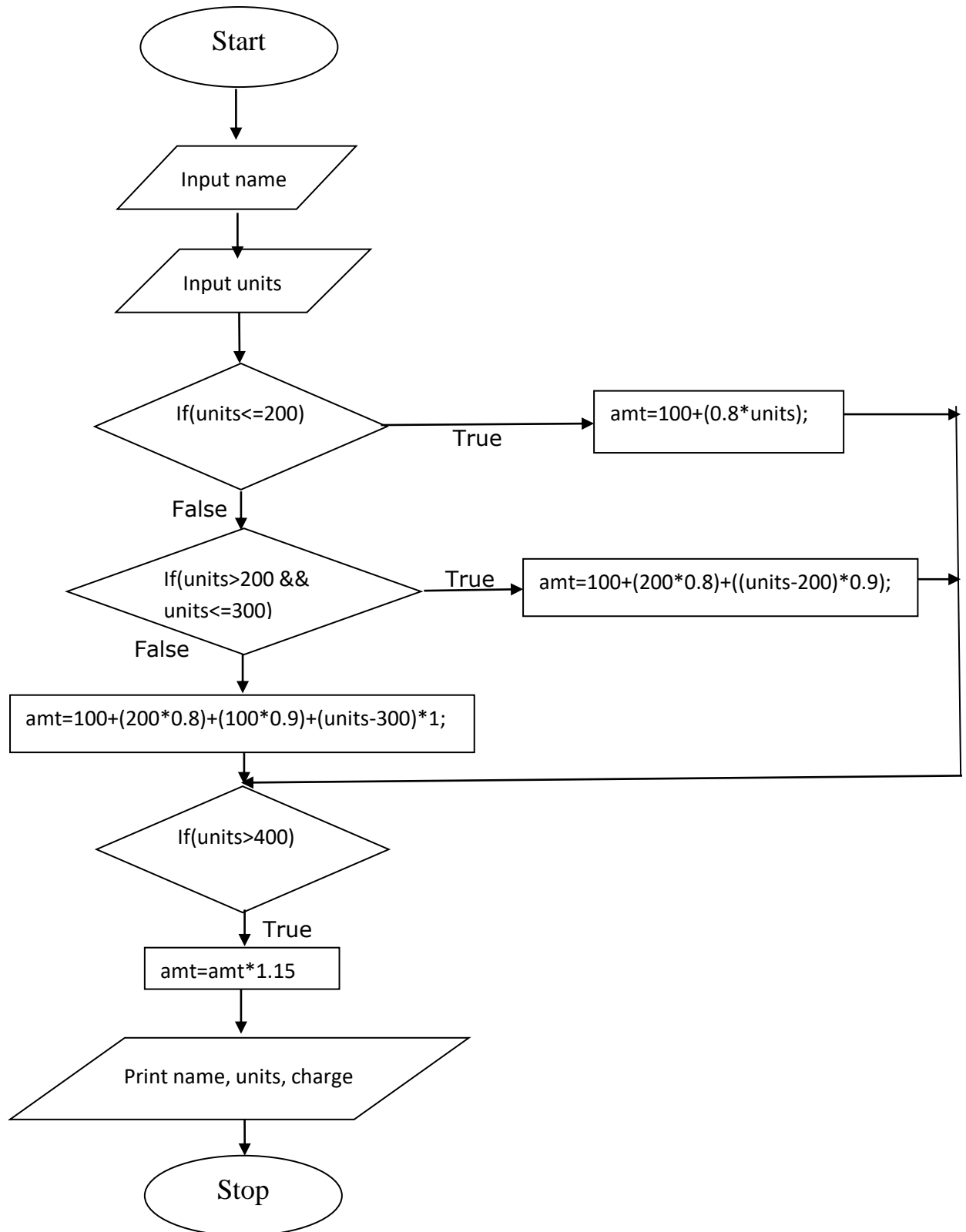
Step 7: Compute $\text{amt} = 100 + (200 * 0.8) + (100 * 0.9) + ((\text{units} - 300) * 1)$, then goto step 8

Step 8: Check if the amt is less than or equal to 400, if true goto step 9 otherwise goto step 10.

Step 9: Print the amount charged and goto step 11.

Step 10: compute $\text{amt} = \text{amt} * 1.15$, and print the amount charged.

Step 11: Stop .

Flow chart

Program 5

```
#include <stdio.h>

void main()
{
    char name[10];
    float unit, amt;
    printf("Enter your name and unit Consumed:");
    scanf("%s %f",name,&unit);
    if(unit<=200)
        amt=unit*0.80+100;
    else if((unit>200)&&(unit<=300))
        amt=200*0.80+((unit-200)*0.90)+100;
    else
        amt=200*0.80+100*0.90+((unit-300)*1)+100;
    if(amt>400)
        amt=1.15*amt;
    printf("Name: %s\n Unit=%f \n charge=%f ",name,unit,amt);
}
```

Output**First Run**

Enter your name and unit Consumed: Siri 52

Name: Siri

Unit=52

charge=141.600000

Second Run

Enter your name and unit Consumed: Rajesh 460

Name: Rajesh

Unit=460

charge=586.500000

Viva Questions:

- Difference between float and double data types.
- Write syntax of for loop?
- What is the use of break statement?
- Difference between continue and break statement?

6) Introduce 1D Array manipulation and implement binary search.**ALGORITHM: BINARY SEARCH**

Step 1: Start

Step 2: Read size of the array **n**

Step 3: Read the list of elements in sorted order **a[i]**.

Step 4: Read the key element in **key**

Step 5: initialize **low=0** and **high= n-1**

Step 6: Check if **low** is less than or equal to **high**. If condition is true, goto step 7, other wise goto Step 11

Step 7: find the middle element.i.e. **mid=(low+high)/2;**

Step 8: if key element is equal to mid element, then initialize loc value, **loc = mid+1;** otherwise goto step 9

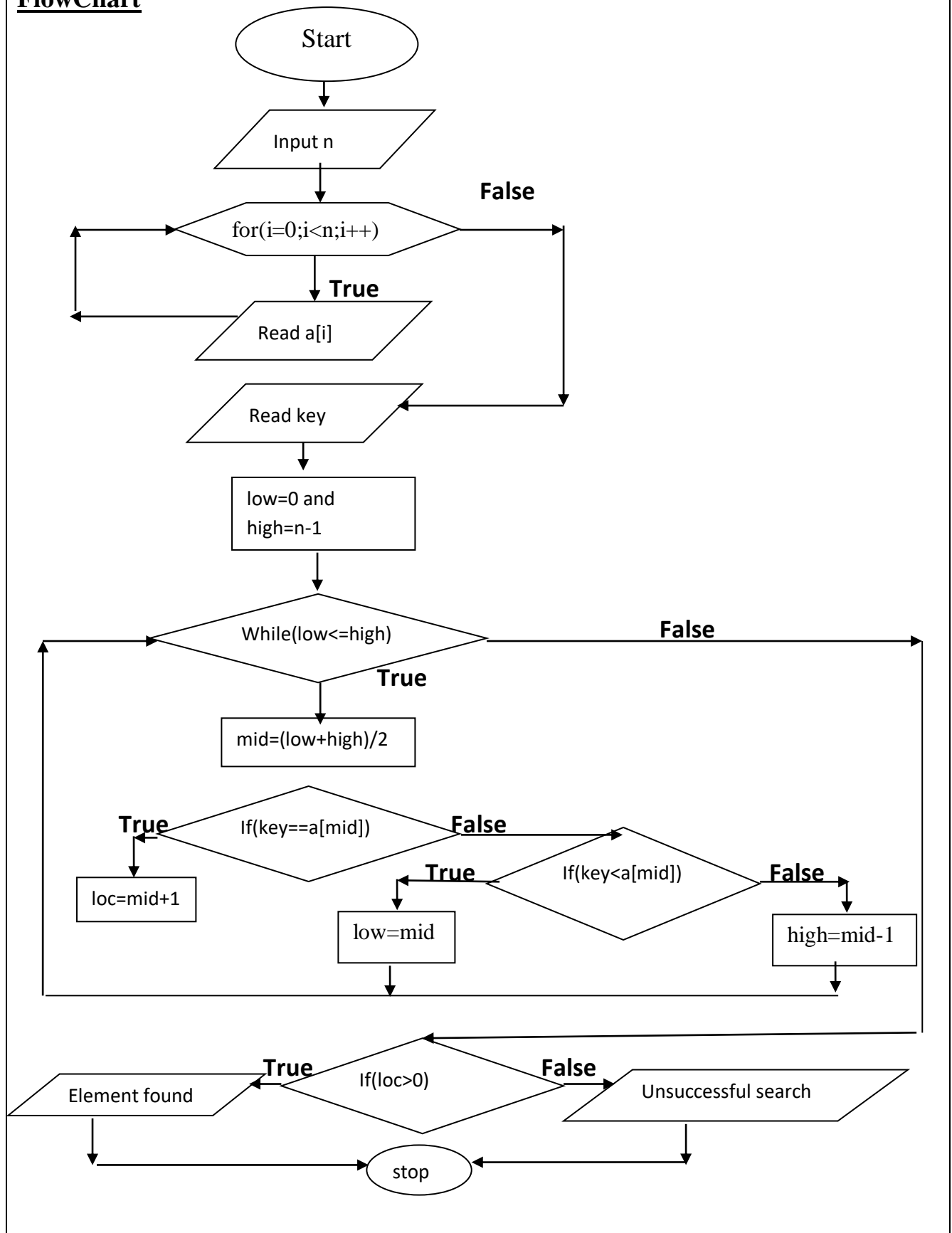
Step 9: Check if key element is less than mid element is true, then initialize **high=mid-1** then goto step 6, if it false goto step10.

Step 10: Check if key element is greater than mid element is true, then initialize **low=mid+1** then goto step 6.

Step 11: Check if **loc** value is greater then **zero** then print the search is successful then goto step 13, otherwise goto step 12.

Step 12: print search is unsuccessful, then goto step 13.

Step 13: Stop

FlowChart

Program 6

```
#include<stdio.h>

void main()
{
    int n, a[100], i, key, high, low, mid, loc=-1;
    printf("Enter the size of the array\n");
    scanf("%d",&n);
    printf("Enter the elements of array in sorted order\n");
    for(i=0;i<n;i++)
        scanf("%d",&a[i]);
    printf("Enter the key element to be searched\n");
    scanf("%d",&key);
    low=0;
    high=n-1;
    while(low<=high)
    {
        mid=(low+high)/2;
        if(key==a[mid])
        {
            loc = mid+1;
            break;
        }
        else
        {
            if(key<a[mid])
                high=mid-1;
            else
                low=mid+1;
        }
    }
    if(loc>0)
        printf("\n The element %d is found at %d ",key,loc);
    else
        printf("\nThe search is unsuccessful");
}
```

Output

First Run

Enter the size of the array

5

Enter the elements of array in sorted order

10

20

30

40

50

Enter the element to be searched

40

The element 40 is found at 4

Viva Questions:

- a. What is an array/definition of array.
- b. What are the types of array?
- c. What is a multidimensional array?
- d. How to declare and initialize one dimensional array?
- e. What are the advantages of an array?
- f. What is the difference between array & string?
- g. Write the syntax of declaring an array.

7) Implement using functions to check whether the given number is prime and display appropriate messages (No built in Math Function).

Algorithm:

Input: Any integer value

Output: The entered value is prime or not.

Step 1: Start

Step 2: Read value.

Step 3: By using for loop find the prime number between n1 and n2 using function.

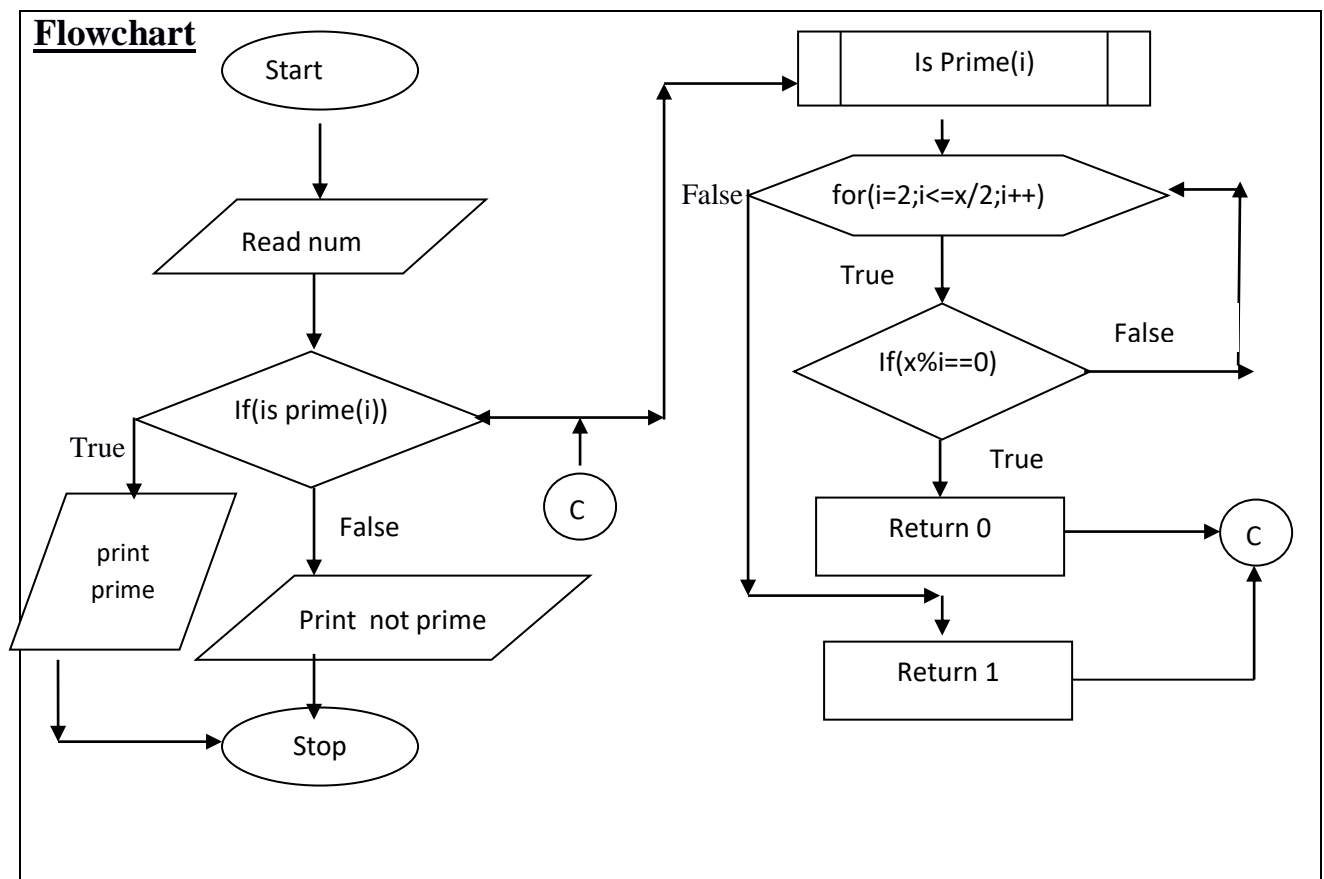
Step 4: Using for loop, check whether the number is divisible or not.

Step 5: If divisible, terminate loop and return 0. If not divisible, terminate loop and return 1.

Step 6: If 1, display number is prime. If 0, display number is not prime.

Step 7: Stop.

Flowchart



Program 7

```
#include<stdio.h>

int isprime(int);

int main()
{
    int num;
    printf("Enter value of num:\n");
    scanf("%d",&num);
    if (isprime(num))
        printf("%d is a prime number",num);
    else
        Printf("%d is not a prime number",num)
}

int isprime(int x)
{
    int i;
    for(i=2;i<=x/2;i++)
        if(x%i == 0)
            return 0;

    return 1;
}
```

Output:**Run1**

```
Enter value of num:
17
17 is a prime number
```

Run1

```
Enter value of num:
10
10 is not a prime number
```

Viva Questions:

- What is prime number?
- Give examples for prime no?
- What are keywords?
- What are conditional statements?

Part B

8) Develop a program to introduce 2D array manipulation and implement matrix multiplication and ensure the rules of multiplication are checked.

ALGORITHM: MATRIX MULTIPLICATION

Step 1: Start

Step 2: Read order of matrix A i.e., m and n.

Step 3: Read order of matrix B i.e., p and q.

Step 4: check if $(n \neq p)$, then display matrix multiplication is not possible goto step 9, otherwise
goto step 5

Step 5: Read elements of matrix A

Step 6: Read elements of matrix B

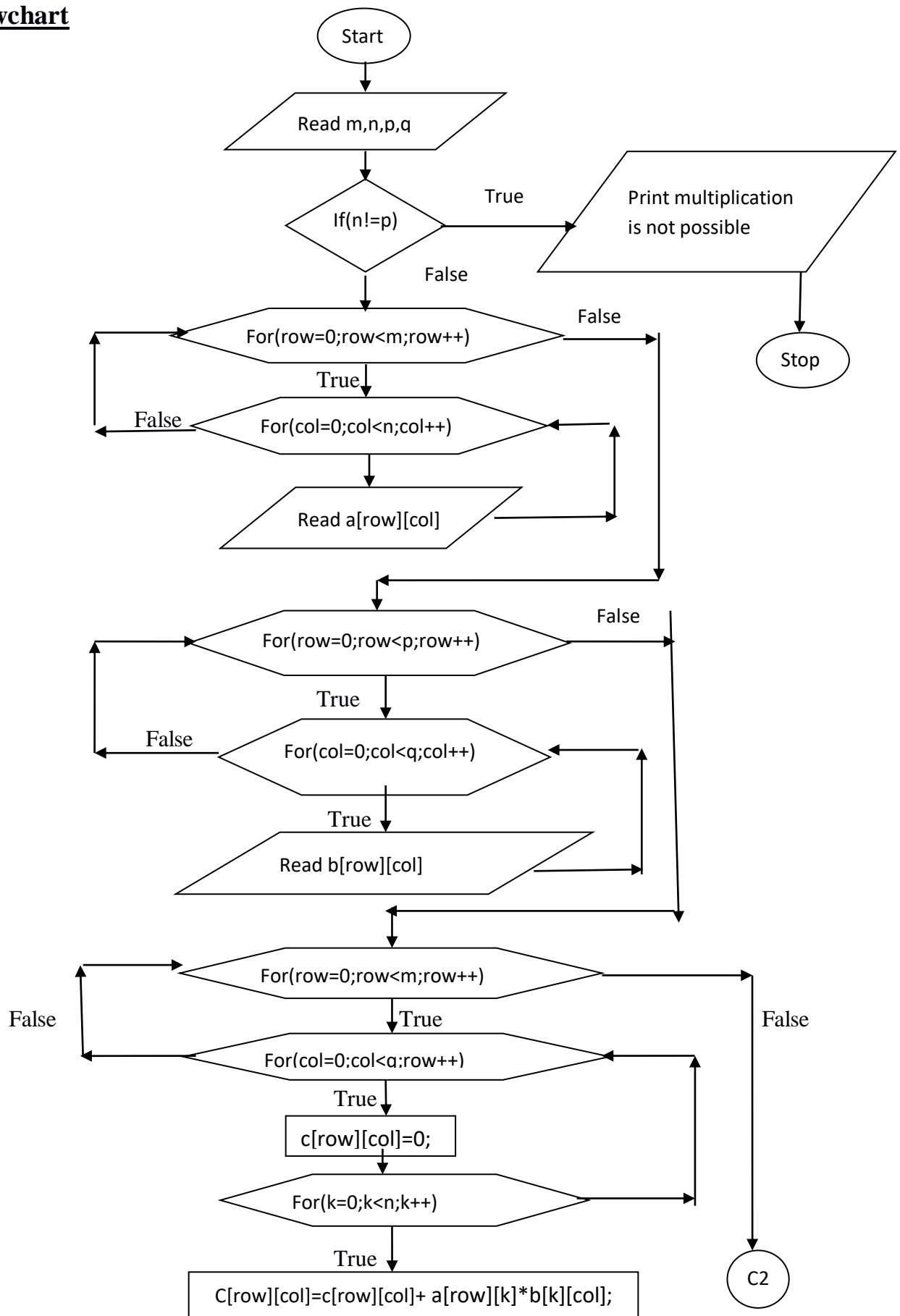
Step 7: Find $A \times B$.by using $c[\text{row}][\text{col}] = c[\text{row}][\text{col}] + a[\text{row}][k] * b[k][\text{col}];$

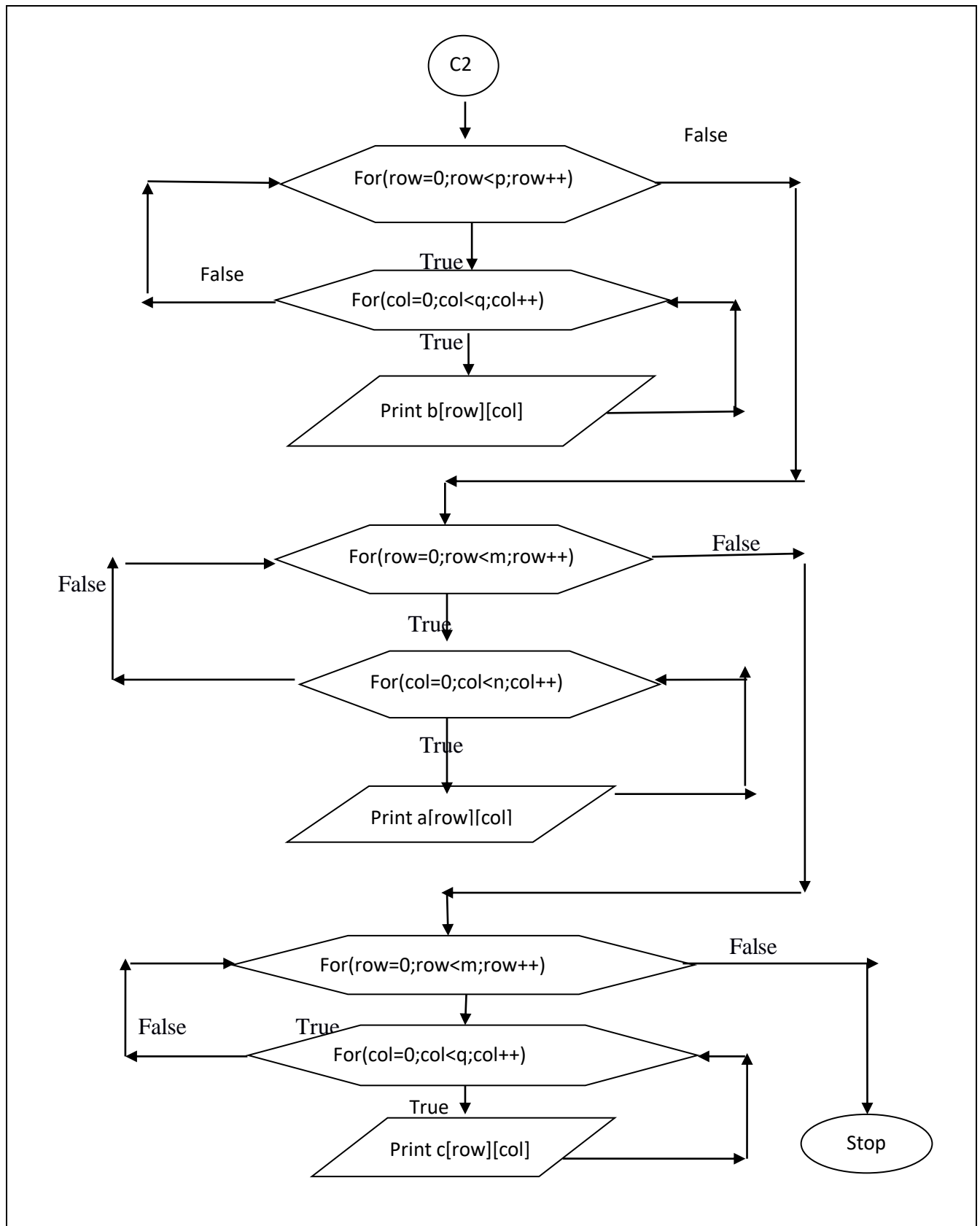
Step 8: Display matrix A

Step 9: Display matrix B

Step 10: display matrix C, i.e $A \times B$.

Step 11:: Stop.

Flowchart



Program 8

```
#include<stdio.h>
#include<stdlib.h>

int main( )
{
    int m,n,p,q,row,col,k,a[3][3],b[3][3],c[3][3];
    printf("Enter the order of matrix A\n");
    scanf("%d%d",&m,&n);
    printf("enter order of matrix B\n");
    scanf("%d%d",&p,&q);
    if(n!=p)
    {
        printf("Matrix Multiplication is not possible\n");
        exit(0);
    }
    printf("Enter the elements into matrix A\n");
    for(row=0;row<m;row++)
    {
        for(col=0;col<n;col++)
        {
            scanf("%d",&a[row][col]);
        }
    }
    printf("Enter the elements into matrix B\n");
    for(row=0;row<p;row++)
    {
        for(col=0;col<q ;col++)
        {
            scanf("%d",&b[row][col]);
        }
    }
    for(row=0;row<m;row++)
    {
        for(col=0;col<q;col++)
        {
            c[row][col]=0;
```

```
        for(k=0;k<n;k++)
        {
            c[row][col]= c[row][col]+a[row][k]*b[k][col];
        }
    }
}
printf("The elements of matrix A are\n");
for(row=0;row<m;row++)
{
    for(col=0;col<n;col++)
    {
        printf("%3d",a[row][col]);
    }
    printf("\n");
}
printf("The elements of matrix B are\n");
for(row=0;row<p;row++)
{
    for(col=0;col<q;col++)
    {
        printf("%3d",b[row][col]);
    }
    printf("\n");
}
printf("Product of Matrix A and B is\n");
for(row=0;row<m;row++)
{
    for(col=0;col<q;col++)
    {
        printf("%3d",c[row][col]);
    }
    printf("\n");
}
}
```

Output:

Enter the order of matrix A

2 2

Enter the order of matrix B

2 2

Enter the elements into matrix A

1 2 3 4

Enter the elements into matrix B

5 6 7 8

Elements of matrix A are

1 2

3 4

Elements of matrix B are

5 6

7 8

Product of matrix A and B is

19 22

43 50

Viva Questions:

- How to initialize two dimensional arrays?
- How to pass a two dimensional array as function parameter?
- How the memory is allocated for two dimensional array
- Write the program to add and subtract two matrices.
- Program to find the transpose of a matrix.
- Program to find determinants of a matrix.
- Program to find the diagonal elements of a matrix.

9) Develop a Program to compute Sin(x) using Taylor series approximation .Compare your result with the built- in Library function. Print both the results with appropriate messages.

ALGORITHM: TAYLOR SERIES

Step 1: Start

Step 2: Read angle x

Step 3: Assign $y=x$

Step 4: Compute the value of x in radians:

$$x = (3.1412/180.0) * \text{degree}$$

Step 5: Initialise:

- i. $\text{sum} = x$
- ii. $t = x$
- iii. $i = 1$

Step 6: Compute:

- i. $i = i + 2$
- ii. $t = (-t * x * x) / ((i-1) * i)$
- iii. $\text{Sum} = \text{sum} + t$

Step 7: Check if the absolute value term is greater than 0.00005. If true goto Step 6 else goto

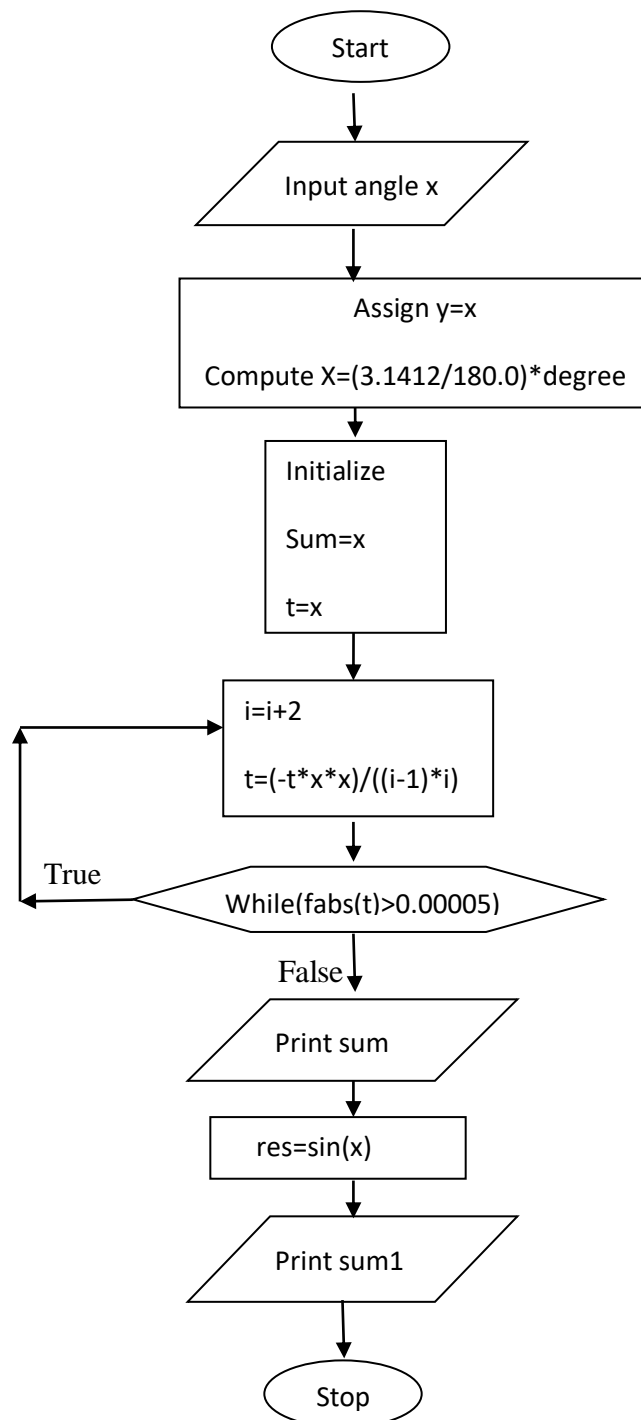
Step 8

Step 8 : Display sine using iteration : sum

Step 9 : Assign $\text{res}=\sin(x)$;

Step 10 : Display sine using library function: $\sin(x)$

Step 11: Stop

Flowchart

Program 9

```
#include<stdio.h>
#include<math.h>

int main( )
{
    int i;
    float x,t,sum,sum1,y;
    printf("Enter the angle\n");
    scanf("%f",&x);

    y=x;
    x=3.1428*(x/180.0);
    sum=x;
    t=x;
    i=1;
    do
    {
        i=i+2;
        t=(-t*x*x)/((i-1)*i);
        sum=sum+t;
    }while(fabs(t)>0.00005);
    printf("sin(%f) using taylor series=%f\n",y,sum);
    sum1=sin(x);
    printf("Using inbuilt function sin(%f)=%f",y,sum1);
}
```

Output:

Enter the angle 30

sin(30.000000) using taylor series=0.500000

Using inbuilt function sin(30.000000)=0.500000

Viva Questions:

- What is pre-processor directive?
- What is difference between `const` and `#define`.
- What is use of `fabs()`.
- What is variable initialization and why is it important?
- What is the difference between the `=` symbol and `==` symbol?
- Can the curly brackets `{ }` be used to enclose a single line of code?

10) Write functions to implement string operations such as compare, concatenate, string length. Convince the parameter passing techniques.

ALGORITHM: STRING OPERATIONS

Step 1: Start

Step 2: press 1-compare 2-concatenate 3-length of string, if press 1 goto step 3, if press 2 goto step 4, if press 3 goto step 5,

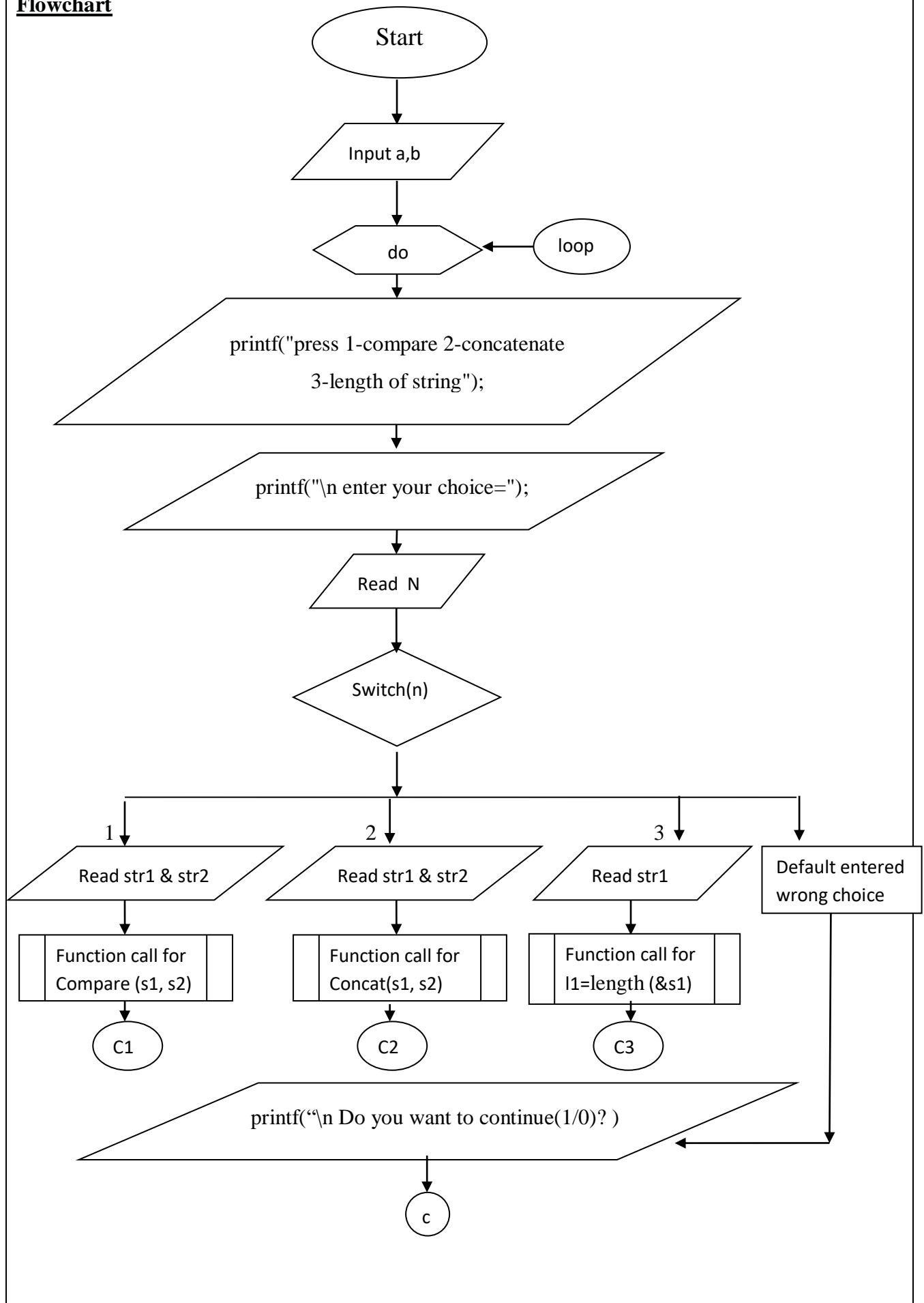
Step 3: Read String1 and String 2 and Comparison function by using strcmp built in function is used. if res=0 then print both strings are equal, else print strings are not equal and goto step 4.

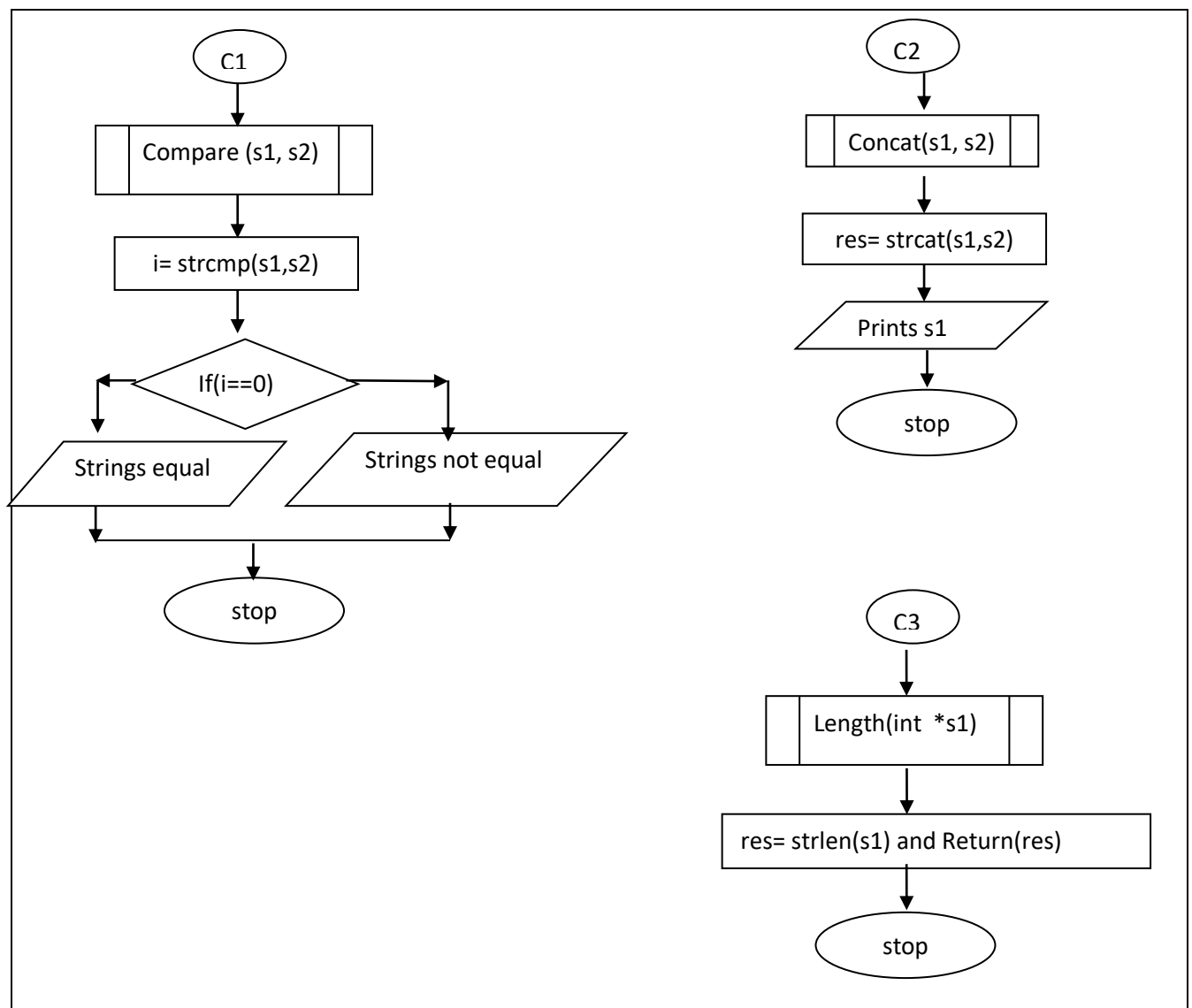
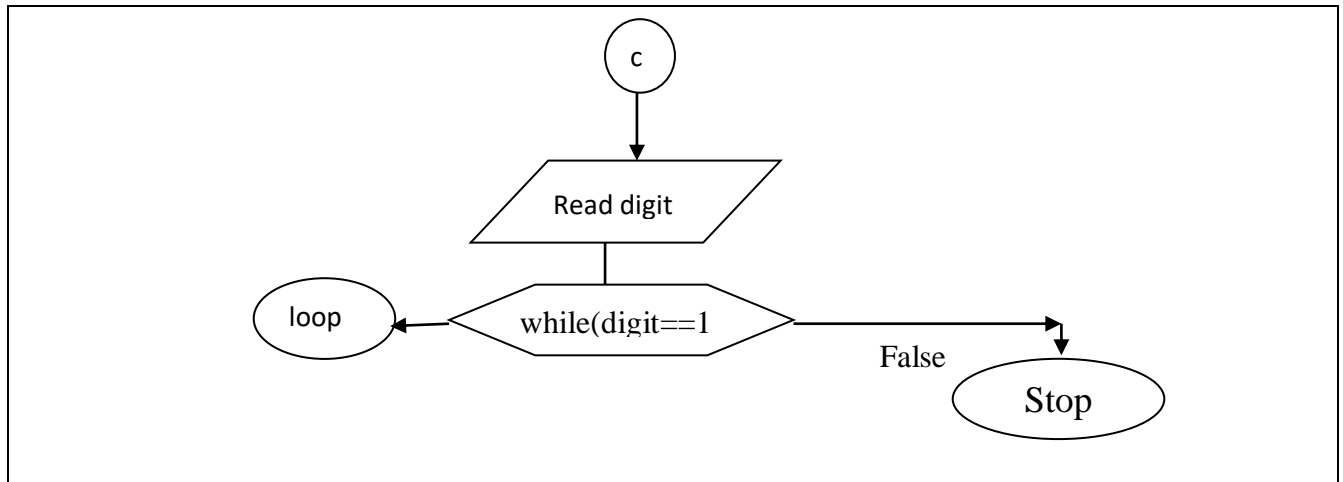
Step 4: Read String1 and String 2 and find concatenation of two strings using string handling function strcat() and display string and return back to main function.

Step 5: Read String1 and call function to find the length of string by calling function length(*string)

Step 6: if digit=1 then goto step 2 otherwise goto step 7

Step 7: stop.

Flowchart



Program 10

```
#include<stdio.h>
#include<string.h>
void compare(char [ ],char [ ]);
void concat(char [ ],char [ ]);
void length(char *[ ]);
void main( )
{
    int n,digit;
    char str1[10],str2[10];
    do
    {
        printf("press 1-compare 2-concatenate 3-length of string");
        printf("\n enter your choice=");
        scanf("%d",&n);
        switch(n)
        {
            case 1:printf("enter first string=");
                    scanf("%s",str1);
                    printf("enter second string=");
                    scanf("%s",str2);
                    compare(str1,str2);
                    break;
            case 2: printf("enter first string=");
                    scanf("%s",str1);
                    printf("enter second string=");
                    scanf("%s",str2);
                    concat(str1,str2);
                    break;
            case 3:printf("enter string=");
                    scanf("%s",str1);
                    length(&str1);
                    break;
            default: printf("wrong choice");
                    break;
        }
    }
}
```

```
        }

        printf("\n Do you want to continue(1/0)? ");
        scanf("%d", &digit);
    }while(digit==1);
}

void compare(char str1[ ],char str2[ ])
{
    int i;
    i=strcmp(str1,str2);
    if(i==0)
        printf("strings are equal\n ");
    else
        printf("string are not equal\n");
}

void concat(char str1[ ],char str2[ ])
{
    strcat(str1,str2);
    printf("concatenate string=%s",str1);
}

void length(char *str1[ ])
{
    int len;
    len=strlen(str1);
    printf("the length of string=%d",len);
}
```

Output**press 1-compare 2-concatenate 3-length of string**

enter your choice=1

enter first string=ram

enter second string=Ram

string are not equal

Do you want to continue(1/0)? 1

press 1-compare 2-concatenate 3-length of string

enter your choice=2

enter first string=RAM

enter second string=kumar

concatenate string=RAMkumar

Do you want to continue(1/0)? 1

press 1-compare 2-concatenate 3-length of string

enter your choice=3

enter string=ram **the length of string=3**

Do you want to continue(1/0)? 1

press 1-compare 2-concatenate 3-length of string

enter your choice=4

wrong choice

Do you want to continue(1/0)? 0

Viva-Voce Question

- What is string?
- How to declare string?
- What are the string manipulation function?
- What is gets() and puts() function in string?

11) Develop a program to sort the given set of N numbers using Bubble sort.

ALGORITHM: BUBBLE SORT

Input: A unsorted array of n numbers

Output: sorted array

Step 1: START

Step 2: Read unsorted array n elements in to a[i], where i is the index value.

Step 3: Initialize index i=0.

Step 4: Check if i is less than n. if true, goto step 5. Otherwise goto step output array.

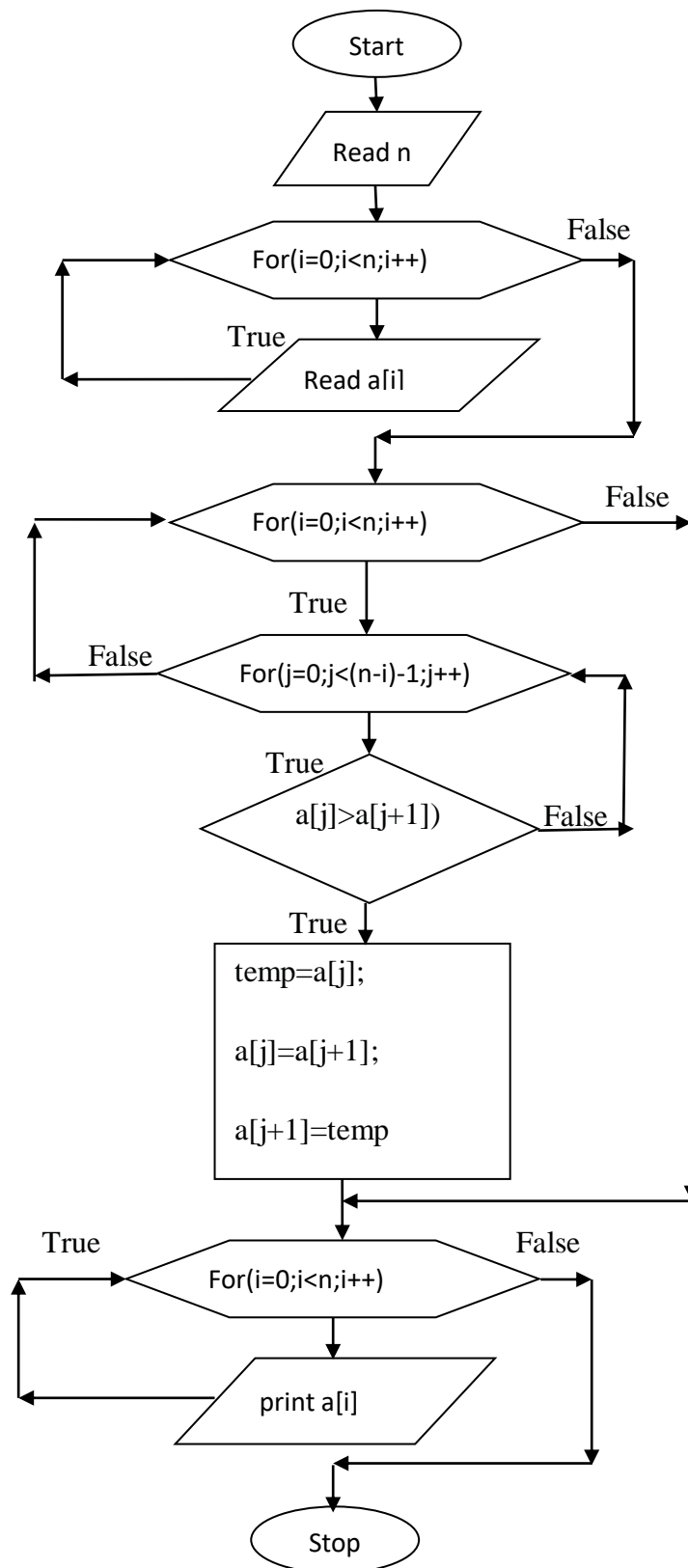
Step 5: initialize index j to zero.

Step 6: check if j is less than (n-i-1). If true goto step 7. Otherwise increment j and goto step 4.

Step 7: inside the for loop check if a[j] is greater than a[j+1](i.e., adjacent elements are compared). If true swap elements using temporary variables. Otherwise goto step 6.

Step 8: Output the sorted array elements using for loop.

Step 9: Stop

Flowchart

Program 11

```
#include<stdio.h>

int main()
{
    int i,j,n,temp;
    int a[20];
    printf("enter the value of n");
    scanf("%d",&n);
    printf("Enter the numbers in unsorted order:\n");
    for(i=0;i<n;i++)
        scanf("%d", &a[i]);
    // bubble sort logic
    for(i=0;i<n;i++)
    {
        for(j=0;j<(n-i)-1;j++)
        {
            if( a[j]>a[j+1])
            {
                temp=a[j];
                a[j]=a[j+1];
                a[j+1]=temp;
            }
        }
    }
    printf("The sorted array is\n");
    for(i=0;i<n;i++)
    {
        printf("%d\n",a[i]);
    }
}
```

Output

Enter the value of n

5

Enter the numbers one by one in unsorted order:

20

10

30

50

40

The sorted array is

10

20

30

40

50

Viva Questions:

- a. Why the name bubble sort?
- b. What are the different types of sorting techniques?
- c. Explain the logic of bubble sort with an example.
- d. What is nested for loop?

12) Develop a program to find the square root of a given number N and execute for all possible inputs with appropriate messages. Note: Don't use library function sqrt(n).

ALGORITHM: SQUARE ROOT

Step 1: START

Step 2: Read the number n

Step 3: Check if the number is less than zero. If true goto Step 4 else goto Step 5

Step 4: Display error and goto step 7

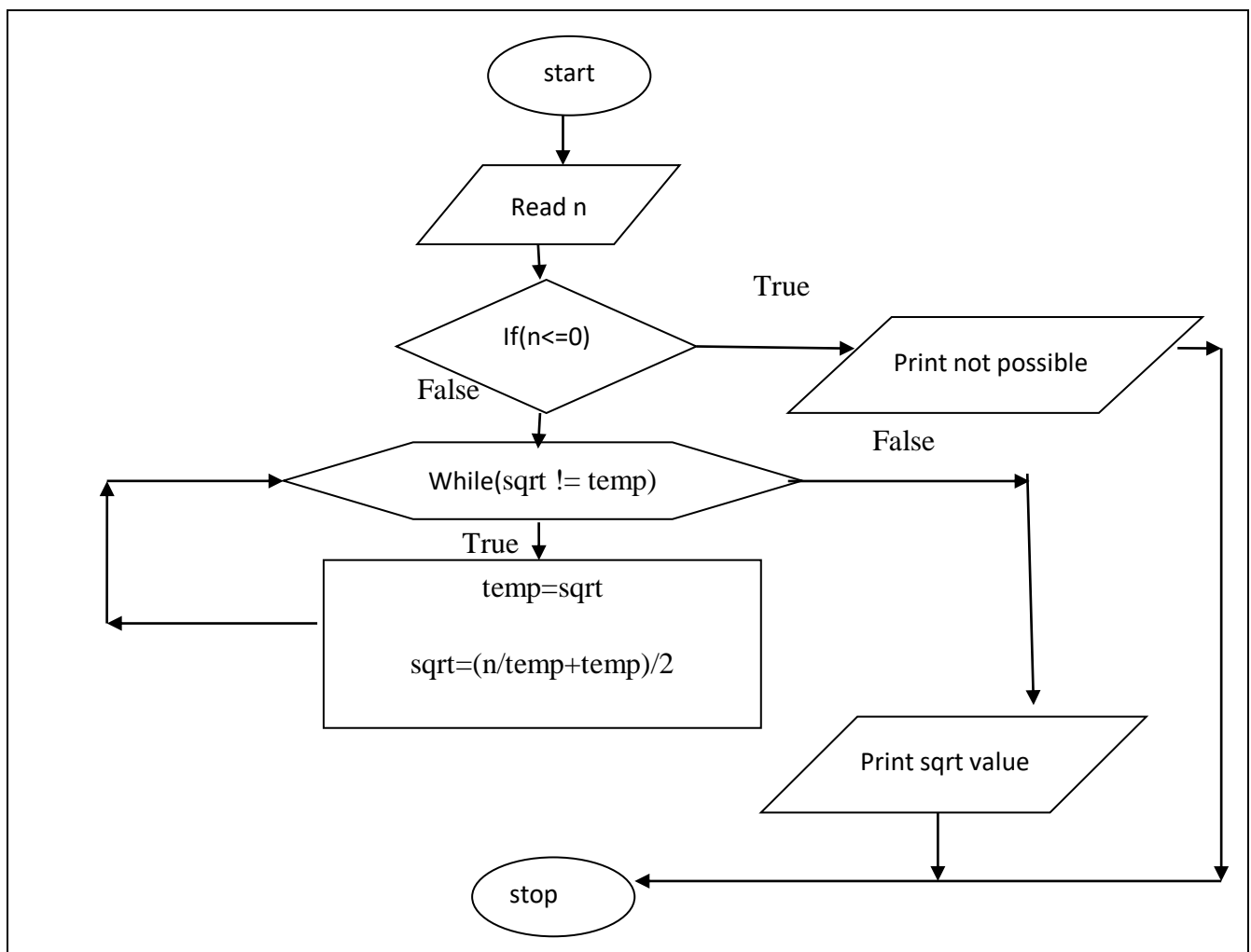
Step 5: In while loop check (sqrt != temp) if true execute, else goto step 6

a. Temp=sqrt

b. Sqrt=(n/temp+temp)/2

Step 6: Display the square root of n

Step 7: STOP



Program 12

```
#include<stdio.h>
#include<math.h>
#include<stdlib.h>
int main()
{
    int n;
    float sqrt, temp;
    printf("Enter a value whose square root has to be calculated\n");
    scanf("%d",&n);
    if(n<=0)
    {
        printf("Sqrt can't be determined");
        exit(0);
    }
    sqrt = n / 2;
    temp = 0;
    while(sqrt != temp)
    {
        temp = sqrt;
        sqrt = ( n/temp + temp) / 2;
    }
    printf("The square root of %d is %f", n, sqrt);
}
```

Output

Run1

```
Enter a value whose square root has to be calculated
9
The Square root of 9.000000=3.0000
```

Viva Questions:

1. Difference between float and double data types.
2. Write syntax of for loop?
3. What is the use of break statement?

13) Implement structures to read, write, compute average- marks and the students scoring above and below the average marks for a class of N students.

ALGORITHM: STUDENT DETAILS

Input: Student details such as student id, name and marks

Output: To print the details of those students scoring above and below average

Step 1: START

Step 2: Read the number of students

Step 3: For each student, read the student id, name and marks for all subjects.

Step 4: Calculate the average of the marks and store in the avg field.

Step 5: Print results.

Step 6: Initialise loop

Step 7: Read the average of each student

Step 8: Check if $\text{avg} > 35.00$

Step 9: If yes print result else go to next iteration

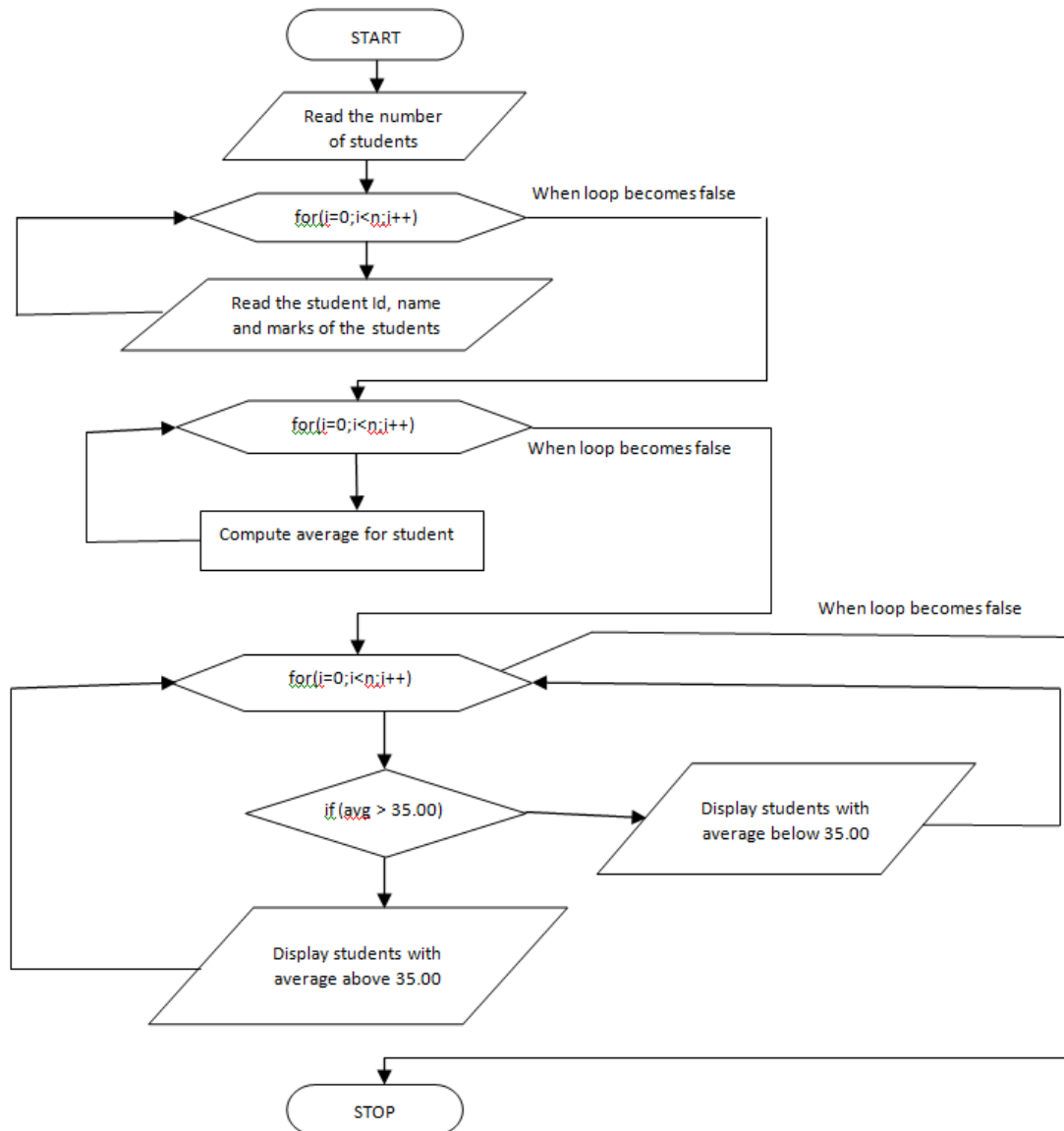
Step 10: Initialise loop

Step 11: Read average of each student

Step 12: Check if $\text{avg} < 35.00$

Step 13: If yes print result else go to next iteration

Step 14: STOP

Flowchart

Program 13

```
#include<stdio.h>

struct student
{
    char usn[10];
    char name[10];
    float m1,m2,m3;
    float avg,total;
};

void main()
{
    struct student s[20];
    int n,i;
    float tavg,sum=0.0;
    printf("Enter the number of student=");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        printf("Enter the detail of %d students\n",i+1);
        printf("\n Enter USN=");
        scanf("%s",s[i].usn);
        printf("\n Enter Name=");
        scanf("%s",s[i].name);
        printf("Enter the three subject score\n");
        scanf("%f%f%f",&s[i].m1,&s[i].m2,&s[i].m3);
        s[i].total=s[i].m1+s[i].m2+s[i].m3;
        s[i].avg=s[i].total/3;
    }
    for(i=0;i<n;i++)
    {
        if(s[i].avg>=35)
            printf("\n %s has scored above the average marks",s[i].name);
        else
            printf("\n %s has scored below the average marks",s[i].name);
    }
}
```

Output

Enter the number of student=2

Enter the detail of 1 students

Enter USN=101

Enter Name=Ram

Enter the three subject score 10 21 15

Enter the detail of 2 students

Enter USN=102

Enter Name=Kumar

Enter the three subject score 11 9 10

Ram has scored above the average marks

Kumar has scored below the average marks

Viva-Voce Questions

- a. What is structure?
- b. How to declare a structure?
- c. What is structure member?
- d. What is difference between array and structure?
- e. What is nested structure?
- f. What is typedef?

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

ALGORITHM: SUM, MEAN, VARIANCE

Input: read numbers to compute sum, mean, variance and deviation

Output: results sum, mean, variance and deviation

Step1: START

Step2: Read n

Step3: for each value of n read x

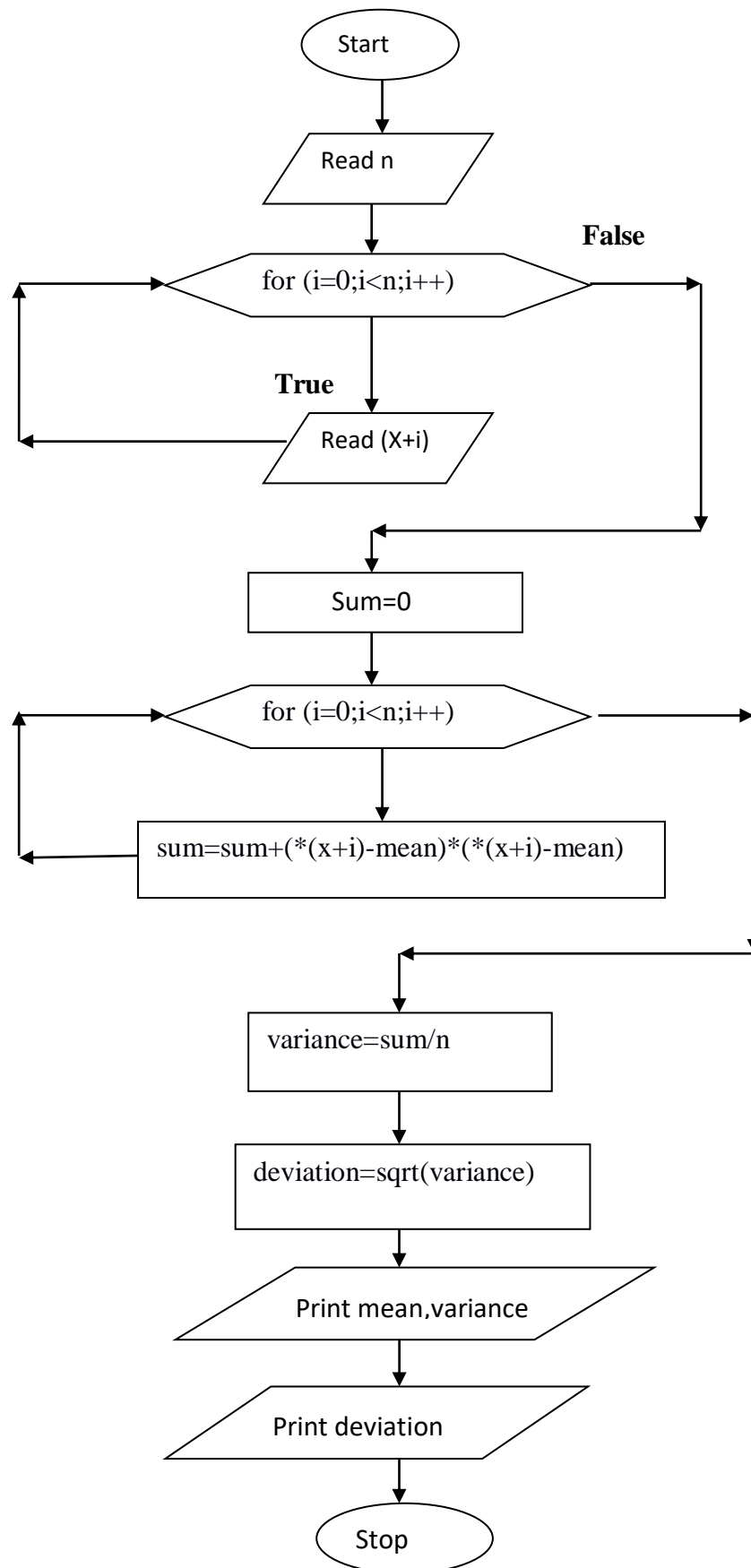
Step4: initialize sum=0, i=0

Step5: for each value of n and i, Compute sum using $\text{sum} = \text{sum} + (x + i) - \text{mean}$

Step 6: using sum value compute $\text{variance} = \text{sum} / n$ and $\text{deviation} = \sqrt{\text{variance}}$

Step 7: display mean, variance, deviation

Step 8: Stop



Program 14

```
#include<stdio.h>
#include<math.h>
int main()
{
    int n , i;
    float x[20],sum,mean;
    float variance , deviation;
    printf("Enter the value of n \n");
    scanf("%d",&n);
    printf("enter %d real values \n",n);
    for (i=0;i<n;i++)
    {
        scanf("%f",(x+i));
    }
    sum=0;
    for(i=0;i<n;i++)
    {
        sum= sum+*(x+i);
    }
    printf("sum=%f\n",sum);
    mean=sum/n;
    sum=0;
    for(i=0;i<n;i++)
    {
        sum=sum+(*(x+i)-mean)*(*(x+i)-mean);
    }
    variance=sum/n;
    deviation=sqrt(variance);
    printf("mean(Average)=%f\n",mean);
    printf("variance=%f\n",variance);
    printf("standard deviation=%f\n",deviation);
}
```

Output:

Entre the value of n

5

Enter 5 real values

3

7

23

1

4

sum=38.000000

mean(Average)=7.600000

variance=63.039997

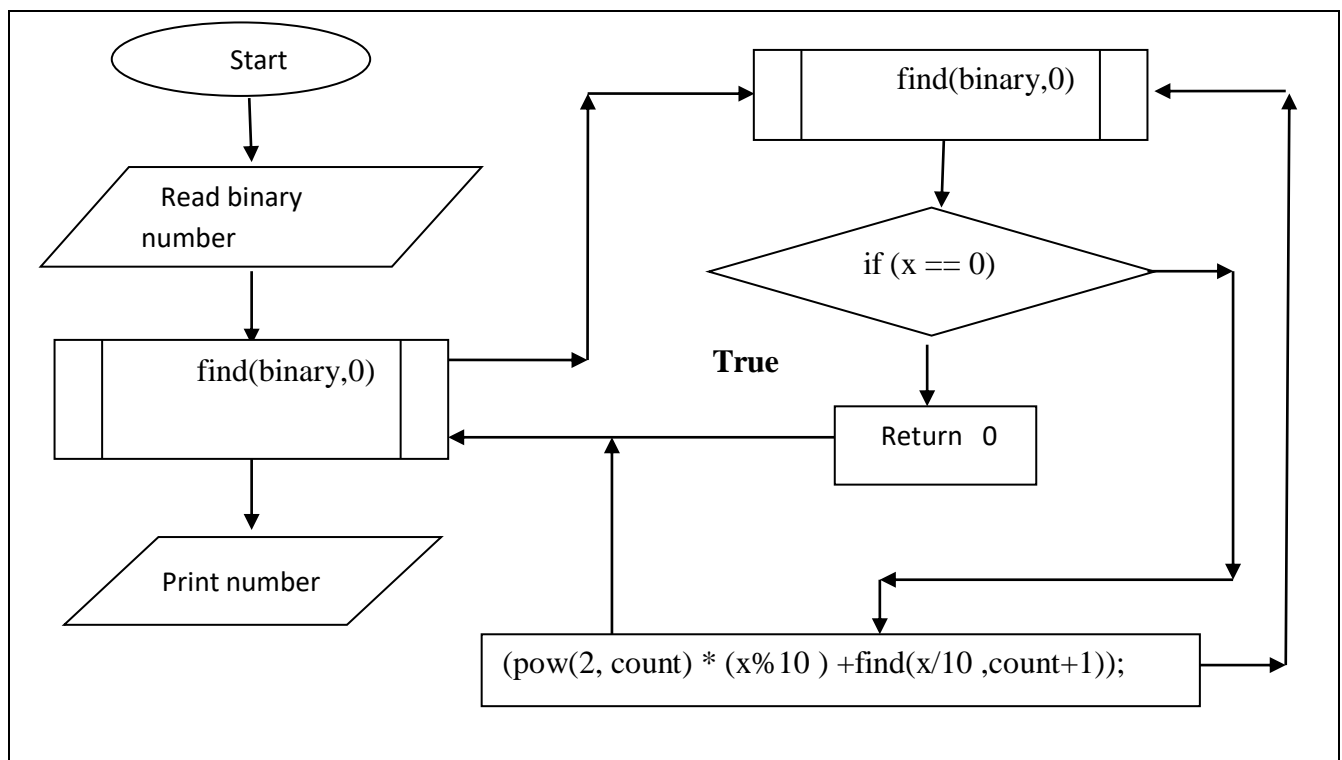
standard deviation=7.939773

Viva Questions:

- a. Define pointer
- b. Define array of pointer
- c. Difference between (x+i) and *(x+i)
- e. Define array

Program 15: Write a Program to Convert a Binary Number into a Decimal Number.**ALGORITHM****Input:** Binary Number**Output:** To Convert binary number to decimal number.**Step 1:** START**Step 2:** Read binary number value**Step 3:** Call the function find by passing arguments.**Step 4:** if number entered is zero then return 0. Otherwise compute using formula

$$(\text{pow}(2, \text{count}) * (\text{x} \% 10) + \text{find}(\text{x}/10, \text{count}+1))$$

Step 5: Repeat step 4 till recursive condition find becomes false**Step 6:** return result of converted number.**Step 7:** STOP

Program 15

```
#include <stdio.h>
#include <math.h>
int find(int x, int count)
{
    if (x == 0)
        return 0;
    else
        return(pow(2, count) * (x%10 ) + find(x/10 ,count+1));
}
void main()
{
    int binary;
    printf("Enter a Binary number");
    scanf ("%d",&binary);
    printf(" The decimal value for binary %d is",binary);
    printf("%d", find(binary,0));
}
```

Output: Enter a Binary number : 1010

The decimal value for binary 1010 is 10

Viva Questions:

- What is the binary?
- What is meant by a binary number system?
- How do you construct an increment statement or decrement statement in C?
- What is variable initialization and why is it important?
- What is the difference between the = symbol and == symbol?

DO'S AND DON'TS**Do's**

1. Do wear ID card and follow dress code.
2. Do log off the computers when you finish.
3. Do ask the staff for assistance if you need help.
4. Do keep your voice low when speaking to others in the LAB.
5. Do ask for assistance in downloading any software.
6. Do make suggestions as to how we can improve the LAB.
7. In case of any hardware related problem, ask LAB in charge for solution.
8. If you are the last one leaving the LAB, make sure that the staff in charge of the LAB is informed to close the LAB.
9. Be on time to LAB sessions.
10. Do keep the LAB as clean as possible.

Don'ts

1. Do not use mobile phone inside the lab.
2. Don't do anything that can make the LAB dirty (like eating, throwing waste papers etc).
3. Do not carry any external devices without permission.
4. Don't move the chairs of the LAB.
5. Don't interchange any part of one computer with another.
6. Don't leave the computers of the LAB turned on while leaving the LAB.
7. Do not install or download any software or modify or delete any system files on any lab computers.
8. Do not damage, remove, or disconnect any labels, parts, cables, or equipment.
9. Don't attempt to bypass the computer security system.
10. Do not read or modify other user's file.
11. If you leave the lab, do not leave your personal belongings unattended. We are not responsible for any theft.