

**CHANNABASVESHWARA INTSITUTE OF TECHNOLOGY,
GUBBI**

**COMPUTER PROGRAMMING LABORATORY
MANUAL (15CPL16/15CPL26)**



Channabasaveshwara Institute of Technology

(An ISO 9001:2008 certified Institution)

NH 206 (B.H. Road), Gubbi, Tumkur – 572 216. Karnataka



Department of Computer Science and Engineering

**COMPUTER PROGRAMMING LABORATORY
15CPL16/15CPL26**

B.E I/II - SEMESTER

LAB MANUAL 2016-17

NAME:

USN:

BRANCH: SECTION:

SYLLABUS

COMPUTER PROGRAMMING LABORATORY

Subject Code: 15CPL16 / 15CPL26

Hrs/Week: 03

Total Hrs: 48

I A Marks: 20

Exam Hours: 03

Exam Marks: 80

Demonstration of Personal Computer and its Accessories.

Demonstration and Explanation on Disassembly and Assembly of a Personal Computer by the faculty-in-charge. Students have to prepare a write-up on the same and include it in the Lab record and evaluated.

Laboratory Session-1: Write-up on Functional block diagram of Computer, CPU, Buses, Mother Board, Chip sets, Operating System & types of OS, Basics of Networking & Topology and NIC.

Laboratory Session-2: Write-up on RAM, SDRAM, FLASH memory, Hard disks, Optical media, CD-ROM/R/RW, DVDs, Flash drives, Keyboard, Mouse, Printers and Plotters. Introduction to flowchart, algorithm and pseudo code.

Note: These TWO Laboratory sessions are used to fill the gap between theory classes and practical sessions. Both sessions are evaluated as lab experiments.

Laboratory Experiments

Implement the programs with WINDOWS / LINUX platform using appropriate C compiler.

1. Design and develop a flowchart or an algorithm that takes three coefficients (**a**, **b**, and **c**) of a Quadratic equation ($ax^2+bx+c=0$) as input and compute all possible roots. Implement a C program for the developed flowchart/algorithm and execute the same to output the possible roots for a given set of coefficients with appropriate messages.

2. Design and develop an algorithm to find the **reverse** of an integer number **NUM** and check whether it is PALINDROME or NOT. Implement a C program for the developed algorithm that takes an integer number as input and output the reverse of the same with suitable messages. Ex: Num: **2014**, Reverse: **4102**, Not a Palindrome.

3a. Design and develop a flowchart to find the square root of a given number **N**. Implement a C program for the same and execute for all possible inputs with appropriate messages. Note: **Don't use library function sqrt(n).**

3b. Design and develop a C program to read a **year** as an input and find whether it is **leap year** or not. Also consider end of the centuries.

4. Design and develop an algorithm for evaluating the polynomial $f(x) = a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0$, for a given value of **x** and its coefficients using Horner's method. Implement a C program for the same and execute the program for different sets of values of coefficients and **x**.

5. Draw the flowchart and Write C Program to compute **Sin(x)** using Taylor series approximation given by $\sin(x) = x - (x^3/3!) + (x^5/5!) - (x^7/7!) + \dots$. Compare the result with the built-in Library function and print both the results with appropriate messages.

6. Develop an algorithm, implement and execute a C program that reads **N** integer numbers and arrange them in ascending order using **Bubble Sort**.

7. Develop, implement and execute a C program that reads two matrices **A** (**m x n**) and **B** (**p x q**) and Compute the product **A** and **B**. Read matrix **A** and matrix **B** in row major order and in column major order respectively. Print both the input matrices and resultant matrix with suitable headings and output should be in matrix format only. Program must check the compatibility of orders of the matrices for multiplication. Report appropriate message in case of incompatibility.

8. Develop, implement and execute a C program to search a Name in a list of names using **Binary searching** Technique.

9. Write and 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.
- Reads a *sentence* and prints frequency of each of the vowels and total count of consonants.

10a. 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.

10b. Design and develop a C function **isprime**(num) that accepts an integer argument and returns 1 if the argument is prime, a 0 otherwise. Write a C program that invokes this function to generate prime numbers between the given range.

11. Draw the flowchart and write a *recursive* C function to find the factorial of a 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 coefficient **nCr**. Tabulate the results for different values of **n** and **r** with suitable messages.

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

Student Name	USN	Heading
Name 1	USN1	
Name 2	USN2	
.....	
.....	

13. Write a C program to maintain a record of "**n**" student details using an array of structures with four fields (Roll number, Name, Marks, and Grade). Assume appropriate data type for each field. Print the marks of the student, given the student name as input.

14. 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.

Reference Book:

1. Reema Thareja, Computer Fundamentals and Programming in C, Oxford Press, 2012.

Practical Examination Procedure :

1. All laboratory experiments (Fourteen) are to be included for practical examination.
2. Students are allowed to pick one experiment from the lot.
3. Strictly follow the instructions as printed on the cover page of answer script for breakup of marks
4. **Change of experiment is allowed only once and 15% Marks allotted to the procedure part to be made zero.**

CONTENTS

SI No.	Titles	Page No.
I	Laboratory Session-1: Write-up on Functional block diagram of Computer, CPU, Buses, Mother Board, Chip sets, Operating System & types of OS, Basics of Networking & Topology and NIC.	1-8
II	Laboratory Session-2: Write-up on RAM, SDRAM, FLASH memory, Hard disks, Optical media, CD-ROM/R/RW, DVDs, Flash drives, Keyboard, Mouse, Printers and Plotters. Introduction to flowchart, algorithm and pseudo code.	9-15
Laboratory Experiments		
1	Design and develop a flowchart or an algorithm that takes three coefficients (a, b, and c) of a Quadratic equation ($ax^2+bx+c=0$) as input and compute all possible roots. Implement a C program for the developed flowchart/algorithm and execute the same to output the possible roots for a given set of coefficients with appropriate messages.	16-19
2	Design and develop an algorithm to find the reverse of an integer number NUM and check whether it is PALINDROME or NOT. Implement a C program for the developed algorithm that takes an integer number as input and output the reverse of the same with suitable messages. Ex: Num: 2014, Reverse: 4102, Not a Palindrome.	20-22
3	3a. Design and develop a flowchart to find the square root of a given number N. Implement a C program for the same and execute for all possible inputs with appropriate messages. Note: Don't use library function sqrt(n). 3b. Design and develop a C program to read a year as an input and find whether it is leap year or not. Also consider end of the centuries.	23-26
4	Design and develop an algorithm for evaluating the polynomial $f(x) = a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0$, for a given value of x and its coefficients using Horner's method. Implement a C program for the same and execute the program for different sets of values of coefficients and x.	27-29
5	Draw the flowchart and Write C Program to compute Sin(x) using Taylor series approximation given by $\sin(x) = x - (x^3/3!) + (x^5/5!) - (x^7/7!) + \dots$. Compare the result with the built-in Library function and print both the results with appropriate messages.	30-32
6	Develop an algorithm, implement and execute a C program that reads N integer numbers and arrange them in ascending order using Bubble Sort.	33-36
7	Develop, implement and execute a C program that reads two matrices A (m x n) and B (p x q) and Compute the product A and B. Read matrix A and matrix B in row major order and in column major order respectively. Print both the input matrices and resultant matrix with suitable headings and output should be in matrix format only. Program must check the compatibility of orders of the matrices for multiplication. Report appropriate message in case of incompatibility.	37-42
8	8. Develop, implement and execute a C program to search a Name in a list of names using Binary searching Technique.	43-46

9	Write and execute a C program that i. Implements string copy operation STRCOPY (str1,str2) that copies a string str1 to another string str2 without using library function. ii. Reads a sentence and prints frequency of each of the vowels and total count of consonants.	47-53
10	10a. 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. 10b. Design and develop a C function isprime(num) that accepts an integer argument and returns 1 if the argument is prime, a 0 otherwise. Write a C program that invokes this function to generate prime numbers between the given range.	54-59
11	Draw the flowchart and write a recursive C function to find the factorial of a 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 coefficient nCr. Tabulate the results for different values of n and r with suitable messages.	60-62
12	Given two university information files "studentname.txt" and "usn.txt" that contains students Name and USN respectively. Write a C program to create a new file called "output.txt" and copy the content of files "studentname.txt" and "usn.txt" into output file in the sequence shown below. Display the contents of output file "output.txt" on to the screen.	63-65
13	Write a C program to maintain a record of "n" student details using an array of structures with four fields (Roll number, Name, Marks, and Grade). Assume appropriate data type for each field. Print the marks of the student, given the student name as input	66-68
14	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.	69-71
	VIVA-VOCE Questions	72

Laboratory Session-1

Write-up on Functional block diagram of Computer, CPU, Buses, Mother Board, Chip sets, Operating System & types of OS, Basics of Networking & Topology and NIC.

Description about Functional block diagram of Computer:

A computer is an electronic device, which mainly performs the four functions as **reading**, **processing**, **displaying** and **storing** on data. These functions of a computer system can be carried out by using the three main units namely input unit, system unit and output unit. The block diagram of a computer system is as follows:

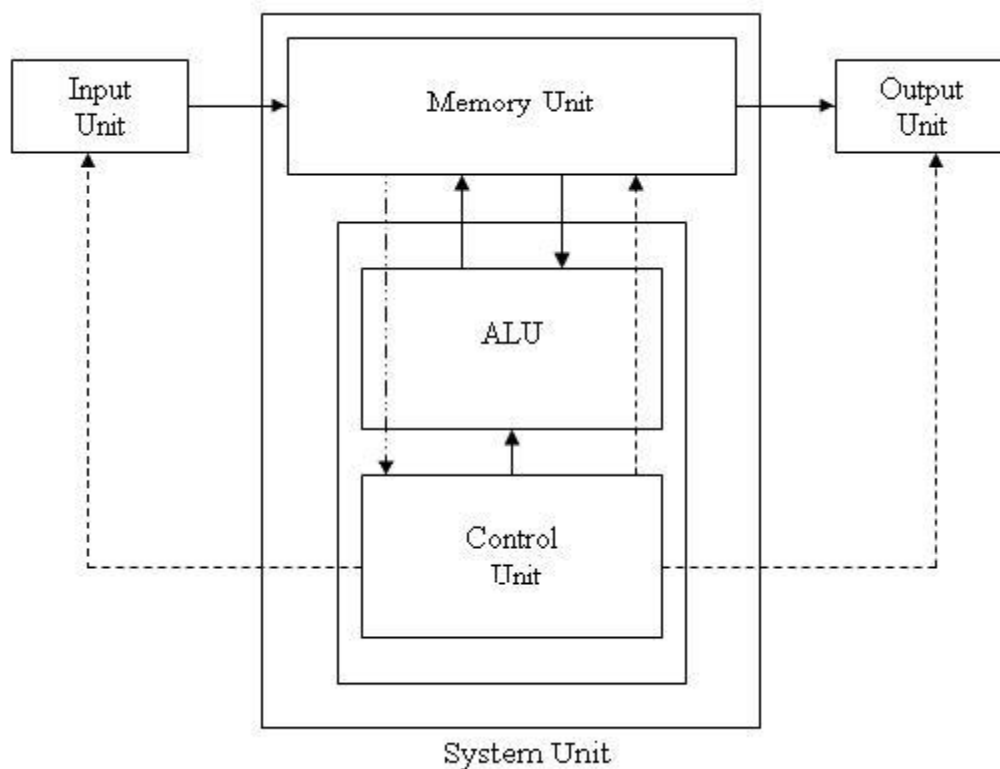


Fig 1: Block Diagram of a Computer

Notations:

- Data and Results flow _____
- Control instructions to other units from control unit -----
- Instructions from memory unit to control unit _____

System or Central Processing Unit (CPU): is commonly known as “processor” that executes the instructions of a computer program. It has Control Unit (CU) and Arithmetic & Logical Unit (ALU). These two units perform the basic arithmetic, logical, and input/output operations.

a) Input unit: is used to enter data and information into a computer. The devices like keyboard, mouse and scanner are commonly used input devices.

- A keyboard is used to enter alphanumeric characters and symbols.
- The mouse is used to pick or select a command from the monitor screen.
- A scanner is used to scan an image or read a barcode and so on.

b) Arithmetic and Logic Unit (ALU): is a digital circuit that perform arithmetic (Add, Sub, Multiplication, Division) and logical (AND, OR, NOT) operations. It helps in fast computation of scientific calculations on floating-point number.

c) Control unit (CU): is the circuitry that controls the flow of information through the processor and coordinates the activities of the other units within the processor.

Functions of Control unit

- Accessing data & instructions from memory unit
- Interpreting instructions
- controlling input and output units
- Overall supervision of a Computer system

d) Memory Unit (MU): is the unit where all the input data and results are stored either temporarily or permanently. The CPU memory is also called as memory register. The memory of a computer has two types:

a. Main Memory / Primary Memory

units i. Random Access Memory (RAM)

ii. Read Only Memory (ROM)

b. Secondary Memory / Auxiliary Memory

e) Output Unit: It is used to display or print results from a computer. Monitor, printer and plotters are commonly used output devices.

f) Bus: A bus is a collection of wires that carries data/Instructions. It connects physical components such as cables, printed circuits, CPU, Memory, Peripherals etc., for sharing of Information and communication with one another. The purpose of buses is to reduce the number of "pathways" needed for communication between the components, by carrying out all communications over a single data channel.

Types of Buses:

- 1. System Buses:** The system buses are used to transfer the data and instructions between Main memory (Random Access Memory) and CPU. These are classified into following three types.

Data Bus	Address Bus	Control Bus
It is used to transfer the data between Processor, Memory and I/O devices.	It is used to transfer the addresses of data and instructions stored in memory.	It is used to transfer the control signals between CPU, Memory and I/O devices.
Bidirectional in nature	Unidirectional in nature	Unidirectional or Bidirectional in nature

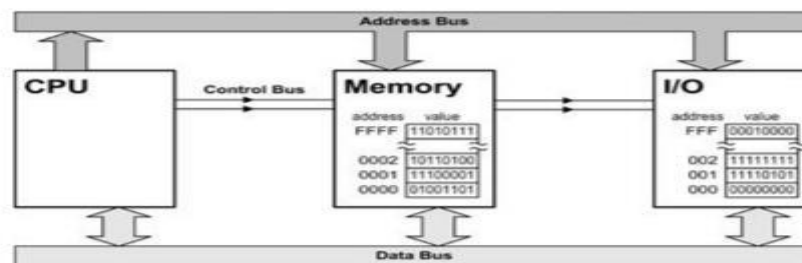


Fig 2: Types of Buses

2. **I/O Buses:** The buses which are used to connect all I/O devices with CPU and Memory are called I/O buses. These are classified into following three types.

PCI Bus	ISA Bus	USB Bus
PCI stands for Peripheral Component Interconnect	ISA stands for Industry Standard Architecture	USB stands for Universal Serial Bus
The motherboard will be having 3 or 4 PCI connectors, so that we can insert various chips.	This is simple and slowest bus used in IBM PCs	It helps to connect various I/O devices like keyboard, mouse, pen drives, printer, etc.
Fastest and presently more powerful bus	Oldest, simplest and slowest bus	Newest and widely used bus

Main Board or Mother Board: Mother Board is a set of Integrated Chips (ICs) which are designed to work together. It controls the flow of data/instructions within our computer. It is the main board on which other hardware components are connected to enable the computer system to work as an integrated unit. It consists of sockets, slots, power connectors and bus.

Chip sets: Chip set is the set of integrated chips that are designed to work together. These set of chips controls the flow of information on computer. The chips may be controllers for memory, cache, hard drive, key board and peripherals.

Operating System and its types: An Operating System (OS) is a system software that **controls** and **supervises** the **hardware components** of a computer system and it provides the services to computer users. Also called as **Resource Manager** that manages the resources such as CPU, Memory, I/O devices, Job/Task/Process etc., a computer cannot run without it.

The major functions of OS includes:

- CPU Management,
- Memory Management,
- File Management,
- Device Management,
- Process/Task/Job Management and Security Management.

The primary goal of an OS is to make the computer system convenient and efficient to use. An OS ensures that the system resources (such as CPU, memory, I/O devices, etc) are

utilized efficiently. For example, there may be many programs residing in the main memory. Therefore, the system needs to determine which programs are active and which need to wait for some I/O operation.

Some of the examples of Operating Systems:

- Windows –XP is an O.S. is used for Personal Computers (PCs)
- Unix and XENIX are the OSs used for multi-user computers.
- Windows 7, Windows 8, Macintosh OS, Fedora, and Android, etc.

Types of Operating Systems: The operating systems are classified into 7 types based on their capability and usage.

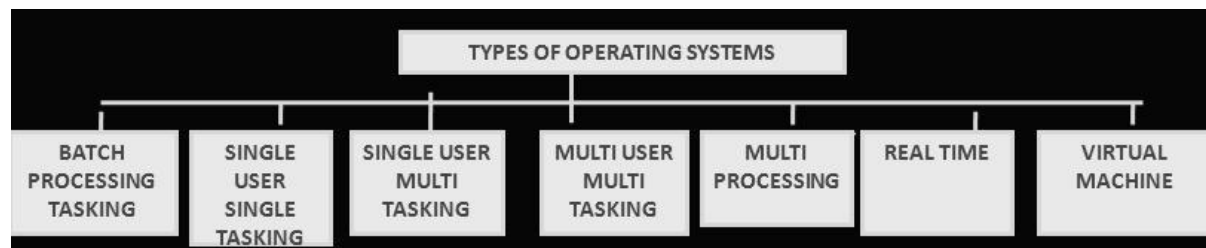


Fig 3: Types of OS

- **Batch Processing Tasking OS:** The data is collected into a group called batch and provides only one batch (one after another) of jobs as input to the computer system at a time. The jobs in a batch are processed on first come first serve basis. In this type, the process takes place at specified time intervals i.e. weekly or monthly without user interaction. **E.g.** Punch cards were using to store the data in batch processing and in payroll preparation in a business batch processing was helpful.
- **Single user and single tasking OS:** The OS that allows only one program to execute at a time is called single user single tasking operating system. Using this operating system user can do only one task at a time. **E.g.** DOS (Disk Operating System).
- **Single user and multi tasking OS:** The OS that allows a single use to perform more than one task at a time is called single user multi tasking operating system. While working with the Ms-Word user can perform other work like print a document, listen music .**E.g.** Windows-XP, Windows Vista, Windows – 7, etc.
- **Multi user and multitasking OS:** The O.S. that allows two or more users to use a main computer system to do more than one task is called multiuser and multitasking operating system.**E.g.** Unix is a multiuser and multitasking operating system.
- **Multiprocessing OS :** The OS that allows multiple programs to be executed by multiple CPUs (Processors) is called multiprocessing operating system. Super and main frame computers have more than one CPU and multiprocessing operating system.
- **Real Time Operating System (RTOS):** The OS that is used for real time applications and to carry out certain calculations within the specified time constraint. This OS is used in applications such as mobile phones, supporting systems in hospitals, nuclear power plants, oil refining, chemical processing, environmental applications and air-traffic control systems, disaster management etc.,

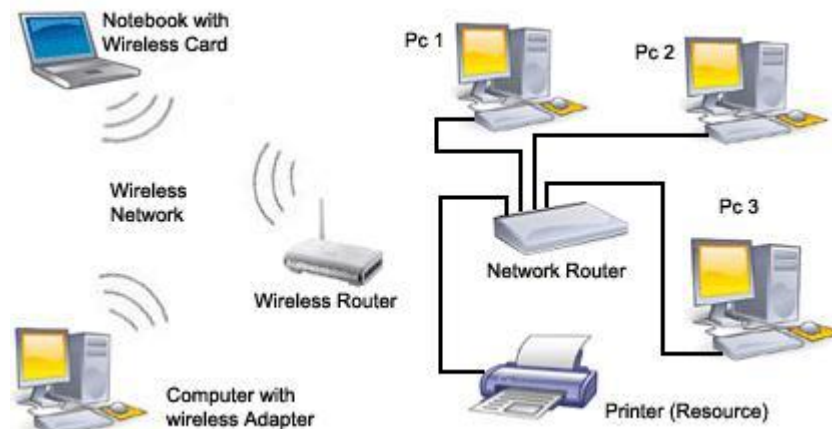
- **Virtual machine OS:** Allows several users of a computer system to operate as if each has the only terminal attached to the computer.

Basics of Networking & Topology and Network Interface Card(NIC):

Introduction to Computer Network:

A computer network is a collection of computers and devices interconnected to facilitate sharing of resources among interconnected devices. Advantages of Computer Networks include File Sharing, Resource Sharing, Increased Storage Capacity, Load Sharing and Facilitate communications.

Computers in a network can be connected by using telephone lines, cables, satellite links, etc., Wireless network will use radio signals to exchange the information.

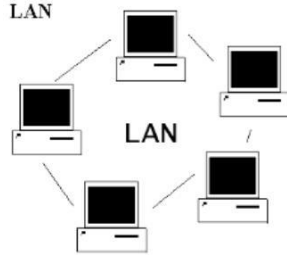
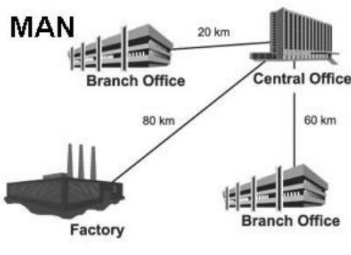
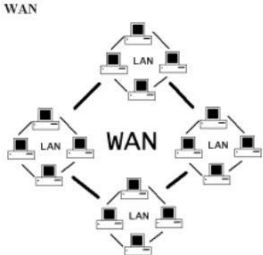


Basic components of a computer network: The basic components of a network are as follows.

1. **Protocol:** Set of rules used during the data transmission.
2. **Transmission Medium:** The media used to connect computer to each other like telephone lines, twisted pair wire, co-axial cable, fiber optics, satellite signals and radio signals, etc.
3. **Processors:** Modem, Multiplexers, bridges, routers, gateways, hub etc. are the processors used in the network for the flow of data.
4. **Channels:** Analog/Digital, Synchronous/Asynchronous, Switched/Non switched, Simplex / duplex, etc.
5. **Topology:** Physical network layout used for networking. For example, bus topology, star topology, ring topology, and mesh topology
6. **Software:** User interface software like Internet Explorer, Netscape Navigator, FTP (File Transfer Protocol), Telnet (Telecommunication Network), PPP (Point to Point Protocol), and SMTP (Simple Mail Transfer Protocol) etc.

Types of Networks: The computer networks are mainly classified into 3 types

LAN	MAN	WAN
Local Area Network	Metropolitan Area Network	Wide Area Network
A group of computers that are connected in a small area such as building, home, office etc i.e. within a small campus	A network which covers large area like a city	A network which covers a large area like a state, country or across a continent.

Distance covered by this network is less than 1 KM Used within a single building like home or office	Distance covered by this network is 5 to 50 KM. Used by private organization like cable television in our city.	Distance covered by this network is 100 to 1000 KM Used all over the world. i.e. good example is internet
Computers are connected through the twisted pair cables and co axial cables.	A network device called router is used to connect the LANs together	It uses fibre optics, cables and even satellite signals as a transmission media.
Transmitting data is cheaper	Transmitting data is costlier	Transmitting data is more costlier
Transmission data is generally error free	Transmission data is generally error prone	Transmission data is generally error free
		

Network Topologies: Topology refers to the schematic description of the arrangement of a network. It is the actual geometric layout of computers and other devices connected to the network.

Types of Network Topologies: These are mainly classified into 4 types.

1. Bus Topology:

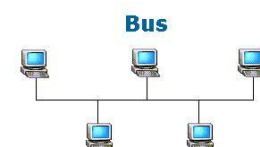
In this network structure, a single cable runs in a building or campus. All the nodes (terminals / computers) are connected to this single cable. It is suitable for Local Area Network.

Advantages:

- Failure of one node will not affect the whole network.
- Well suited for quick setup
- Easy to install and expand
- High rate of data transmission as compare to star and ring topology

Disadvantages:

- A cable break can disable the entire network
- Trouble shooting is very difficult
- Only a single message can travel at a time



2. Ring Topology:

In this network structure, all the computers are connected to each other in the form of a ring. i.e. first node is connected to second, second to third and so on. Finally last node is connected to first one.



Advantages:

- All the nodes have equal chance to transfer the data
- These are easily extensible
- It can span longer distance than other type of networks

Disadvantages:

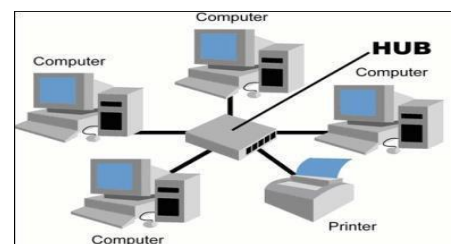
- Difficult to install
- Expensive
- Difficult to troubleshoot
- Adding or removing computer can disturb the entire network

3. Star Topology:

In this network structure, all the computers are connected with a centralized system called server. The central computer is also called a hub. To transmit information from one node to another node, it should be transmitted through a central hub. The central hub manages and controls all the functions of network.

Advantages:

- Easy to install and expand.
- Addition or deletion of a node is easier.
- Failure of one node will not affect the entire network.
- Well suited for quick setup
- Easier to debug network problems through a hub

**Disadvantages:**

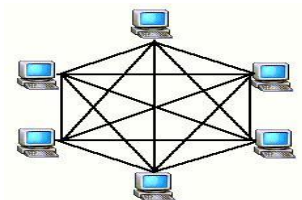
- Failure of a central system i.e. hub will affect the whole network
- Cost of hub is expensive.

4. Mesh Topology:

In this network structure, all the computers and network devices are interconnected with one another like a mesh. Every node has a connection to every other node in the network. This topology is not commonly used for most computer networks because of its installation difficulty and expensive.

Advantages:

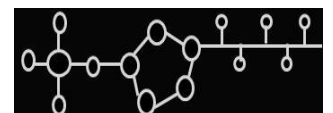
- Failure of a single node will not affect the entire network
- Data transfer rate is very fast because all the nodes are connected to each other.

**Disadvantages:**

- Installation and re configuration is very difficult
- Costlier

5. Hybrid Topology:

Each of the topologies has their own advantages and disadvantages. So in the real world, a pure star or pure ring or bus is rarely used. Rather a combination of two or more topologies is used. Hence, hybrid network topology uses a combination of any two or more topologies in such a way



that the resulting network does not exhibit one of the standard topologies (e.g., bus, star, ring, etc.). Two very commonly used hybrid network topologies include the *star ring network* and *star bus network*.

Network Interface Cards (NICs):

This is the important hardware component, which connects the machine to the computer network. This will be fixed into one of the free slot on the mother board. It has one port for the connection of a network cable. These cards typically use an Ethernet connection and are available in 10, 100, and 1000 Base-T configurations.

A 100 Base-T card can transfer data at 100 Mbps. The cards come in ISA and PCI versions and are made by companies like 3Com and LinkSys.



Laboratory Session-2

Write-up on RAM, SDRAM, FLASH memory, Hard disks, Optical media, CD-ROM/R/RW, DVDs, Flash drives, Keyboard, Mouse, Printers and Plotters. Introduction to flowchart, algorithm and pseudo code.

Random Access Memory (RAM):

RAM is basically main memory of the computer.

RAM is a semiconductor memory made up of small memory chips that form a memory module. These modules are installed in the RAM slots on the motherboard of computer. Every time you open a program, it gets loaded from the hard drive into the RAM. This is because reading data from the RAM is much faster than reading data from the hard drive.

**Synchronous Dynamic Random Access Memory (SDRAM):**

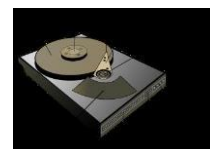
It is an improvement to standard DRAM because it retrieves data alternately between two sets of memory. This eliminates the delay caused when one bank of memory addresses is shut down while another is prepared for reading. It is called "Synchronous" DRAM because the memory is synchronized with the clock speed that the computer's CPU bus speed is optimized for. The faster the bus speed, the faster the SDRAM can be. SDRAM speed is measured in Megahertz.

FLASH memory:

Flash memory is a type of Electrically Erasable Programmable Read-Only Memory (EEPROM). The name comes from how the memory is designed -- a section of memory cells can be erased in a single action or in a "flash.". Flash memory cards used for digital cameras, cellular phones, networking hardware, and PC cards.

Hard disks: Hard disk is prime unit of storage of the computer. Huge amount of data can be stored and accessed in few milliseconds. The hard disk consists of more number of disks arranged in the cylindrical order, one above another on a spindle.

The read/write heads are attached to single access mechanism so that they cannot move independently. All read/write heads are moved together to position that heads on the required track. The hard disks available today ranges from 200 GB to 2TB and so on. The present day hard disk ranges from 3600 rpm to more than 10000 rpm and so on.



Advantages: High storage capacity, high data accessing rate and permanent storage medium.

Disadvantages: It is not portable.

Optical media: An optical storage media is kind of storage, which is coated with thin metal on which bits are stored. The data can be stored in to optical storage media or read form the optical storage media.

- The devices which perform read or write operation on optical storage media are called optical storage media.

- The laser technology is used to read the data or write the data on optical storage devices.

Examples: CD-ROM, DVD etc.

Compact Disc Read-Only-Memory (CD-ROM):

It is a type of optical disc that uses laser technology to read and write data on the disc. The information stored on CDROM becomes permanent and cannot be altered. This means that the stored information can only be read for processing. A CD-ROM uses the round shaped optical disk to store data, applications, games and audio files. It can store up to 700 MB of data. It has become integral part of every organization due to its features like reliability, reasonable, storage capacity and easy to use of carry. CD-Drive will be with motor to rotate the disks to perform read and write operations. A CD-drive will consists of the components like Disc drive, disk drive motor, laser pick up assembly tracking drive and tracking motor and so on.



Compact Disk Recordable (CD-R):

The CD-R allows you to create your own CD. CD-R drives have the ability to create CDs but they can write data on the disk only once. CD-R technology also called as Write Once-Read much (WORM) technology. Laser technology is used to write the data on the compact disk. CD-R drives come in IDE, SCSI and USB models.

Compact Disc Rewritable (CD-RW):

CD-RW is an erasable optical disk which is used to write data multiple times on a disk, CD-RW disks are good for data backup, data archiving or data distribution on CDs. The disk normally holds 700MB of data. Technology to write data multiple times on a CD was known as the Phase change Dual (PD) technology. The reflective properties of a CD-RW are different than regular CD-ROM disks.

Digital Video Disk or Digital Versatile Disc (DVD-ROM):

A DVD is a small optical disk having high density medium and capable of storing a full-length movie on a single disk. The high density is achieved by using both sides of the disk, special data-compression technology, and extremely small tracks to store the data.

Advantages: Storage capacity is more compared to CDs.

Flash Drives (Pen drives):

USB flash drives are removable, rewritable, and physically much smaller drives weighing even less than 30 g. A flash drive consists of a small printed circuit board carrying the circuit elements and a USB connector, insulated electrically and protected inside a plastic, metal, or rubberized case which can be carried in a pocket or on a key chain.

Advantages

- Data stored on flash drives is impervious to scratches and dust
- Mechanically very robust
- Easily portable
- Have higher data capacity than any other removable media.
- Compared to hard drives, flash drives use little power



- Flash drives are small and light-weight devices
- Flash drives can be used without installing device drivers.

Disadvantages

- Can sustain only a limited number of write and erase cycles before the drive fails.
- Most flash drives do not have a write-protect mechanism
- Flash drives are very small devices that can easily be misplaced, left behind, or otherwise lost.
- The cost per unit of storage in a flash drive is higher than that of hard disks

Keyboard:

A keyboard is the primary input device used in all computers. Keyboard has a group of switches resembling the keys on an ordinary typewriter machine. Normally keyboard has around 101 keys. The keyboard includes key that allows us to type letters, numbers and various special symbols such as *, /, [, % etc.

Mouse:

The mouse is the key input device to be used in a Graphical User Interface (GUI). The users can use mouse to handle the cursor pointer easily on the screen to perform various functions like opening a program or file. With mouse, the users no longer need to memorize commands, which was earlier a necessity when working with text-based command line environment such as MS-DOS.

Advantages:

- Easy to use; Cheap; Can be used to quickly place the cursor anywhere on the screen
- Helps to quickly and easily draw figures
- Point and click capabilities makes it unnecessary to remember certain commands

Disadvantages:

- Needs extra desk space to be placed and moved easily
- The ball in the mechanical mouse needs to be cleaned very often for smooth movements

Printers:




The printer is an output device, which is used to get hard copy of the text displayed on the screen. The printer is an external optional device that is connected to the computer system using cables. The printer driver software is required to make the printer working. The performance of a printer is measured in terms of Dots Per Inch (DPI) and Pages Per Minute (PPM) produced by the printer.

Types of Printers:

1) Impact Printers: Impact printers are those printers in which a physical contact is established between the print head, ribbon (cartridge) and paper. **E.g.** Dot Matrix Printers

2) Non-Impact Printers: No physical contact is established between the print head, ribbon (cartridge) and paper. **E.g.** Inkjet Printers and Laser Printer

Dot matrix, Inkjet and Laser printers

Sl. No	Dot Matrix Printer	Inkjet Printer	Laser Printer
1	Impact Printer	Non-impact Printer	Non-impact printer
2	It uses metal pins in its head to create text and graphics in the form of dots.	Its print head does not have metal pins; instead it has several tiny nozzles that spray ink onto the paper. Each nozzle is thinner than hair.	The laser printer uses a beam of laser for printing.
3	The process of printing involves striking a pin against a ribbon to produce its output.	The ink cartridges are attached to the printer head that moves horizontally from left to right.	The printer uses a cylindrical drum, a toner and the laser beam.
4	Printing speed is slower than laser printer,	Printing speed is slower than laser dot matrix.	Printing speed is higher than both.
5	Character by character printing	Line by line printing	It is a page printer
6	Low quality printing	High quality printing	High quality printing
7	Less expensive	High expensive	High expensive
8	Generates much noise while printing	Generates less noise while printing	No noise
9	Speed is measured in DPI (Dots Per Inch)	Speed is measured in CPI (Characters Per Inch)	Speed is measured in PPM (Pages Per Minute)
10	Monochrome (Black & White) Printers	Colour printer	Monochrome and colour printer
11			

Plotters: A plotter is similar to printer that produces hard-copy output with high-quality colour graphics. Plotters are generally more expensive than printers, ranging from about \$1000 to \$75000.



Problem Solving Techniques:

The process of working through details of a problem to reach a solution. There are three approaches to problem solving:

- **Algorithm**
- **Flowchart**
- **Pseudo Code**

Algorithm: The algorithm is a ***step-by-step procedure*** to be followed in solving a problem. It provides a scheme to solve a particular problem in ***finite number of unambiguous steps***. It helps in implementing the solution of a problem using any of the *programming languages*. In order to qualify as an algorithm, a sequence of instructions must possess the following characteristics:

- **Definiteness:** Instructions must be ***precise and unambiguous*** i.e. each and every instruction should be clear and should have only one meaning.
- **Finiteness:** ***Not*** even a single instruction must be ***repeated infinitely***. i.e., each instruction should be performed in finite time.
- **Termination:** After the algorithm gets executed, the user should get the desired result

Key features of an algorithm:

Any algorithm has a finite number of steps and some steps may involve decision making, repetition. Broadly speaking, an algorithm exhibits three key features that can be given as:

- **Sequence:** Sequence means that each step of the algorithm is executed in the specified order.
- **Decision:** Decision statements are used when the outcome of the process depends on some condition.
- **Repetition:** Repetition which involves executing one or more steps for a number of times can be implemented using constructs like the while, do-while and for loops. These loops executed one or more steps until some condition is true.

Example: To compute the Area of Rectangle

ALGM: AREA_of_RECTANGLE [This algorithm takes length and breadth, the sides of the rectangle as input and computes the area of rectangle using the formula ***area=length * breadth***. Finally it prints the area of rectangle]

STEPS:

Step 1:[Initialize]

Start

Step 2: [Input the sides of Rectangle]

Read **length, breadth**

Step 3:[Compute the area of rectangle]

Area=length*breadth




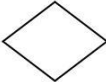
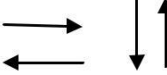



Step 4:[Display the Area]

Print **Area**

Step 5: [Finished]

Stop

Flowcharts: A flowchart is a graphical or symbolic representation of an algorithm. They are basically used to design and develop complex programs to help the users to visualize the logic of the program so that they can gain a better understanding of the program and find flaws, bottlenecks, and other less-obvious features within it. Basically, a flowchart depicts the “**flow**” of a program. The following table shows the symbols used in flowchart along with its descriptions.

Symbol	Name	Description
	oval	Represents the terminal point
	Rectangle	Represents the process steps defined in algorithm
	Parallelogram	Indicate the reading Operation used for input/output or data or information from/to any device
	Diamond	Indicates the decisions (questions) and consequently the branch points or the paths to be followed based on the result of the question
	Arrows	Shows the flowchart direction and connects the various flow chart symbols
	Small circle	Shows the continuation from one point in the process flow to another
	Hexagon	Represents Looping structures
	Predefined Process	Indicates Subroutines

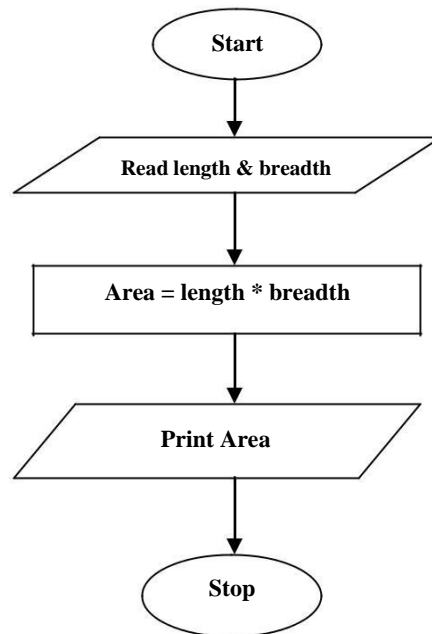
Advantages of Flowcharts:

- A flowchart is a diagrammatic representation that illustrates the sequence of steps that must be performed to solve a problem. They are usually drawn in the early stages of formulating computer solutions to facilitate communication between programmers and business people.
- Flowcharts help programmers to understand the logic of complicated and lengthy problems.
- They help to analyse the problem in a more effective manner
- Flowchart can be used to debug programs that have error(s).

E.g.: To compute the Area of Rectangle

Limitations of using Flowcharts:

- Drawing flowcharts is a laborious and a time consuming activity.
- Flowchart of a complex program becomes, complex and clumsy. At times, a little bit of alteration in the solution may require complete re-drawing of the flowchart
- Essentials of what is done may get lost in the technical details of how it is done.
- There are no well-defined standards that limits the details that must be incorporated in a flowchart

**Pseudocode:**

It is a form of structured English that describes algorithms. It facilitates the designers to focus on the logic of the algorithm without getting bogged down by the details of language syntax. **Pseudocode** is a compact and informal high-level description of an algorithm that uses the structural conventions of a programming language. It is meant for human reading rather than machine reading, so it omits the details that are not essential for humans. Such details include keywords, variable declarations, system-specific code and subroutines. There are no standards defined for writing a **pseudocode** because it is not an executable program. Flowcharts can be considered as a graphical alternative to **pseudocode**, but are more spacious on paper.

E.g.: To compute the area of Rectangle

Begin

Input length, breadth

Area=length*breadth

Print Area

End

Experiment No. 1: Design and develop a flowchart or an algorithm that takes three coefficients (**a**, **b**, and **c**) of a Quadratic equation ($ax^2+bx+c=0$) as input and compute all possible roots. Implement a C program for the developed flowchart/algorithm and execute the same to output the possible roots for a given set of coefficients with appropriate messages.

Algorithm: *To find and output all the roots of a given quadratic equation for non zero coefficients*

Step 1: [Start]

Begin

Step 2: [Input the co-efficients of the quadratic equation]

Read a,b,c

Step 3:[Check for the non-zero coefficient of a]

If $a = 0$ then print "Invalid input", go to step 2

Step 4: [Find the value of

disc] $disc = b^2 - 4 * a * c$

Step 5: [Find the type and the values of roots of a given quadratic

equation] If $(disc = 0)$ then

Print "The roots are

equal" $root1 = root2 = -b /$

$2.0*a$ Go to step 6

Else If $(disc > 0)$ then

Print "The roots are real and distinct"

$root1 = (-b + \sqrt{disc}) / 2.0*a$

$root2 = (-b - \sqrt{disc}) / 2.0*a$

Go to step 6

else

Print "The roots are

imaginary" $root1 = -b / 2.0*a$

$root2 = \sqrt{\text{fabs}(disc)} / 2.0*a$

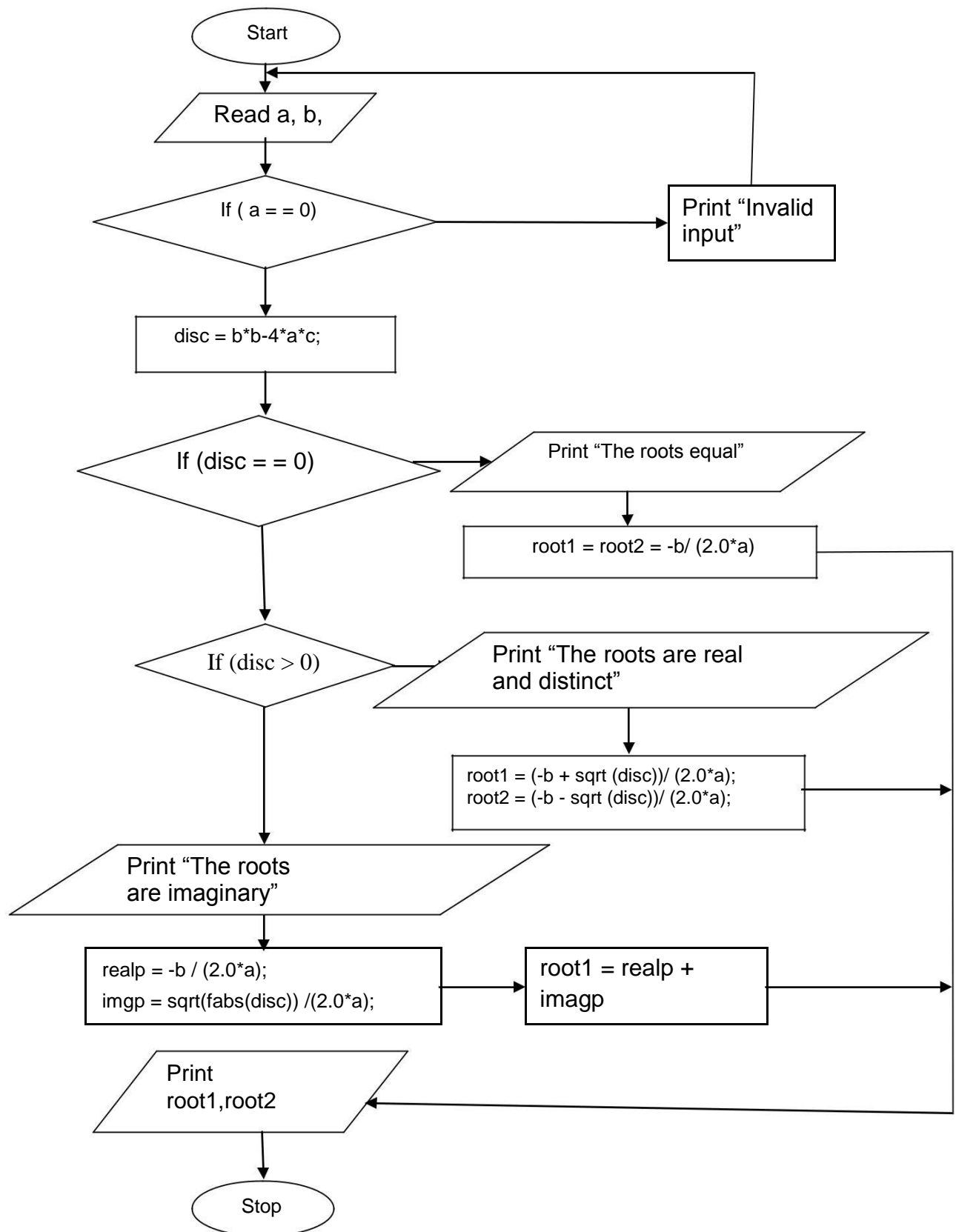
Step 6: [Output]

Print root1, root2

Step 7: [Stop]

End

Flow chart



Program 1:

```
// Preprocessor directives
#include<stdio.h>
#include<conio.h>
#include<math.h>

// Main program
main()
{
    // Variable(s) declaration
    float a,b,c;
    float root1, root2, realp, imgp, disc;

    clrscr();

    // Program statements and expressions

    printf("\n C program to find and output all the roots of a given quadratic equation");
    printf("\n for non-zero coefficients");

    printf("\n\n Enter the value of coefficient a (Note: Non zero only) ...");
    scanf("%f", &a);

    if(a == 0)
    {
        printf("\n Invalid input...Retry again"); getch();
        exit(0);
    }

    printf("\n Enter the value of coefficient b → ");
    scanf("%f", &b);
    printf("\n Enter the value of coefficient c → ");
    scanf("%f", &c);

    disc = b*b-4*a*c;    // compute discriminant

    // find the type and values of the roots of a given quadratic equation
    if(disc == 0)
    {
        printf("\n\n The roots are equal...");
        root1 = root2 = -b / (2.0*a);

        printf("\n Root1 = Root2 = %.2f", root1);
    } //endif
    else
        if(disc > 0)
        {
            printf("\n\n The roots are real and distinct...");
            root1 = (-b + sqrt(disc))/(2.0*a);
            root2 = (-b - sqrt(disc))/(2.0*a);
            printf("\nRoot1 = %.2f", root1);
            printf("\nRoot2 = %.2f", root2);
        } //endif
```

```
        else
        {
            printf("\n\n The roots are
            imaginary..."); realp = -b/(2.0*a);
            imgp = sqrt(fabs(disc))/(2.0*a);

            printf("\nRoot1 = %.2f + i %.2f",realp,
            imgp); printf("\nRoot2 = %.2f - i %.2f",realp,
            imgp); } //end else

    getch();

} // end main
```

Sample Output:

```
1.
Enter the value of coefficient a...: 0
Invalid input...Retry again

2.
Enter the value of coefficient a...: 1.0
Enter the value of coefficient b...: -4.0
Enter the value of coefficient c...: 4.0

The roots are equal...
Root1 = Root2 = 2.0000

3.
Enter the value of coefficient a...: 1
Enter the value of coefficient b...: -5.0
Enter the value of coefficient c...: 6.0

The roots are real and distinct...
Root1 = 3.0000
Root2 = 2.0000

4.
Enter the value of coefficient a...: 2.0
Enter the value of coefficient b...: 3
Enter the value of coefficient c...: 4

The roots are imaginary...
Root1 = -0.750000 +i 1.198958
Root2 = -0.750000 -i 1.198958
```

Experiment No. 2: Design and develop an algorithm to find the **reverse** of an integer number **NUM** and check whether it is PALINDROME or NOT. Implement a C program for the developed algorithm that takes an integer number as input and output the reverse of the same with suitable messages. Ex: Num: **2014**, Reverse: **4102**, Not a Palindrome.

Algorithm: *To reverse a given integer number & check for palindrome*

Step 1: [Start]

Begin

Step 2: [Input an integer number]

Read num

Step 3: [Initialize]

temp \leftarrow num

rev \leftarrow 0

Step 4: [Repeat step 4 until num become 0]

digit \leftarrow num mod 10

rev \leftarrow rev *10 + digit

num \leftarrow num / 10.

Step 5: [Check for palindrome]

If (temp == rev)

Print (" num is a palindrome")

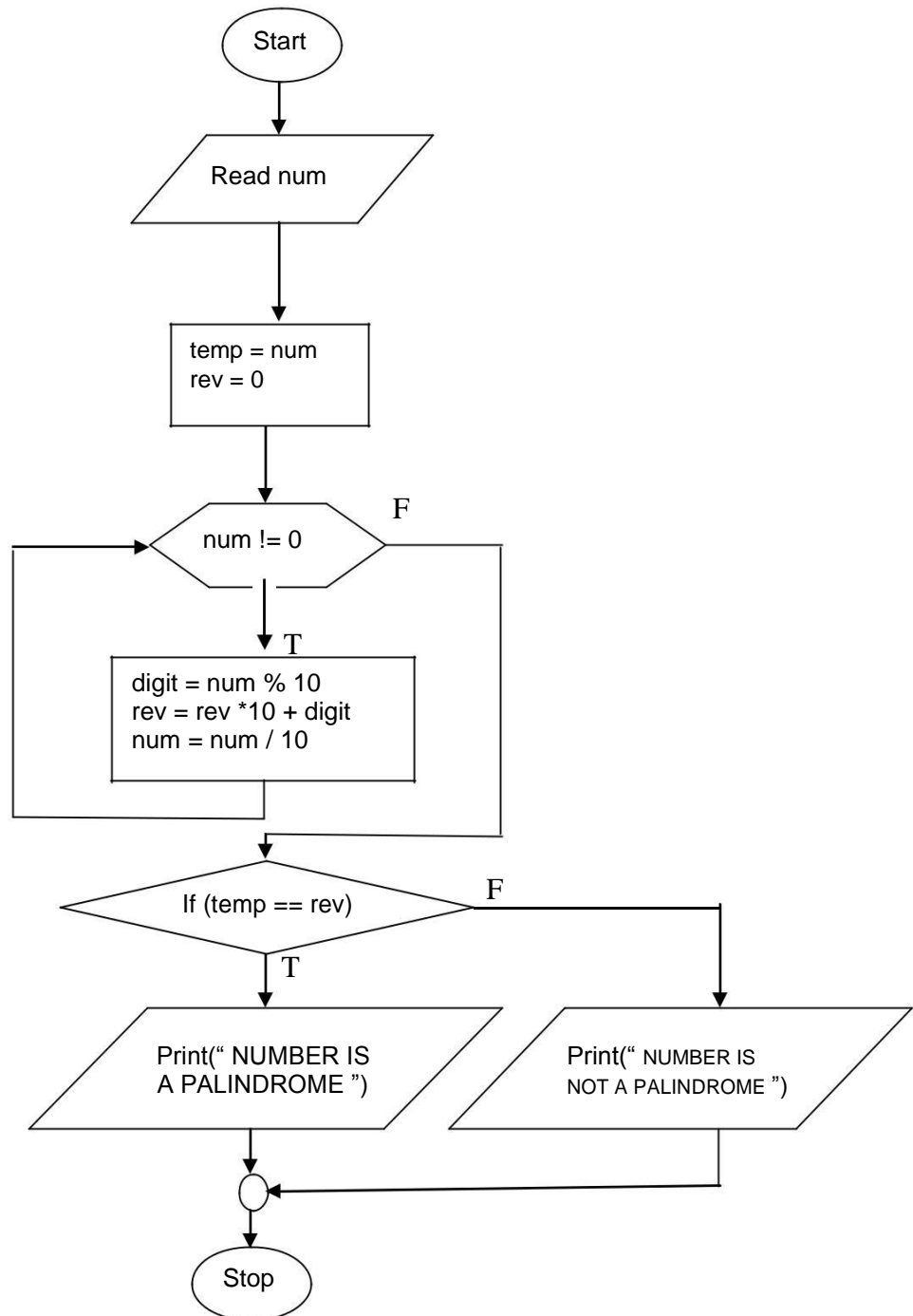
else

Print ("num is not a palindrome")

Step 6: [Stop]

End

Flow chart



Program 2:

```
// Preprocessor directives
#include<stdio.h>
#include<conio.h>

// Main program
main()
{
    // Variable declaration
    int num, temp, rev=0, digit;
    clrscr();

    // Statements and expressions
    printf("\n C program to reverse a given integer number and check whether it is a");
    printf("\n palindrome or not. Output the given number with suitable message");

    printf("\n\n Enter a valid integer number → ");
    scanf("%d",&num);

    temp = num;

    while(num != 0)
    {
        digit = num % 10;
        rev = rev * 10 + digit;
        num = num / 10;
    } // end while

    printf("\n\n The reversed number of  %d is = %d",temp,rev);
    // check for palindrome
    if(temp == rev)
        printf("\n The number is palindrome\n");
    else
        printf("\n The number is not palindrome\n");

    getch();
} // end main
```

Sample Output

```
Enter a valid integer number → 1234
The reversed number of 1234 is = 4321
The number is not palindrome

Enter a valid integer number → 1221
The reversed number of 1221 is = 1221
The number is palindrome
```

Experiment No.3:

3a. Design and develop a flowchart to find the square root of a given number **N**. Implement a C program for the same and execute for all possible inputs with appropriate messages. Note: **Don't use library function $\text{sqrt}(n)$.**

Algorithm: To find the square root of a given number N

Step 1: [Start]

Begin

Step 2: [Input an integer
n] Read n

Step 3: [Calculate square root]

$s = n/2;$

For i ← 0 thru n with step 1
 $s = (s + (n/s))/2;$

Step 4: [Output]

If $n > 0$

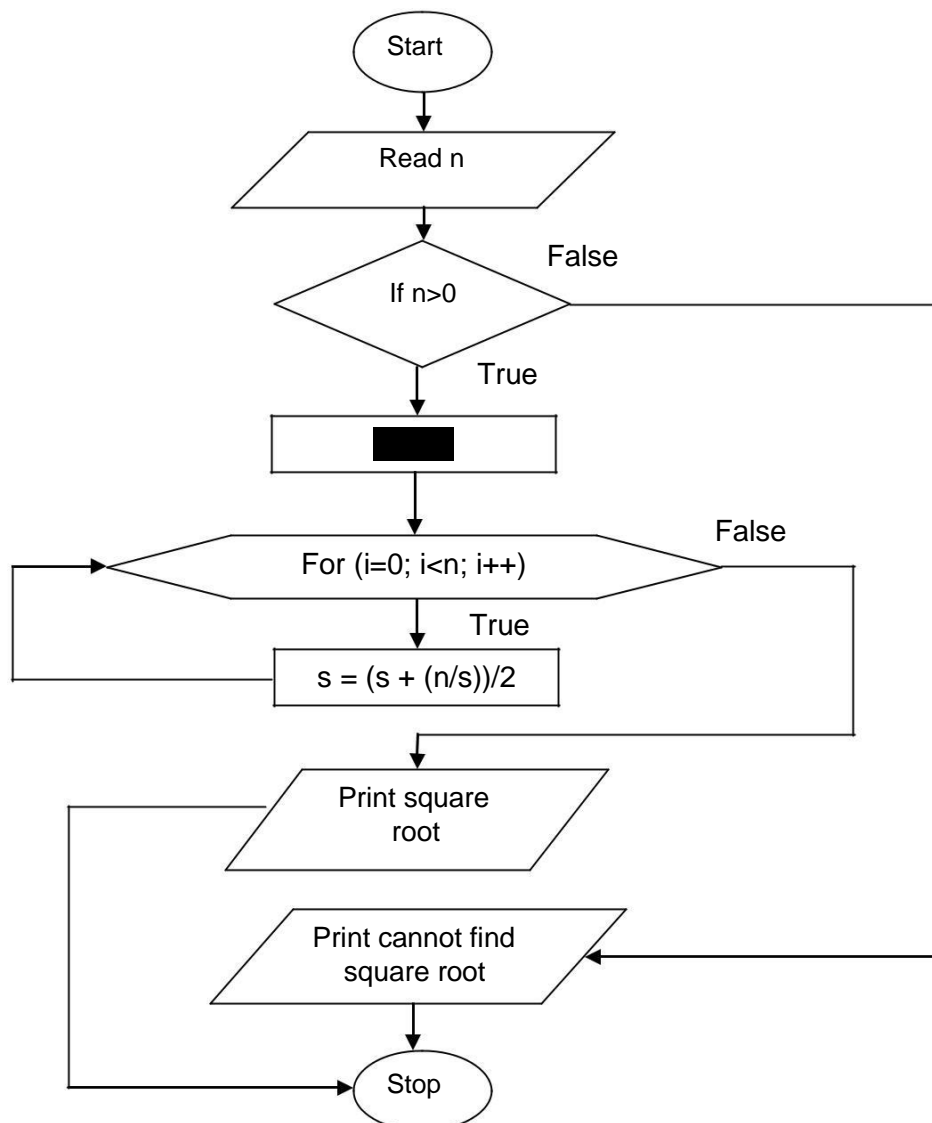
Print square root

Else

Print not possible to find square root

Step 5: [Stop]

End

Flowchart

Program 3a:

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main()
{
    float n,s;
    int i;
    clrscr();
    printf("\n Enter the number to find the square
    root\n"); scanf("%f",&n);
    if(n>0)
    {
        s = n/ 2;
        for(i = 0; i <n; i++)
            s = (s + (n/s))/2;
        printf("Square root of %f is %f\n",n,s);
    }
    else
        printf("\n Not possible to find the square root");
    getch();
}
```

Sample Output:

Enter the number to find the square
root 4
Square root of 4.000000 is 2.000000

Enter the number to find the square
root 0
Not possible to find square the root

Enter the number to find the square
root 23
Square root of 23.000000 is 4.795832

3b. Design and develop a C program to read a **year** as an input and find whether it is **leap year** or not. Also consider end of the centuries.

Algorithm: To find whether the year is leap or not

Step 1: [Start]

Begin

Step 2: [Input Year]

Read year

Step 3: [check for leap year]

If (year%4 ==0 && year%100!=0 || year%400==0)

Print the given year is leap year

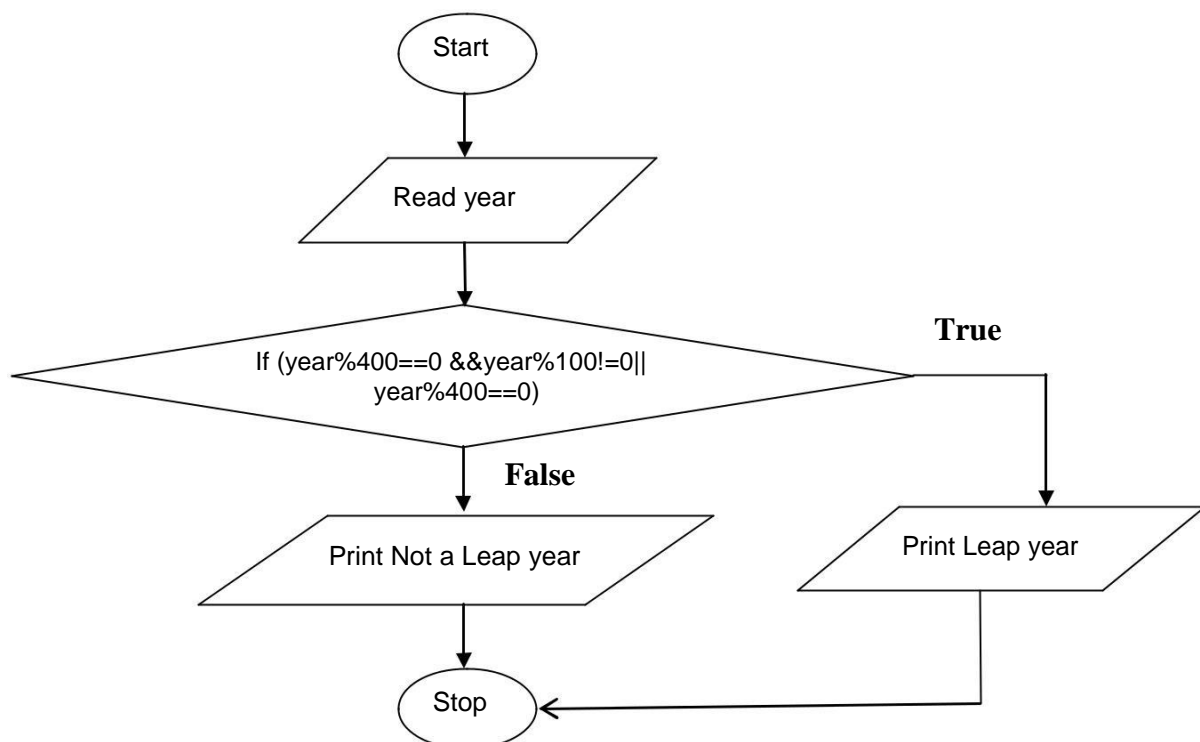
Else

Print the given year is not leap year

Step 4: [Stop]

End

Flowchart



Program 3b:

```
#include<stdio.h>
#include<conio.h>
//main void
main()
{
    int year;
    clrscr();
    printf("Enter any year:
"); scanf("%d",&year);

    if(((year%4==0)&&(year%100!=0))||((year%400==0))
        printf("%d is a leap year",year);
    else
        printf("%d is not a leap year",year);

    getch();
}
```

Sample Output:

Enter any year: 2013
2013 is not a leap year

Enter any year: 2000
2000 is a leap year

Enter any year: 1996
1996 is a leap year

Experiment No.4: Design and develop an algorithm for evaluating the polynomial $f(x) = a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0$, for a given value of x and its coefficients using Horner's method. Implement a C program for the same and execute the program for different sets of values of coefficients and x .

Algorithm: Horner's method to evaluate the given polynomial $f(x) = a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0$ for a given value of x .

Step 1: [Start]

Begin

Step 2: [Input (integer / float) the coefficients of a polynomial of degree 4]

For i ← 0 thru 4 with step 1
Read coeff[i]
End for.

Step 4: [Input x (integer / float)] Read x

Step 5: [Initialize]

fx = 0

Step 6: [Repeat step 6 until i is greater than or equal to 0]

For i ← 4 thru 0 with step -1
fx = fx * x + coeff[i];
End for

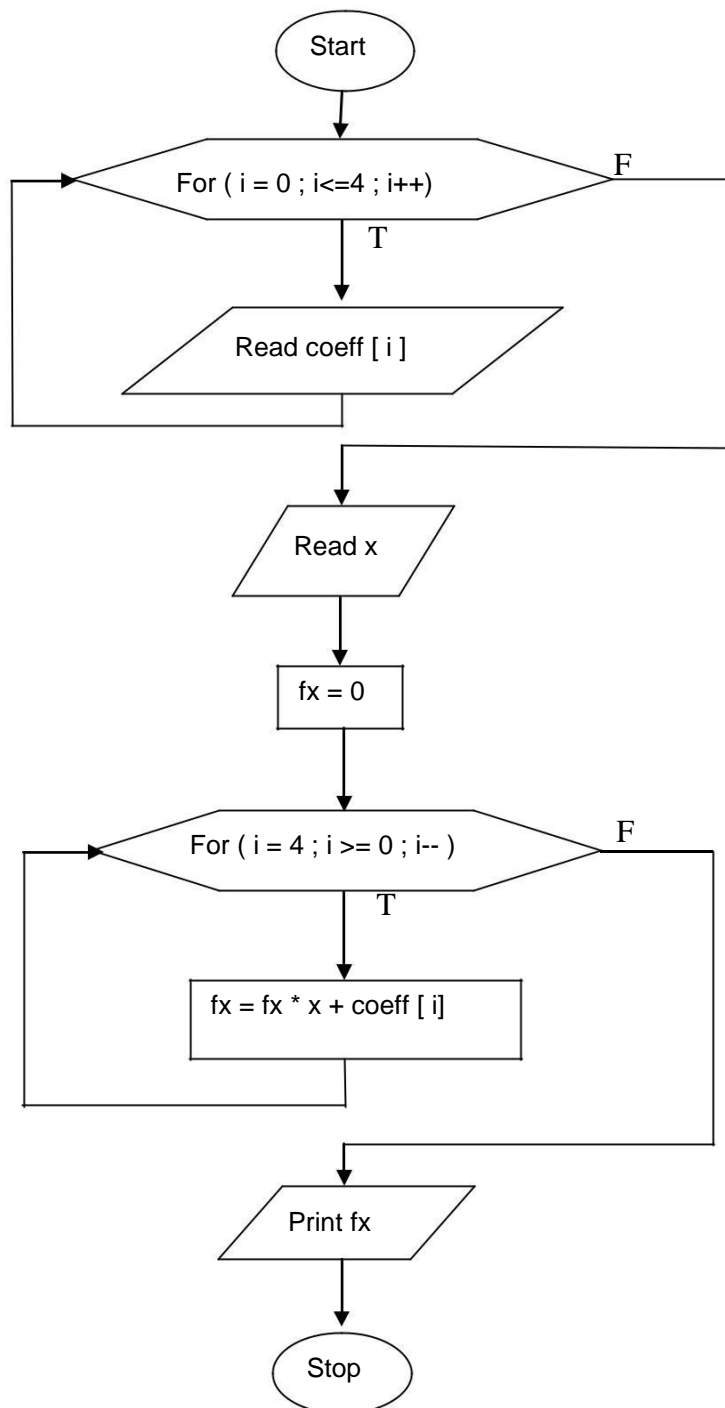
Step 7: [Output]

Print fx

Step 8: [Stop]

End

Flow chart



Program 4:

```
// Preprocessor directives
#include<stdio.h>
#include<conio.h>

// Main program
void main()
{
    // Variable declaration
    float coeff[10], x, fx=0;
    int i;
    clrscr();
    // Statements and expressions
    printf("\n C program to evaluate the given polynomial  $f(x) = a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0$ 
for"); printf("\n given value of x and the coefficients using Horner's method");
    printf("\n\n Enter the coefficients (integer/float) of a given polynomial (i.e., a0,a1,a2,a3,a4
    )...\n"); for(i = 0 ; i <= 4 ; i++)
    {
        printf("\n coeff [%d] = ",i);
        scanf("%f", &coeff[ i] );
    } //end for

    printf("\n Enter the value of x (integer or float )...:");
    scanf("%f",&x);

    for( i = 4 ; i >= 0 ; i--)
        fx = fx * x + coeff[ i];

    printf("\n\n f(%f) = %f",x,fx);
    getch();
} // end main
```

Sample Output

```
1.
Enter the coefficients (integer or float) of a given polynomial (i.e., a0,a1,a2,a3 and a4
)... coeff [0] = 1
coeff [1] = 2
coeff [2] = 3
coeff [3] = 4
coeff [4] = 5

Enter the value of x (integer or float )...: 2

f(2) = 129
```

Experiment No. 5: Draw the flowchart and Write C Program to compute **Sin(x)** using Taylor series approximation given by $\text{Sin}(x) = x - (x^3/3!) + (x^5/5!) - (x^7/7!) + \dots$. Compare the result with the built-in Library function and print both the results with appropriate messages.

Algorithm 5: To compute Sin(x) using Taylor series approximation

Step 1: [Start]

Begin

Step 2: [Read the value of x in degrees] Read x

Step 3: [Initialization & Radians

conversion] Temp=x

$x = x * (3.142/180.0)$

Term=x

Sinx=term

n=1

Step 4: [compute sin value]

Repeat While (term > FLT_EPSILON)

Fact $\leftarrow 2 * n * (2 * n + 1)$

Term $\leftarrow -\text{term} * x * x / \text{fact}$ Sinx

Sinx+term

n \leftarrow n+1

Step 5: [Output]

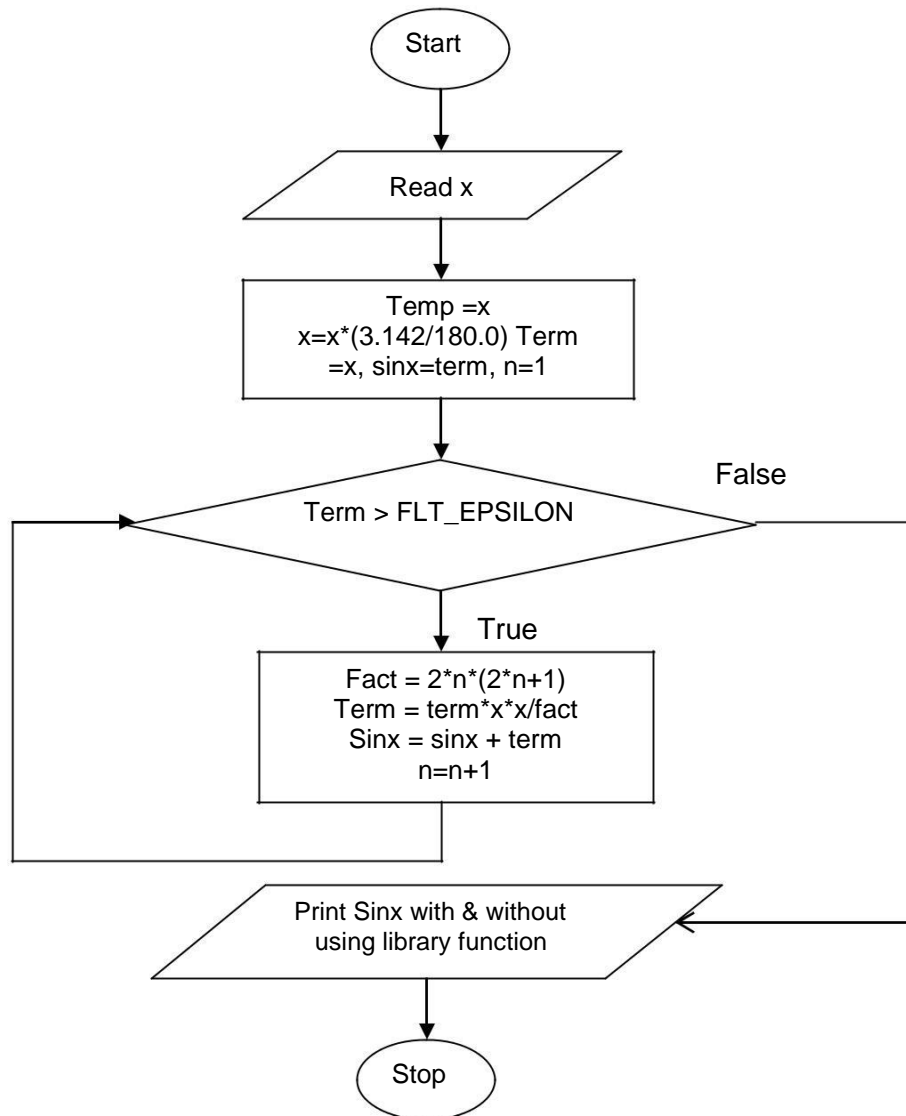
Print sin(x) \rightarrow without using library function Print

sin(x) \rightarrow with using library function

Step 6: [Stop]

End

Flow chart



Program 5:

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

void main()
{
    int n, temp;
    float term, fact, x, sinx=0;
    clrscr();
    printf("Enter the value of x (in degrees)\n");
    scanf("%f", &x);

    temp = x;

    x = x*(3.142/180.0);    /* Converting degrees to radians*/
    term=x;
    sinx=term;
    n=1;
    while(term>FLT_EPSILON)
    {
        fact = 2*n*(2*n+1);
        term = - term * x * x /fact;
        sinx = sinx + term;
        n = n + 1;
    }
    printf ("Sum of the sine series without using library function sin(%d)= %f\n", temp, sinx);
    printf ("Sum of the sine series with using library function sin(%d) = %f\n", temp, sin(x));
}
```

Sample output:

Enter the value of x (in degrees)

30

Sum of the sine series without using library function sin(30)= 0.499733

Sum of the sine series with using library function sin(30) = 0.500059

Enter the value of x (in degrees)

60

Sum of the sine series without using library function sin(60)= 0.855862

Sum of the sine series with using library function sin(60) = 0.866093

Experiment No. 6: Develop an algorithm, implement and execute a C program that reads **N** integer numbers and arrange them in ascending order using **Bubble Sort**.

Algorithm: Bubble sort technique to sort N integer numbers in to ascending order.

Step 1: [Start]

Begin

Step 2: [Input the number of elements in the array]

Read n

Step 3: [Input the n integer numbers]

For i ← 0 thru n-1 in steps of 1
1 Read num[i]
End for

Step 4: [Display the original array]

For i ← 0 thru n-1 in steps of 1
1 Print num[i]
End for

Step 5: [Repeat step 5 for i varies from 0 thru n-1]

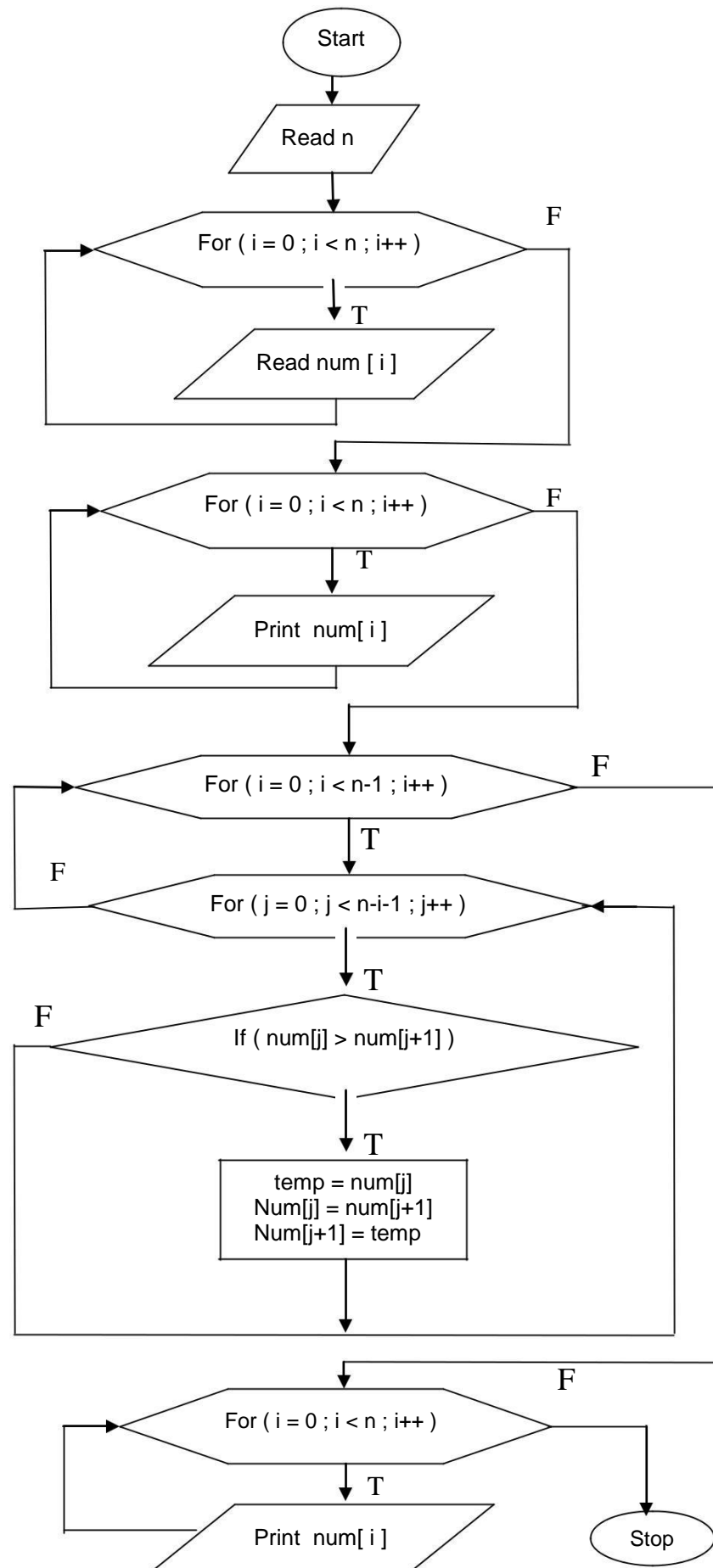
For i ← 0 thru n-1 in steps of 1
For j ← 0 thru n-i-1 in steps of 1
IF (num[j] > num[j + 1]) then
temp= num[j]
num[j] = num[j + 1]
num[j+1] = temp
End if
End for

End for

Step 6: [Output the sorted array]

For i ← 0 thru n-1 in steps of 1
1 Print num[i]
End for

Flow Chart



Program 6:

```
// Preprocessor directives
#include<stdio.h>
#include<conio.h>

// Main program
Void main()
{
    // Array and variable
    declaration int num[100], n, i, j,
    temp; clrscr();

    // Statements and expressions

    printf("\n C program to input N integer numbers into a single dimension array, sort them in");
    printf("\n to ascending order using BUBBLE SORT technique and print both the given");
    printf("\n array and the sorted array with suitable headings");

    printf("\n\n Enter the number of elements in the
    array...:"); scanf("%d",&n);
    printf("\n\n Enter %d integer
    numbers...",n); for(i = 0; i < n; i++)
    {
        printf("\n num[%d] → ",i);
        scanf("%d",&num[i]);
    }// end if

    // Display the input array
    printf("\n\n The unsorted/input array
    is...\n"); for(i = 0; i < n; i++)
        printf("%d\t",num[ i]);

    // Bubble sort
    for(i = 0; i < n-1; i++)                // outer loop
    {
        for(j = 0; j < (n-i-1); j++)        // inner loop
        {
            if( num[j] > num[j+1] )

                temp = num[j];
                num[j] = num[j+1];
                num[j+1] = temp;
            }// end if
        }// end for – inner loop
    }// end for – outer loop

    // Display the sorted array
    printf("\n\n The sorted / output array is...\n");
    for(i = 0; i < n; i++)
        printf( "%d\t",num[ i ] );

    getch();

} // end main
```

Sample Output

```
1.
Enter the number of elements in the array...: 5
Enter 5 integer numbers...
num[0] → 3
num[1] → 6
num[2] → 1
num[3] → 0
num[4] → 9
The unsorted/input array is...
3      6      1      0      9

The Sorted / output array is...
0      1      3      6      9
```

Experiment No. 7: Develop, implement and execute a C program that reads two matrices **A** ($m \times n$) and **B** ($p \times q$) and Compute the product **A** and **B**. Read matrix **A** and matrix **B** in row major order and in column major order respectively. Print both the input matrices and resultant matrix with suitable headings and output should be in matrix format only. Program must check the compatibility of orders of the matrices for multiplication. Report appropriate message in case of incompatibility.

Algorithm: Matrix multiplication

Step 1: [Start]

Begin

Step 2: [Input the order of matrix A and matrix B]

Read m, n, p, q

Step 3: [Check whether two matrices are multiplicable]

If ($n = p$) then go to step 4

Else

Print ("MULTIPLICATION NOT ALLOWED...TRY AGAIN")

Go to step 2

End if

Step 4: [Read matrix A]

For i ← 0 thru m-1 in steps of 1

For j ← 0 thru n-1 in steps of 1

Read A[i][j]

End for

End for

Step 5: [Read matrix B]

For i ← 0 thru p-1 in steps of 1

For j ← 0 thru q-1 in steps of 1 Read B[i][j]

End for

End for

Step 6: [Compute C[i][j]]

For i ← 0 thru m-1 in steps of 1

For j ← 0 thru q-1 in steps of 1 C[i][j] ← 0

j] ← 0

For k ← 0 thru n-1 in steps of 1

C[i][j] ← C[i][j] + A[i][k] * B[k][j]

End for

End for

End for

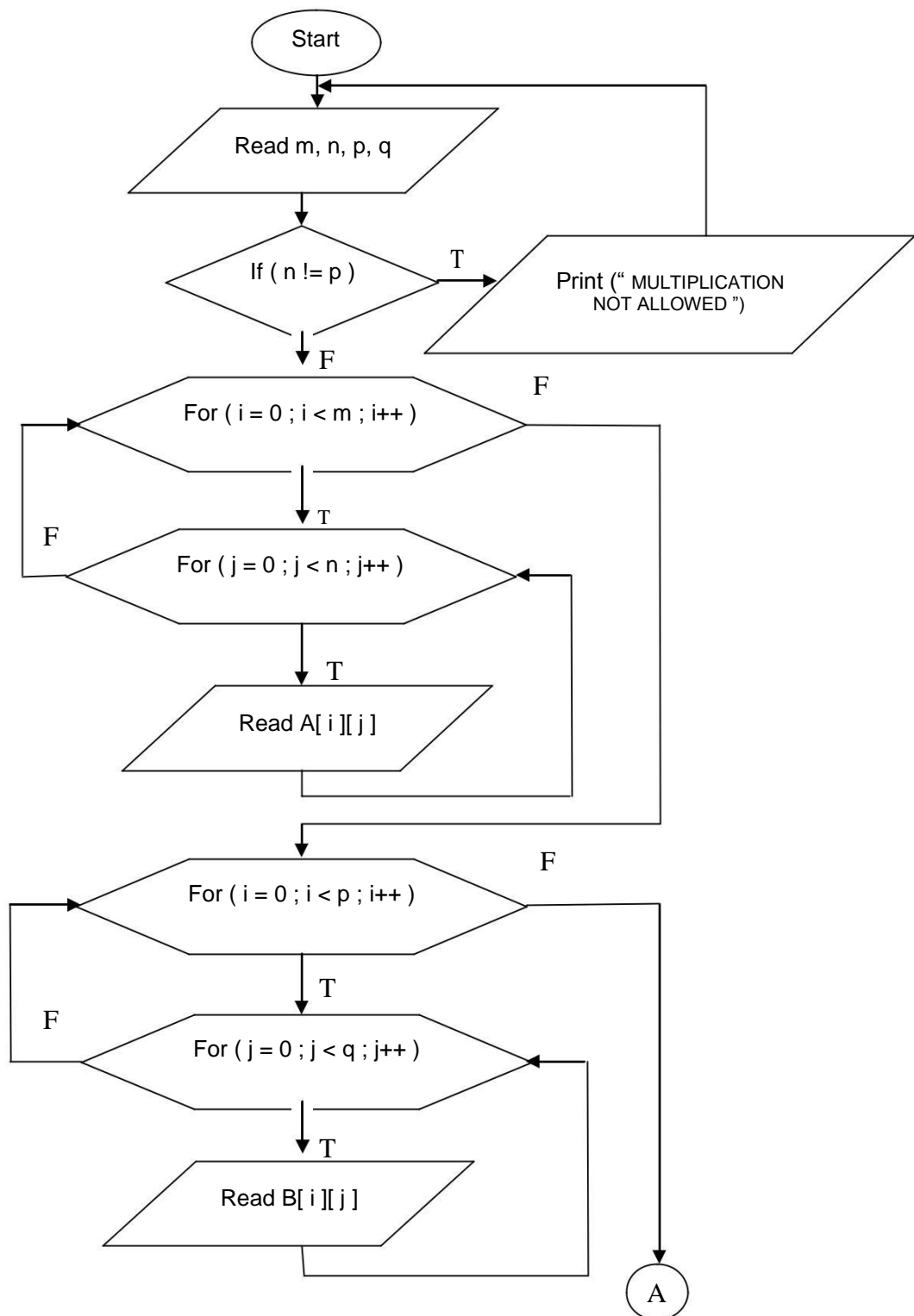
Step 7: [Output]

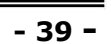
Print A[m][n], B[p][q], C[m][q]

Step 8: [Stop]

End

Flow Chart





Program 7:

```
// Preprocessor directives
```

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

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

```
// Main program
```

```
Void main()
```

```
{
```

```
    // Matrix declaration and variables
```

```
    declaration int A[5][5], B[5][5], C[5][5],
```

```
    m,n,p,q,i,j,k; clrscr();
```

```
    // Statements and declarations
```

```
printf("\n C program to read two matrices A(m x n) and B(p x q) and to compute the ");
printf("\n product of A and B if the matrices are compatible for multiplication. The program");
printf("\n is to print the input matrices and the resultant matrix with suitable headings and");
printf("\n format if the matrices are compatible for multiplication, otherwise the program");
printf("\n must print a suitable message");
```

```
printf("\n\n Enter the order of matrix A(m and
n)..."); scanf("%d%d", &m,&n);
```

```
printf("\n Enter the order of matrix B(p and q)...");
scanf("%d%d", &p,&q);
```

```
// Check compatibility of two
matrices if(n!=p)
```

```
{
```

```
    printf("Matrix multiplication is not possible...Try again");
```

```
    getch();
```

```
    exit(0);
```

```
}// end if
```

```
Else
```

```
// Read matrix A
```

```
printf("\n Enter the elements of matrix A(m *
n)... \n"); for(i=0;i<m;i++)
```

```
    for(j=0;j<n;j++)
```

```
        scanf("%d", &A[i][j]);
```

```
// Read matrix B
```

```
printf("\n Enter elements of matrix B(p * q)... \n");
```

```
for(i=0;i<p;i++)
```

```
    for(j=0;j<q;j++)
```

```
        scanf("%d", &B[i][j]);
```

```
// Matrix multiplication block
```

```
for(i=0;i<m;i++)
```

```
    { // Outer loop
```

```
        for(j=0;j<q;j++)
```

```
            { // Inner loop1
```

```
                C[i][j]=0;
```

```
                for(k=0;k<n;k++)
```



```
                { // Inner loop2 C[i][j]=
                  C[i][j] +A[i][k]*B[k][j]; }//
                end for
            }// end
        for }// end for

// Output

printf("\n\n The Matrix A
is..."); for(i=0;i<m;i++)
{
    for(j=0;j<n;j++) printf("%4d",
        A[i][j]);
    printf("\n");
}// end for

printf("\n\n The Matrix B
is..."); for(i=0;i<p;i++)
{
    for(j=0;j<q;j++) printf("%4d",
        B[i][j]);
    printf("\n");
}// end for

printf("\n\n The Resultant Matrix C is...");
for(i=0;i<m;i++)
{
    for(j=0;j<q;j++)
        printf("%4d", C[i][j]);
    printf("\n");
}// end for
getch();

}// end main
```

Sample Output

1.

Enter the order of matrix A(m and n)...2 2

Enter the order of matrix B(p and q)...3 2

Matrix multiplication is not possible...Try again

2.

Enter the order of matrix A(m and n)...2 2

Enter the order of matrix B(p and q)...2 2

Enter the elements of matrix A(m * n)...

1

2

3

4

Enter elements of matrix B(p * q)...

2

3

4

5

The Matrix A is...

1 2

3 4

The Matrix B is...

2 3

4 5

The Resultant Matrix C is...

10 13

22 29

Experiment No 8: Develop, implement and execute a C program to search a Name in a list of names using **Binary searching** Technique.

Algorithm: Binary search to search a given name in list of names & report success or failure.

Step 1: [Start]

Begin

Step 2: [Input the total number of elements in the list of names]

Read n

Step 3: [Input the n names]

For i \leftarrow 0 thru n-1 in steps of 1

Read name [i]

End for

Step 4: [Input the name to be searched]

Read key

Step 5: [Initialize]

low \leftarrow 0, high \leftarrow n -1

Step 6: [Repeat step 6 thru step 7 until (low <=

high)] mid \leftarrow (low + high) / 2

Step 7: [search for the given name]

If (strcmp(key , name[mid])==0)

then Found = 1, go to step 8;

Else if (strcmp(key, name [mid])>0)

Low \leftarrow mid + 1

Else

high \leftarrow mid - 1

End if

Step 8: [Output]

If (Found = 1) then

Print("Successful Search")

Else

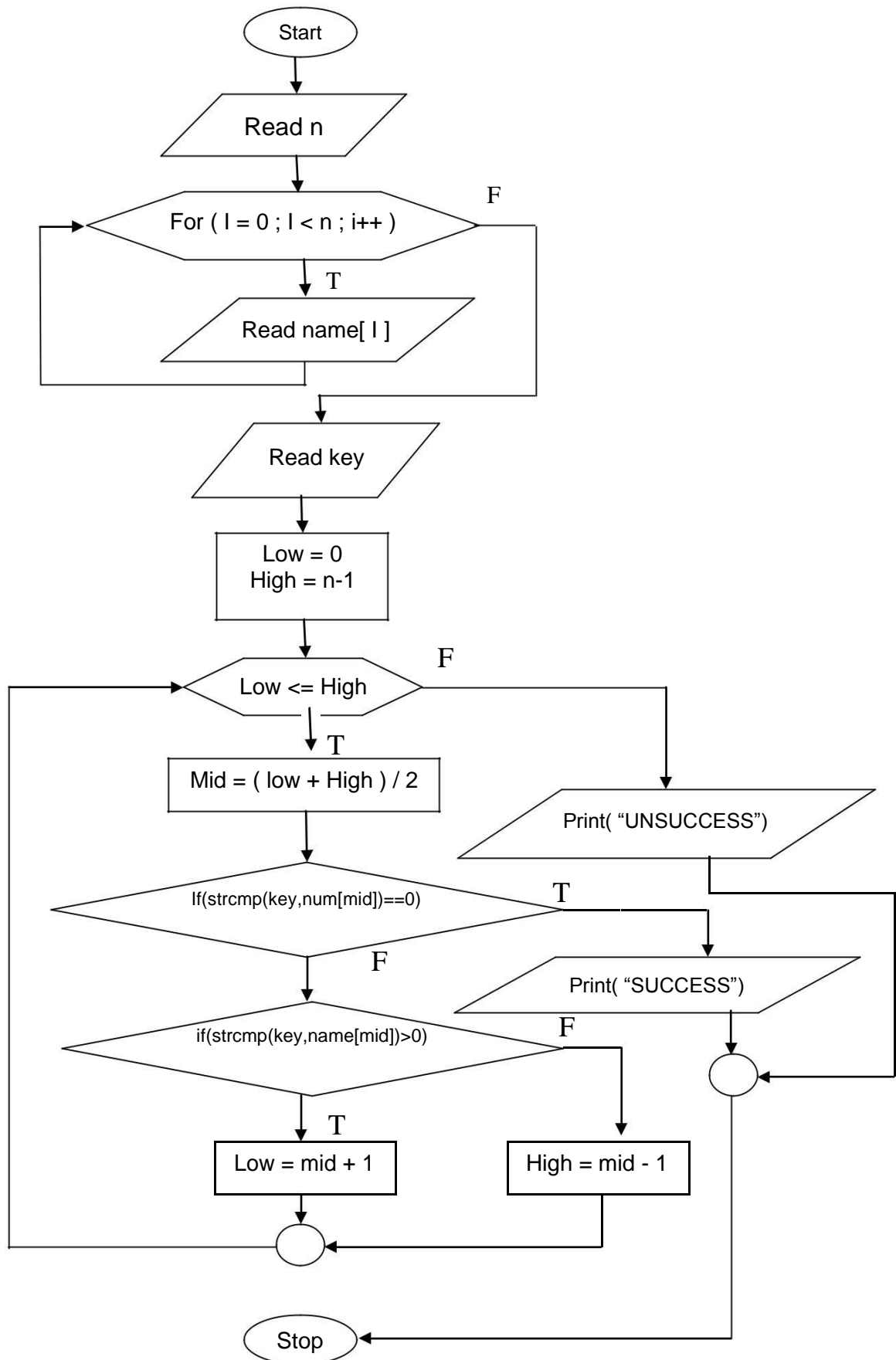
Print("Successful Search")

Endif

Step 9: [Stop]

End

Flowchart



Program 8:

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#include<stdlib.h>
// Main program
void main()
{
// variable decleration
    int n, i, mid, low, high;
    char name[100][100], key[100];
    clrscr();
// Statements and expressions
    printf("\n C program to search a name in list of names using binary search");
    printf("\n Enter the size of the array...:");
    scanf("%d", &n);
    printf("\n Enter %d names in alphabetical order only", n);
// Reading names
    for(i = 0; i < n; i++)
    {
        printf("\n\n name[%d]:",i);
        scanf("%s", name[i]);
    }// end for
    printf("\n\n Enter the name to be searched:");
    scanf("%s", key);
    low=0;
    high=n-1;
// Binary search
    while ( low <=high)
    {
        mid = (low+high)/2;
        if(strcmp(key,name[mid])==0)
            break;           // Successful search
        else
            if(strcmp(key,name[mid])>0)
                low = mid+1; // Initialize low to 1st element after middle
            else
                high = mid-1; // Initialize high to element one less than middle
    }// end while

    if(strcmp(key,name[mid])==0)
        printf("\n\n Search Successful:%s found", key);
    else
        printf("\n\n Search Unsuccessful:%s not found", key);
    getch();
}// end main
```

Sample Output

1.
Enter the size of the array...: 3
Enter 3 names in alphabetical order only

asha
pallavi
yamuna

Enter the name to be searched: yamuna

Search Successful: yamuna found

2.
Enter the size of the array...: 2
Enter 2 names in alphabetical order only

cit
college

Enter the name to be searched: gubbi

Search Successful: gubbi not found

9: Write and execute a C program that

- a. Implements string copy operation **STRCOPY**(str1,str2) that copies a string *str1* to another string *str2* without using library function.
- b. Reads a *sentence* and prints frequency of each of the vowels and total count of consonants.

Algorithm 9 a: To implement string copy operation to copy str1 to str2

Step 1: [Start]

Begin

Step 2: [Input any string]

Read str1

Step 3: [Copy str1 to str2]

For i ← 0 thru str1 not equal to NULL in steps of 1
Str2[i]=Str1[i]

End For

Step 4: [Initialize Null at end of str2 after copying]

Str2='\0'

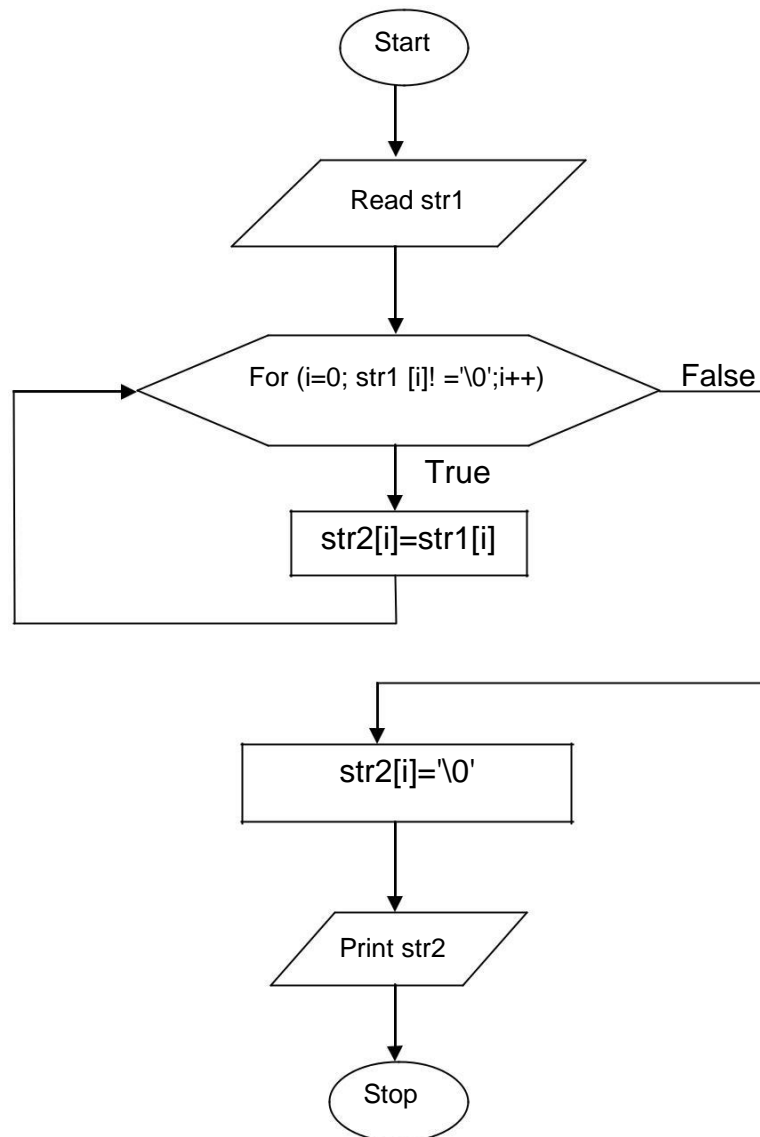
Step 5: [Output]

Print str2

Step 6: [Stop]

End

FLOW CHART



Program 9 a:

```
#include<stdio.h>
#include<conio.h>
void main()
{
    char str1[100],str2[100];
    int i;
    clrscr();
    printf("Enter any string str1: ");
    scanf("%s",str1);
    for(i=0;str1[i]!='\0';i++)
        str2[i]=str1[i];
    str2[i]='\0';
    printf("After copying str1 to str2, the value of str2 is: %s",str2);
    getch();
}
```

Sample Output:

Enter any String Str1: cse

After copying str1 to str2, the value of str2 is: cse

ii.

Algorithm 9b: To find frequency of each of the vowels and total count of consonants.

Step 1: [Start]

Begin

Step 2: [Input any string]

Read a string

Step 3: [Initialize all count variables:]

a=0,e=0,i=0,o=0,u=0,con=0

Step 4: [Increment the count variables]

```
repeat While ((c=getchar())!='\n')
if (c=='a' || c=='A') a=a+1;
if (c=='e' || c=='E') e=e+1;
if (c=='i' || c=='I') i=i+1;
if (c=='o' || c=='O') o=o+1;
if (c=='u' || c=='U') u=u+1;
if((c>='a'&&c<='z') || (c>='A'&&c<='Z')) con++;
```

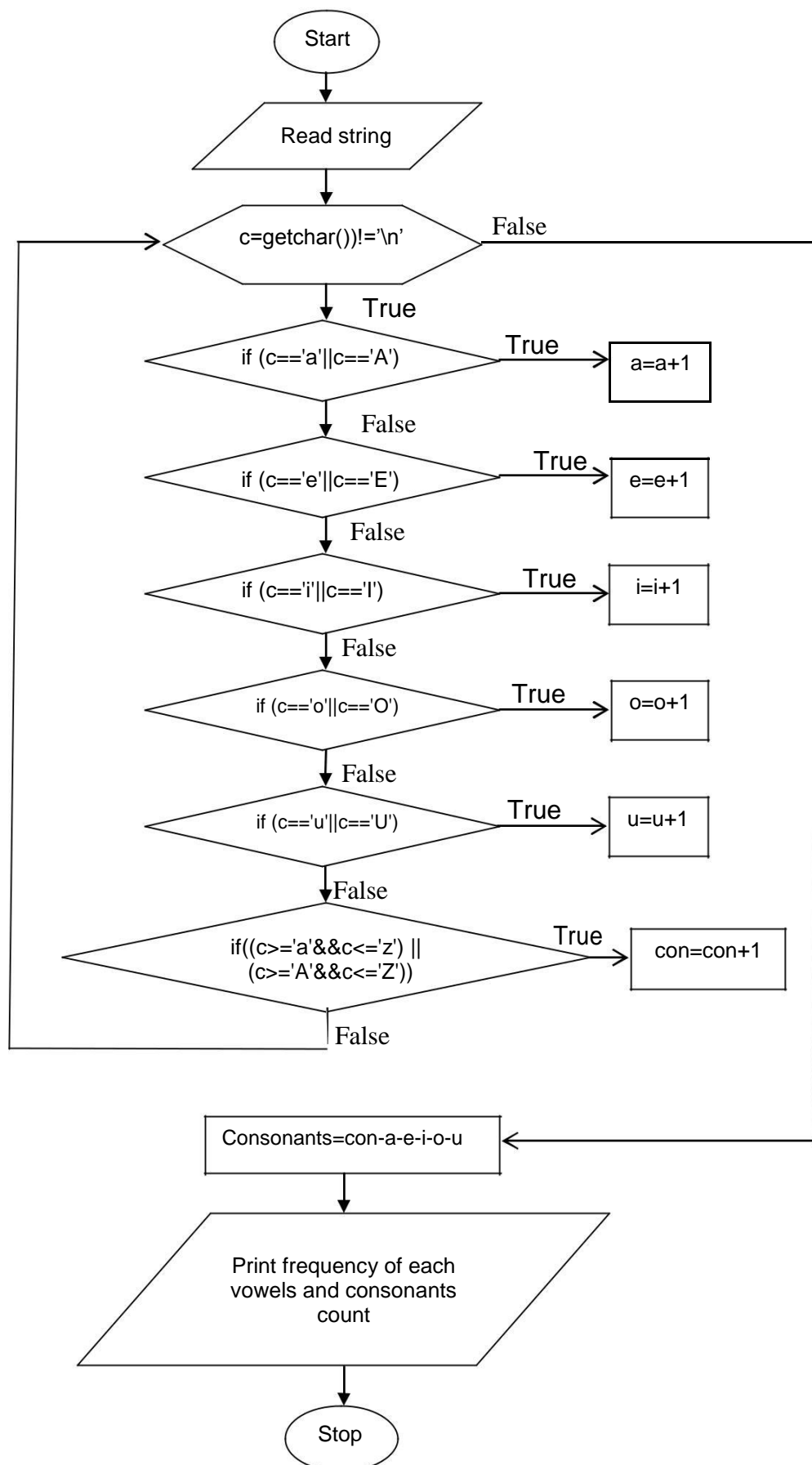
Step 5: [Output]

print a, e, i, o, u count and consonants count

Step 6: [Stop]

End

FLOW CHART



Program 9 b:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a=0,e=0,i=0,o=0,u=0,con=0,
consonants; char c,line[100];
clrscr();
printf("\n Enter the sentence:");
printf("Enter a line of string:\n");
setbuf(stdin,NULL);

while((c=getchar())!='\n')
{
if (c=='a' || c=='A')
a=a+1;
if (c=='e' || c=='E')
e=e+1;
if (c=='i' || c=='I')
i=i+1;
if (c=='o' || c=='O')
o=o+1;
if (c=='u' || c=='U')
u=u+1;
if ((c>='a' && c<='z') || (c>='A' && c<='Z'))
con++;
}
consonants=con-a-e-i-o-u;

printf("Frequency of vowel 'a' is %d.\n",a);
printf("Frequency of vowel 'e' is %d.\n",e);
printf("Frequency of vowel 'i' is %d.\n",i);
printf("Frequency of vowel 'o' is %d.\n",o);
printf("Frequency of vowel 'u' is %d.\n",u);
printf("Consonants count is %d.\n", consonants);
getch();
}
```

Sample output

Enter the sentence: welcome to cit

Frequency of vowel 'a' is 0.

Frequency of vowel 'e' is 2.

Frequency of vowel 'i' is 1.

Frequency of vowel 'o' is 2.

Frequency of vowel 'u' is 0.

Consonants count is 7.

Experiment No 10:

a. 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.

b. Design and develop a C function **isprime**(num) that accepts an integer argument and returns 1 if the argument is prime, a 0 otherwise. Write a C program that invokes this function to generate prime numbers between the given range.

Algorithm 10a: A function rightshift(x, n) returns the value of the integer x rotated to the right by n bit positions.

Step 1: [Start]

Begin

Step 2: [Input the unsigned integer X, and N ($N > 0$)

] Read x, n

Step 3: [Invoke function rightshift(x, n)]

$X1 \leftarrow$ rightshift(x, n)
return

Step 4: [Output]

Print $X1$

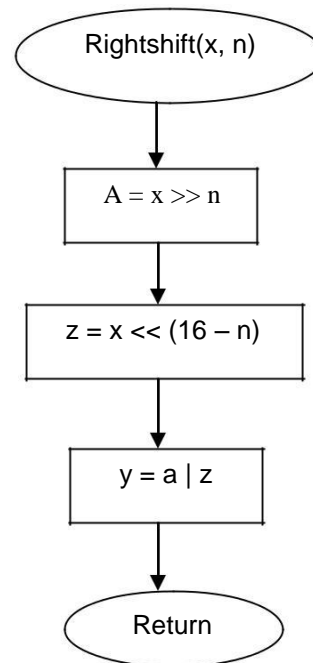
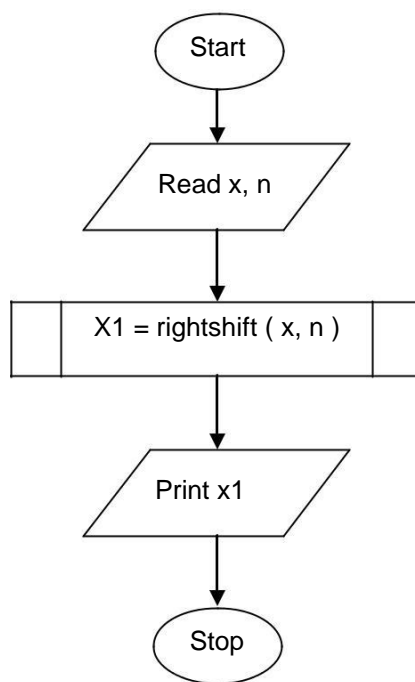
Step 5: [Stop]

End

Pseudo code: function rightshift(x, n)

Compute $a \leftarrow x \gg n$
Compute $z \leftarrow x \ll (16 - n)$
Compute $y \leftarrow a | z$
Return y

Flow chart



Program 10 a:

```
// Preprocessor directives
#include<stdio.h>
#include<conio.h>

Unsigned int rightshift(unsigned int, unsigned int); // function prototype

// Main program
void main()
{
    // Variable declaration and initialization
    unsigned int x,x1,n;
    clrscr();

    // Statements and expressions
    printf("\n C program to develop a function rightshift(x,n) in C that returns the value of the");
    printf("\n integer x rotated to the right by n bit positions as an unsigned integer. Invoke the");
    printf("\n function from the main with different values for x and n and print the results with");
    printf("\n suitable headings");

    printf("\n Enter the value of x (unsigned integer only)...");
    scanf("%u",&x);
    printf("\n\n Enter the no of bits x will rotate right (n)...");
    scanf("%u",&n);
    x1 = rightshift(x,n); // function call
    printf("\n\n After rotating %d bits to the right, %u become → %u", n, x, x1);
    getch();

} // end main

// Function definition
unsigned int rightshift(unsigned int x,unsigned int n)
{
    unsigned int y,z,a;
    a=x>>n;
    z=x<<(16-n);
    y=a|z;
    return(y);
} // end rightshift
```

Sample Output

Enter the value of x (unsigned integer only)...5

Enter the no of bits x will rotate right (n)...4

After rotating 4 bits to the right, 5 become → 20480

ii.

Algorithm 10b: To determine the given integer number is prime or not and to generate prime numbers between the given range.

Step 1: [Start]

Begin

Step 2: [Input the range of integer
number] Read a,b

Step 3: [Invoke function isprime(num)]

For i ← a thru i ≤ b in steps of 1

res ← isprime(num)

Step 4: [Output]

If (res = 1)

Print("Prime numbers between range a and b")

Else

Print("No prime numbers exists")

Step 5: [Stop]

End

Pseudo code

Function isprime(num)

For i ← 2 thru num / 2 in steps of 1

{

If ((num mod i) = 0) then

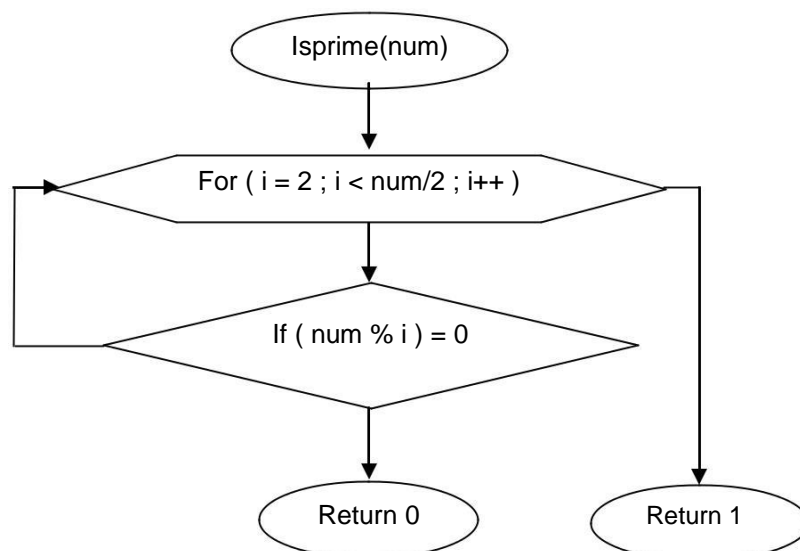
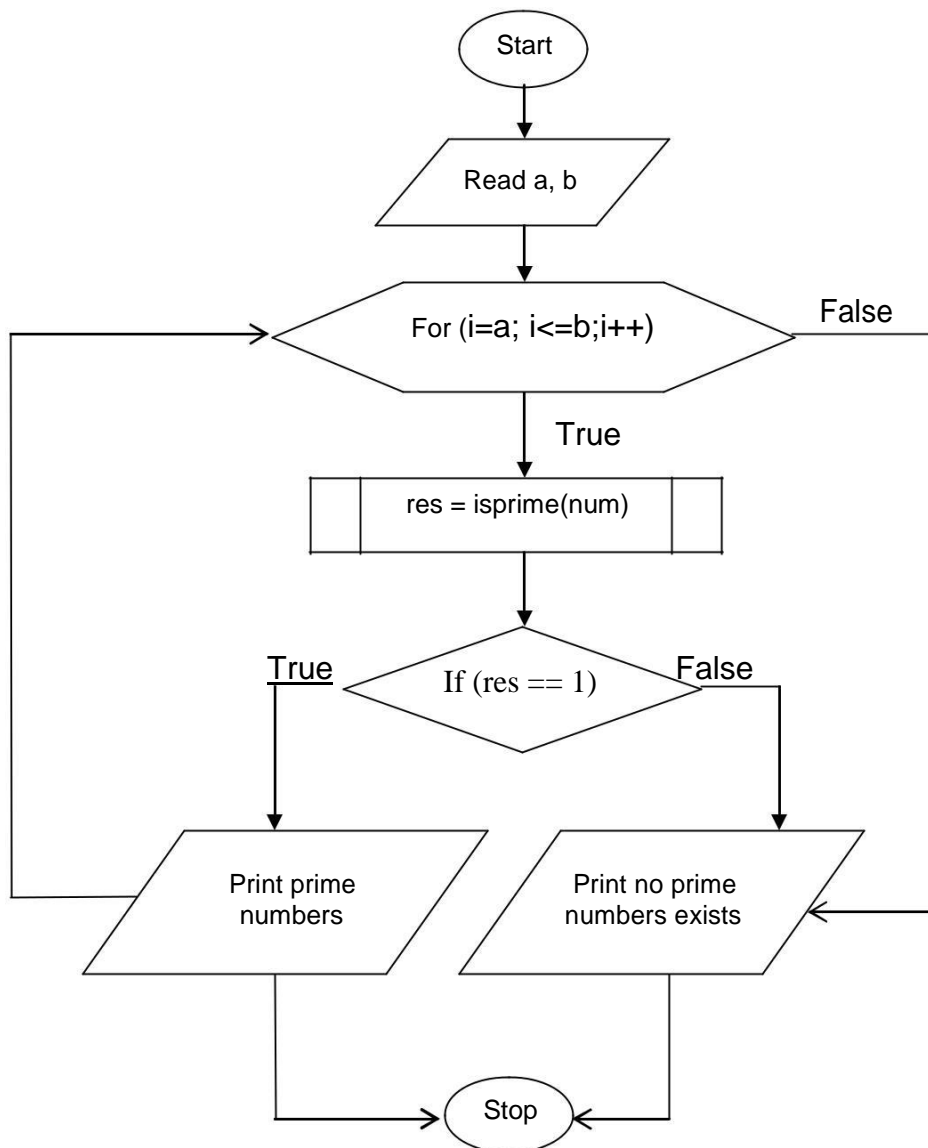
Return 0

}

Return

1 End For

Flow chart



Program 10b:

```
// Preprocessor directives
# include<stdio.h>
# include<conio.h>
int isprime(int);           // function prototype
// Main program
void main()
{
    // Variable
    declaration int num,
    res,a,b,i,f=0; clrscr();
    // Statements and expressions
    printf("\n C program to develop a function isprime(x) that accepts an integer argument
    and"); printf("\n returns 1 if the argument is prime and 0 otherwise");
    printf("\n It that invokes this function to generate prime numbers between the given range");

    printf("\n Enter the range of integer number to generate prime numbers");
    scanf("%d%d",&a,&b);
    for(i=a; i<=b;i++)
    {
        res = isprime(i);      // function call
        if ( res == 1)
        {
            printf("\t%d",i);
            f=1;
        }
    } // end for
    if(f==0)
        printf("No prime numbers exists\n");
    getch();
} // end main
// Function definition
int isprime(int n)
{
    int j;
    for(j = 2 ; j<= n / 2 ;j++)
    {
        if ( n % j == 0)
            return 0;          // not a prime
    }
    return 1;                  // a prime
} // end isprime
```

Sample output

Enter the range of integer number to generate prime
numbers 1 10 1 2 3 5 7

Experiment 11: Draw the flowchart and write a *recursive* C function to find the factorial of a 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 coefficient. Tabulate the results for different values of n and r with suitable messages.

Algorithm 11: To determine factorial of a number and compute the binomial coefficient using factorial function.

Step 1: [Start]

Begin

Step 2: [Input the

number] Read n, r

Step 3: [Invoke function factorial and

binomialCoefficient] Factorial (n)

binomialCoeff(n, r)

Step 4: [Output]

Print factorial and binomialCoefficient

Step 5: [Stop]

End

Pseudo code

Function factorial (n)

if($n \leq 0$)

return 1;

else

return $n * \text{factorial}(n -$

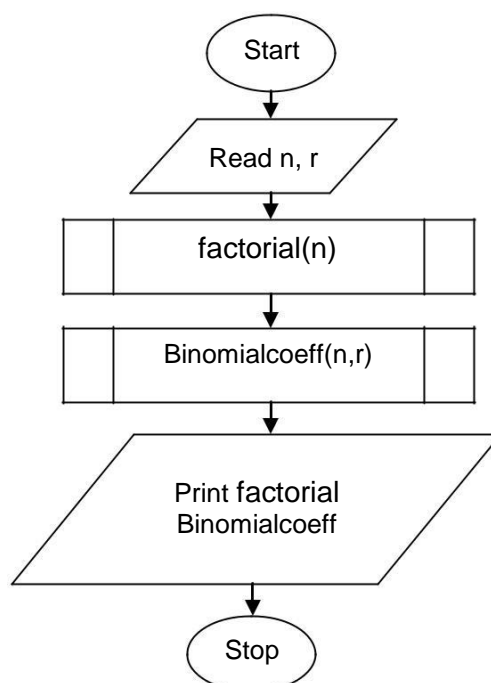
1); End if

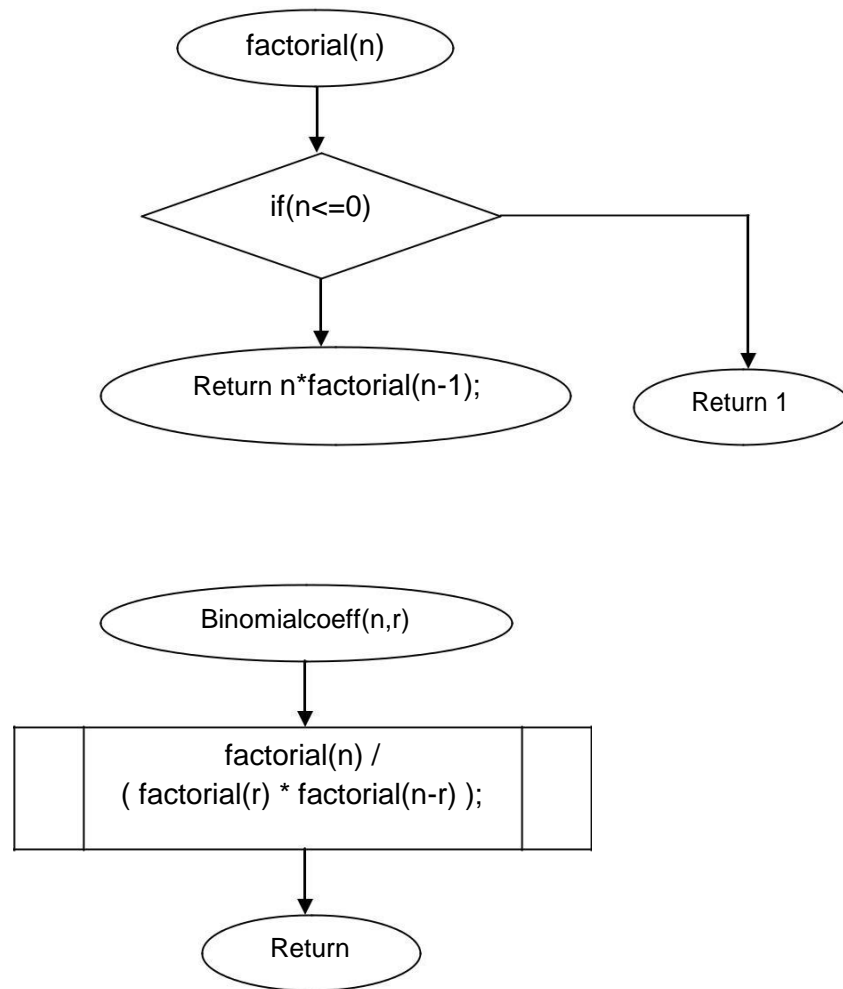
Pseudo code

Function binomialCoeff(n, r)

return $\text{factorial}(n) / (\text{factorial}(r) * \text{factorial}(n - r))$;

Flow chart





Program 11:

```
#include<stdio.h>
#include<conio.h>
int factorial(int n);
float binomialCoeff(const float n, const float r)
; void main()
{
    int n,r,fact;
    clrscr();
    printf("Enter the value of n & r: ");
    scanf("%d%d",&n,&r);
    printf("Factorial of %d = %d", n, factorial(n));
    printf("\nBinomial coefficient =%f",binomialCoeff(n,r));
    getch();
}
int factorial(int n)
{
    if(n<=0)
    return 1;
    else
    return n*factorial(n-1);
}

float binomialCoeff(const float n, const float r)
{
    return factorial(n) / ( factorial(r) * factorial(n-r) );
}
```

SAMPLE OUTPUT:

```
Enter the value of n &
r: 6 3
Factorial of 6 = 720
Binomial coefficient = 20.0000
```

Experiment No. 12: Given two university information files “**studentname.txt**” and “**usn.txt**” that contains students Name and USN respectively. Write a C program to create a new file called “**output.txt**” and copy the content of files “**studentname.txt**” and “**usn.txt**” into output file in the sequence shown below. Display the contents of output file “**output.txt**” on to the screen.

Student Name	USN
Name 1	USN1
Name 2	USN2
.....
.....



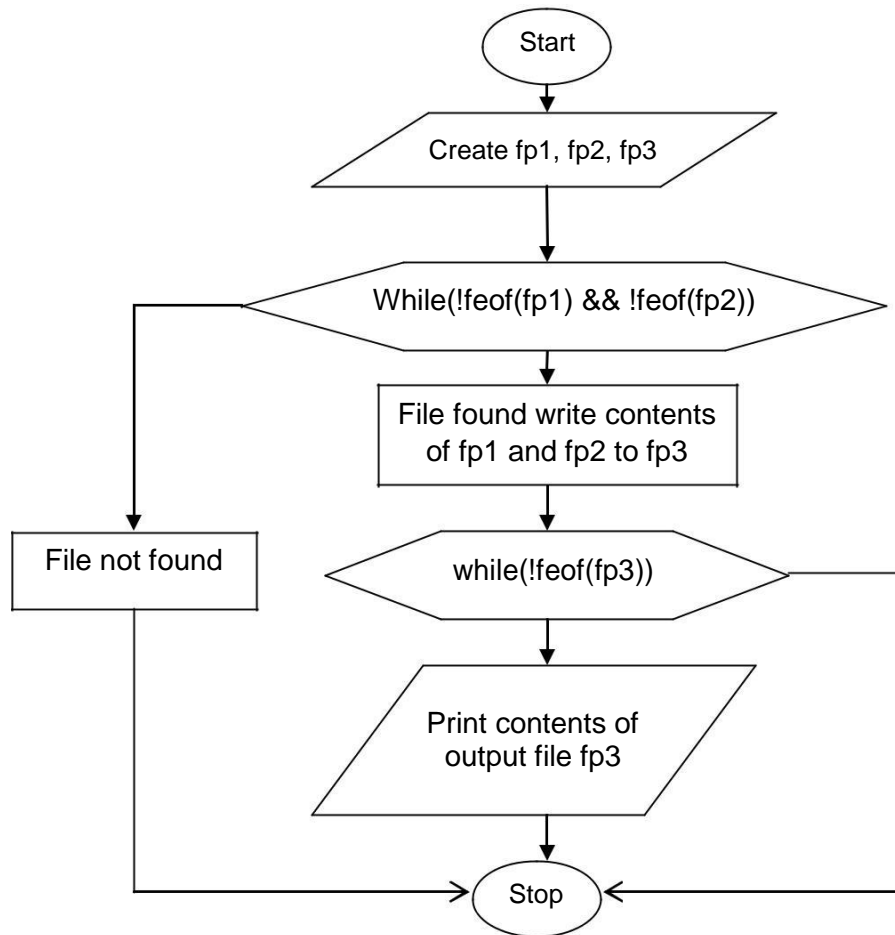
Note : Students are required to create two files “**studname.txt**” and **studusn.txt**” with the Contents

studname.txt	studusn.txt
Asha	1CG14CS001
Bharath	1CG14CS002
Uma	1CG14CS003
Shilpa	1CG14CS004

Algorithm 12: To copy the content of files “**studentname.txt**” and “**usn.txt**” into output file and display the contents of output file “**output.txt**” on to the screen.

```
Step 1: [Start]
        Begin
Step 2: [Create the
        files] fp1,fp2,fp3
Step 3: If files not created print file not
found Step 4: [Repeat]
        while( !feof(fp1) && !feof(fp2) )
            write contents of fp1 and fp2 to fp3
Step 4: [Output]
        while(!feof(fp3))
            Print contents of output file fp3
Step 5: [Stop]
        End
```

Flow chart



Program 12:

```
#include <stdio.h>
#include <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();
}
```

SAMPLE OUTPUT:

Name	USN
Asha	1CG14CS001
Bharath	1CG14CS002
Uma	1CG14CS003
Shilpa	1CG14CS004

Experiment No. 13: Write a C program to maintain a record of “n” student details using an array of structures with four fields (Roll number, Name, Marks, and Grade). Assume appropriate data type for each field. Print the marks of the student, given student name as input.

Algorithm 13: To maintain a record of “n” student details with four fields and print marks of the student given student name as input

Step 1: [Start]
Begin

Step 2: [Input the number of students]
Read n

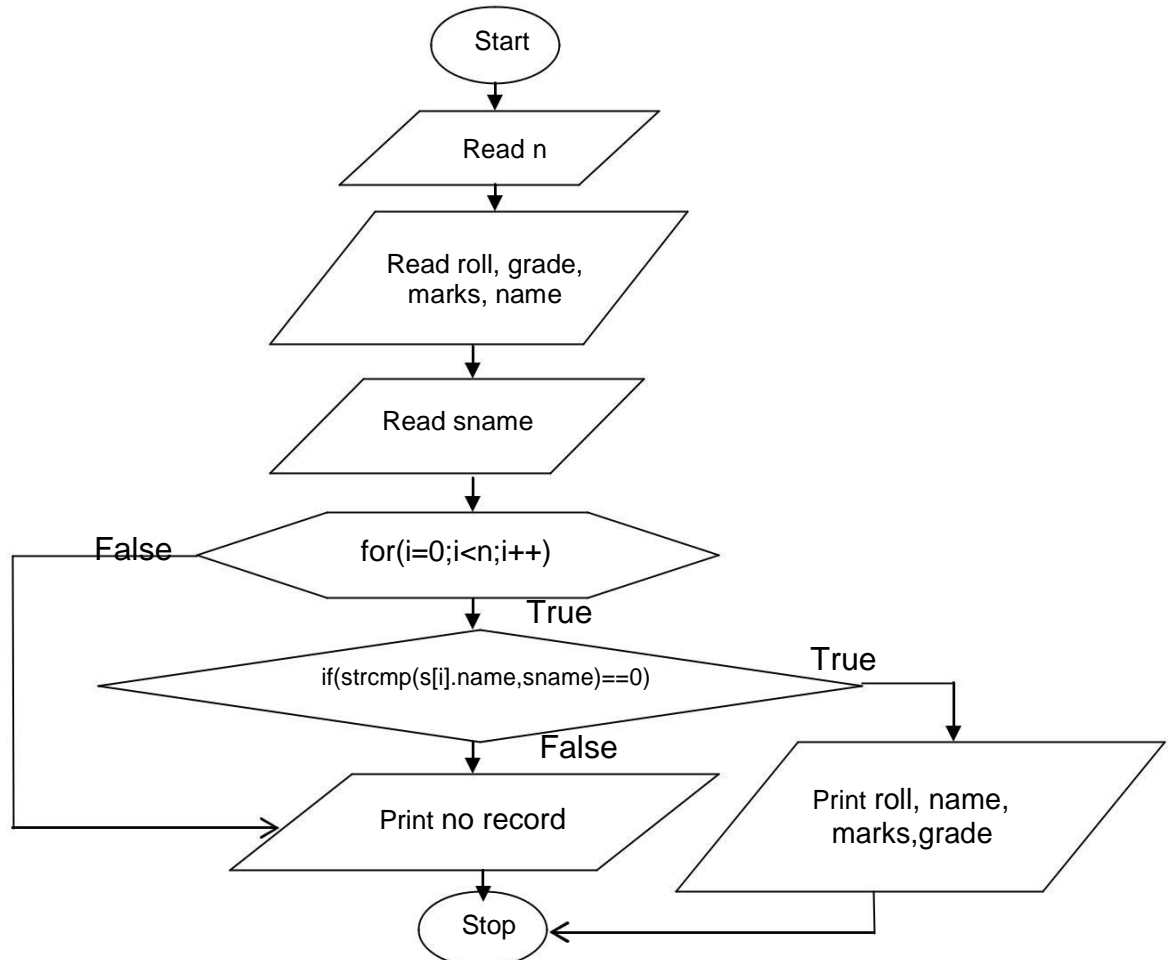
Step 3: [Input roll number,name,marks,grade]
Read roll,grade,marks,name

Step 4: [Input name of student whose marks to be displayed]
Read sname

Step 5: [Search student details]
For i ← 0 thru n in steps of 1
If (strcmp (sname, s[i].name)==0)
Print roll, name, grade marks
Else
Print record not found

Step 6: [Stop]
End

Flow chart



Program 13:

```
#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("\nMarks of the student is : %d",s[i].marks); found=1;
        }
    }
    if(found ==0)
        printf("\nGiven student name not found\n");
    getch();
}
```

SAMPLE 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: archana
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: yamuna
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:
bharghavi enter the marks : 45
enter the grade : C

Roll	Name	Marks	Grad
100	archana	90	A
101	yamuna	50	B
102	bharghavi	45	C

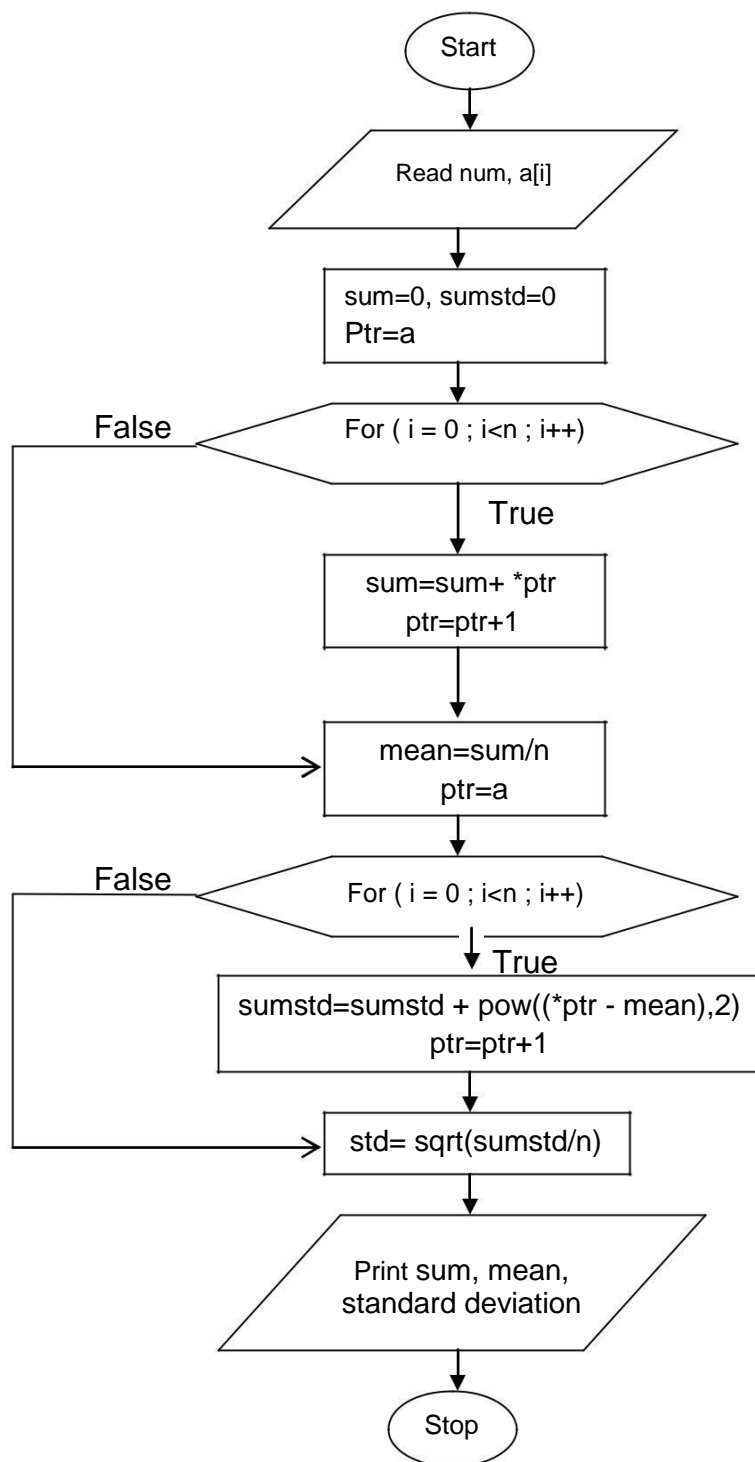
Enter the student name to print the marks: yamuna
Marks of the student is: 50

Experiment No. 14: 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.

Algorithm 14: To compute the sum, mean and standard deviation of all elements stored in an array of **n** real numbers.

```
Step 1: [Start]
        Begin
Step 2: [Input the elements]
        Read n, a[i]
Step 3: [Initialize] sum=0,
        sumstd=0
Step 4: [Compute sum, mean, standard
        deviation] Ptr=a
        For i ← 0 thru n in steps of 1
            Sum=sum+*ptr
            Ptr++
            Mean=sum/n
            Ptr=a
        For i ← 0 thru n in steps of 1
            Sumstd=sumstd+pow((*ptr-mean),2)
            Ptr++
        std=
        sqrt(sumstd/n)
Step 5: [Output]
        Print sum, mean, standard deviation
Step 6: [Stop]
        End
```

Flowchart



Program 14:

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
void 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();
}
```

SAMPLE 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

VIVA-VOCE Questions

1. What is a computer system?
2. What is hardware and software?
3. What is integrated circuit? Explain.
4. What is the function of cpu?
5. What is a chip set?
6. What is system bus?
7. What are input and output devices?
8. What is an operating system?
9. What is an algorithm?
10. What is flow chart?
11. What are identifiers, keywords, constants ,variables in c?
12. What are data types? List the data types available in c?
13. What is type casting?
14. What is the purpose of continue, goto and break statement in c?
15. What are loops? What is the difference b/w pretest and posttest loops?
16. What are functions?
17. Give the difference b/w call by value and call by reference?
18. Name few built in functions in c.
19. What are header files?
20. What are recursive functions?
21. What is an array?
22. What is a string?
23. What is actual and formal parameter?
24. What are operators? Mention the different types of operators.
25. What is the purpose of getch(),getchar(),gets(),putchar(), puts(),scanf(), printf() functions?
26. What are local and global variables?
27. What is a pointer?
28. What is LAN, WAN, MAN?
29. Differentiate: primary memory and secondary memory, RAM and ROM,ROM and PROM.
30. What is a plotter?
31. Which are the primary storage and secondary storage devices?
32. What is sorting? List different sorting techniques, which is the best sorting method?
33. What is searching? List different types of searching. Which are the best searching technique?
34. Name some of the string handling functions.
35. What is the minimum and maximum value of int data type?
36. What is parallel computing?
37. What is a thread?