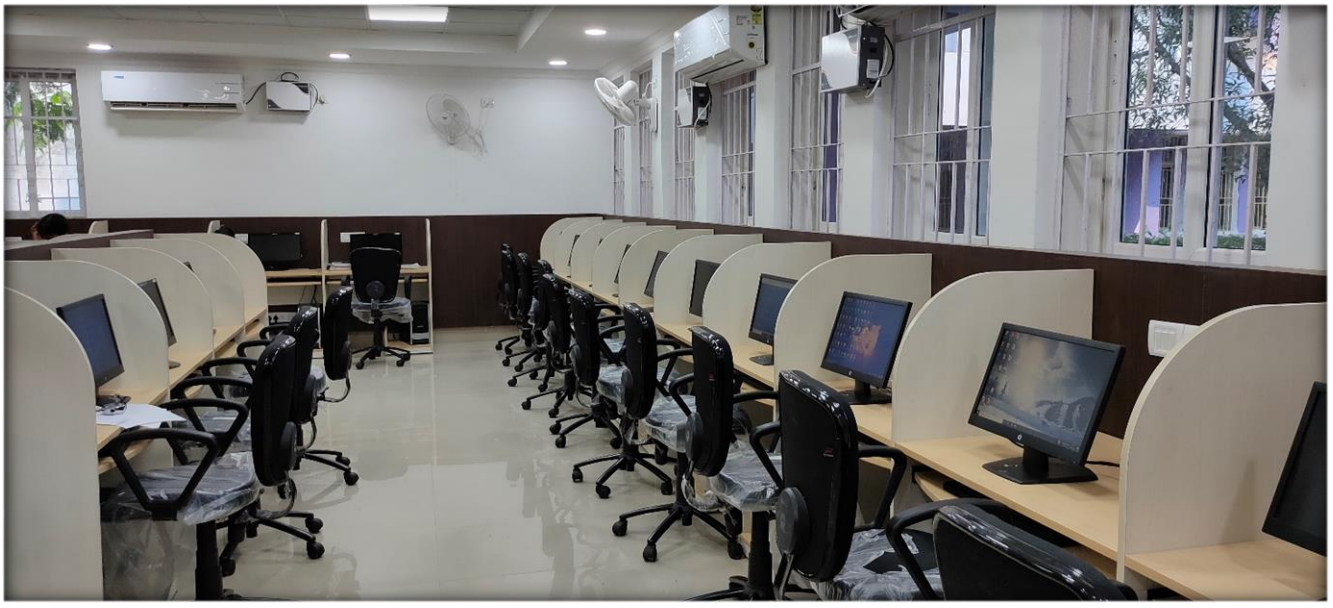
E-Learning Material

on

Computer Application

of

1st/2nd semester of all Engineering Branches



State Council for Technical Education and Vocational Training, Odisha

Bhubaneswar-751012

E-Learning Material

Computer Application

of

1st/2nd semester of all Engineering Branches of Diploma courses of SCTE&VT, Odisha

Contents Written by

Mr. Ajay Ku. Panda, Lecturer CSE, BOSE, Cuttack
Ms. D. Susmita, Lecturer CA, BOSE, Cuttack
Mr. Ajit Ku. Behera, Lecturer CSE, BOSE, Cuttack

Reviewed and validated by

Ms. Kalpana Panigrahi, Sr. Lecturer CA, GP, Bhubaneswar
Ms. Swetalina Das, Lecturer CA, GP, Bhubaneswar

Published by
SCTE&VT, Odisha,
Bhubaneswar-12
https://sctevtodisha.nic.in/en/
secretarysctevt@gmail.com,material.sctevt@gmail.com

Syllabus

Th.1b. COMPUTER APPLICATION
(1st / 2nd sem Common)

1. COMPUTER ORGANISATION

Introduction to Computer Evolution of Computers Generation of Computers Classification of Computers
Basic Organisation of Computer (Functional Block diagram) Input Devices, CPU & Output Devices.
Computer Memory and Classification of Memory

2. COMPUTER SOFTWARE

Software concept, System software, Application software
Overview of Operating System Objectives and Functions of O.S ,
Types of Operating System: Batch Processing, Multiprogramming, Time Sharing OS Features of DOS, Windows and UNIX
Programming Languages Compiler, interpreter Computer Virus
Different Types of computer virus
Detection and prevention of Virus
Application of computers in different Domain

3. COMPUTER NETWORK AND INTERNET

Networking concept, Protocol, Connecting Media, Date Transmission mode
Network Topologies, Types of Network
Networking Devices like Hub, Repeater, Switch, Bridge, Router, Gateway & NIC
Internet Services like E-Mail, WWW, FTP, Chatting, Internet Conferencing, Electronic Newspaper & Online Shopping
Different types of Internet connectivity and ISP

4. FILE MANAGEMENT AND DATA PROCESSING

Concept of File and Folder
File Access and Storage methods. Sequential, Direct, ISAM
Data Capture, Data storage
Data Processing and Retrieval

5. PROBLEM SOLVING METHODOLOGY

Algorithm, Pseudo code and Flowchart Generation of Programming Languages Structured Programming Language
Examples of Problem solving through Flowchart

6. OVERVIEW OF C PROGRAMMING LANGUAGE

Constants, Variables and Data types in C Managing Input and Output operations.
Operators, Expressions, Type conversion & Typecasting
Decision Control and Looping Statements (If, If-else, If-else-if, Switch, While, Do-while, For, Break, Continue & Goto)
Programming Assignments using the above features.

7. ADVANCED FEATURES OF C

Functions and Passing Parameters to the Function (Call by Value and Call by Reference) Scope of Variables and Storage Classes
Recursion Function and Types of Recursion
One Dimensional Array and Multidimensional Array
String Operations and Pointers
Pointer Expression and Pointer Arithmetic Programming Assignments using the above features. Structure and Union (Only concepts, No Programming)

Contents

CHAPTER -1: COMPUTER ORGANISATION

1.1 Introduction to Computer

- Computer is an electronic device that operates (works) under the control of programs stored in its own memory unit.
- A computer is an electronic machine that processes raw data to give information as output.
- An electronic device that accepts data as input, and transforms it under the influence of a set of special instructions called Programs, to produce the desired output (referred to as Information).

Explanations:

- A computer is described as an electronic device because; it is made up of electronic components and uses electric energy (such as electricity) to operate.
- A computer has an internal memory, which stores data & instructions temporarily awaiting processing, and even holds the intermediate result (information) before it is communicated to the recipients through the Output devices.
- It works on the data using the instructions issued, means that, the computer cannot do any useful job on its own. It can only work as per the set of instructions issued.
- A computer will accept data in one form and produce it in another form. The data is normally held within the computer as it is being processed.

Program:

- A computer Program is a set of related instructions written in the language of the computer & is used to make the computer perform a specific task (or, to direct the computer on what to do).
- A set of related instructions which specify how the data is to be processed.
- A set of instructions used to guide a computer through a process.

Data:

- Data is a collection of raw facts, figures or instructions that do not have much meaning to the user.
- Data may be in form of numbers, alphabets/letters or symbols, and can be processed to produce information.

Types of Data

There are two types/forms of data:

a). Digital (discrete) data:

Digital data is discrete in nature. It must be represented in form of numbers, alphabets or symbols for it to be processed by a computer. Digital data is obtained by counting.

E.g. 1, 2, 3…

b). Analogue (continuous) data:

Analogue data is continuous in nature. It must be represented in physical nature in order to be processed by the computer. • Analogue data is obtained by measurement. E.g. Pressure, Temperature, Humidity, Lengths or currents, etc. The output is in form of smooth graphs from which the data can be read.

Data Processing:

- It is the process of collecting all items of data together & converting them into information.
- Processing refers to the way the data is manipulated (or handled) to turn it into information.
- The processing may involve calculation, comparison or any other logic to produce the required result. The processing of the data usually results in some meaningful information being produced.

Information:

- Information is the data which has been refined, summarized & manipulated in the way you want it, or into a more meaningful form for decision-making.
- The information must be accurate, timely, complete and relevant.

1.2 Evolution of Computers

Computer evolution refers to the change in computer technology right from the time computers were first used to the present. As computing evolves to higher system levels, so its design also changes, from technical to socio-technical design.

The series of the evolution of computers are given below.

- ✓ Abacus
- ✓ Pascaline
- ✓ Difference engine
- ✓ Punched card equipment
- ✓ ABC
- ✓ UNIVAC - I

Abacus

- The present day computers are a result of an evolutionary process which started way back in 500 B.C. when Egyptian used a machine which is an early form of Abacus.
- However the present form of Abacus is attributed to the Chinese and Japanese.
- This is a machine, which was used for addition, subtraction, multiplication and division operation.

Pascaline

- In 1645 a device known as Pascaline was invented by French mathematician Blaise Pascal.
- The machine was also used per addition and subtraction purpose.
- The device was operated by dialing a set of wheels.
- In 1671 Leibniz improved on Pascal's adding machine and invented the Leibniz's Calculator.

Difference engine

- In 1822 Charles Babbage invented a Difference Engine.
- The purpose of this device was to calculate the roots of polynomial equations and prepare astronomy table for the British Navy.
- He upgraded this to, invent an Analytical engine, which could store program instructions initially coded on punched cards and subsequently shared internally.
- Therefore Charles Babbage is known as the father of computers.

Punched card equipment

- In 1890 Dr. H. Hollerith developed punched card equipment.
- This equipment read the holes punched in the card and mechanically performed the statistical analysis.

ABC (Atanasoff-Berry Computer)

- The first pure electronic computer was invented by J. V. Atanasoff and C. Berry which is known as Atanasoff-Berry Computer or ABC.
- It used vacuum tubes for both data storage and data computation.
- Subsequently Electronic Numerical Integrator and Calculator (ENIAC) was designed and accepted as the general purpose computer.

UNIVAC

- In 1945 John Von Neumann first gave the idea of sharing the same internal memory for storing both data and instruction, which was subsequently adopted in every computer organization.
- On this principle subsequently Universal Automatic Computer (UNIVAC-1) was invented

1.3 Generation of Computers

Computers are devices that accomplish tasks or calculations in accordance to a set of

directions, or programs. The first fully electronic computers, introduced in the 1940s, were voluminous devices that required teams of people to handle. In comparison to those new

machines, today's computers are astounding. They are not only thousands of times more expeditious, but also they can fit on your desk, on your lap, or even in your pocket. Computers are such an integral part of our everyday life now most people take them for granted.

Computers work through an interaction of hardware and software. The whole picture of the computer goes back to decades. However there are five apparent generations of computers. Each generation is defined by a paramount technological development that changes necessarily how computers operate – leading to more compressed, inexpensive, but more dynamic, efficient and booming machines.

First Generation – Vacuum Tubes (1940 – 1956)

These ancient computers utilized vacuum tubes as circuitry and magnetic drums for recollection. As a result they were huge, actually taking up entire rooms and costing resources to run. These were ineffective materials which produce a huge amount of heat, sucked enormous electricity and subsequently engendered an abundance of heat which caused perpetual breakdowns.

These first generation computers relied on 'machine language' (which is the most fundamental programming language that can be understood by computers). These computers were limited to solving one problem at a time. Input was predicated on punched cards and paper tape. Output emerged on print-outs. The two eminent machines of this era were the UNIVAC and ENIAC machines – the UNIVAC is the first ever commercial computer which was purchased in 1951 by a business named as the US Census Bureau.

Second Generation – Transistors (1956 – 1963)

The supersession of vacuum tubes by transistors, visualized the onset of the second generation of computing. Although first invented in 1947, transistors weren't used considerably in computers until the cessation of the 1950s. They were a huge development over the vacuum tube, despite the fact still subjecting computers to destroying different levels of heat. However they were extremely superior to the vacuum tubes, making computers smaller, more expeditious, inexpensive and less burdensome on electricity use. They still count on punched card for input/printouts.

The language emerged from strange binary language to symbolic ('assembly') languages. These meant programmers could discover instructions in words. Meanwhile during the same time high caliber programming languages were being developed (early versions of COBOL and FORTRAN). Transistor-driven machines were the first computers to store instructions into their recollections, peregrinating from magnetic drum to magnetic core 'technology'. The anticipatory versions of these machines were created for the atomic energy industry.

Third Generation – Integrated Circuits (1964 – 1971)

By this phase, transistors were now being miniaturized and put on silicon chips. This led to a huge improvement in speed and effectiveness of these machines. These were the first computers where users interacted utilizing keyboards and monitors who interfaced with an operating system, a consequential leap up from the punch cards and printouts. This facilitated these machines to run various applications at once utilizing a central program which functioned to monitor memory.

As a result of these advances which again made machines more reasonable and more tiny, a brand new group of users emerged during the '60s.

Fourth Generation – Microprocessors (1972 – 2010)

This innovation can be defined in one word: Intel. The chip-maker accomplished the Intel 4004 chip in 1971, which located all components of computer such as CPU, recollection, input/output controls onto a single chip. What overcrowded a room in the 1940s now gets fit in the palm of the hand. The Intel chip contained thousands of unified circuits. The year 1981 saw the first ever computer (IBM) categorically designed for home use and 1984 saw the Macintosh introduced by Apple. Microprocessors even transformed beyond the realm of computers and into an incrementing number of everyday products.

The incremented power of these small computers denoted they could be linked, establishing networks. Which eventually led to the expansion, birth and rapid evolution of the Internet? Other primary advances during this period have been the Graphical user interface (GUI), the mouse and more of late the startling advances in laptop capability and hand-held contrivances.

Fifth Generation – Artificial Intelligence (2010 Onwards)

Computer devices with artificial potentiality are still in development, but some of these technologies are commencing to emerge and be used such as voice recognition. AI is an authenticity, made possible by adopting parallel processing and superconductors. Inclining to the future, computers will be thoroughly revolutionized again by quantum computation, molecular and nano technology. The essence of fifth generation will be utilizing these technologies to ultimately engender machines which can proceed and acknowledge natural language, and have efficiency to determine and organize themselves.

Summary of Generation of computers

Features	1st Generation Computer	2nd Generation Computer	3rd Generation Computer	4th Generation Computer
Main Switching Device	Vacuum Tube	Transistor	Integrated Circuit(IC)	Large Scale Integration(LSI) & VLSI
Component Size	6000 circuits/cubic foot	10000 circuits/cubic foot	10 millions circuits/cubic foot	Over 500 billion circuits/cubic foot
Number of instructions/sec.	250	30,000	2,00,000	80 Millions
Meantime between failure	Hour	Days	Weeks	Months

Internal memory capacity	4000 characters	30000 characters	5,12,000 characters	Over 4 million characters

1.4 Classification of Computers

All the modern computers are broadly classified into the following three categories.

a) Analog Computer.

b) Digital Computer and

c) Hybrid Computer.

Analog computers

- Are mostly used in industries in process control activities.
- These computers work on analog data such as variation in temperature, pressure, speed, voltage etc.
- They are not general purpose computers; rather they are specific to a particular application area. Therefore the cost of such computer varies from application to application depending on the complexity.
- The uses of such computers are very limited.

Digital computers

- These computers are general purpose computers, which work on digital / binary data.
- The speed and accuracy with which these computers work are very high.
- Digital computers are also having several ranges from super computers to personal computers.

Hybrid computers

- Practically Hybrid computer are used to control the entire process.
- The analog feature of such computer enables it to measure the physical quantities such a temperature, pressure, voltage level etc. and convert them to digital data.
- These data are then processed by the computer by using its digital data processing capability.
- The output from this computer may be taken in a paper as hardcopy, may be seen on a display device or may be converted into analog form to automatically control various processes.

Digital computers

Digital computers are classified into the following four categories:

- Super Computers.
- Mainframe Computers.
- Mini Computers.
- Micro Computer

Super computers

- These computers are specifically designed to maximize the processing of floating point instructions.
- This is possible because of parallel processing technique which implements multiple processors to work in parallel manner.
- Such computers are very expensive and used in very high-end numerical processing, geographical information system, etc.
- Some of the popular super computers are Cray, Param, Anupam etc.
- The speed of processing of super computers are measured in GFLOPS i.e., Giga Floating Point Operations Per Second.
- These computers used their own operating system and programming language and hence vary from computer to computer.

Mainframe computers

These computers are intended for substantial high volume data processing.

These computers are characterized by–

- Large primary memory.
- Substantial processing capabilities. (MIPS)
- Substantial amount of peripheral devices that can be attached.
- A high data communication capability i.e. ability to connect thousands of terminals.
- Wide variety of memory size and terminal support option.
- Ability to handle large type computer application.

Application of Mainframe Computer

- space research,
- university connectivity,
- Wide area network (WAN) implementation etc.

Specification of mainframe computers.

- Processing speed – 30 to 100 million instruction per second (MIPS)
- Word length – More than 64 bits.
- I/O device –Wide range of peripheral devices.
- Internal Storage – More than 1 GB.

Mini Computers

- Fairly large primary memory.
- Medium scale processing capability i.e., lesser than mainframe but higher than personal computers.
- Can connect up to 500 terminals on LAN.
- Supports wide range of application areas.
- Affordable, unlike mainframe computers by small business organization.

The application of mini computers

- The field of engineering and scientific organizations,
- Educational Institutes,
- Universities,
- Small/medium business organizations.

These computers are mainly used for medium or large volume data processing activity.

Specification of mini computer

- Processing speed – 10 to 30 MIPS.
- Word length – 32 bits.
- I/O device – Wide range of I/O devices can be connect.
- Internal storage – 66 MB to 512 MB.

Micro computers

This is the smallest and least expensive computers are or personal computers popularly known as PC.

Typical features.

- These computers are portable.
- They require minimum power.
- Processing power is appropriate for handling most of the tasks.
- Memory capacity is sufficient to handle most of the tasks.
- Ease of use and support to various types of operating systems and application softwares.
- Affordable price tag.
- These micro computers are further classified into three categories i.e., PC, PC-XT and PC-AT.

Typical Specifications of PC

- Processor – I 8086 / I 8088 microprocessor.
- Memory – 640 KB of RAM
- Two 360 K Floppy Disk drive.
- Numerical Processor – I 8087.
- System bus –8 bit data bus & 16 bit address bus.
- Clock speed – 8 MHz

Personal Computer

- A PC-XT or Personal Computer with extended technology is an Up gradation of the PC. It is having all the features discussed above. Apart from these the concept of secondary memory / mass storage in the form of hard disk drive, was introduced for the first time.
- The present days PC are PC-AT or personal computer with advanced technology.

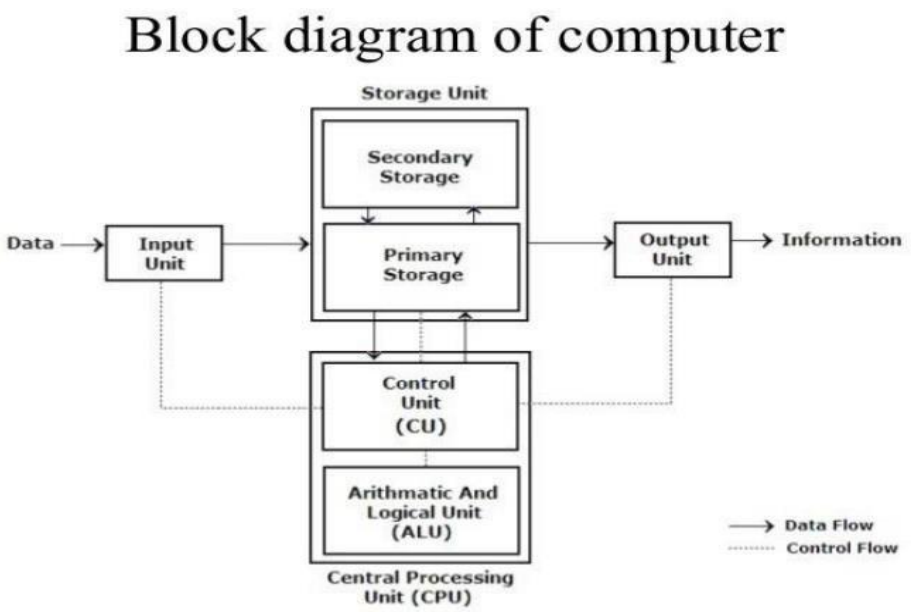
Features of PC-AT

- Processor –I 80386 / 80486 / Pentium.
- Memory – 2 MB to 512 MB.
- Floppy disk drive 1.44 MB
- Hard disk drive 1.2 MB to 80 GB.
- System bus–32 bit to 64 bit.
- Clock speed–Up to 3 GHz
- Operating System –MS-DOS, Windows, UNIX, Linux etc.

1.5 Basic Organization of Computer (Functional Block diagram), Input Devices, CPU & Output Devices.

A digital computer is considered to be a calculating device that can perform arithmetic operations at enormous speed. It is defined as a device that operates upon information/data. To be able to process data the computer is made of various functional units to perform its specified task.

Block Diagram of Computer



Input Device:

Computers need to receive data and instruction in order to solve any problem. Therefore, we need to input the data and instructions into the computers. The input unit consists of one or more input devices. **Keyboard** is the one of the most commonly used input device. Other commonly used input devices are the **Mouse, Scanner, Microphone** etc. All the input devices perform the following functions.

- Accept the data and instructions from the outside world.

- Convert it to a form that the computer can understand.
- Supply the converted data to the computer system for further processing.

Central Processing Unit:

The Control Unit (CU) and Arithmetic Logic Unit (ALU) of the computer are together known as the Central Processing Unit (CPU). The CPU is like brain performs the following functions:

- It performs all calculations.
- It takes all decisions.
- It controls all units of the computer.

Arithmetic Logical Unit:

- All calculations are performed in the Arithmetic Logic Unit (ALU) of the computer. It also does comparison and takes decision. The ALU can perform basic operations such as addition, subtraction, multiplication, division, etc and does logic operations viz, >, <, =, 'etc. Whenever calculations are required, the control unit transfers the data from storage unit to ALU once the computations are done, the results are transferred to the storage unit by the control unit and then it is send to the output unit for displaying results.

Control Unit:

- It controls all other units in the computer. The control unit instructs the input unit, where to store the data after receiving it from the user. It controls the flow of data and instructions from the storage unit to ALU. It also controls the flow of results from the ALU to the storage unit.

- The control unit is generally referred as the central nervous system of the computer that control and synchronizes it's working.

Output Device:

- The output unit of a computer provides the information and results of a computation to outside world. Printers, Visual Display Unit (VDU) are the commonly used output devices. Other commonly used output devices are Speaker, Headphone, and Projector etc.

Storage Unit:

The storage unit of the computer holds data and instructions that are entered through the input unit, before they are processed. It preserves the intermediate and final results before these are sent to the output devices. It also saves the data for the later use. The various storage devices of a computer system are divided into two categories.

a) **Primary Storage:** Stores and provides very fast. This memory is generally used to hold the program being currently executed in the computer, the data being received from the input unit, the intermediate and final results of the program. The primary memory is temporary in nature. The data is lost, when the computer is switched off. In order to store the data permanently, the data has to be transferred to the secondary memory. The cost of the primary storage is more compared to the secondary storage. Therefore, most computers shave limited primary storage capacity.

b) **Secondary Storage:** Secondary storage is used like an archive. It stores several programs, documents, data bases etc. The programs that you run on the computer are first transferred to the primary memory before it is actually run. Whenever the results are saved, again they get stored in the secondary memory. The secondary memory is slower and cheaper than the primary memory. Some of the commonly used secondary memory devices are Hard disk, CD, etc.,

Memory Size:

- All digital computers use the binary system, i.e. 0's and 1's. Each character or a number is represented by an 8-bit code.
- The set of 8 bits is called a byte. A character occupies 1-byte space. A numeric occupies 2- byte space. Byte is the space occupied in the memory.
- The size of the primary storage is specified in KB (Kilobytes) or MB (Megabyte). One KB is equal to 1024 bytes and one MB is equal to 1000KB. The size of the primary storage in a typical PC usually starts at 16MB. PCs having 32 MB, 48MB, 128 MB, 256MB memory are quite common.

1.6 Computer Memory and Classification of Memory

- Computer memory is a generic term for all of the different types of data storage technology that a computer may use, including RAM, ROM, and flash memory.
- Some types of computer memory are designed to be very fast, meaning that the central processing unit (CPU) can access data stored there very quickly. Other types are designed to be very low cost, so that large amounts of data can be stored there economically.
- Another way that computer memory can vary is that some types are non-volatile, which means they can store data on a long term basis even when there is no power. And some types are volatile, which are often faster, but which lose all the data stored on them as soon as the power is switched off.
- A computer system is built using a combination of these types of computer memory, and the exact configuration can be optimized to produce the maximum data processing speed or the minimum cost, or some compromise between the two.

Types of Computer Memory: Primary and Secondary

- Although many types of memory in a computer exist, the most basic distinction is between primary memory, often called system memory, and secondary memory, which is more commonly called storage.
- The key difference between primary and secondary memory is speed of access.
- Primary memory includes ROM and RAM, and is located close to the CPU on the computer motherboard, enabling the CPU to read data from primary memory very quickly indeed. It is used to store data that the CPU needs imminently so that it does not have to wait for it to be delivered.
- Secondary memory by contrast, is usually physically located within a separate storage device, such as a hard disk drive or solid state drive (SSD), which is connected to the computer system either directly or over a network. The cost per gigabyte of secondary memory is much lower, but the read and write speeds are significantly slower.
- Over several periods of computer evolution, a wide of array of computer memory types has been deployed, each with its own strengths and weaknesses.

Primary Memory Types: RAM and ROM

There are two key types of primary memory:

- **RAM, or random access memory**
- **ROM, or read-only memory**

Let's look in-depth at both types of memory.

1) RAM Computer Memory

The acronym RAM stems from the fact that data stored in random access memory can be accessed– as the name suggests – in any random order. Or, put another way, any random bit of data can be accessed just as quickly as any other bit.

The most important things to understand about RAM are that RAM memory is very fast, it can be written to as well as read, it is volatile (so all data stored in RAM memory is lost when it loses power) and, finally, it is very expensive compared to all types of secondary memory in terms of cost per gigabyte. It is because of the relative high cost of RAM compared to secondary memory types that most computer systems use both primary and secondary memory.

Data that is required for imminent processing is moved to RAM where it can be accessed and modified very quickly, so that the CPU is not kept waiting. When the data is no longer required it is shunted out to slower but cheaper secondary memory, and the RAM space that has been freed up is filled with the next chunk of data that is about to be used.

Types of RAM

DRAM: DRAM stands for Dynamic RAM, and it is the most common type of RAM used in computers. The oldest type is known as single data rate (SDR) DRAM, but newer computers use faster dual data rate (DDR) DRAM. DDR comes in several versions including DDR2 , DDR3, and DDR4, which offer better performance and are more energy efficient than DDR. However different versions are incompatible, so it is not possible to mix DDR2 with DDR3 DRAM in a computer system. DRAM consists of a transistor and a capacitor in each cell.
SRAM: SRAM stands for Static RAM, and it is a particular type of RAM which is faster than DRAM, but more expensive and bulker, having six transistors in each cell. For those reasons SRAM is generally only used as a data cache within a CPU itself or as RAM in very high-end server systems. A small SRAM cache of the most imminently-needed data can result in significant speed improvements in a system

The key differences between DRAM and SRAM are that SRAM is faster than DRAM - perhaps two to three times faster - but more expensive and bulkier. SRAM is usually available in megabytes, while DRAM is purchased in gigabytes.

DRAM uses more energy than SRAM because it constantly needs to be refreshed to maintain data integrity, while SRAM - though volatile – does not need constant refreshing when it is powered up.

2) ROM Computer Memory

ROM stands for read-only memory, and the name stems from the fact that while data can be read from this type of computer memory, data cannot normally be written to it. It is a very fast type of computer memory which is usually installed close to the CPU on the motherboard.

ROM is a type of non-volatile memory, which means that the data stored in ROM persists in the memory even when it receives no power – for example when the computer is turned off. In that sense it is similar to secondary memory, which is used for long term storage.

When a computer is turned on, the CPU can begin reading information stored in ROM without the need for drivers or other complex software to help it communicate. The ROM usually contains "bootstrap code" which is the basic set of instructions a computer needs to carry out to become aware of the operating system stored in secondary memory, and to load parts of the operating system into primary memory so that it can start up and become ready to use.

ROM is also used in simpler electronic devices to store firmware which runs as soon as the device is switched on.

Types of ROM

ROM is available in several different types, including PROM, EPROM, and EEPROM.

PROM: PROM stands for Programmable Read-Only Memory, and it is different from true ROM in that while a ROM is programmed (i.e. has data written to it) during the manufacturing process, a PROM is manufactured in an empty state and then programmed later using a PROM programmer or burner.

EPROM: EPROM stands for Erasable Programmable Read-Only Memory, and as the name suggests, data stored in an EPROM can be erased and the EPROM reprogrammed. Erasing an EPROM involves removing it from the computer and exposing it to ultraviolet light before re-burning it.

EEPROM: EEPROM stands for Electrically Erasable Programmable Read-Only Memory, and the distinction between EPROM and EEPROM is that the latter can be erased and written to by the computer system it is installed in. In that sense EEPROM is not strictly read-only. However in many cases the write process is slow, so it is normally only done to update program code such as firmware or BIOS code on an occasional basis.

Secondary Memory

We know that processor memory, also known as primary memory, is expensive as well as limited. The faster primary memory is also volatile. If we need to store large amount of data or programs permanently, we need a cheaper and permanent memory. Such memory is called secondary memory. Here we will discuss secondary memory devices that can be used to store large amount of data, audio, video and multimedia files.

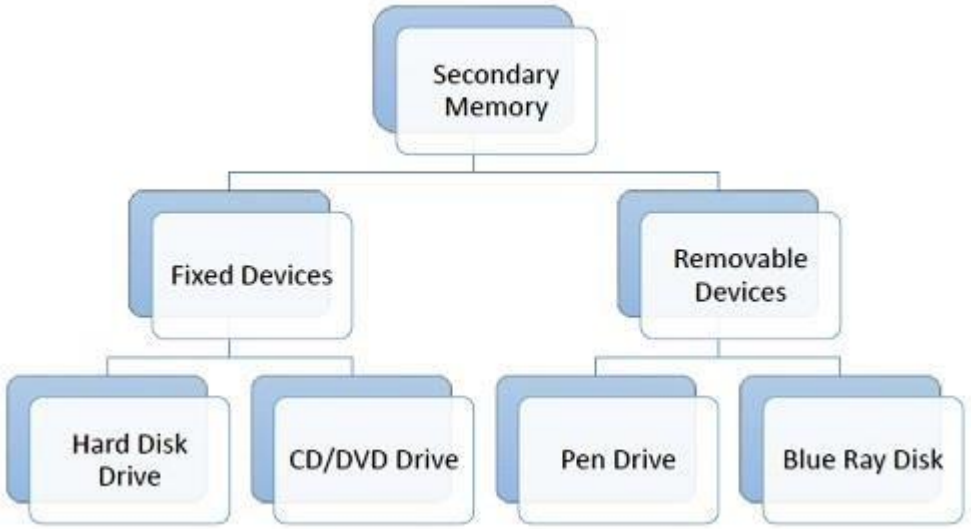
Characteristics of Secondary Memory

These are some characteristics of secondary memory, which distinguish it from primary memory

- It is non-volatile, i.e. it retains data when power is switched off
- It is large capacities to the tune of terabytes

- It is cheaper as compared to primary memory

Depending on whether secondary memory device is part of CPU or not, there are two types of secondary memory – fixed and removable.



Let us look at some of the secondary memory devices available.

Hard Disk Drive

Hard disk drive is made up of a series of circular disks called platters arranged one over the other almost ½ inches apart around a spindle. Disks are made of non-magnetic material like aluminum alloy and coated with 10-20 nm of magnetic material.

Standard diameter of these disks is 14 inches and they rotate with speeds varying from 4200 rpm (rotations per minute) for personal computers to 15000 rpm for servers. Data is stored by magnetizing or demagnetizing the magnetic coating. A magnetic reader arm is used to read data from and write data to the disks. A typical modern HDD has capacity in terabytes (TB).

CD Drive

CD stands for Compact Disk. CDs are circular disks that use optical rays, usually lasers, to read and write data. They are very cheap as you can get 700 MB of storage space for less than a dollar. CDs are inserted in CD drives built into CPU cabinet. They are portable as you can eject the drive, remove the CD and carry it with you. There are three types of CDs −

- CD-ROM (Compact Disk – Read Only Memory) − The data on these CDs are recorded by
  the manufacturer. Proprietary Software, audio or video are released on CD-ROMs.
- CD-R (Compact Disk – Recordable) − Data can be written by the user once on the CD-R. It cannot be deleted or modified later.
- CD-RW (Compact Disk – Rewritable) − Data can be written and deleted on these optical disks again and again.

DVD Drive

DVD stands for Digital Video Display. DVD are optical devices that can store 15 times the data held by CDs. They are usually used to store rich multimedia files that need high storage capacity. DVDs also come in three varieties – read only, recordable and rewritable.

Pen Drive

Pen drive is a portable memory device that uses solid state memory rather than magnetic fields or lasers to record data. It uses a technology similar to RAM, except that it is nonvolatile. It is also called USB drive, key drive or flash memory.

Blu Ray Disk

Blu Ray Disk (BD) is an optical storage media used to store high definition (HD) video and other multimedia filed. BD uses shorter wavelength laser as compared to CD/DVD. This enables writing arm to focus more tightly on the disk and hence pack in more data. BDs can store up to 128 GB data.

Solved Questions

Short Answer Type Questions.

Q.1 Outline the key features of 1st Generation Computer in brief.*(2019-Winter)*

Ans. The period of first generation was from 1946-1959. The computers of first generation used vacuum tubes as the basic components for memory and circuitry for CPU (Central Processing Unit). These tubes, like electric bulbs, produced a lot of heat and the installations used to fuse frequently. Therefore, they were very expensive and only large organizations were able to afford it. In this generation, mainly batch processing operating system was used. Punch cards, paper tape, and magnetic tape was used as input and output devices. The computers in this generation used machine code as the programming language.

Q.2 What is CPU?*(2018-Summer)*

Ans. A central processing unit (CPU), also called a central processor, main processor or just processor, is the electronic circuitry within a computer that executes instructions that make up a computer program. The CPU performs basic arithmetic, logic, controlling, and input/output (I/O) operations specified by the instructions in the program.

Q.3 What is ALU?*(2014-Winter)*

Ans. Stands for "Arithmetic Logic Unit." An ALU is an integrated circuit within a CPU or GPU that performs arithmetic and logic operations. Arithmetic instructions include addition, subtraction, and shifting operations, while logic instructions include boolean comparisons, such as AND, OR, XOR, and NOT operations.

Q.4 Name three input devices used in PC.*(2016-Summer)*

Ans. The three input devices used in PC are Keyboard, Mouse and Scanner.

Long Answer Type Questions.

Q.1 Discuss about the generation of computers? Explain the key features of computers of each generation?*(2013-Winter)*

Ans. A computer is an electronic device that manipulates information or data. It has the ability to store, retrieve, and process data.

Nowadays, a computer can be used to type documents, send email, play games, and browse the Web. It can also be used to edit or create spreadsheets, presentations, and even videos. But the evolution of this complex system started around 1940 with the first Generation of Computer and evolving ever since.

There are five generations of computers.

FIRST GENERATION

Features:

- 1946-1959 is the period of first generation computer.
- J.P.Eckert and J.W.Mauchy invented the first successful electronic computer called ENIAC, ENIAC stands for “Electronic Numeric Integrated and Calculator”.
- Few Examples are: ENIAC, EDVAC, UNIVAC, etc.

Advantages:

- It made use of vacuum tubes which are the only electronic component available during those days.
- These computers could calculate in milliseconds.

Disadvantages:

- These were very big in size, weight was about 30 tones.
- These computers were based on vacuum tubes.
- These computers were very costly.

SECOND GENERATION

Features:

- 1959-1965 is the period of second-generation computer.
- Second generation computers were based on Transistor instead of vacuum tubes.
- Few Examples are: Honeywell 400, IBM 7094, etc.

Advantages:

- Due to the presence of transistors instead of vacuum tubes, the size of electron component decreased. This resulted in reducing the size of a computer as compared to first generation computers.
- Less energy and not produce as much heat as the first generation.
- Assembly language and punch cards were used for input.

Disadvantages:

- A cooling system was required.
- Constant maintenance was required.
- Only used for specific purposes.

THIRD GENERATION

Features:

- 1965-1971 is the period of third generation computer.
- These computers were based on Integrated circuits.
- IC was invented by Robert Noyce and Jack Kilby In 1958-1959.
- IC was a single component containing number of transistors.
- Few Examples are: PDP-8, PDP-11, ICL 2900, etc.

Advantages:

- These computers were cheaper as compared to second-generation computers.
- They were fast and reliable.
- Use of IC in the computer provides the small size of the computer.

Disadvantages:

- IC chips are difficult to maintain.
- The highly sophisticated technology required for the manufacturing of IC chips.
- Air conditioning is required.

FOURTH GENERATION

Features:

- 1971-1980 is the period of fourth generation computer.
- This technology is based on Microprocessor.
- A microprocessor is used in a computer for any logical and arithmetic function to be performed in any program.
- Graphics User Interface (GUI) technology was exploited to offer more comfort to users.
- Few Examples are: IBM 4341, DEC 10, STAR 1000, etc.

Advantages:

- Fastest in computation and size get reduced as compared to the previous generation of computer.
- Heat generated is negligible.
- Small in size as compared to previous generation computers.

Disadvantages:

- The Microprocessor design and fabrication are very complex.
- Air conditioning is required in many cases due to the presence of ICs.
- Advance technology is required to make the ICs.

FIFTH GENERATION

Features:

- The period of the fifth generation in 1980-onwards.
- This generation is based on artificial intelligence.

- The aim of the fifth generation is to make a device which could respond to natural language input and are capable of learning and self-organization.
- This generation is based on ULSI (Ultra Large Scale Integration) technology resulting in the production of microprocessor chips having ten million electronic components.
- Few Examples are: Desktop, Laptop, etc.

Advantages:

- It is more reliable and works faster.
- It is available in different sizes and unique features.
- It provides computers with more user-friendly interfaces with multimedia features.

Disadvantages:

- They need very low-level languages.
- They may make the human brains dull and doomed.

Q2 Discuss about the various input devices used in PC platform?*(2013-Summer)*

Ans. Following are some of the important input devices which are used in a computer −

Keyboard

Keyboard is the most common and very popular input device which helps to input data to the computer. The layout of the keyboard is like that of traditional typewriter, although there are some additional keys provided for performing additional functions.

Keyboards are of two sizes 84 keys or 101/102 keys, but now keyboards with 104 keys or 108 keys are also available for Windows and Internet.

Mouse

Mouse is the most popular pointing device. It is a very famous cursor-control device having a small palm size box with a round ball at its base, which senses the movement of the mouse and sends corresponding signals to the CPU when the mouse buttons are pressed.

Generally, it has two buttons called the left and the right button and a wheel is present between the buttons. A mouse can be used to control the position of the cursor on the screen, but it cannot be used to enter text into the computer.

Joystick

Joystick is also a pointing device, which is used to move the cursor position on a monitor screen. It is a stick having a spherical ball at its both lower and upper ends. The lower spherical ball moves in a socket. The joystick can be moved in all four directions.

Joystick

The function of the joystick is similar to that of a mouse. It is mainly used in Computer Aided Designing (CAD) and playing computer games.

Light Pen

Light pen is a pointing device similar to a pen. It is used to select a displayed menu item or draw pictures on the monitor screen. It consists of a photocell and an optical system placed in a small tube.
When the tip of a light pen is moved over the monitor screen and the pen button is pressed, its photocell sensing element detects the screen location and sends the corresponding signal to the CPU.

Track Ball

Track ball is an input device that is mostly used in notebook or laptop computer, instead of a mouse. This is a ball which is half inserted and by moving fingers on the ball, the pointer can be moved.

Since the whole device is not moved, a track ball requires less space than a mouse. A track ball comes in various shapes like a ball, a button, or a square.

Scanner

Scanner is an input device, which works more like a photocopy machine. It is used when some information is available on paper and it is to be transferred to the hard disk of the computer for further manipulation.

Scanner captures images from the source which are then converted into a digital form that can be stored on the disk. These images can be edited before they are printed.

Digitizer

Digitizer is an input device which converts analog information into digital form. Digitizer can convert a signal from the television or camera into a series of numbers that could be stored in a computer. They can be used by the computer to create a picture of whatever the camera had been pointed at.

Digitizer is also known as Tablet or Graphics Tablet as it converts graphics and pictorial data into binary inputs. A graphic tablet as digitizer is used for fine works of drawing and image manipulation applications.

Microphone

Microphone is an input device to input sound that is then stored in a digital form.

The microphone is used for various applications such as adding sound to a multimedia presentation or for mixing music.

Magnetic Ink Card Reader (MICR)

MICR input device is generally used in banks as there are large number of cheques to be processed every day. The bank's code number and cheque number are printed on the cheques with a special type of ink that contains particles of magnetic material that are machine readable.

This reading process is called Magnetic Ink Character Recognition (MICR). The main advantages of MICR is that it is fast and less error prone.

Optical Character Reader (OCR)

OCR is an input device used to read a printed text.

OCR scans the text optically, character by character, converts them into a machine readable code, and stores the text on the system memory.

Bar Code Readers

Bar Code Reader is a device used for reading bar coded data (data in the form of light and dark lines). Bar coded data is generally used in labelling goods, numbering the books, etc. It may be a handheld scanner or may be embedded in a stationary scanner.

Bar Code Reader scans a bar code image, converts it into an alphanumeric value, which is then fed to the computer that the bar code reader is connected to.

Optical Mark Reader (OMR)

OMR is a special type of optical scanner used to recognize the type of mark made by pen or pencil. It is used where one out of a few alternatives is to be selected and marked.

It is specially used for checking the answer sheets of examinations having multiple choice questions.

EXERCISE

Short Answer Type Questions.

Q.1 Outline the key features of 4th generation computer in brief.

Q.2 Differentiate between analog computer and digital computer.

Q.3 Define hybrid computer.

Q.4 Differentiate between data and information?

Q.5 Differentiate between softcopy and hardcopy output?

Q.6 Define Auxiliary memory used in PC.

Q.7 Define primary memory used in PC.

Q.8 Define memory hierarchy. State the different memory present in the different levels. ***(2015-Winter, 2013-Winter)***

Q.9 Define a cache memory.

Q.10 What is an OMR?

Q.11 What is a MICR?***(2014-Winter)***

Q.12 Define Hardcopy output and Softcopy output from a PC.

Q.13 Differentiate between volatile and non-volatile computer memory.

Q.14 What are the different types of plotters used in PC?

Long Answer Type Questions.

Q.1 Discuss about the various output devices used in a PC platform?

Q.2 What is classification of computer? Compare and contrast the features of different classes of computers? ***(2013-Winter)***

Q.3 Discuss about the evolution of computers? Give suitable examples of various computers?

Q.4 What is memory hierarchy? Explain the main features of the various types of memory present at different levels of this hierarchy? ***(2016-Winter)***

Q.5 Differentiate between primary memory and secondary memory used in PC. Give proper example to substantiate your answer? ***(2017-Winter)***

****

CHAPTER – 2: COMPUTER SOFTWARE

2.1 Software concept

Software is a program which helps the human user to give instruction to computer hardware.

Types of Software

- Systems software
- Application software

2.1.1 System software

- System Software is a set of programs that manage the resources of a computer system.
- System Software is a collection of system programs that perform a variety of functions such as -
  - File Editing
  - Resource Accounting
  - I/O Management
  - Storage, Memory Management access
- System Software can be broadly classified into three types as:
  - System control programs
  - System support programs
  - System development programs
    - System control programs
      - controls the execution of programs
      - manage the storage & processing resources of the computer
      - perform other management & monitoring function
      - The most important of these programs is the operating system, DBMS & communication monitors.
    - System support programs
      - Provide routine service functions to the other computer programs & computer users:
        E.g. Utilities, libraries, performance monitors & job accounting.
    - System development programs
      - Assists in the creation of application
      - Programs. e.g., language translators such as BASIC interpreter & application generators.

2.1.2 Application software

- Application software, or app for short, is software that performs specific tasks for an end- user.
- Effectively, if the user is interacting directly with a piece of software it is application software. For example, Microsoft Word or Excel is application software, as are common web browsers such as Firefox or Google Chrome.
- It also includes the category of mobile apps, including communication apps such as WhatsApp or games such as Candy Crush Saga. There are also app versions of common services such as those providing weather or transport information or apps for customers to interact with companies.

- Application software is distinct from system software, which refers to the software that actually keeps the systems running such as the operating system, computational science software, game engines, industrial automation, and software as service applications.
- Instead of interacting with the user, the system software interacts with other software or hardware.

2.2 Overview of Operating System

- An operating system (OS) is software that manages computer hardware resources and provides common services for computer programs. The operating system is an essential component of the system software in a computer system.
- Application programs usually require an operating system to function. Time-sharing operating systems schedule tasks for efficient use of the system and may also include accounting software for cost allocation of processor time, mass storage, printing, and other resources.
- For hardware functions such as input and output and memory allocation, the operating system acts as an intermediary between programs and the computer hardware, although the application code is usually executed directly by the hardware and will frequently make a system call to an OS function or be interrupted by it. Operating systems can be found on almost any device that contains a computer—from cellular phones and video game consoles to supercomputers and web servers.
- Examples of popular modern operating systems include Android, BSD, iOS, Linux, OS X, QNX, Microsoft Windows, Windows Phone, and IBM z/OS. All these, except Windows, Windows Phone and z/OS, share roots in UNIX.

Types of operating systems

- Real-time
- Multi-user
- Multi-tasking vs. single-tasking
- Distributed

2.2.1 Objectives and Functions of O.S

OS typically provides services in the following areas:

1. **Program development -** The OS provides a variety of facilities and services, such as editors and debuggers, to assist the programmer in creating programs.
2. **Program execution -** A number of steps need to be performed to execute a program. Instructions and data must be loaded into main memory, I/O devices and files must be initialized, and other resources must be prepared. The OS handles these scheduling duties for the user.
3. **Access to I/O devices -** Each I/O device requires its own peculiar set of instructions or control signals for operation. The OS provides a uniform interface that hides these details so that programmers can access such devices using simple reads and writes.
4. **Controlled access to files -** For file access, the OS must reflect a detailed understanding of not only the nature of the I/O device (disk drive, tape drive) but also the structure of the data contained in the files on the storage medium. In the case of a system with multiple users, the OS may provide protection mechanisms to control access to the files.

5. **System access -** For shared or public systems, the OS controls access to the system as a whole and to specific system resources. The access function must provide protection of resources and data from unauthorized users and must resolve conflicts for resource contention.
6. **Error detection and response -** A variety of errors can occur while a computer system is running. These include internal and external hardware errors, such as a memory error, or a device failure or malfunction; and various software errors, such as division by zero, attempt to access forbidden memory location etc. In each case, the OS must provide a response that clears the error condition with the least impact on running applications. The response may range from ending the program that caused the error, to retrying the operation, to simply reporting the error to the application.
7. **Accounting -** A good OS will collect usage statistics for various resources and monitor performance parameters such as response time.

The major functions of Operating System are it acts as –

- The Resource Manager for Computer.
- The Memory Manager for computer.
- The Device Manager for computer.
- The User manager.

2.2.2 Types of Operating System: Batch Processing, Multiprogramming &Time Sharing OS

Following are some major types of Operating system.

Real-time operating system

A real-time operating system is a multitasking operating system that aims at executing real-time applications. Real-time operating systems often use specialized scheduling algorithms so that they can achieve a deterministic nature of behavior. The main objective of real-time operating systems is their quick and predictable response to events. They have an event-driven or time sharing design and often aspects of both. An event-driven system switches between tasks based on their priorities or external events while time-sharing operating systems switch tasks based on clock interrupts.

Multi-user operating system

A multi-user operating system allows multiple users to access a computer system at the same time. Time-sharing systems and Internet servers can be classified as multi-user systems as they enable multiple-user access to a computer through the sharing of time. Single-user operating systems have only one user but may allow multiple programs to run at the same time.

Multi-tasking vs. single-tasking Operating System

A multi-tasking operating system allows more than one program to be running at the same time, from the point of view of human time scales. A single-tasking system has only one running program.

Distributed Operating system

A distributed operating system manages a group of independent computers and makes them appear to be a single computer. The development of networked computers that could be linked and communicate with each other gave rise to distributed computing. Distributed computations are carried out on more than system.

Batch Processing

Batch processing is the execution of a series of programs ("jobs") on a computer without manual intervention. Jobs are set up so they can be run to completion without human interaction. All input parameters are predefined through scripts, command-line arguments, control files, or job control language. This is in contrast to "online" or interactive programs which prompt the user for such input. A program takes a set of data files as input, processes the data, and produces a set of output data files. This operating environment is termed as "batch processing" because the input data are collected into batches or sets of records and each batch is processed as a unit.

Multiprogramming

Multiprogramming is a rudimentary form of parallel processing in which several programs are run at the same time on a uniprocessor. Since there is only one processor, there can be no true simultaneous execution of different programs. Instead, the operating system executes part of one program, then part of another, and so on. To the user it appears that all programs are executing at the same time.

Time Sharing OS

In computing, time-sharing is the sharing of a computing resource among many users by means of multiprogramming and multi-tasking. Its introduction in the1960s, and emergence as the prominent model of computing in the 1970s, represented a major technological shift in the history of computing. By allowing a large number of users to interact concurrently with a single computer, time-sharing dramatically lowered the cost of providing computing capability, made it possible for individuals and organizations to use a computer without owning one, and promoted the interactive use of computers and the development of new interactive applications.

2.2.3 Features of DOS, Windows and UNIX

Features of DOS

The operating system is a system software used for management of computer hardware and application software. It is an interactive interface of computer.MS-DOS is a very popular operating system for PC, and it is replaced by its extension Windows operating system. In the Windows environment, DOS is also Boss because many utilities programs of MS-DOS are used in trouble shooting of Windows Operating system.

MS-DOS

MS-DOS stands for Microsoft Disk Operating System. Tim Paterson developed this operating system in 1980. The IBM (International Business Machine) released first PC (Personal Computer) in 1981. MS-DOS version 1.0 was used as operating system in IBM-PC and become talk of town in overnight. The father of PC Operating System is Gary Kildall of Digital Research. He had his Ph.D in computer and designed more successful operating System called CP/ M. The selling of CP / M is more than 600,000 copies proves its popularity. The Microsoft Disk Operating System or MS- DOS was based on QDOS, the

Quick and Dirty Operating System written by Tim Peterson of Seattle Computer Products, for their prototype Intel 8086 based computer. QDOS was based on Gray Kildall's CF/M. Paterson had bought a CP/ M manual and used it as the basis to write his operating system six weeks, QBOS was different enough from CP / M. MS-DOS version 7.0 is lunched in 1997 which is a hidden with Window95/98 OS.

It has three essential files and many command files. These essential files are: IO.SYS, MSDOS.SYS, and COMMAND.COM. These files are called system files of MS-DOS. The DOS is Boss in real sense, because in the age of Windows operating system, hardware utilities are dependent on the DOS.

The heart of MS-DOS:

- IO.SYS: This let DOS communicate with the hardware through the BIOS (Basic Input / Output System).
- MSDOS.SYS: This is a DOS kernel.
- COMMAND.COM: This is where all the DOS commands are stored and interpreted.
- COFIG.SYS: Hardware configuration information is stored here.
- AUTOEXEC.BAT: All the programs that are supposed to run at startup are called here.
- Booting: The booting is a process of loading system files into main memory (RAM). There are two types of booting: cold booting and warm booting.

Windows and UNIX

It is an operating system, extension of MS-DOS with user friendly GUI and several facilities to control memory, hardware, text, graphics, audio, video, internet connection etc.

Version

Comments

Windows 1.0

This operating system with user interface is a notification of MSDOS. The nifty mouse is used to click on desired program to open. It was first called interface manager, but then changed in to the more appealing Windows. Windows 1.0lanched in November 1985.

Windows 2.0

It was released in 1987 to take advantage of the awesome processing power of the Intel 286processor. The first version of Microsoft Word and Excel are introduced in this version.

Windows 3.0

It was released in May 1990. It came with a prettier 16-colorinterface, and new technological bells and whistles that let it make better use of the memory. In 1991, Microsoft brought multimedia support for Windows 3.0, called feature, though, is the active desktop, which integrates the web browser (internet Explorer), with operating system.

Windows 2000

Its interface is similar to interface for windows 98. It has new security protocol with an encryption facility to authenticate users logging in to the network. It supports 32 FAT file system along with NTFC (New Technology File System), making it easier for users to upgrade for Windows 98. It has quite robustly brid kernel architecture, make it more stable

version.

Windows Me

Later in 2000, Windows Millennium (windows me) Edition was released for the home user. This was the last version of Windows to be based on Windows 98. Windows Me had always been regarded as Microsoft' sway of keeping users busy while they waited for Windows XP.

Windows XP

Here, XP stands for experience. It bought together the robust Kernel of windows 2000 and all the friendless and multimedia support of Windows Me, and painted on a new face for it. Apart from the merger of Windows2000 and me, Windows Xp also added new features to enhance its performance. The first of These Throttling, Usually, Windows likes to do many things at once, but when memory falls short, it will throttle its memory access, doing fewer at a time. This shows the system down considerably, but prevents it from crashing.

Windows 7

Windows 7 is an operating system released by Microsoft on October 22, 2009. It follows the previous (sixth) version of Windows, called Windows Vista.

Like previous versions of Windows, Windows 7 has a graphical user interface (GUI) that allows you to interact with items on the screen using a keyboard and mouse. However, Windows 7 is also includes a feature called "Windows Touch" that supports touch screen input and multi touch functionality

Windows 8

Windows 8 is the latest version of Microsoft's Windows operating system. It was released on October 26, 2012, and is the first major update to Windows since Windows 7, which was released over three years earlier.

The goal of the new Windows 8 interface is to function on both traditional desktop PCs, such as desktop computers and laptops, as well as tablet PCs. Windows 8 supports both touch screen input as well as traditional input devices, such as a keyboard and mouse.

Windows 10

Windows 10 is a major version of the Microsoft Windows operating system that was released on July 29, 2015. It is built on the Windows NT kernel and follows Windows 8.

While Windows 10 includes many new features, it also brings back the Start Menu, which was dropped in Windows 8. The new and improved Start Menu provides quick access to settings, folders, and programs and also includes tiles from the Windows 8 interface. The bottom of the Windows 10 Start Menu includes a search bar that allows you to search both your local PC and the web.

UNIX Operating System

The UNIX (pronounced as YEW-nihks) is a powerful, flexible, multi-user Operating system with GUI and several utilities. Ken Thompson and Dennis Ritchie wrote Compiler under UNIX in 1969 at Bell Labs. In 1973, Thompson and Ritchie rewrote the UNIX kernel using C language. It is based on MULTICS operating system. Its first user was Bell patent department. XENIX, VENIX, MICRONIX, LINUX, UNIXWARE-7 etc are version of UNIX operating system.

The UNIX operating system is made up from three parts:

(a) Kernel: It is a hub of operating system dedicated for memory management, file management and communication within system.

(b) Shell: It is an interface between kernel and users. When a user logs in, the login program matches the username and password, and then starts shell. The shell is a command line interpreter (CLI) of UNIX.

(c) Program or command: It is used to accomplish specific tasks. When one command is terminated, the shell displays prompt % to accept next command for execution.

2.3 Programming Languages

A programming language is a formal constructed language designed to communicate instructions to a machine, particularly a computer. Programming languages can be used to create programs to control the behavior of a machine or to express algorithms so that the computer hardware can run with a proper step by step instruction.

The earliest programming languages preceded the invention of the computer and were used to direct the behavior of machines such as Jacquard looms and player pianos. Thousands of different programming languages have been created, mainly in the computer field, and many more still are being created every year. Many programming languages require computation to be specified in an imperative form (i.e., as a sequence of operations to perform), while other languages utilize other forms of program specification such as the declarative form (i.e. the desired result is specified, not how to achieve it).

The description of a programming language is usually split into the two components of syntax (form) and semantics (meaning). Some languages are defined by a specification document (for example, the C programming languages specified by an ISO Standard), while other languages (such as Perl) have dominant implementation that is treated as a reference.
There are two major types of programming languages I,e. Procedural programming language and Object Oriented programming language.

Procedural programming

In procedural programming our code is organized into small "procedures" that use and change our data. In ColdFusion, we write our procedures as either custom tags or functions. These functions typically take some input, do something, then produce some output. Ideally your functions would behave as "black boxes “where input data goes in and output data comes out.

The key idea here is that our functions have no intrinsic relationship with the data they operate on. As long as you provide the correct number and type of arguments, the function will do its work and faithfully return its output.

Sometimes our functions need to access data that is not provided as a parameter, i.e., we need access data that is outside the function. Data accessed in this way is considered "global" or "shared" data.

So in a procedural system our functions use data they are "given" (as parameters) but also directly access any shared data they need.

Object oriented programming

In object oriented programming, the data and related functions are bundled together into an "object". Ideally, the data inside an object can only be manipulated by calling the object's functions. This means that your data is locked away inside your objects and your functions provide the only means of doing something with that data. In a well-designed object oriented system objects never access shared or global data, they are only permitted to use the data they have, or data they are given.

2.4 Compiler, Interpreter

Compiler: It is a program which translates a high level language program into a machine language program. A compiler is more intelligent than an assembler. It checks all kinds of limits, ranges, errors etc. But its program run time is more and occupies a larger part of the memory. It has slow speed. Because a compiler goes through the entire program and then translates the entire program into machine codes. If a compiler runs on a computer and produces the machine codes for the same computer then it is known as a self-compiler or resident compiler. On the other hand, if a compiler runs on a computer and produces the machine codes for other computer then it is known as a cross compiler.

Interpreter: An interpreter is a program which translates statements of program into machine code. It translates only one statement of the program at a time. It reads only one statement of program, translates it and executes it. Then it reads the next statement of the program again translates it and executes it. In this way it proceeds further till all the statements are translated and executed. On the other hand, a compiler goes through the entire program and then translates the entire program into machine codes. A compiler is 5 to 25 times faster than an interpreter. By the compiler, the machine codes are saved permanently for future reference. On the other hand, the machine codes produced by interpreter are not saved. An interpreter is a small program as compared to compiler. It occupies less memory space, so it can be used in a smaller system which has limited memory space.

2.5 Computer Virus, Different Types of computer virus

A computer virus is a malware program that, when executed, replicates by inserting copies of itself (possibly modified) into other computer programs, data files, or the boot sector of the hard drive. When this replication succeeds, the affected areas are then said to be "infected". Viruses often perform some type of harmful activity on infected hosts, such as stealing hard disk space or CPU time, accessing private information, corrupting data, displaying political or humorous messages on the user's screen, spamming their contacts, or logging their keystrokes. However, not all viruses carry a destructive payload or attempt to hide themselves—the defining characteristic of viruses is that they are self-replicating computer programs which install themselves without the user's consent.

Virus writers use social engineering and exploit detailed knowledge of security vulnerabilities to gain access to their hosts' computing resources. The vast majority of viruses target systems running Microsoft Windows, employing a variety of mechanisms to infect new hosts, and often using complex anti detection/stealth strategies to evade antivirus software. Motives for creating viruses can include seeking profit, desire to send a political message, personal amusement, to demonstrate that a vulnerability exists in software, for sabotage and denial of service, or simply because they wish to explore artificial life and evolutionary algorithms.

Computer viruses currently cause billions of dollars worth of economic damage each year, due to causing systems failure, wasting computer resources, corrupting data, increasing maintenance costs, etc. In response, free, open source antivirus tools have been developed, and a multi-billion dollar industry of antivirus software vendors has cropped up, selling virus protection to users of various operating systems of which Android and Windows are among the most victimized. Unfortunately, no currently existing antivirus software is able to catchall computer viruses (especially new ones); computer security researchers are actively searching for new ways to enable antivirus solutions to more effectively detect emerging viruses, before they have already become widely distributed.

Different Types of computer virus

There are different types of viruses which can be classified according to their origin, techniques, types of files they infect, where they hide, the kind of damage they cause, the type of operating system, or platform they attack. Let us have a look at few of them.

Memory Resident Virus-These viruses fix themselves in the computer memory and get activated whenever the OS runs and infects all the files that are then opened. Hideout: This type of virus hides in the RAM and stays there even after the malicious code is executed. It gets control over the system memory and allocates memory blocks through which it runs its own code, and executes the code when any function is executed. Target: It can corrupt files and programs that are opened, closed, copied, renamed, etc. Examples: Randex, CMJ, Meve, and MrKlunky, Protection: Install an antivirus program.

Direct Action Viruses-The main purpose of this virus is to replicate and take action when it is executed. When a specific condition is met, the virus will go into action and infect files in the directory or folder that are specified in theAUTOEXEC.BAT file path. This batch file is always located in the root directory of the hard disk and carries out certain operations when the computer is booted. Find First/Find Next technique is used where the code selects a few files as its victims. It also infects the external devices like pen drives or hard disks by copying itself on them. Hideout: The viruses keep changing their location into new files whenever the code is executed, but are generally found in the hard disk's root directory. Target: It can corrupt files. Basically, it is a file- infector virus. Examples: Vienna virus. Protection: Install an antivirus scanner. However, this type of virus has minimal effect on the computer's performance.

Overwrite Viruses-A virus of this kind is characterized by the fact that it deletes the information contained in the files that it infects, rendering them partially or totally useless once they have been infected. Hideout: The virus replaces the file content. However, it does not change the file size. Examples: Way, Trj.Reboot, Trivial.88.D. Protection: The only way to clean a file infected by an overwrite virus is to delete the file completely, thus losing the original content. However, it is very easy to detect this type of virus, as the original program becomes useless.

Boot Sector Virus-This type of virus affects the boot sector of a hard disk. This is a crucial part of the disk, in which information of the disk itself is stored along with a program that makes it possible to boot (start) the computer from the disk. This type of virus is also called Master Boot Sector Virus or Master Boot Record Virus. Hideout: It hides in the memory until DOS accesses the floppy disk, and whichever boot data is accessed, the virus infects it. Examples: Polyboot.B, AntiEXE. Protection: The best way of avoiding boot sector viruses is to ensure that floppy disks are write-protected. Also, never start your computer with an unknown floppy disk in the disk drive.

Macro viruses infect files that are created using certain applications or programs that contain macros, like .doc, .xls, .pps, .mdb, etc. These mini programs make it possible to automate series of operations so that they are performed as a single action, thereby saving the user from having to carry them out one by one. These viruses automatically infect the file that contains macros, and also infects the templates and documents that the file contains. It is referred to as a type of e-mail virus. Hideout: These hide in documents that are shared via e-mail or networks. Examples: Relax, Melissa.A, Bablas, O97M/Y2KProtection: The best protection technique is to avoid opening e- mails from unknown senders. Also, disabling macros can help to protect your useful data.

Directory viruses (also called Cluster Virus/File System Virus)-It infects the directory of your computer by changing the path that indicates the location of a file. When you execute a program file with an extension .EXE or .COM that has been infected by a virus, you are unknowingly running the virus program, while the original file and program is previously moved by the virus. Once infected, it becomes impossible to locate the original files. Hideout: It is usually located in only one location of the disk, but infects the entire program in the directory. Examples: Dir- 2virusProtection: All you can do is, reinstall all the files from the backup that are infected after formatting the disk.

Polymorphic viruses encrypt or encode themselves in a different way (using different algorithms and encryption keys) every time they infect a system. This makes it impossible for antivirus software to find them using string or signature searches (because they are different in each encryption). The virus then goes onto creates a large number of copies. Examples: Elkern, Marburg, Satan Bug and Tuareg, Protection: Install a high-end antivirus as the normal ones are incapable of detecting this type of virus.

Companion viruses: It can be considered as a type of file infector virus, like resident or direct action types. They are known as companion viruses because once they get into the system they 'accompany' the other files that already exist.

In other words, to carry out their infection routines, companion viruses can wait in memory until a program is run (resident virus), or act immediately by making copies of themselves (direct action virus).

Hideout: These generally use the same filename and create a different extension of it. For example: If there is a file "Me.exe", the virus creates another file named "Me.com" and hides in the new file. When the system calls the filename "Me", the ".com" file gets executed (as ".com" has higher priority than ".exe"), thus infecting the system. Examples: Stator, Asimov.1539 and Terrax.1069 Protection: Install an antivirus scanner and also download Firewall.

FAT Virus: The file allocation table (FAT) is the part of a disk used to store all the information about the location of files, available space, unusable space, etc. Hideout: FAT virus attacks the FAT section and may damage crucial information. It can be especially dangerous as it prevents access to certain sections of the disk where important files are stored. Damage

caused can result in loss of information from individual files or even entire directories.

Examples: Link Virus, Protection: Before the virus attacks all the files on the computer, locate all the files that are actually needed on the hard drive, and then delete the ones that are not needed. They may be files created by viruses.

Multipartite Virus-These viruses spread in multiple ways possible. It may vary in its action depending upon the operating system installed and the presence of certain files.

Hideout: In the initial phase, these viruses tend to hide in the memory as the resident viruses do; then they infect the hard disk. Examples: Invader, Flip and Tequila.

Protection: You need to clean the boot sector and also the disk to get rid of the virus, and then reload all the data in it. However, ensure that the data is clean.

Web Scripting Virus-Many web pages include complex codes in order to create an interesting and interactive content. This code is often exploited to bring about certain undesirable actions. Hideout: The main sources of web scripting viruses are the web browsers or infected web pages. Examples: JS.Fortnight is a virus that spreads through malicious e-mails. Protection: Install the Microsoft tool application that is a default feature in Windows 2000, Windows 7 and Vista. Scan the computer with this application.

Worms: A worm is a program very similar to a virus; it has the ability to self-replicate and can lead to negative effects on your system. But they can be detected and eliminated by an antivirus software. Hideout: These generally spread through emails and networks. They do not infect files or damage them, but they replicate so fast that the entire network may collapse. Examples: PSWBugbear.B, Lovgate.F, Trile.C, Sobig.D, Mapson. Protection: Install an updated version of antivirus.

Trojans are another unsavory breed of malicious code are Trojans or Trojan horses, which unlike viruses, do not reproduce by infecting other files, nor do they self-replicate like worms. In fact, it is a program which disguises itself as a useful program or application.

Beware of the fact that these viruses copy files in your computer (when their carrier program is executed) that can damage your data, and even delete it. The attacker can also program the Trojans in such a manner that the information in our computer is accessible to them.

Logic Bombs are not considered viruses because they do not replicate. They are not even programs in their own right, but rather camouflaged segments of other programs. They are only executed when certain predefined condition is met. Their objective is to destroy data on the computer once certain condition shave been met. Logic bombs go undetected until launched, the results can be destructive, and your entire data can be deleted.

2.5.1 Detection and prevention of Virus

Following steps may be taken for Virus Detection and Prevention.

1. Do not open any files attached to an email from an unknown, suspicious or untrustworthy source.
2. Do not open any files attached to an email unless you know what it is, even if it appears to come from a dear friend or someone you know. Some viruses can replicate themselves and spread through email. Better be safe than sorry and confirm that they really sent it.
3. Delete chain emails and junk email. Do not forward or reply to any to them. These types of email are considered spam, which is unsolicited, intrusive mail that clogs up the network.
4. Exercise caution when downloading files from the Internet. Ensure that the source is a legitimate and a reputable one. Verify that an anti-virus program checks the files on the download site. If you're uncertain, don't download the file at all.
5. Update your anti-virus software regularly. Thousands of viruses are discovered each month, so you'll want to be protected.
6. Back up your files on a regular basis. If a virus destroys your files, at least you can replace them with your back-up copy. You should store your backup copy in a separate location from your work files, one that is preferably not on your computer.
7. When in doubt, always err on the side of caution and do not open, download, or execute any files or email attachments. Not executing the files is especially important. Check with your product vendors for updates which include those for your operating system web browser, and email. One example is the security site section of Microsoft located at http://www.microsoft.com/security.
8. Stay away from Bit torrent sites. Some of the more popular ones include Lime wire, Bit Torrent, Frost wire and Pirate Bay. These are heavily laden with viruses, malware and spyware. Downloading material from these websites is one of the easiest ways to become infected. It's in your best interest to just avoid these websites completely.
9. Be careful when searching on the internet, the links that come up from your search engine may contain a virus. Never go to sites that sound suspicious.
10. Due to the popularity of the social networking websites such as MySpace, Face book, and Twitter, virus makers target them more than any other website. Online gaming and gambling websites also are high risk websites. It's best to avoid these kinds of websites altogether.
11. If you happen to see a popup message when on the internet about being infected and to buy their software to protect yourself, do not fall for it! Most of the time these messages are easy to see as they tend to have bad grammar and spelling errors. Common names are XP Antivirus, Security Tools, ThinkPoint, Security Shield, Win 7 Security 2011, and similar variations. If do see one of these popup, do not click on them, immediately shut down your computer. If you click on any part of those windows you will give the virus permission to install and bypass your antivirus program.
12. If you see any suspicious pop-ups appear on your screen, do not click on them. If you do, it is very likely you will infect your computer. Instead use the following keyboard command, which will allow you to close the pop-up, without having the click on it or infecting yourself. The keyboard command is ALT + F4. If that fails, then shut down the computer.

2.6 Application of computers in different Domain

Computer is a device through which you can perform a variety of jobs. You can use your computer system for different applications by changing the software packages

Uses of Computer at Home

Computer can be used at home in the following ways.

Home Budget

Computer can be used to manage Home Budget. You can easily calculate your expenses and income. You can list all expenses in one column and income in another column. Then you can apply any calculation on these columns to plan your home budget. There are also specialize software that can manage your income and expenses and generate some cool reports.

Computer Games

An important use of computers at home is playing games. Different types of games are available. These games are a source of entertainment and recreation. Many games are available that are specially developed to improve your mental capability and thinking power.

Working from Home

People can manage the office work at home. The owner of a company can check the work of the employees from home. He can control his office while sitting at home.

Entertainment

People can find entertainment on the internet. They can watch movies, listen to songs, and watch videos download different stuff. They can also watch live matches on the internet.

Information

People can find any type of information on the internet. Educational and informative websites are available to download books, tutorials etc. to improve their knowledge and learn new things.

Chatting & Social Media

People can chat with friends and family on the internet using different software like Skype etc. One can interact with friends over social media websites like Face book, Twitter & Google Plus. They can also share photos and videos with friends.

Uses of Computers in Education

CBT are different programs that are supplied on CD-ROM. These programs include text, graphics and sound. Audio and Video lectures are recorded on the CDs. CBT is a low cost solution for educating people. You can train a large number of people easily.

Benefits of CBT

Some benefits of CBT are as follows:

1. The students can learn new skills at their own pace. They can easily acquire knowledge in any available time of their own choice.
2. Training time can be reduced.
3. Training materials are interactive and easy to learn. It encourages students to learn the topic.
4. Planning and timing problems are reduced or eliminated.
5. The skills can be taught at any time and at any place.
6. It is very cost effective way to train the large number of students.
7. Training videos and audios are available at affordable prices.

Computer Aided Learning (CAL)

Computer aided learning is the process of using information technology to help teaching and enhance the learning process. The use of computer can reduce the time that is spent on preparing teaching material. It can also reduce the administrative load of teaching and research. The use of multimedia projector and PowerPoint presentations has improved the quality of teaching. It has also helped the learning process.

Distance Learning

Distance learning is a new learning methodology. Computer plays the key role in this kind of learning. Many institutes are providing distance learning programs. The student does not need to come to the institute. The institute provides the reading material and the student attends virtual classroom. In virtual classroom, the teacher delivers lecture at his own workplace. The student can attend the lecture at home by connecting to a network. The student can also ask questions to the teacher.

Online Examination

The trend of online examination is becoming popular. Different examination like GRE, GMAT and SAT are conducted online all over the world. The questions are marked by computer. It minimizes the chance of mistakes. It also enables to announce the result in time.

Uses of Computers in Business

The use of computer technology in business provides many facilities. Businessmen are using computers to interact with their customers anywhere in the world. Many business tasks are performed more quickly and efficiently. Computers also help them to reduce the overall cost of their business. Computer can be used in business in the following ways.

Marketing

An organization can use computers for marketing their products. Marketing applications provide information about the products to customers. Computer is also used to manage distribution system, advertising and selling activities. It can also be used in deciding pricing strategies. Companies can know more about their customers and their needs and requirements etc.

Stock Exchange

Stock Exchange is the most important place for businessmen. Many stock exchanges use computers to conduct bids. The stockbrokers perform all trading activities electronically. They connect with the computer where brokers match the buyers with sellers. It reduces cost

as no paper or special building is required to conduct these activities.

Uses of computers in Medical

Field Hospital Management

System

Specialized hospital management softwares are used to automate the day today procedures and operations at hospitals. These tasks may be online appointments, payroll admittance and discharge records etc.

Patient History

Hospital management systems can store data about patients. Computers are used to store data about patients, their diseases & symptoms, the medicines that are prescribed.

Patients Monitoring

Monitoring systems are installed in medical wards and Intensive care units to monitoring patients continuously. These systems can monitor pulse, blood pressure and body temperature and can alert medical staff about any serious situations.

Life Support Systems

Specialized devices are used to help impaired patients like hearing aids.

Diagnosis Purpose

A variety of software are used to investigate symptoms and prescribed medication accordingly. Sophisticated systems are used for tests like CT Scan, ECG, and other medical tests.

Solved Questions

Short Answer Type Questions.

Q.1 Define an operating system. Give examples of three operating system used on PC.

Ans. An Operating System (OS) is an interface between computer user and computer hardware. An operating system is software which performs all the basic tasks like file management, memory management, process management, handling input and output, and controlling peripheral devices such as disk drives and printers.

Some popular Operating Systems include Linux Operating System, Windows Operating System, VMS, OS/400, AIX, z/OS, etc.

Q.2 Define a batch processing system.

Ans. Batch processing is the processing of transactions in a group or batch. No user interaction is required once batch processing is underway. This differentiates batch processing from transaction processing, which involves processing transactions one at a time and requires user interaction.

Q.3 Define a multi programming operating system.

Ans. Multiprogramming is a rudimentary form of parallel processing in which several programs are run at the same time on a uniprocessor. Instead, the operating system executes part of one program, then part of another, and so on. To the user it appears that all programs are executing at the same time.

Q4 Define a time sharing operating system.

Ans. Time-sharing enables many people, located at various terminals, to use a particular computer system at the same time. Multitasking or Time-Sharing Systems is a logical extension of multiprogramming. Processor's time is shared among multiple users simultaneously is termed as time-sharing.

Long Answer Type Questions

Q.1 **Define an operating system. Discuss about the major function of any operating system.** ***(2013-Winter) (2017-Winter)***

Ans.

An operating system is a program on which application programs are executed and acts as a communication bridge (interface) between the user and the computer hardware.

The main task an operating system carries out is the allocation of resources and services, such as allocation of: memory, devices, processors and information. The operating system also includes programs to manage these resources, such as a traffic controller, a scheduler, memory management module, I/O programs, and a file system.

Functions of an operating System:

Security –

The operating system uses password protection to protect user data and similar other techniques. It also prevents unauthorized access to programs and user data.

Control over system performance –

Monitors overall system health to help improve performance records the response time between service requests and system response to have a complete view of the system health. This can help improve performance by providing important information needed to troubleshoot problems.

Job accounting –

Operating system Keeps track of time and resources used by various tasks and users, this information can be used to track resource usage for a particular user or group of user.

Error detecting aids –

Operating system constantly monitors the system to detect errors and avoid the malfunctioning of computer system.

Coordination between other software and users –

Operating systems also coordinate and assign interpreters, compilers, assemblers and other software to the various users of the computer systems.

Memory Management –

The operating system manages the Primary Memory or Main Memory. Main memory is made up of a large array of bytes or words where each byte or word is assigned a certain address. Main memory is a fast storage and it can be accessed directly by the CPU. For a program to be executed, it should be first loaded in the main memory. An Operating System performs the following activities for memory management:

It keeps tracks of primary memory, i.e., which bytes of memory are used by which user program. The memory addresses that have already been allocated and the memory addresses of the memory that has not yet been used. In multi programming, the OS decides the order in which process are granted access to memory, and for how long. It allocates the memory to a process when the process requests it and deallocates the memory when the process has terminated or is performing an I/O operation.

Processor Management –

In a multi programming environment, the OS decides the order in which processes have access to the processor, and how much processing time each process has. This function of OS is called process scheduling. An Operating System performs the following activities for processor management.

Keeps tracks of the status of processes. The program which perform this task is known as traffic controller. Allocates the CPU that is processor to a process. De-allocates processor when a process is no more required.

Device Management –

An OS manages device communication via their respective drivers. It performs the following activities for device management. Keeps tracks of all devices connected to system. Designates a program responsible for every device known as the Input/output controller. Decides which process gets access to a certain device and for how long. Allocates devices in an effective and efficient way. Deallocates devices when they are no longer required.

File Management –

A file system is organized into directories for efficient or easy navigation and usage. These directories may contain other directories and other files. An Operating System carries out the following file management activities. It keeps track of where information is stored, user access settings and status of every file and more. These facilities are collectively known as the file system.

Q2 **What is the difference between Application Software and System Software?** ***(2015-Winter)***

Ans

Sr. No.	Key	System Software.	Application Software.
1	Definition	System Software is the type of software which is the interface between application software and system.	On other hand Application Software is the type of software which runs as per user request. It runs on the platform which is provide by system software.
2	Development Language	In general System software are developed in low level language which is more compatible with the system hardware in order to interact with.	While in case of Application software high level language is used for their development as they are developed as some specific purpose software.
3	Usage	System software is used for operating computer hardware.	On other hand Application software is used by user to perform specific task.
4	Installation	System software are installed on the computer when operating system is installed.	On other hand Application software are installed according to user's requirements.
5	User interaction	As mentioned in above points system software are specific to system hardware so less or no user interaction available in case of system software.	On other hand in application software user can interacts with it as user interface is available in this case.
6	Dependency	System software can run independently. It provides platform for running application Software.	On other hand in application software can't run independently. They can't run without the presence of system software..
7	Examples	Some examples of system software's are compiler, assembler, debugger, driver, etc.	On other hand some examples of application software's are word processor, web browser, media player, etc.

EXERCISE

Short Answer Type Questions.

Q.1 What are the four major functions of an operating system?

Q.2 What are the various types of operating system used on PC?

Q.3 Define a multi-tasking operating system.

Q.4 Define GUI.

Q.5 What is computer security?

Q.6 What is software? ***(2015-Summer)***

Q.7 Write types of software?

Q.8 What is application software?

Q.9 What is system software? Quote some examples of system software?

Q.10 What is an interpreter? How is it different from compiler?

Long Answer Type Questions

Q.1 Discuss the various objectives of an operating system. Illustrate your answer with proper examples.

Q.2 What is DOS? Discuss about the main features of DOS with appropriate examples.

Q.3 Compare and contrast between the features of UNIX operating system.

Q.4 Compare the various features of windows and UNIX operating system. Give suitable example to substantiate your answer. ***(2015-Summer)***

Q.5 Write a short note on Single user Vs. Multi-user O.S.

Q.6 What is Virus? How does virus spread and what are the symptoms of virus attack? How can you prevent virus attack? ***(2017-Winter) (2013-Winter) (2014-Winter)***

Q.7 Distinguish between Compiler and Interpreter? ***(2016-Summer)***

Q.8 Define Software. Describe various types of software and explain them?

****

CHAPTER –3: COMPUTER NETWORK AND INTERNET

3.1 Networking concept, Protocol, Connecting Media, Data Transmission mode

Networking concept

- A computer network is a collection of two or more computers, which are connected two or more computers, which are connected together to share information and resources.
- It is a combination of hardware and software that allows communication between computers over a network.

3.1.1 Protocol

- A protocol is a set of rules that the communication set of rules that the communication between computers on a network.
- Most important sets of internet protocols are TCP/IP, HTTPS, SMTP, and FTP.

3.1.2 Connecting Media

- Connecting media of a network refer to the transmission media used in the network.
- It refers to the physical media through which communication signals can be transmitted from one point to another.
- It can be divided into two broad categories
    1) Guided Media
    2) Unguided Media

Guided Media:

- The data signal in guided medium is bound by the cabling system that guide the data signal along a specific path.
- It consists of a cable composed of materials like copper, tin or silver.
- Basically, they are divided into three categories:

(a) Ethernet cable or Twisted Pair:

- In this pair, wires are twisted together, which are surrounded by an insulating material and an outer layer called jacket.
- A twisted pair consists of two conductors (copper)
- Ex- LAN Cable

(b) Coaxial Cable:

- It carries the signal of higher frequency data communication through the network.
- It is commonly used in transferring multi-channel television signals in cities.
- Ex- cable TV network

(c) Fiber – Optics Cable:

- It is made up of glass or plastic and transmits signals in the form of light from a source at one end to another end.
- The speed of Optical fiber is hundreds of times of faster than coaxial cables.

- Ex-Wavelength Division Multiplexing, SONET network

Unguided Media:

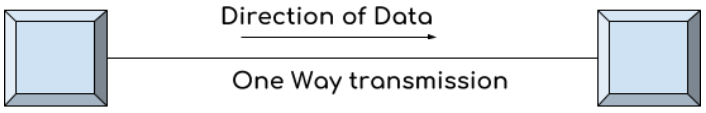
- It is the transfer of information over a distance without the use of enhanced electrical conductors or wires.
- When the computers in a network interconnected and data is transmitted through waves, then they are said to be connected through unguided media.
- Some commonly used unguided media of transmission are –
    1. Radio wave transmission
    2. Micro wave transmission
    3. Satellite communication
    4. Infrared wave transmission
    5. Bluetooth.

3.1.3 Date Transmission mode

- The way in which data is transmitted from one place to another is called Data transmission mode or data communication mode or directional modes.
- There are mainly 3 types of data transmission modes are-
    1. Simplex mode
    2. Half –duplex Mode
    3. Full –duplex mode

Simplex mode-

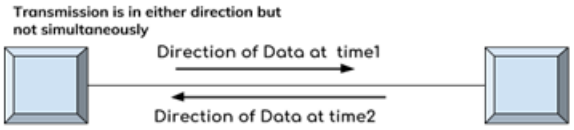
- In simplex mode, data can flow in only one direction.
- In this mode, a sender can only send data and cannot receive it.
- Similarly, a receiver can only receive data but cannot send it.
- Ex- Radio, Television etc.



Simplex Mode

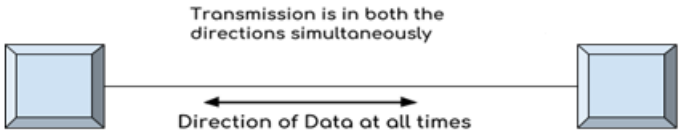
Half –duplex Mode-

- In half-duplex mode, data can flow in both directions but not at a same time.
- In this mode, data is sent and received alternatively.
- Ex- walkie-Talkie



Half-Duplex Mode

Full-Duplex Mode



Full-Duplex Mode

- In full Duplex-mode, data can flow in both directions at the same time.
- Ex- Mobile phone.

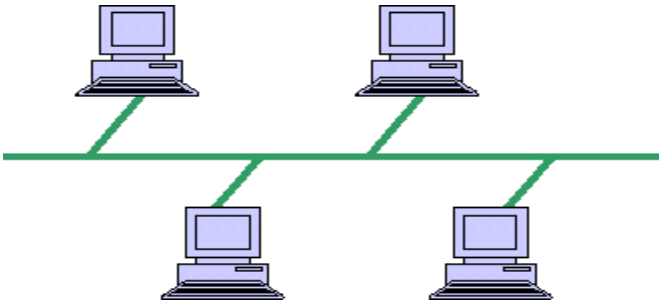
3.2 Network Topologies, Types of Network

3.2.1 Network Topologies

- Network topology is determined only by the configuration of connections between nodes.
- In a fully connected network with n nodes, there are n(n-1)/2 direct links.
- The most commonly used topology are described below:

1. Bus Topology

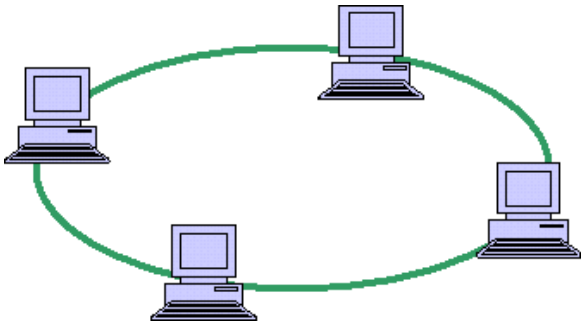
- In case of Bus topology, all devices share single communication line or cable.
- It is one of the simple forms of networking where a failure of a device does not affect the other devices.
- But failure of the shared communication line can make all other devices stop functioning.



[Bus Topology Diagram]

2. Ring Topology

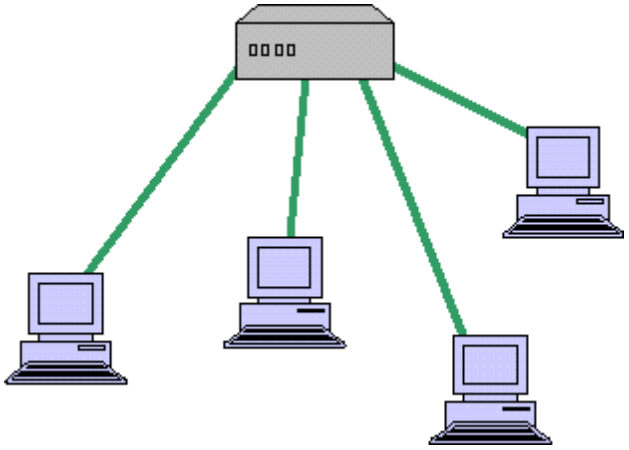
- In ring topology, each host machine connects to exactly two other machines, creating a circular network structure.
- Failure of any host results in failure of the whole ring.
- Thus, every connection in the ring is point of failure.



[Ring Topology Diagram]

3. Star Topology

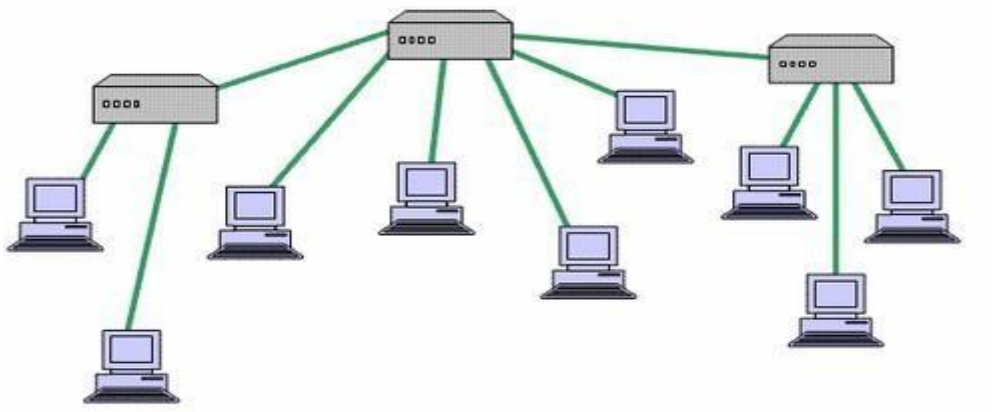
- All hosts in star topology are connected to a central device, known as hub device, using a point-to-point connection.
- If the central hub fails, then whole network fails.



[Star Topology Diagram]

4. Tree Topology

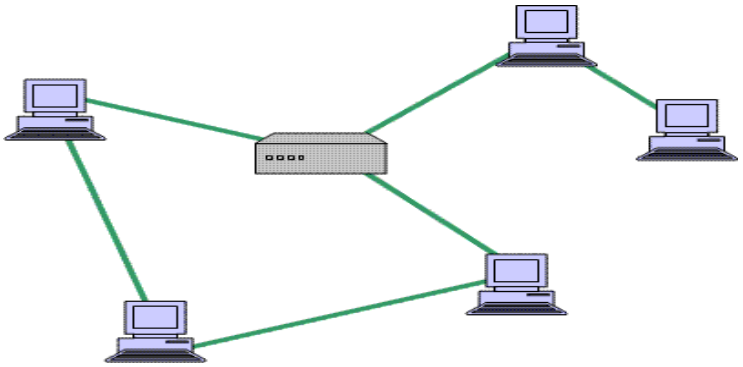
- This is also known as hierarchical topology.
- It is a network topology in which nodes are arranged as a tree.
- The function of the central node in this topology may be distributed.
- It is an extension and variation of star topology.
- A tree topology combines characteristics of linear bus and star topologies.



[Tree Topology Diagram]

5. Mesh Topology

- It is also known as completely interconnected topology.
- In mesh topology, every node has a dedicated point to point link to every other node.



[Mesh Topology Diagram]

3.2.2 Types of Network

- There are broadly classified into three types of computer network.

A. LAN
B. WAN
C. MAN
D. PAN

LAN (Local Area Network)

- LAN is a small and single site network.
- A LAN connects network devices over a relatively short distance.
- It is a system in which computers area interconnected and the geographical area such as home, office, building, school may be within a building.
- On most LANs cables are used to connect the computers.
- Data transfer rate in LAN is of the order 10 to 100Mbps.

WAN (Wide Area Network)

- A WAN is a geographically dispersed collection of LANs
- A WAN like the internet spans most of the world.
- A network device called a router connects LAN to a WAN.
- These kind of networks use telephone lines, Satellite links and other long-range communication technologies to connect.
- WAN use technology like ATM for connectivity.

MAN (Metropolitan Area Network)

- It is a data network designed for a town or city.
- It connects an area larger than a LAN, but smaller than a WAN, such as a city, with high performance hardware.
- Its main purpose is to share hardware and software resources by the various users.
- Ex- Cable TV network.
- The computers in a WAN are connected using coaxial cables or fiber optic cables.

PAN (Personal Area Network)

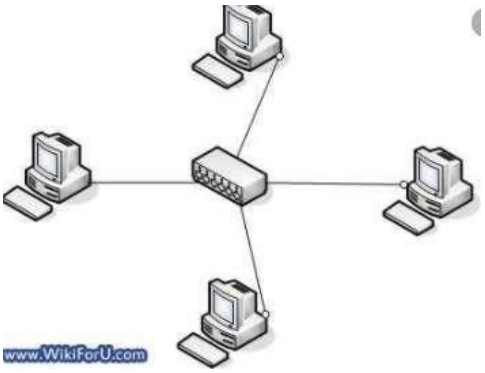
- A personal area network (PAN) is a computer network used for communication among computer devices close to one person.
- Some examples of devices that are used in a PAN are printers, fax machines, telephones, PDAs or scanners.
- The reach of a PAN is typically within about 20-30 feet (approximately 6-9 meters).
- Personal area networks may be wired with computer buses such as USB and FireWire.
- A wireless personal area network (WPAN) can also be made possible with network technologies such as IrDA and Bluetooth..

3.3 Networking Devices like Hub, Repeater, Switch, Bridge, Router, Gateway & NIC

- Network devices are required to provide an interface to connect multiple computers in a network.
- There are many types of network devices used in networking. These are Hub, Repeater, Switch, Bridge, Router, Gateway & NIC

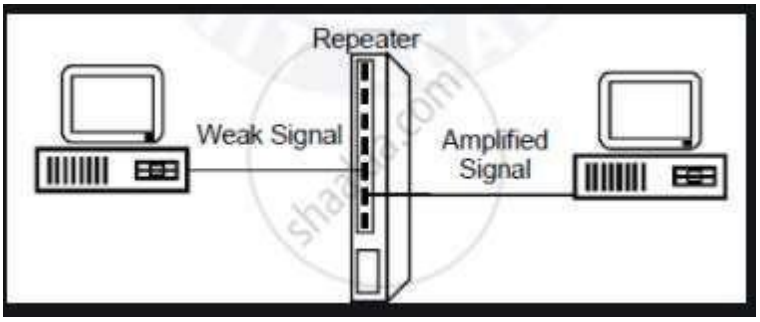
HUB-

- Networking using a star topology requires a central point for the devices to connect.
- It is like a repeater with multiple ports to connect the network channels.



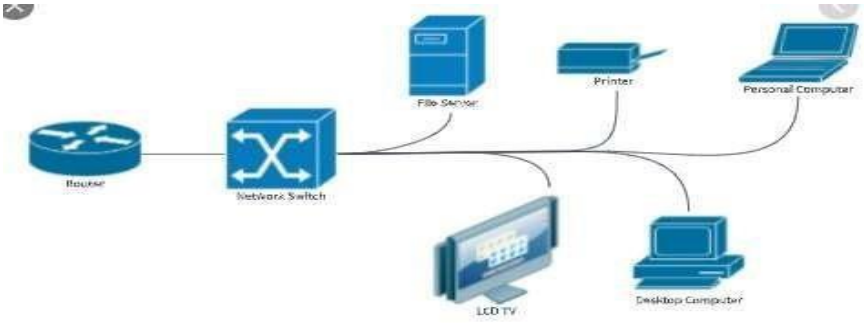
REPEATER

- Repeaters have two ports and can connect two segments of a LAN.
- It is an electronic device that receives a signal and retransmits it.



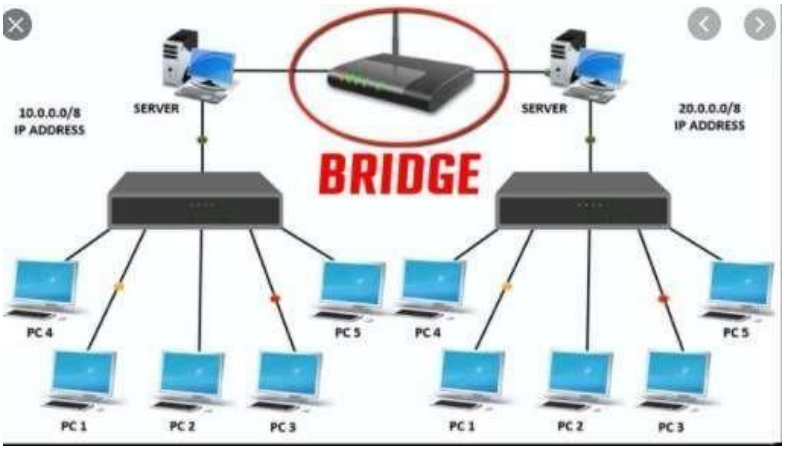
SWITCH

- It is a small hardware device that joins multiple computers together one LAN.
- It helps to reduce overall network traffic.



BRIDGE

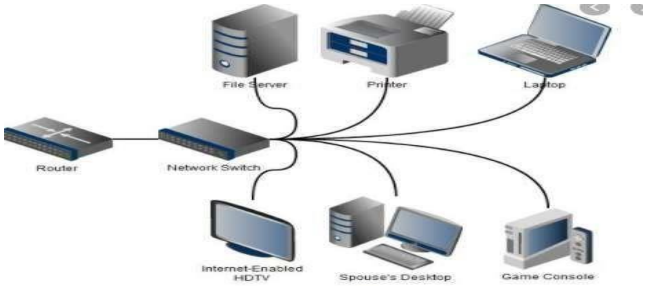
- A bridge is used to join two network segments together; it allows computers on either segment to access resources on the other.
- They can also be used to divided large networks into smaller segment.



- A bridge is used to join two network segments together; it allows computers on either segment to access resources on the other.
- They can also be used to divide large networks into smaller segment.

ROUTER

- Routers are networking devices used to extend or segment packets from one logical network to another.
- Routers are most often used in large internetworks that use LANs to the internet using dedicated leased lines.

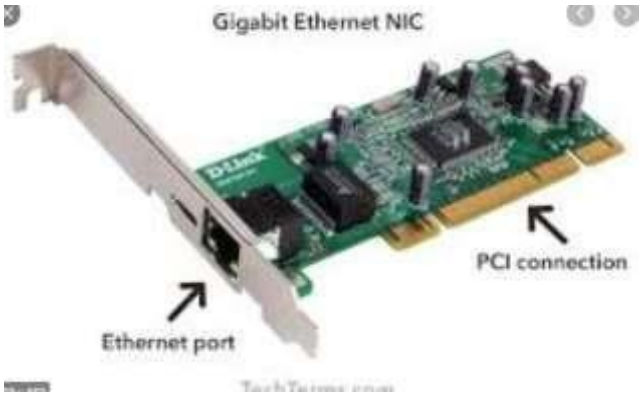


GATEWAY

- It is an internetworking device, which joins two different network protocols together.
- They are also known as protocol converters.

NIC (Network Interface Card)

- It is a hardware Card installed in a computer so it can communicate on a network.
- The network adapter provides one or more ports for the network cable to connect to and it transmits and receives data onto the network cable.



3.4 Internet Services like E-Mail, WWW, FTP, Chatting, Internet Conferencing, Electronic Newspaper & Online Shopping

- Internet is a network of networks that consists millions of private and public network of local to global scope.
- An internet user can access to a wide variety of services such as

E-MAIL (Electronic Mail)

- E-mail is an electronic version of sending and receiving letter.
- To use E-mail, a user must have an Email address.
- The Email address contains all information required to send or receive a message is called mail box.
- Email address consists of two parts separated by @ symbol
  - 1st part is user name
  - 2nd part is host name (domain name) Example:

    principalbose@rediffmail.com

WWW (World Wide Web)

- The World Wide Web is a system of internet servers that supports hypertext and multimedia to access several internet protocols on a single interface.
- It is a way of exchanging information between computers on the internet.

Example:

https://www.google.com/

http://www.bosecuttack.in/

FTP (File Transfer Protocol)

- FTP is the internet file transfer between any computers that have an internet connection and also works between computers using totally different operating systems.

- It is a protocol through which internet users can upload files from their computers to a website or download files from a website to their PC.
- It is the easiest way to transfer files between computers via the internet and utilities TCP/IP systems to perform uploading and downloading tasks.

Chatting

- Chatting is the online textual or multimedia conversation.
- Chatting i.e., a virtual means of communication that involves the sending and receiving of messages, share audio and video between users located in any part of the world.

Internet Conferencing

- Internet conferencing allows users to carry on business meetings and seminars make presentation, provide online education and offer direct customer support.
- Internet conferencing solutions require high speed internet connection at all user site.

Electronic Newspaper

- An electronic newspaper is a self-contained, reusable and refreshable version of a traditional newspaper that acquires and holds information electronically.
- Information to be displayed will be downloaded through some wireless internet connections.

Online Shopping

- It is the process of buying goods and services from merchants who sell on the internet.
- The main components of online shopping are product, selling price, accessibility to people, placement of orders, mode of payments, delivery mechanism.

3.5 Different types of Internet connectivity and ISP

Different types of Internet connectivity

There are different types of connections and speeds to get on the information super high way.

1) Dial-Up Connections

- A dial up is a method of connecting to the internet using an existing telephone.
- Dial up connection uses the telephone line to connect to the internet.
- The modem must dial the telephone every time it wants to connect to the internet hence the name Dial up.

a) Modem Dial-Up Connections-

- The modem connects the computer through the standard phone line which serves as the data transfer medium.
- A modem changes the digital data from your computer into analog data, a format that can be carried by telephone lines.

b) ISDN Dial-Up Connections-

- The second type of dial up connection is through an ISDN (Integrated services digital network).
- It is a digital telephone service that can transmit voice, data and control information over an existing single telephone line.

2) ADSL Connections

- ADSL (Asymmetric Digital Subscribers Line) connections are becoming more and more widely available and can provide an excellent internet connection.
- The connections work by splitting your phone line two separate channels, one for data (Internet) and one for voice (phone calls), which means you can talk on the phone and be connected to the internet at the same time.

3) Cable Connections

- Cable connection are considered one of the best types of internet connections available to the home user, they offer very fast and reliable connections with a fixed monthly fee.
- A cable connection uses a totally separate medium to transfer that it doesn't affect your ability to make/receive phone calls.
- Cable connections are always on, eliminating long wait to make a connection.

ISP (Internet Service Provider)

- When a user initiates a Dial up connection, the modem dials a phone number of an internet service provider (ISP) that is designated to receive Dial up calls.
- The ISP then establishes the connection, which usually takes about ten seconds and is accompanied by several beeping and buzzing sounds.
- ISP refers to the company that provides internet connections to the users.
- Some popular ISP's, are Airtel, MTNL, Vodafone etc.

Solved Questions

Short Answer Type Questions.

Q.1 **Define WWW.** ***(2017-Winter)***

Ans. The World Wide Web, commonly known as the Web, is an information system where documents and other web resources are identified by Uniform Resource Locators, which may be interlinked by hypertext, and are accessible over the Internet.

Q.2 **Define HTTP.**

Ans. The Hypertext Transfer Protocol is an application layer protocol for distributed, collaborative, hypermedia information systems.

Q.3 **Define a DNS.**

Ans. The Domain Name System is a hierarchical and decentralized naming system for computers, services, or other resources connected to the Internet or a private network. It associates various information with domain names assigned to each of the participating entities.

Q.4 **Define a FTP.** ***(2016-Winter)***

Ans. The File Transfer Protocol is a standard network protocol used for the transfer of computer files from a server to a client on a computer network.

Long Answer Type Questions

Q.1 Explain e-mail message format? *(2017-Winter)*

Ans. Electronic Mail (e-mail) is one of the most widely used services of the Internet. This service allows an Internet user to send a message in a formatted manner (mail) to other

Internet users in any part of the world. Message in the mail not only contain text, but it also contains images, audio and videos data. The person who is sending mail is called sender and person who receives mail is called the recipient. It is just like postal mail service.

Format of E-mail:

An e-mail consists of three parts that are as follows:

1. Envelope
2. Header
3. Body

These are explained as following below.

1. Envelope:

The envelope part encapsulates the message. It contains all information that is required for sending any e-mail such as destination address, priority and security level. The envelope is used by MTAs for routing message.

2. Header:

The header consists of a series of lines. Each header field consists of a single line of ASCII text specifying field name, colon and value. The main header fields related to message transport are:

To: It specifies the DNS address of the primary recipient(s).

CC: It refers to carbon copy. It specifies address of secondary recipient(s).

BCC: It refers to blind carbon copy. It is very similar to Cc. The only difference between Cc and Bcc is that it allow user to send copy to the third party without primary and secondary recipient knowing about this.

From: It specifies name of person who wrote message.

Sender: It specifies e-mail address of person who has sent message.

Received: It refers to identity of sender's, data and also time message was received. It also contains the information which is used to find bugs in routing system. Return-Path: It is added by the message transfer agent. This part is used to specify how to get back to the sender.

3. Body:

The body of a message contains text that is the actual content/message that needs to be sent, such as "Employees who are eligible for the new health care program should contact their supervisors by next Friday if they want to switch." The message body also may include signatures or automatically generated text that is inserted by the sender's email system.

Q.2 What do you mean by network topologies? What are the major types of network topologies? ***(2016-Winter) (2015-Winter) (2013-Winter)***

Ans.

There are five types of topology in computer networks: These are

1. Mesh Topology
2. Star Topology
3. Bus Topology
4. Ring Topology
5. Hybrid Topology

Mesh Topology

In mesh topology each device is connected to every other device on the network through a dedicated point-to-point link. When we say dedicated it means that the link only carries data for the two connected devices only. Let's say we have n devices in the network then each device must be connected with (n-1) devices of the network. Number of links in a mesh topology of n devices would be n(n-1)/2.

Advantages of Mesh topology

1. No data traffic issues as there is a dedicated link between two devices which means the link is only available for those two devices.
2. Mesh topology is reliable and robust as failure of one link doesn't affect other links and the communication between other devices on the network.
3. Mesh topology is secure because there is a point to point link thus unauthorized access is not possible.
4. Fault detection is easy.

Disadvantages of Mesh topology

1. Amount of wires required to connected each system is tedious and headache.
2. Since each device needs to be connected with other devices, number of I/O ports required must be huge.
3. Scalability issues because a device cannot be connected with large number of devices with a dedicated point to point link.

Star Topology

In star topology each device in the network is connected to a central device called hub. Unlike Mesh topology, star topology doesn't allow direct communication between devices; a device must have to communicate through hub. If one device wants to send data to other device, it has to first send the data to hub and then the hub transmit that data to the designated device.

Advantages of Star topology

1. Less expensive because each device only need one I/O port and needs to be connected with hub with one link.
2. Easier to install
3. Less amount of cables required because each device needs to be connected with the hub only.
4. Robust, if one link fails, other links will work just fine.

5. Easy fault detection because the link can be easily identified.

Disadvantages of Star topology

1. If hub goes down everything goes down, none of the devices can work without hub.
2. Hub requires more resources and regular maintenance because it is the central system of star topology.

Bus Topology

In bus topology there is a main cable and all the devices are connected to this main cable through drop lines. There is a device called tap that connects the drop line to the main cable. Since all the data is transmitted over the main cable, there is a limit of drop lines and the distance a main cable can have.

Advantages of bus topology

1. Easy installation, each cable needs to be connected with backbone cable.
2. Less cables required than Mesh and star topology

Disadvantages of bus topology

1. Difficultly in fault detection.
2. Not scalable as there is a limit of how many nodes you can connect with backbone cable.

Ring Topology

In ring topology each device is connected with the two devices on either side of it. There are two dedicated point to point links a device has with the devices on the either side of it. This structure forms a ring thus it is known as ring topology. If a device wants to send data to another device then it sends the data in one direction, each device in ring topology has a repeater, if the received data is intended for other device then repeater forwards this data until the intended device receives it.

Advantages of Ring Topology

1. Easy to install.
2. Managing is easier as to add or remove a device from the topology only two links are required to be changed.

Disadvantages of Ring Topology

1. A link failure can fail the entire network as the signal will not travel forward due to failure.
2. Data traffic issues, since all the data is circulating in a ring.

Hybrid topology

A combination of two or more topology is known as hybrid topology. For example a combination of star and mesh topology is known as hybrid topology.

Advantages of Hybrid topology

1. We can choose the topology based on the requirement for example, scalability is our concern then we can use star topology instead of bus technology.
2. Scalable as we can further connect other computer networks with the existing networks with different topologies.

Disadvantages of Hybrid topology

1. Fault detection is difficult.
2. Installation is difficult.
3. Design is complex so maintenance is high thus expensive.

EXERCISE

Short Answer Type Questions.

Q.1 Define computer network. ***(2014-Summer)***
Q.2 What are the various types of networks?
Q.3 What is a MAN?
Q.4 What are the different topologies used in computer network? ***(2018-Summer)***
Q.5 Name some Internet service providers in India?
Q.6 What do you mean by a web page?
Q.7 What is a website?
Q.8 What is e-mail? What are the uses of e-mail?
Q.9 What are the benefits of e-mail?
Q.10 What are the limitations of e-mail?
Q.11 Define a URL. ***(2013-Summer)***
Q.12 What do you mean by Network topology? ***(2016-Winter)***
Q.13 What are the advantages and Disadvantages of Bus Topology?
Q.14 What is a modem?

Long Answer Type Questions

Q.1 Discuss about the various categories of computer network. Give a comparisons between LAN and WAN.
Q.2 Describe various categories of network with example. ***(2016-Summer)***

****

CHAPTER – 4: FILE MANAGEMENT AND DATA PROCESSING

4.1 Concept of File and Folder

Concept of file and folder: -

- In earlier DOS based system, we had organized data into files and directories.
- In GUI based operating system, such as window, we have file and folders, in which data are organized during storage in computer memory.
- However, the unit of raw data in binary format is either byte (B) or kilobyte(KB) or megabyte(MB) or gigabyte (GB).
- A byte is smallest unit of information. It is used to measure the size of our documents.

1Byte=1B=8bits

1KB=2^{10}Bytes=1024Bytes

1MB= 2^{10}KB= 1024KB

1GB=2^{10}MB= 1024MB

1TB=2^{10}GB= 1024GB

Files:

- Files are the most basic unit of data that user can store on a disk. A file is the common storage unit in a computer.
- All program and data are contained in a file and the computer needs and writes files.
- In every program, image, video, song and document are stored in a file.
- It is possible to move a file from one folder to another.
- One can create, save, open, move and delete files.
- There are different types of files depending on the type of information they contain. There are image files etc.
- The files are assigned a type of file which can be known from the extension of the file name.
- The file name can have up to 255 character, it can contain letter, number, blank space and special characters like dashes, underlines, etc. but there is a group that cannot be used (“, /,
- >, <, |)
- File extension files are identified by a short "extension" at the end of their name.
- Ex: Soumya.jpg is a JPEG image
- Chandan.doc is a Microsoft word doc
- ABC.exe is an executable application in windows.

Folders:

- A folder is a collection of multiple files.
- A folder holds one or more files and it can be empty with just a name.
- Folder can also store other folders called subfolders.
- Folders were also called "directories" in operating systems before windows.
- It would become almost possible to manage hundreds of files in your computer.

Different between a file and a folder:

File:

- File store data, whether text, music or item.
- Files are taking spaces on computer memory.
- Each file has its own extension.
- Easily move or copy data from one file to another.
- Cannot create any folder or subfolder with in a file.

Folder:

- A folder stores files and other folders.
- Folders are not taking space on computer memory.
- Folders do not have any extension.
- Copy or move files from one folder to another folder.
- Can create different types of files or sub folders in a folder.

4.2 File Access and Storage methods. Sequential, Direct, ISAM

File Access and storage method:

- An access method defines the technique that is used to store and receive data.
- An access method is a function of a main frame operating system that enableaccess to data on click or other external device.
- In computing an access method is a program or a hardware mechanism thatmoves the data between the computer and an outing device such as a hard diskor a display terminal.
- It is also used to describe the way that data is located within a large unit of datasuch as a data set or files.
- There are various types of access methods

1) Random Access or Direct Access:

- Direct access method is based on a disk model if a file, since disks allowrandom access to any file block.
- This types of access method provides a speedy access to the file. It provides immediate access to large amount of information.
- It allows the programs to read and write the records in a rapid manner in on particular orders.
- For direct access, we can view the file as a number sequence of blocks or records.
- This method is usually used in database.

2) Sequential Access:

- This is the most common method.
- Here the information present in the file is accessed in a sequential fashion, on record after the other.
- It is very common approach which is used by editors and compiler usually.
- The Read and write operation from the major part of the operations done on a file.
- A read operation reads the next portion of the file and automatically advances the file pointer, which tracks the I/O Locations.
- A write operation appends to the end of files and advances to the and, if the newly written material.

3) Indexed Sequential Access Method (ISAM)-

- This method is built on top of direct access method.
- Here an index contains the pointers to various blocks of the file.
- So, to find a record inside a file, we firstly search the index and later use the pointer obtained to access the file directly and find the record we have been searching for.
- The records of the data file are stored in sequential order according to some data attributes.
- Since ISAM is static, it doesn't change its structure if records are added or deleted from the data file.
- ISAM is available in many variations on microcomputer, mini computers, and main frame computers.

4.3 Data Capture, Data storage, Data Processing and Retrieval

4.3.1 Data Capture: -

- Data capture is the process of identification and extraction of data from ascanned document, often to be sent to a workflow for routing and action aspart of business process.
- Multiple methods are available for capturing data from unstructured documents (letters, invoices, email, fax, forms etc.)
- Methods of capture from documents in electronic format are identified below:

Single click:

- It is an optical character recognition (OCR) tool that can be used to capture machine produced characters in low volume ad-hoc capture application and populating a line of business application.

OCR:

- OCR as technology provides the ability to successfully capture machine produced characters in full page.
- OCR systems can recognize many different OCR fonts as well as typewriter and computer printed characters.

ICR (Intelligent character recognition):

- ICR is the computer translation hand printed and written characters.
- Data is entered from hand printed forms through a scanner and the image of the capture data is then analyzed and then translated by sophisticated ICR software.

Barcode Recognition:

- Dependent upon the type of barcodes that is used, the amount of meta data that can be included is high, as is the level of recognition.
- The application of single and multiple bar codes to particular document types such as proof of delivery notes.

IDR (Intelligent document recognition):

- The level of capability is dependent upon the individual product.
- These application are used to capture metadata from documents that is ruled based.
- Ex: The product will identify post codes, logos, keywords.

4.3.2 Data Storage:

- Data storage is the holding of data in an electromagnetic from for access by a computer processor.
- There are two kinds of storage:

a) Primary storage is data that is held in RAM and other memory devices that are built into computers.
b) Secondary storage is data that is stored on external storage devices such as hard disks, tapes, CD, Pen drive etc.

Following are some main devices for data storage:

- Hard disks
- Floppy disks
- Optical disks
- CD
- Pen drives
- Flat memory card/memory card

4.3.3 Data Processing:

- Data processing must be processed in order to convert it into information.
- For this purpose, different operations may be performed on data.
- Data processing is defined as a sequence of operations on data to convert it into useful information.
- The data processing can be accomplished through following methods:

1. Manual Data Processing:

- In this method, data is processed manually without using any machine or tool to get required results.
- In manual data processing, all the calculations and logical operations are performed manually on the data.
- Ex: Mark sheets, fee receipts

2. Mechanical Data Processing:

- In this method, data is processed by using different devices like type writers, mechanical printers or other mechanical devices.
- Examination board and printing press use mechanical data processing devices frequently.

3. Electronic Data Processing:

- It is the modern technique to process data.
- The data is processed through computer; data and set of instructions are given to the computer as input and the data according to the given set of instructions.
- The computer is also known as electronic data processing machine.
- Ex: results of students are prepared through computers.

4.3.4 Data Retrieval:

- Data is one of the most important assets of any business.
- Data recovery refers to the whole process of salvaging this lost data that is corrupted, failed, damaged or inaccessible.
- Lost files can occur because of any of the below possibilities.

1) File was mistakenly deleted.

2) File was corrupted or deleted by scandisk.

3) Another program deleted the file.

4) File is password protected.

- Following are some different methods of data recovery: -

1) Physical damage to storage devices:

- Different failure can cause physical damage to your storage media.

2) Media errors and corrupt partitions and file systems:

- In some cases, media errors or damage to the file system or partition table can make the data on a hard drive to be unreadable.

3) Online data recovery:

- This is another popular method of data recovery Sydney business use to restore deleted or lost files.

- It is a method of data recovery that is performed over the internet without necessarily having the computer or the drive in possession.

Solved Questions

Short Answer Type Questions.

Q. 1 What is the difference between file and folder.

Ans:-.

Sl. No.	Key	File	Folder
1	Extension	Files may or may not have extensions.	Folders do not have extensions.
2	Container	A File can not contain another file/folder.	A folder can contain any number of file/folders.
3	Memory size	A file has certain size and memory consumption.	A folder has no size of its own. It derives the size from the files it contains.
4	Attributes	Name, Extension, Date, Time, Length and Protection (Read-Only, hidden etc.)	Name, Date, Time and Protection (Read-Only, hidden etc.)

Q. 2 What do you mean by ISAM? ***(2016-Winter)***

Ans:-i.

I. It stands for Indexed Sequential Access Method.

II. ISAM is a method for creating, maintaining, and manipulating computer files of data so that records can be retrieved sequentially or randomly by one or more keys.

III. In this method, each record has the address of its data block, searching a record in a huge database is quick and easy.

IV. This method supports range retrieval and partial retrieval of records.

Q. 3 Define OCR.

Ans:-

I. Stands for "Optical Character Recognition.
II. OCR is a technology that recognizes text within a digital image.
III. It is commonly used to recognize text in scanned documents, but it serves many other purposes as well.

Q. 4 What is folder. ***(2017-Winter)***

Ans:-A Folder is a collection of tables, charts, or other outputs in the Report tree.

- It is the same idea as having folders on your computer.
- They created by right-clicking on the Report tree and selecting Add Folder.
- Folders are also called directories because of the way they organize data within the file system of a storage device.

Long Answer Type Questions.

Q. 1 What is file access? Explain the various type of file access method. ***(2014-Winter)*** ***(2015- Winter)***

Ans:- A file access definition can control access to data in specified tables and columns, or to tables and columns for which access is not granted explicitly. You define access permissions by creating an access list for a table, column, or the default.

When a file is used, information is read and accessed into computer memory and there are several ways to access this information of the file. Some systems provide only one access method for files.

There are three ways to access a file into a computer system: Sequential-Access, Direct Access, Index sequential Method.

1. **Sequential Access –**

   Data is accessed one record right after another record in an order. When we use read command, it move ahead pointer by one. When we use write command, it will allocate memory and move the pointer to the end of the file. Such a method is reasonable for tape.

2. **Direct Access –**
   Another method is direct access method also known as relative access method. A filed- length logical record that allows the program to read and write record rapidly. in no particular order. The direct access is based on the disk model of a file since disk allows random access to any file block. For direct access, the file is viewed as a numbered sequence of block or record. Thus, we may read block 14 then block 59 and then we can write block 17. There is no restriction on the order of reading and writing for a direct access file.
3. **Index sequential method –**
   It is the other method of accessing a file which is built on the top of the sequential access method. These methods construct an index for the file. The index, like an index in the back of a book, contains the pointer to the various blocks. To find a record in the file, we first search the index and then by the help of pointer we access the file directly.

Q. 2 Discuss about the Data storage devices.

Ans:- A storage device is a piece of computer hardware used for saving, carrying and pulling out data. It can keep and retain information short-term or long-term. It can be a device inside or outside a computer or server. Other terms for storage device is storage medium or storage media.

A storage device is one of the basic elements of any computer device. It almost saves all data and applications in a computer except for hardware firmware. It comes in different shapes and sizes depending on the needs and functionalities.

There are two different types of storage devices:

	Primary Storage Device	**Secondary Storage Device**
Size	Smaller	Larger
Data Retention	Temporary	Permanent
Location	Internal	Internal / External
E.g	RAM, ROM	Magnetic Storage Device, Floppy diskette, Hard drive, Magnetic strip , Cassette tape, etc.

EXERCISE

Short Answer Type Questions.

Q.1 What is the difference between Random Access method and sequential access method?
(2017-Winter)

Q.2 What is Data processing and discuss about some operations that can be performed on data?

Long Answer Type Questions.

Q.1 What do you mean by file access? Explain the various types of file access techniques. ***(2017- Summer)***

Q.2 Discuss about the Data Retrieval.

****

CHAPTER –5: PROBLEM SOLVING METHODOLOGY

Problem solving

Solving problems is the core of computer science. Programmers must first understand how a human solves a problem, then understand how to translate this "algorithm" into something a computer can do, and finally how to "write" the specific syntax (required by a computer) to get the job done. It is sometimes the case that a machine will solve a problem in a completely different way than a human.

Computer Programmers are problem solvers. In order to solve a problem on a computer you must:

1. Know how to represent the information (data) describing the problem.
2. Determine the steps to transform the information from one representation into another

5.1 Algorithm, Pseudo code and Flowchart

5.1.1 Algorithm

An algorithm is a set of specific steps to solve a problem. Think of it this way: if you were to tell your 3 year old niece to play your favorite song on the piano (assuming the niece has never played a piano), you would have to tell her where the piano was, and how to sit on the bench, and how to open the cover, and which keys to press, and which order to press them in, etc, etc, etc.

Definition:

- An algorithm is a well-defined procedure that allows a computer to solve a problem.
- Algorithm is defined as the step-by-step solution of problem in user's language.
- It is considered as an effective procedure for solving a problem in finite number of steps.
- Another way to describe an algorithm is a sequence of unambiguous instructions.
- In fact, it is difficult to think of a task performed by your computer that does not use algorithms.

The characteristics of Algorithm are

- Precise
- Unambiguous
- Finite termination
- Unique solution

Example:

1. Algorithm to find out sum of two numbers to be taken as input.

   Step-1 Read the 1st number x
   Step-2 Read the 2nd number y
   Step-3Sum=x+y
   Step-4 Print Sum

   This is an example where only sequence is exhibited

2. Algorithm to find out larger between two numbers to be taken as input.

   Step-1 Read the 1st number x
   Step-2 Read the 2nd number y
   Step-3 If x > y
   Then Print x
   Else if x<y
   Then Print y
   Else Print "Both are Equal "

   This is an example where Branching is exhibited

3. Algorithm to find out sum of first 10 natural numbers.
   Step-1 i=1, Sum=0
   Step-2 Repeat step 3 and 4 while i<>10
   Step-3 Sum= Sum+i
   Step-4 i=i+1 Step-5 Print Sum

   This is an example where Repetition is exhibited

5.1.2 Pseudocode

It is a concise description algorithm in English language that uses programming language constructs. It contains outlines of the program that can be easily converted to program. It focuses on the logic of the algorithm without giving stress on the syntax of programming language. This is meant for understanding the logic of the program easily. Flowchart can be considered as an alternative to pseudo code. Several constructs/key words of programming language can be used in the algorithm to write the pseudo code.

Some of them are

If ... Endif
Do while ... end do
While ... end while
Repeat ... until
For ... end for
Case end case
Call
Return

5.1.3 Flowchart

Flowchart is a graphical or symbolic representation of the process of solution to a problem or algorithm. It helps to visualize the complex logic of the solution of the problem in a simplified manner through diagrammatic representation. Each step of the algorithm is presented using a symbol and a short description. The different symbols used for the flowchart are

Symbol	Purpose	Description
	Flow line	Indicates the flow of logic by connecting symbols.
	Terminal(Stop/Start)	Represents the start and the end of a flowchart.
	Input/output	Used for input and output operation.
	Processing	Used for arithmetic operations and data-manipulations.
	Decision	Used for decision making between two or more alternatives.
	On-page Connector	Used to join different flowline
	Off-page Connector	Used to connect the flowchart portion on a different page.
	Predefined Process/Function	Represents a group of statements performing one processing task.

5.2 Generation of Programming Languages

Programming Language

Programming language is a tool to express the logic or instructions for understanding of the computer. Any programming language has two components:

a) Syntax
b) Semantics

Syntax refers to the rules to be followed for writing valid program statements. Compiler can detect errors in syntax while compiling the program. Semantics is associated with logic of the program. Compiler cannot detect the semantic error. The user of programmer can diagnose semantic error.

There are good numbers of High level languages, each meant for specific area of data processing. Commonly known languages are BASIC, FORTRAN, COBOL, Pascal, C, C++ etc. While FORTRAN is good for Numerical and scientific calculation, COBOL is good for Business applications involving large amount of data handling.

Generations of Programming Language

The Programming languages can be classified into 4 generations:

1stGeneration: Machine Language

2ndGeneration: Assembly Language

3rdGeneration: High Level Language

4thGeneration: Very High Level Language

Machine Level language contains instructions in binary form i.e. in 0s and 1s. Thus writing instruction was very difficult and needs heavy expertise. This was used in early days computers.

Assembly level language instructions were written using symbolic codes known as mnemonics. In comparison to Machine language, it is relatively easier to write program, but still it requires lot of expertise. A translator called assembler is used to translate assembly language program to machine level language.

High level language contains instructions in English like words so that user will feel easier to formulate and write the logical statements of the program. Here the logic may spread over multiple statements as against a single statement in assembly language. It uses a translator called compiler for translation of High level language program to machine level language program. There are many High level languages used for programming such as BASIC, FORTRAN, COBOL, PASCAL, C, C++ etc.

Very High Level language otherwise called as 4GL uses nonprocedural logical statements.

A typical example of 4GL is the query language such as SQL.

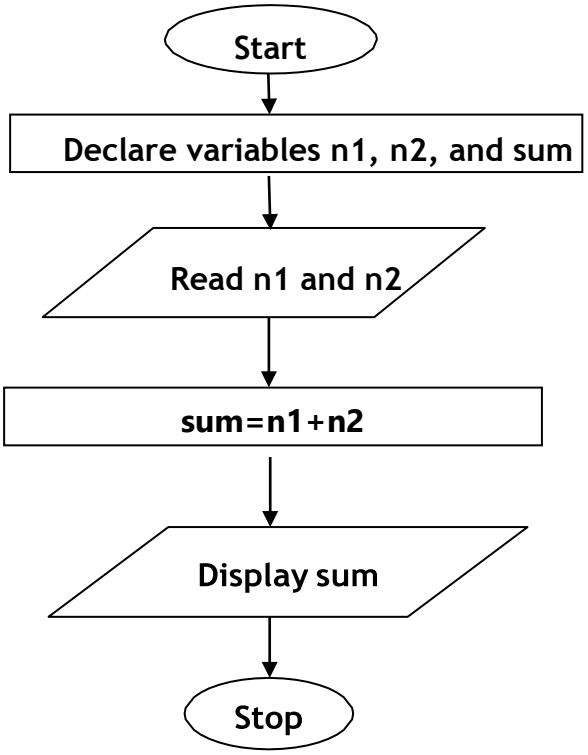
5.3 Structured Programming Language

Structured Programming is also known as Modular Programming. In this type of programming technique, the program shall be broken into several modules. This helps in managing memory efficiently as the required module of the program will be loaded into the memory only and not the entire program. This will also enhance code reuse. Writing, understanding, debugging and modifying the individual module of the program is also easier.

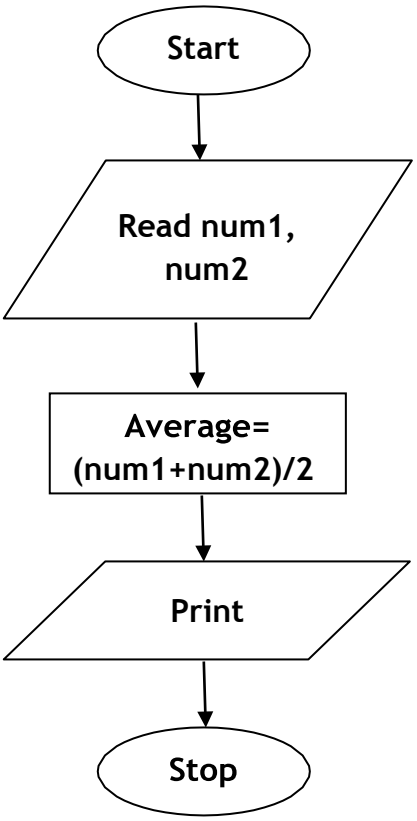
5.4 Examples of Problem solving through Flowchart

Example

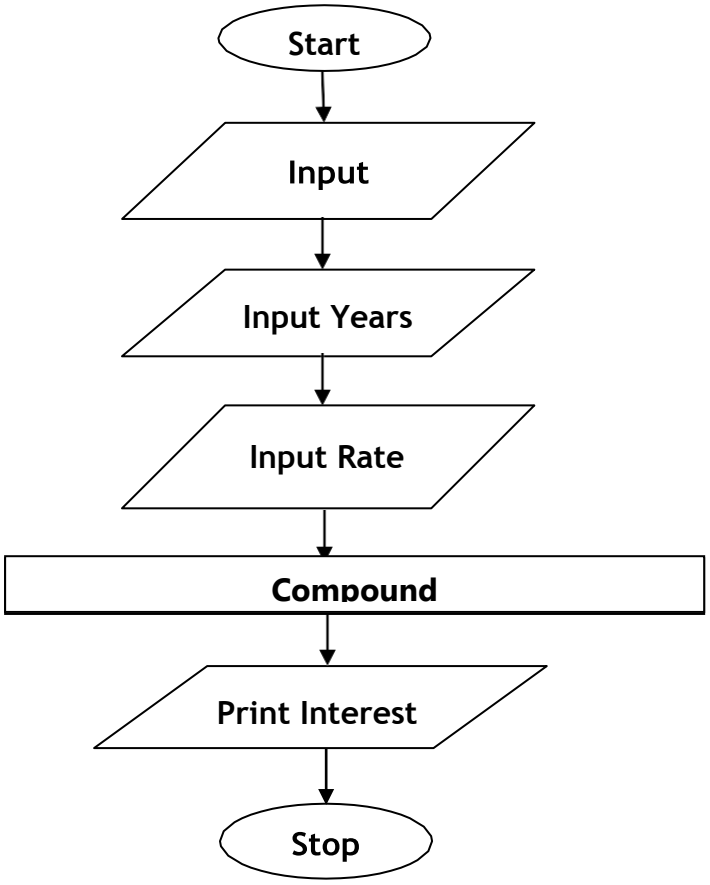
1. **Add two numbers entered by the user.**



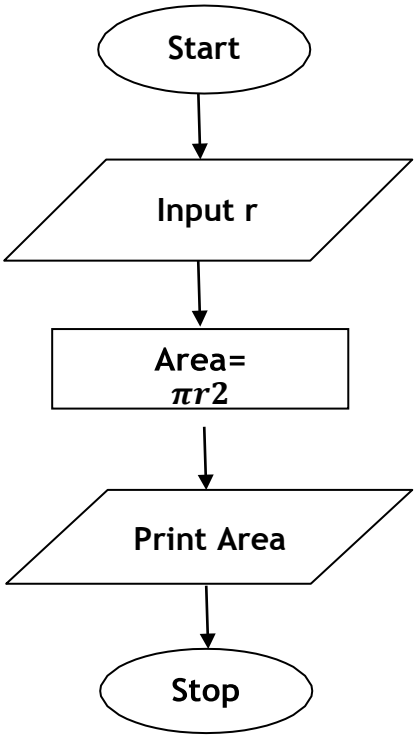
2. **Flowchart to calculate the average of two numbers**



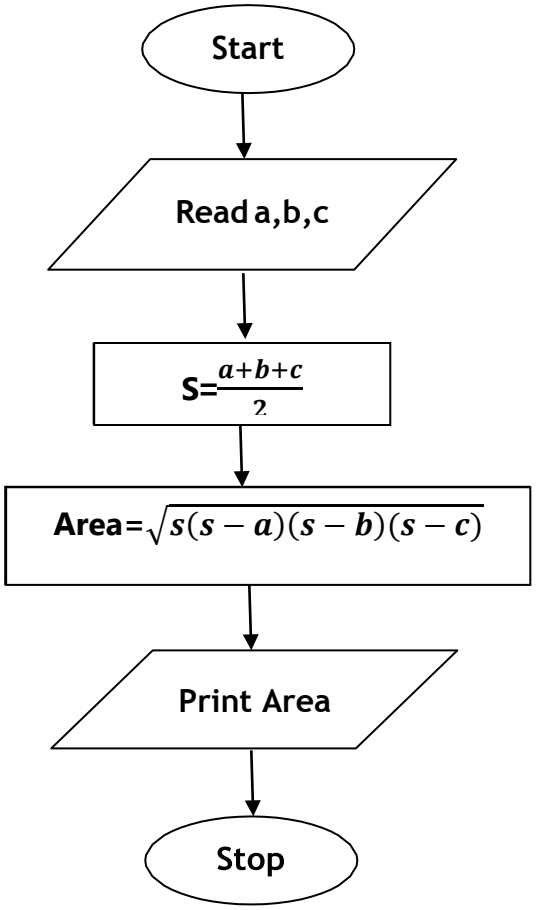
3. **Flowchart to Calculate the Interest of a Bank Deposit**



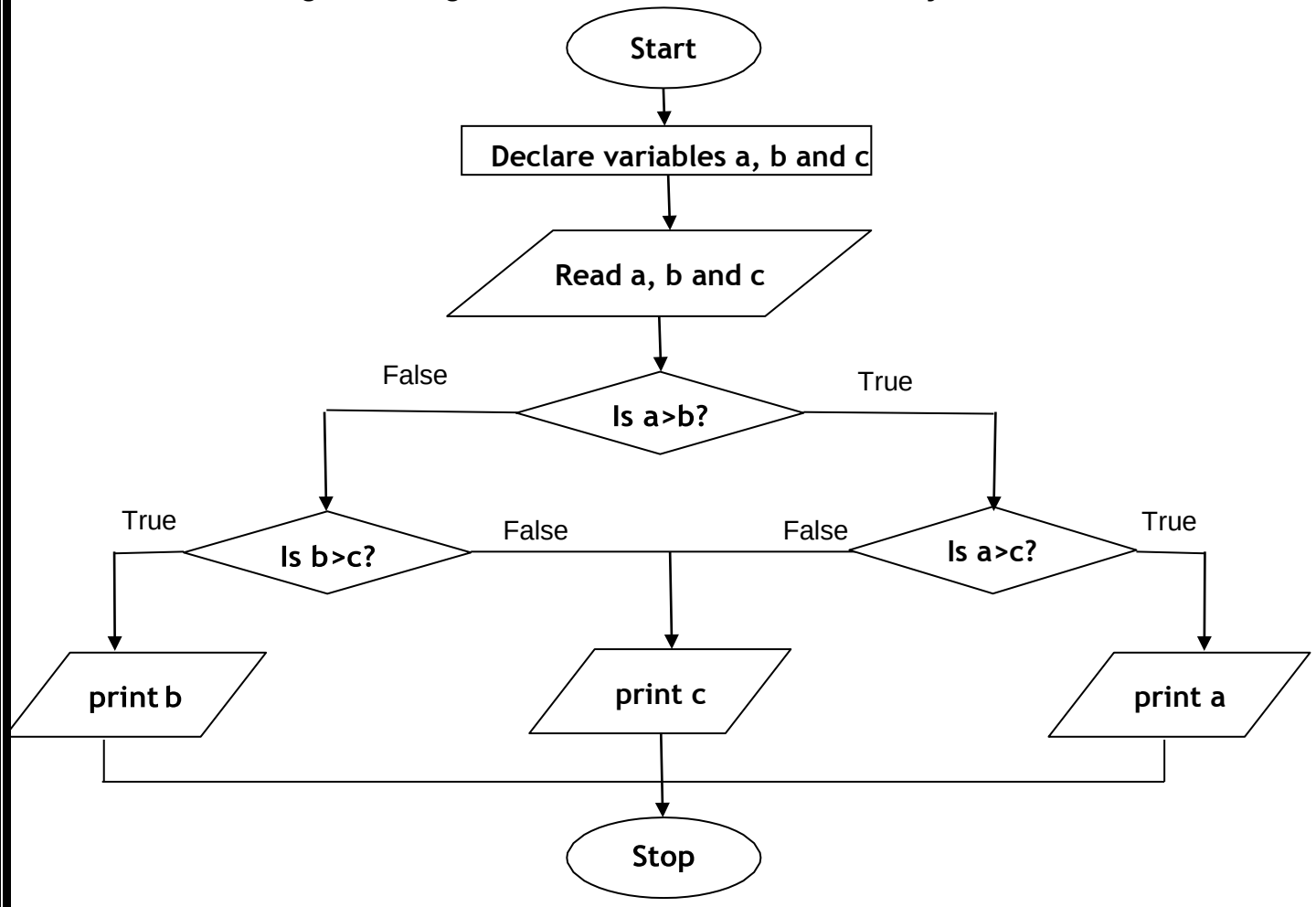
4. **Flowchart to calculate the area of a circle.**



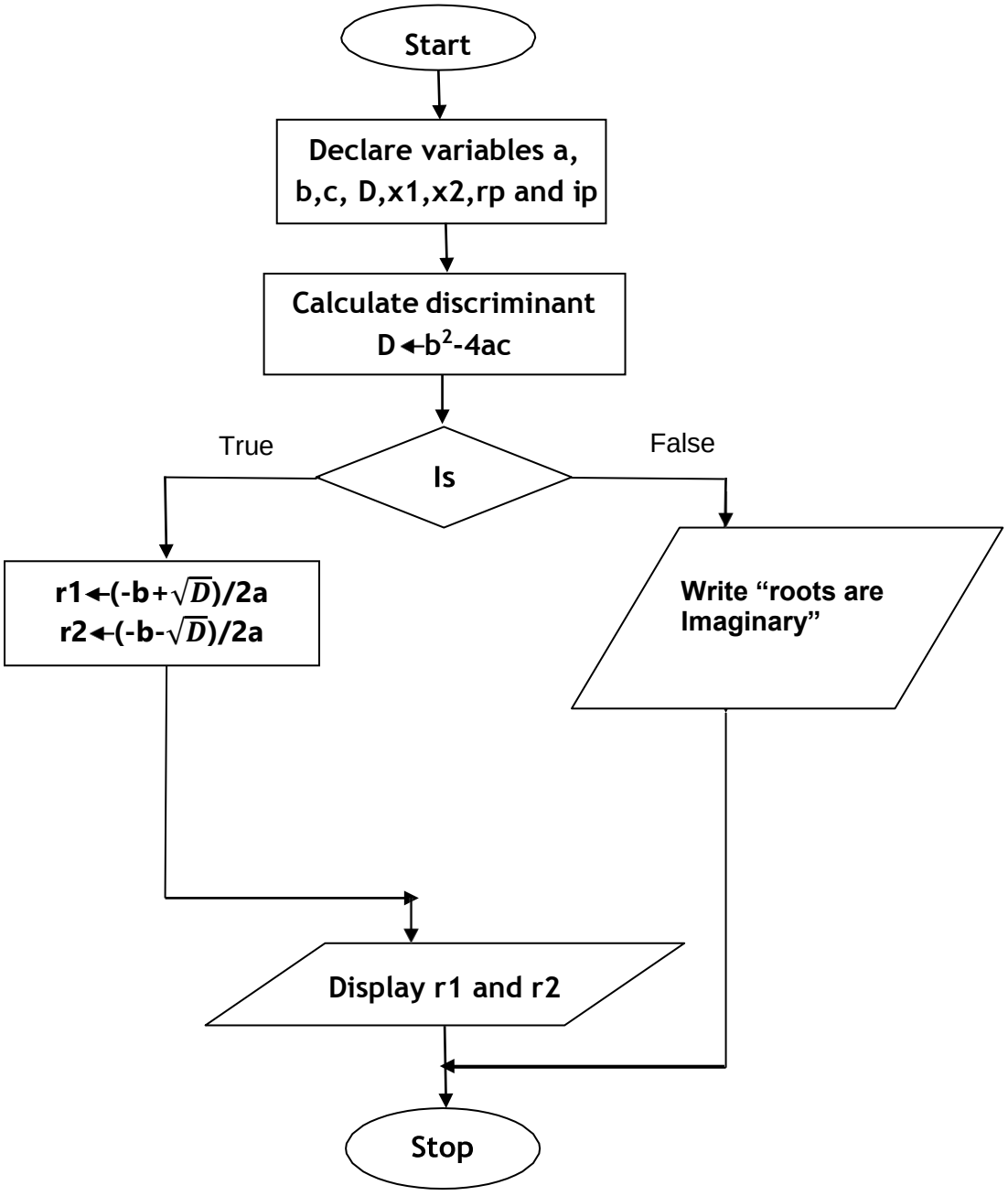
5. **Flowchart to calculate the area of a triangle.**



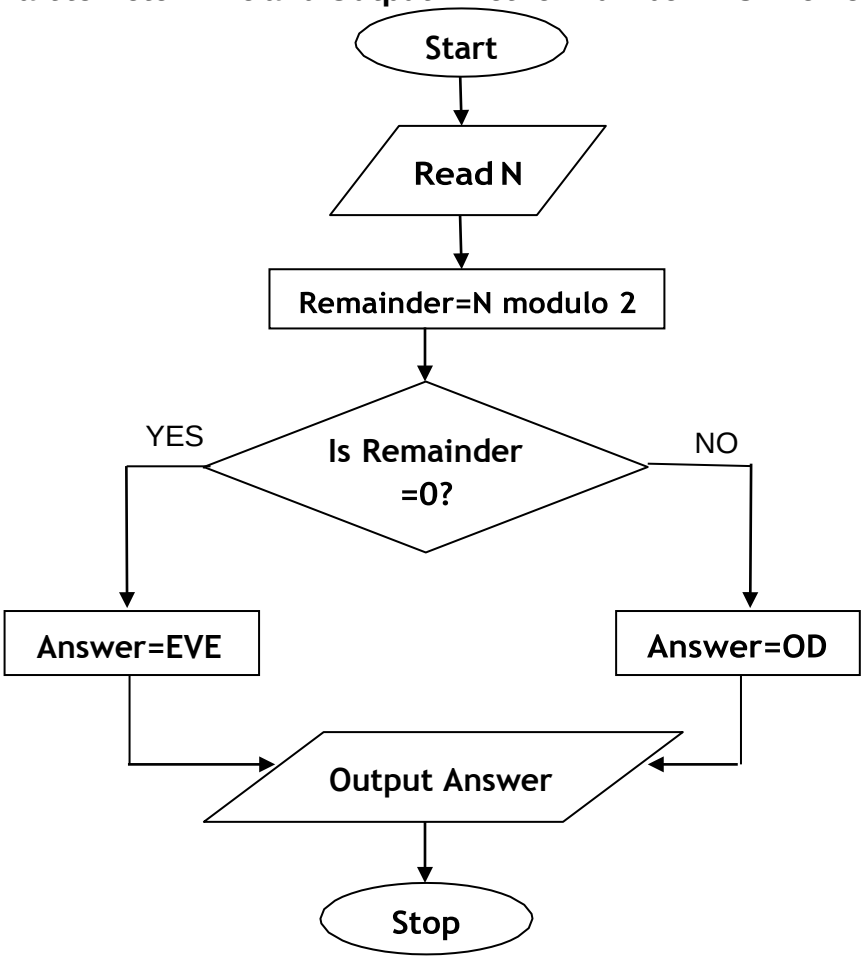
6. **Find the largest among three different numbers entered by the user.**



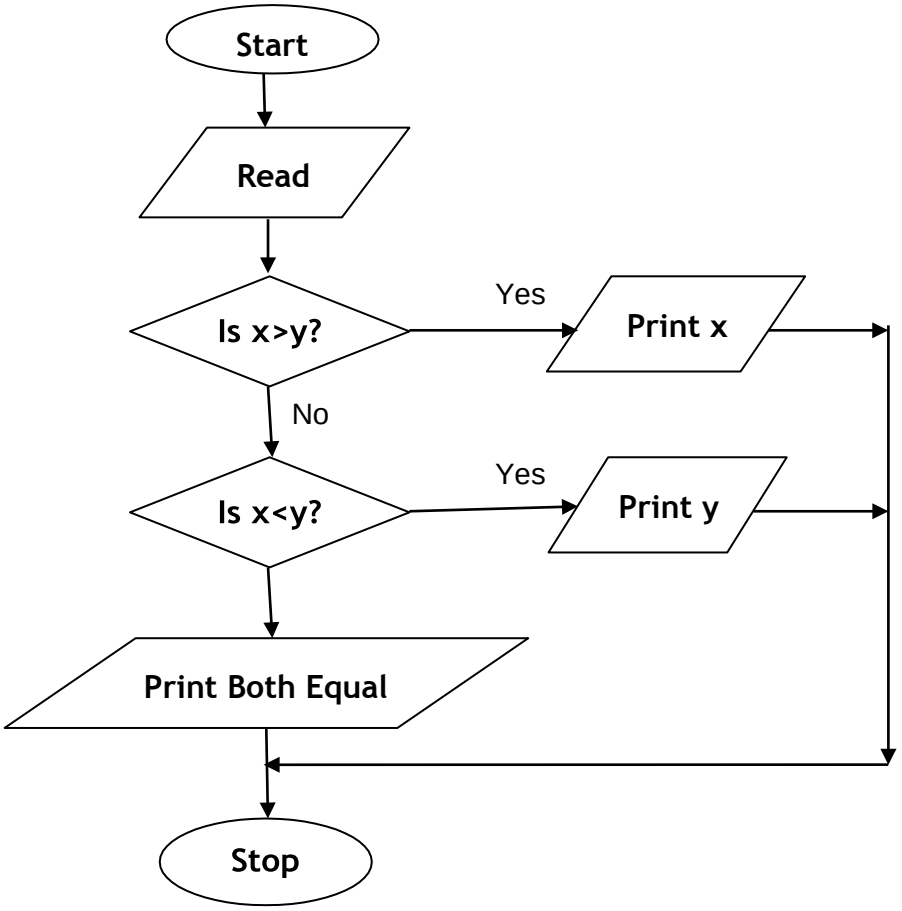
7. **Find all the roots of a quadratic equation $ax^2+bx+c=0$**



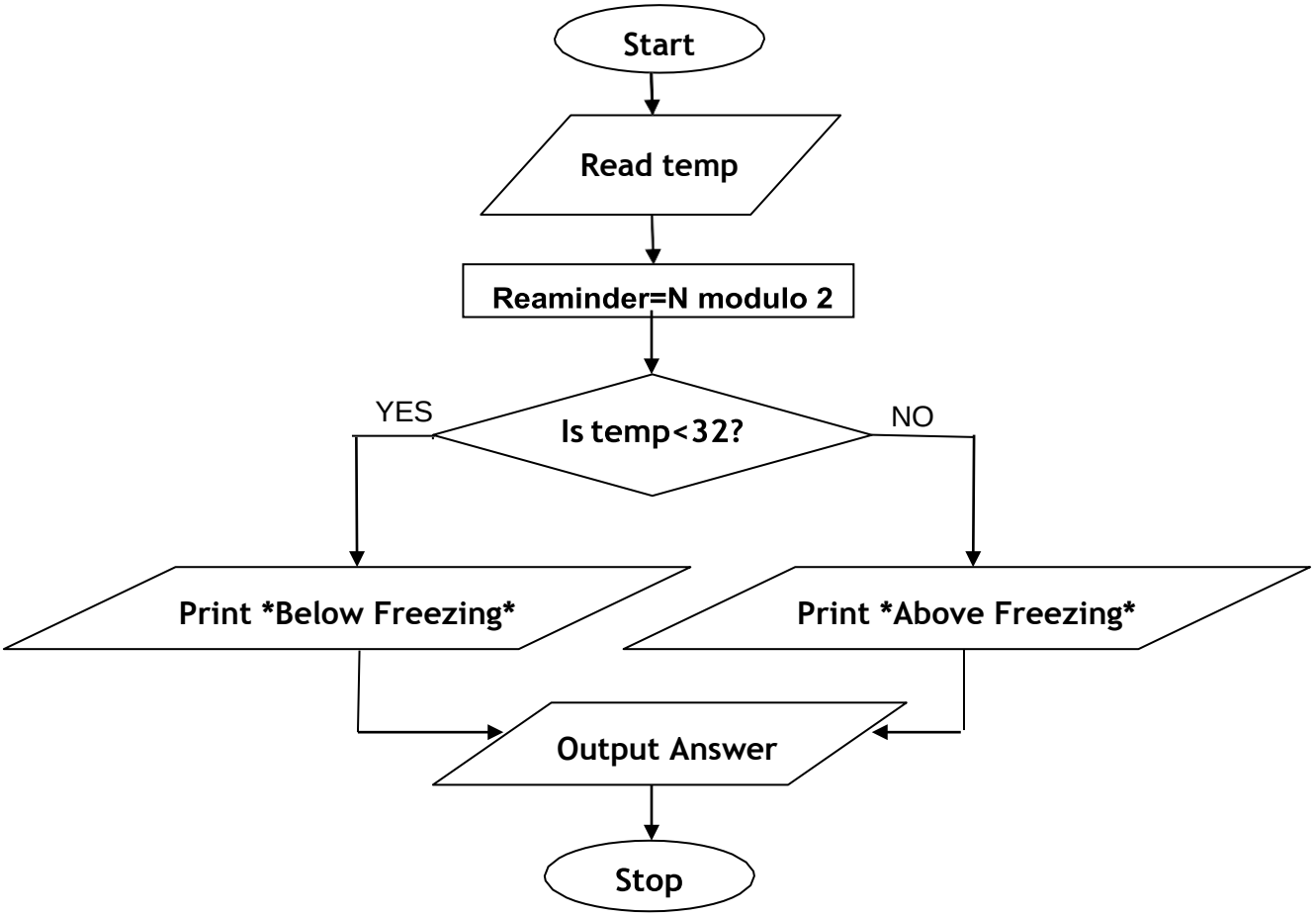
8. **Flowchart to Determine and Output Whether Number N is Even or Odd**



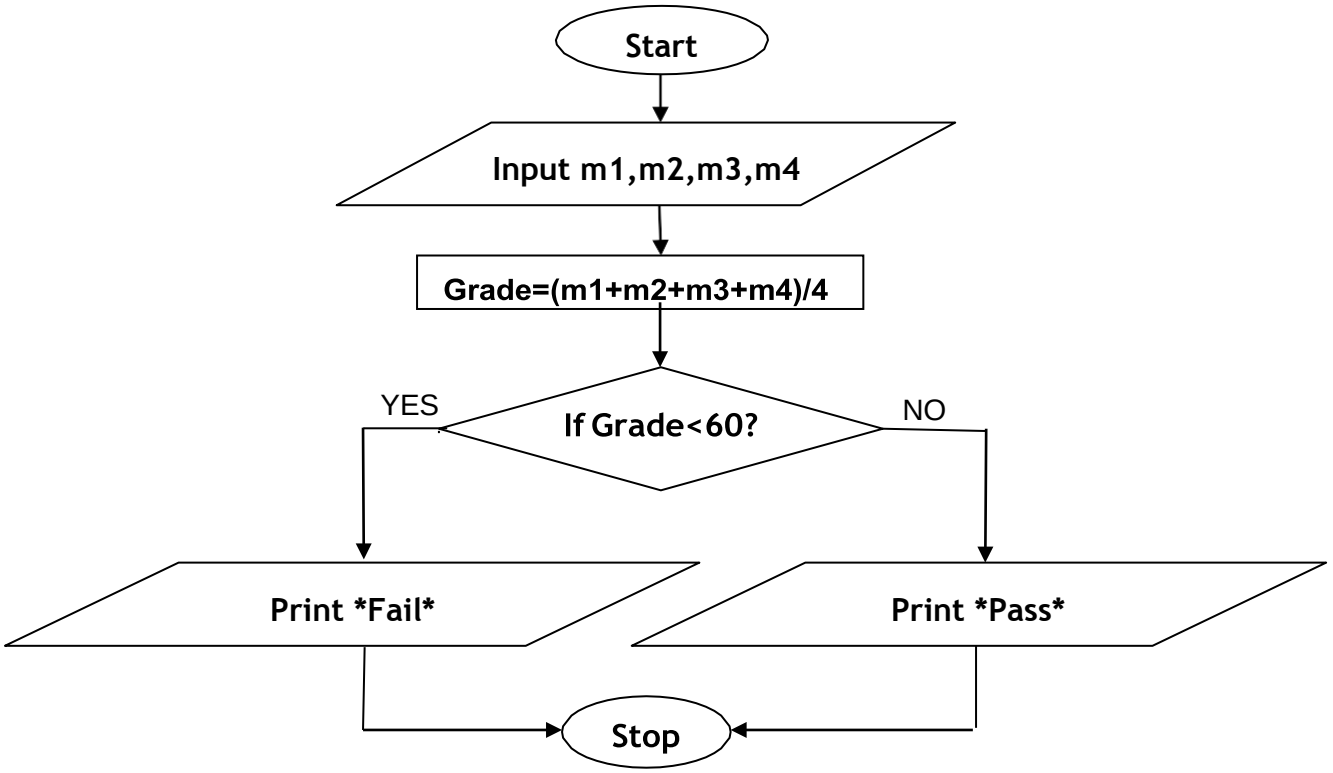
9. **Flowchart to find out larger between two numbers to be taken as input**



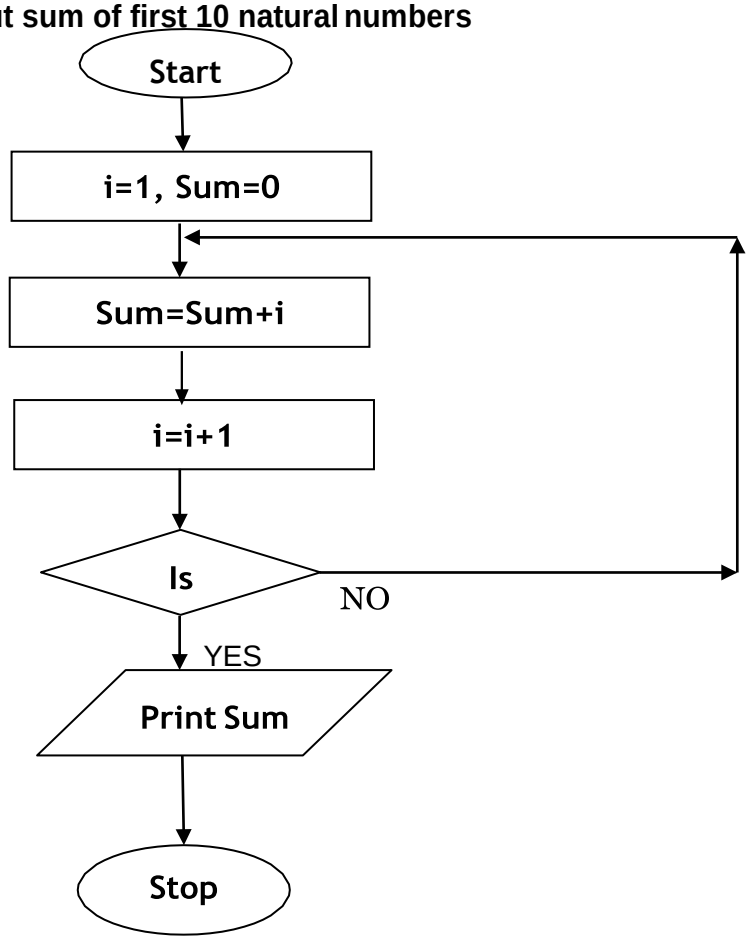
10. Flowchart to Determine Whether a Temperature is Below or Above the Freezing Point



11. Flowchart to Determine Whether A Student Passed the Exam or Not



12. Flowchart to find out sum of first 10 natural numbers



Solved Questions

Short Answer Type Questions.

Q. 1 Define an algorithm. ***(2013-Winter)***

Ans:- An algorithm is a set of instructions designed to perform a specific task. This can be a simple process, such as multiplying two numbers, or a complex operation, such as playing a compressed video file. Algorithm is defined as the step-by-step solution of problem in user's language. It is considered as an effective procedure for solving a problem in finite number of step.

Q. 2 What are the characteristics of an Algorithm?

Ans:-

1. **Input specified:**
   The input is the data to be transformed during the computation to produce the output. An algorithm should have 0 or more well-defined inputs.

2. **Output specified:**
   The output is the data resulting from the computation (your intended result). An algorithm should have 1 or more well-defined outputs, and should match the desired output.

3. **Effectiveness:**
   For an algorithm to be effective, it means that all those steps that are required to get to output must be feasible with the available resources.

4. **Independent:**

An algorithm should have step-by-step directions, which should be independent of any programming code. It should be such that it could be run on any of the programming languages.

Q. 3 What is the function of a Flow Chart? What information does it depict?

Ans:-

- A flowchart describes the steps software takes to process information, from the beginning data inputs, through processing and logical decisions, to the point where the program ends.
- Software developers use flowcharts to plan out how computer applications work before programmers write the code.
- It helps to visualize the complex logic of the solution of the problem in a simplified manner through diagrammatic representation
- Each step of the algorithm is presented using a symbol and a short description.

What information It depicts.:-

A flowchart is a picture of the separate steps of a process in sequential order. It is a generic tool that can be adapted for a wide variety of purposes, and can be used to describe various processes, such as a manufacturing process, an administrative or service process, or a project plan.

Q. 4 Differentiate between algorithm and flowchart ***(2015-Winter)*** **Ans:-**

S.NO	Algorithm	Flowchart
1.	Algorithm is step by step procedure to solve the problem.	Flowchart is a diagram created by different shapes to show the flow of data.
2.	Algorithm is complex to understand.	Flowchart is easy to understand.
3.	In algorithm plain text are used.	In flowchart, symbols/shapes are used.
4.	Algorithm is easy to debug.	Flowchart it is hard to debug.
5.	Algorithm is difficult to construct.	Flowchart is simple to construct.

Long Answer Type Questions.

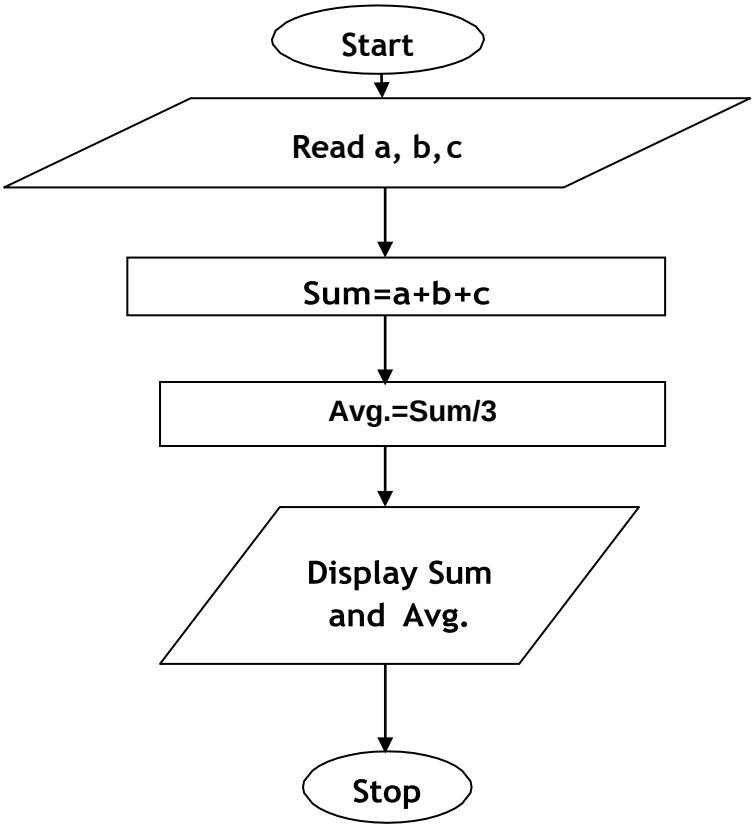
Q. 1 Write an algorithms and flow chart of the followings:

a. To find sum & average of 3 numbers.

Ans:-Algorithm

- Step 1: Start.
- Step 2 :Read the three number suppose "a","b","c" from the user.
- Step 3 : Sum=a+b+c;
- Step 4 : Avg.=Sum/3.
- Step 5 :Display "Sum " and "Avg".
- Step 6 :End .

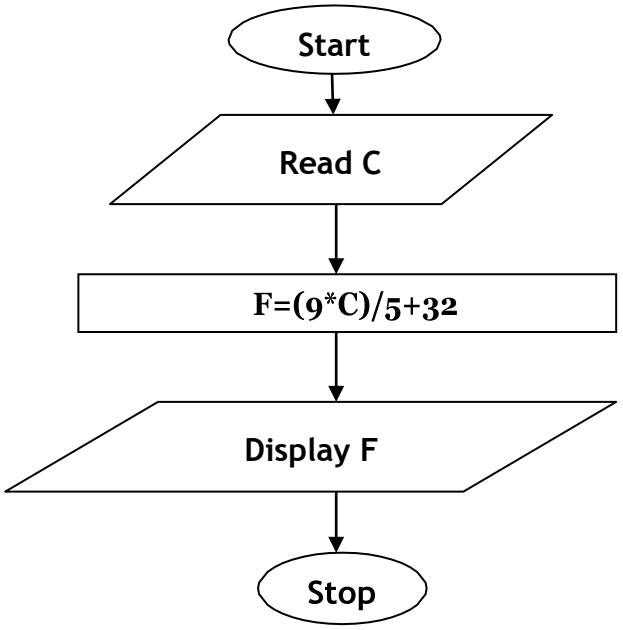
Flowchart:



b. To convert temperature from degree Celsius to Fahrenheit. Algorithm

- Step1 :Start
- Step2 :Read the input of temperature in Celsius (say C)
- Step3 :F=(9*C)/5+32
- Step4 :Print temperature in Fahrenheit is F
- Step5 :Stop

Flowchart



EXERCISE

Short Answer Type Questions.

Q.1 What is a flow chart? ***(2014-Summer)***

Q.2 Give the flowchart symbols for keeping, I/O statement and decision statement.

Long Answer Type Questions

Q.1 Draw a flowchart to find sum of 10 random numbers. ***(2017-Winter)***

Q.2 Write an algorithms and flow chart of the followings:

a. To calculate square of number n.
b. To identify whether entered numbered is positive (+) or negative (-).
c. To generate natural numbers which are divisible by 5 in between 1 and 200?
d. To find sum of digits of a three-digit number (say 364).

****

CHAPTER – 6: OVERVIEW OF C PROGRAMMING LANGUAGE

6.1 Constants, Variables and Data types in C

The C Language

C is a professional programmer's language. It was designed to get in one's way as little as possible. Kernighan and Ritchie wrote the original language definition in their book, *The C Programming Language* (below), as part of their research at AT&T. Unix and C++ emerged from the same labs. For several years I used AT&T as my long distance carrier in appreciation of all that CS research, but hearing "thank you for using AT&T" for the millionth time has used up that goodwill.

Important Points

- The C Language is developed by Dennis Ritchie for creating system applications that directly interact with the hardware devices such as drivers, kernels, etc.
- C programming is considered as the base for other programming languages, that is why it is known as mother language.
- It can be defined by the following ways:
    1. Mother language
    2. System programming language
    3. Procedure-oriented programming language
    4. Structured programming language
    5. Mid-level programming language

1) C as a mother language

C language is considered as the mother language of all the modern programming languages because most of the compilers, JVMs, Kernels, etc. are written in C language, and most of the programming languages follow C syntax, for example, C++, Java, C#, etc. It provides the core concepts like the array, strings, functions, file handling, etc. that are being used in many languages like C++, Java, C#, etc.

2) C as a system programming language

A system programming language is used to create system software. C language is a system programming language because it can be used to do low-level programming (for example driver and kernel). It is generally used to create hardware devices, OS, drivers, kernels, etc. For example, Linux kernel is written in C.

It can't be used for internet programming like Java, .Net, PHP, etc.

3) C as a procedural language

A procedure is known as a function, method, routine, subroutine, etc. A procedural language specifies a series of steps for the program to solve the problem.

A procedural language breaks the program into functions, data structures, etc.

C is a procedural language. In C, variables and function prototypes must be declared before being used.

4) C as a structured programming language

A structured programming language is a subset of the procedural language. Structure means to break a program into parts or blocks so that it may be easy to understand.

In the C language, we break the program into parts using functions. It makes the program

easier to understand and modify.

5) C as a mid-level programming language

C is considered as a middle-level language because it supports the feature of both low-level and high-level languages. C language program is converted into assembly code, it supports pointer arithmetic (low-level), but it is machine independent (a feature of high-level).

A Low-level language is specific to one machine, i.e., machine dependent. It is machine dependent, fast to run. But it is not easy to understand.

A High-Level language is not specific to one machine, i.e., machine independent. It is easy to understand.

Quick History of C

- Developed at Bell Laboratories in the early seventies by Dennis Ritchie.
- Born out of two other languages – BCPL (Basic Control Programming Language) and B.
- C introduced such things as character types, floating point arithmetic, structures, unions and the preprocessor.
- The principal objective was to devise a language that was easy enough to understand to be "high-level" – i.e. understood by general programmers, but low-level enough to be applicable to the writing of systems-level software.
- The language should abstract the details of how the computer achieves its tasks in such a way as to ensure that C could be portable across different types of computers, thus allowing the UNIX operating system to be compiled on other computers with a minimum of re-writing.
- C as a language was in use by 1973, although extra functionality, such as new types, was introduced up until 1980.
- In 1978, Brian Kernighan and Dennis M. Ritchie wrote the seminal work The C Programming Language, which is now the standard reference book for C.
- A formal ANSI standard for C was produced in 1989.
- In 1986, a descendant of C, called C++ was developed by Bjarne Stroustrup, which is in wide use today. Many modern languages such as C#, Java and Perl are based on C and C++.
- Using C language scientific, business and system-level applications can be developed easily.

6.1.1 Constants

In C programming language, a constant is similar to the variable but the constant hold only one value during the program execution. That means, once a value is assigned to the constant, that value can't be changed during the program execution. Once the value is

assigned to the constant, it is fixed throughout the program. A constant can be defined as follows...

A constant is a named memory location which holds only one value throughout the program execution.

In C programming language, a constant can be of any data type like integer, floating-point, character, string and double, etc.,

Integer constants

An integer constant can be a decimal integer or octal integer or hexadecimal integer. A decimal integer value is specified as direct integer value whereas octal integer value is prefixed with 'O' and hexadecimal value is prefixed with 'OX'.

An integer constant can also be unsigned type of integer constant or long type of integer constant. Unsigned integer constant value is suffixed with 'u' and long integer constant value is suffixed with 'l' whereas unsigned long integer constant value is suffixed with 'ul'.

Example

125---- > Decimal Integer Constant

O76 ---- > Octal Integer Constant

OX3A----- >Hexa Decimal Integer Constant

50u-----> Unsigned Integer Constant

30l-----> Long Integer Constant

100ul ----> Unsigned Long Integer Constant

Floating Point constants

A floating-point constant must contain both integer and decimal parts. Sometimes it may also contain the exponent part. When a floating-point constant is represented in exponent form, the value must be suffixed with 'e' or 'E'.

Example

The floating-point value 3.14 is represented as 3E-14 in exponent form.

Character Constants

A character constant is a symbol enclosed in single quotation. A character constant has a maximum length of one character.

Example

'A'

'2'

'+'

In the C programming language, there are some predefined character constants called escape sequences. Every escape sequence has its own special functionality and every escape sequence is prefixed with '\' symbol. These escape sequences are used in output function called 'printf()'.

String Constants

- A string constant is a collection of characters, digits, special symbols and escape sequences that are enclosed in double quotations.
- We define string constant in a single line as follows... "This is Diploma smart class"

- We can define string constant using multiple lines as follows...

```
" This\
is\
Diploma smart class "
```

- We can also define string constant by separating it with white space as follows...

"This" "is" "Diploma smart class"

All the above three defines the same string constant.

Creating constants in C

- In a c programming language, constants can be created using two concepts...
  - Using the "**const**" keyword
  - Using "**#define**" preprocessor

Using the "const" keyword

- We create a constant of any data type using 'const' keyword. To create a constant, we prefix the variable declaration with 'const' keyword.
- The general syntax for creating constant using 'const' keyword is as follows...

```
const datatype constantName ;
            OR
const data type constant Name = value ;
```

Example

```
const int x = 10 ;
```

Here, 'x' is a integer constant with fixed value 10.

Example Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i = 9 ;
const int x = 10 ;
i = 15 ;
x = 100 ; // creates an error
printf("i = %d\nx = %d", i, x ) ;
}
```

The above program gives an error because we are trying to change the constant variable value (x = 100).

Using '#define' preprocessor

We can also create constants using '#define' preprocessor directive. When we create constant using this preprocessor directive it must be defined at the beginning of the program (because all the preprocessor directives must be written before the global declaration).

We use the following syntax to create constant using '#define' preprocessor directive...

#define CONSTANTNAME value

Example

#define PI 3.14

Here, PI is a constant with value 3.14

Example Program

```
#include<stdio.h>
#include<conio.h>
#define PI 3.14
void main()
{
int r, area ;
printf("Please enter the radius of circle : ") ;
scanf("%d", &r) ;
area = PI * (r * r) ;
printf("Area of the circle = %d", area) ;
}
```

6.1.2 Variables:

Variables in a c programming language are the named memory locations where the user can store different values of the same datatype during the program execution. That means a variable is a name given to a memory location in which we can store different values of the same data type. In other words, a variable can be defined as a storage container to hold values of the same data type during the program execution. The formal definition of a data type is as follows.

Variable is a name given to a memory location where we can store different values of the same datatype during the program execution.

Every variable in c programming language must be declared in the declaration section before it is used. Every variable must have a data type that determines the range and type of values be stored and the size of the memory to be allocated.

A variable name may contain letters, digits and underscore symbol. The following are the rules to specify a variable name...

- Variable name should not start with a digit.
- Keywords should not be used as variable names.
- A variable name should not contain any special symbols except underscore (_).
- A variable name can be of any length but compiler considers only the first 31 characters of the variable name.

Declaration of Variable

Declaration of a variable tells the compiler to allocate the required amount of memory with the specified variable name and allows only specified datatype values into that memory location. In C programming language, the declaration can be performed either before the function as global variables or inside any block or function. But it must be at the beginning of block or function.

Declaration Syntax:

datatype variableName;

Example

int number;

The above declaration tells to the compiler that allocates 2 bytes of memory with the name

"number" and allows only integer values into that memory location.

6.1.3 Data types

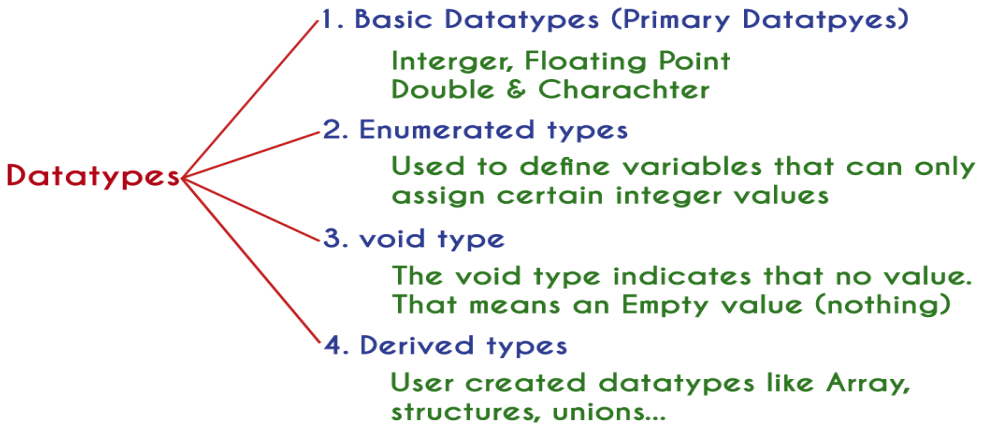
Data used in c program is classified into different types based on its properties. In the C programming language, a data type can be defined as a set of values with similar characteristics. All the values in a data type have the same properties.

Data types in the c programming language are used to specify what kind of value can be stored in a variable. The memory size and type of the value of a variable are determined by the variable data type. In a c program, each variable or constant or array must have a data type and this data type specifies how much memory is to be allocated and what type of values are to be stored in that variable or constant or array. The formal definition of a data type is as follows...

The Data type is a set of value with predefined characteristics. Data types are used to declare variable, constants, arrays, pointers, and functions.

In the c programming language, data types are classified as follows...

- Primary data types (Basic data types OR Predefined data types)
- Derived data types (Secondary data types OR User-defined data types)
- Enumeration data types
- Void data type

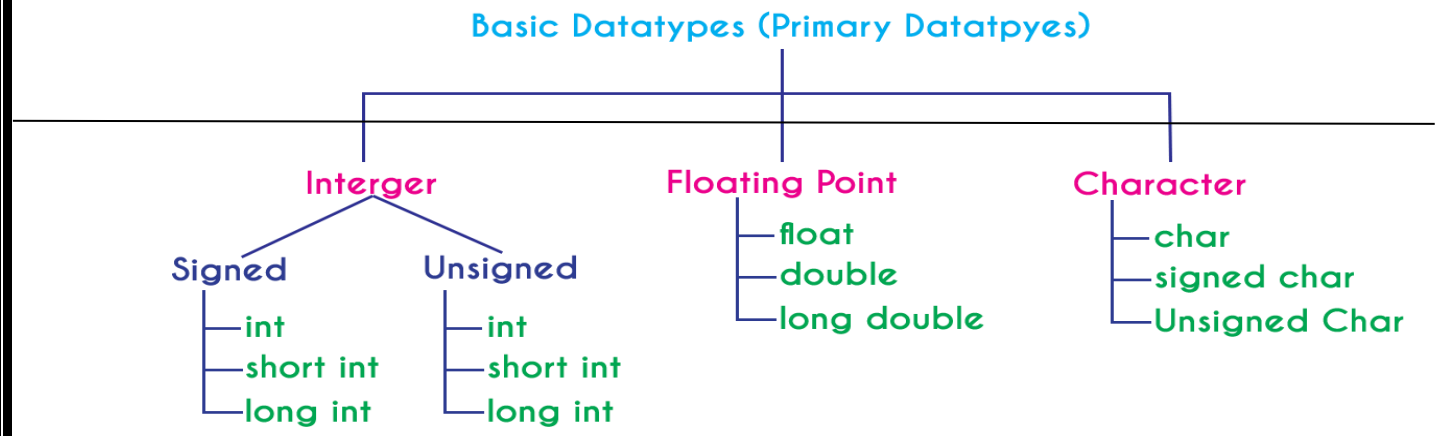


Primary data types

The primary data types in the C programming language are the basic data types. All the primary data types are already defined in the system. Primary data types are also called as Built-In data types. The following are the primary data types in c programming language.

- Integer data type
- Floating Point data type
- Double data type

- Character data type



Integer Data type (int)

The integer data type is a set of whole numbers. Every integer value does not have the decimal value. We use the keyword "int" to represent integer data type in c. We use the keyword int to declare the variables and to specify the return type of a function. The integer data type is used with different type modifiers like short, long, signed and unsigned. The following table provides complete details about the integer data type.

Type	Size (Bytes)	Range	Specifier
int (signed short int)	2	-32768 to +32767	%d
short int (signed short int)	2	-32768 to +32767	%d
long int (signed long int)	4	-2,147,483,648 to +2,147,483,647	%d
unsigned int (unsigned short int)	2	0 to 65535	%u
unsigned long int	4	0 to 4,294,967,295	%u

Floating Point data types

Floating-point data types are a set of numbers with the decimal value. Every floating-point value must contain the decimal value. The floating-point data type has two variants...

- float
- double

We use the keyword "float" to represent floating-point data type and "double" to represent double data type in c. Both float and double are similar but they differ in the number of decimal places. The float value contains 6 decimal places whereas double value contains 15 or 19 decimal places. The following table provides complete details about floating-point data type.

Character data type

The character data type is a set of characters enclosed in single quotations. The following table provides complete details about the character data type.

Type	Size (Bytes)	Range	Specifier
char (signed char)	1	-128 to +127	%c
unsigned char	1	0 to 255	%c

The following table provides complete information about all the data types in c programming language...

	Integer	Floating Point	Double	Character
What is it?	Numbers without decimal value	Numbers with decimal value	Numbers with decimal value	Any symbol enclosed in single quotation
Keyword	int	float	double	char
Memory Size	2 or 4 Bytes	4 Bytes	8 or 10 Bytes	1 Byte
Range	-32768 to +32767 (or) 0 to 65535 (Incase of 2 bytes only)	1.2E - 38 to 3.4E + 38	2.3E-308 to 1.7E+308	-128 to + 127 (or) 0 to 255
Type Specifier	%d or %i or %u	%f	%ld	%c or %s
Type Modifier	short, long signed, unsigned	No modifiers	long	signed, unsigned
Type Qualifier	const, volatile	const, volatile	const, volatil	const, volatile

Void data type

The void data type means nothing or no value. Generally, the void is used to specify a function which does not return any value. We also use the void data type to specify empty parameters of a function.

Type	Size (Bytes)	Range	Specifier
float	4	1.2E - 38 to 3.4E + 38	%f
double	8	2.3E-308 to 1.7E+308	%ld
long double	10	3.4E-4932 to 1.1E+4932	%ld

Enumerated data type

An enumerated data type is a user-defined data type that consists of integer constants and each integer constant is given a name. The keyword "enum" is used to define the enumerated data type.

Derived data types

Derived data types are user-defined data types. The derived data types are also called as user- defined data types or secondary data types. In the c programming language, the derived data types are created using the following concepts...

- Arrays
- Structures
- Unions

6.2 Managing Input and Output operations.

Output Functions

C programming language provides built-in functions to perform output operation. The output operations are used to display data on user screen (output screen) or printer or any file. The c programming language provides the following built-in output functions...

- printf()
- putchar()
- puts()
- fprintf()

printf() function

The printf() function is used to print string or data values or a combination of string and data values on the output screen (User screen). The printf() function is built-in function defined in a header file called "stdio.h". When we want to use printf() function in our program we need to include the respective header file (stdio.h) using the #include statement. The printf() function has the following syntax...

Syntax:

```
printf("message to be display!!!");
```

Example Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
printf("Hello! Welcome to Diploma smart class!!!");
}
```

In the above example program, we used the printf() function to print a string on to the output screen.

Output of the program is **Hello! Welcome to Diploma smart class!!!**

The printf() function is also used to display data values. When we want to display data values we use format string of the data value to be displayed.

Syntax:

```
printf("format string",variableName);
```

Example Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i = 10; float x = 5.5;
printf("%d %f",i, x);
}
```

In the above example program, we used the printf() function to print data values of variables i and x on to the output screen. Here i is an integer variable so we have used format string %d and x is a float variable so we have used format string %f.

The printf() function can also be used to display string along with data values.

To display the output in different lines or as we wish, we use some special characters called escape sequences. Escape sequences are special characters with special functionality used in printf() function to format the output according to the user requirement. In the C programming language, we have the following escape sequences...

Escape sequence	Meaning
\n	Moves the cursor to New Line
\t	Inserts Horizontal Tab (5 characters space)
\v	Inserts Vertical Tab (5 lines space)
\a	Beep sound
\b	Backspace (removes the previous character from its current position)
****	Inserts Backward slash symbol
\?	Inserts Question mark symbol
\'	Inserts Single quotation mark symbol
\"	Inserts Double quotation mark symbol

Consider the following example program...

Example Program:

```
#include<stdio.h>
#include<conio.h>
void main()
{
printf("Welcome to\n");
printf("Diploma smart class\n");
printf("the perfect website for learning");
}
```

Output:-

```
Welcome to
Diploma smart class
the perfect website for learning
```

putchar() function

The putchar() function is used to display a single character on the output screen. The putchar() functions prints the character which is passed as a parameter to it and returns the same character as a return value. This function is used to print only a single character. To print multiple characters we need to write multiple times or use a looping statement. Consider the following example program...

```
#include<stdio.h>
#include<conio.h>
void main()
{
char ch = 'A';
putchar(ch);
}
```

puts() function

The puts() function is used to display a string on the output screen. The puts() functions prints a string or sequence of characters till the newline. Consider the following example program...

Example Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
char name[30];
printf("\nEnter your favourite website: ");
gets(name);
puts(name);
}
```

fprintf() function

The fprintf() function is used with the concept of files. The fprintf() function is used to print a line into the file. When you want to use fprintf() function the file must be opened in writing mode.

Input Functions

C programming language provides built-in functions to perform input operations. The input operations are used to read user values (input) from the keyboard. The c programming language provides the following built-in input functions.

- scanf()
- getchar()
- getch()
- gets()
- fscanf()

scanf() function

The scanf() function is used to read multiple data values of different data types from the keyboard. The scanf() function is built-in function defined in a header file called "stdio.h". When we want to use scanf() function in our program, we need to include the respective header file (stdio.h) using #include statement. The scanf() function has the following syntax...

Syntax:

```
scanf("format strings",&variableNames);
```

Example Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i;
printf("\nEnter any integer value: ");
scanf("%d",&i);
printf("\nYou have entered %d number",i);
}
```

In the above example program, we used the scanf() function to read an integer value from the keyboard and store it into variable 'i'.

Output:-

```
Enter any integer value: 53
You have entered 53 number
```

The scanf() function is also used to read multiple data values of different or the same data types. Consider the following example program...

Example Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i; float x;
printf("\nEnter one integer followed by one float value : ");
scanf("%d%f",&i, &x);
printf("\ninteger value = %d, float value = %f",i, x);
}
```

Output:-

```
Enter one integer followed by one float value : 59 32.8
integer value = 59, float value =32.8
```

In the above example program, we used the scanf() function to read one integer value and one float value from the keyboard. Here 'i' is an integer variable so we have used format string
%d, and 'x' is a float variable so we have used format string %f.

The scanf() function returns an integer value equal to the total number of input values read using scanf function.

Example Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i,a,b;
float x;
printf("\nEnter two integers and one float : ");
i = scanf("%d%d%f",&a, &b, &x);
printf("\nTotal inputs read : %d",i);
}
```

getchar() function:

The getchar() function is used to read a character from the keyboard and return it to the program. This function is used to read a single character. To read multiple characters we need to write multiple times or use a looping statement. Consider the following example program...

Example Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
char ch;
printf("\nEnter any character : ");
ch = getchar();
printf("\nYou have entered : %c\n",ch);
}
```

Output:-

Enter any character :H
You have entered :H

getch() function

The getch() function is similar to getchar function. The getch() function is used to read a character from the keyboard and return it to the program. This function is used to read a single character. To read multiple characters we need to write multiple times or use a looping statement. Consider the following example program...

Example Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
char ch;
printf("\nEnter any character : ");
```

```
ch = getch();
printf("\nYou have entered : %c",ch);
}
```

gets() function

The gets() function is used to read a line of string and stores it into a character array. The gets() function reads a line of string or sequence of characters till a newline symbol enters. Consider the following example program...

Example Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
char name[30];
printf("\nEnter your favourite website: ");
gets(name);
printf("%s",name);
}
```

fscanf() function

The fscanf() function is used with the concept of files. The fscanf() function is used to read data values from a file. When you want to use fscanf() function the file must be opened in reading mode.

6.3 Operators, Expressions, Type conversion & Typecasting

6.3.1 Operators

An operator is a symbol used to perform arithmetic and logical operations in a program. That means an operator is a special symbol that tells the compiler to perform mathematical or logical operations. C programming language supports a rich set of operators that are classified as follows.

- Arithmetic Operators
- Relational Operators
- Logical Operators
- Increment & Decrement Operators
- Assignment Operators
- Bitwise Operators
- Conditional Operator
- Special Operators

Arithmetic Operators (+, -, *, /, %)

The arithmetic operators are the symbols that are used to perform basic mathematical operations like addition, subtraction, multiplication, division and percentage modulo. The following table provides information about arithmetic operators.

Operator	Meaning	Example
+	Addition	10 + 5 = 15
-	Subtraction	10 - 5 = 5
*	Multiplication	10 * 5 = 50
/	Division	10 / 5 = 2
%	Remainder of the Division	5 % 2 = 1

- The addition operator can be used with numerical data types and character data type. When it is used with numerical values, it performs mathematical addition and when it is used with character data type values, it performs concatenation (appending).
- The remainder of the division operator is used with integer data type only.

Relational Operators (<, >, <=, >=, ==, !=)

The relational operators are the symbols that are used to compare two values. That means the relational operators are used to check the relationship between two values. Every relational operator has two results TRUE or FALSE. In simple words, the relational operators are used to define conditions in a program. The following table provides information about relational operators.

Operator	Meaning	Example
<	Returns TRUE if the first value is smaller than second value otherwise returns FALSE	10 < 5 is FALSE
>	Returns TRUE if the first value is larger than second value otherwise returns FALSE	10 > 5 is TRUE
<=	Returns TRUE if the first value is smaller than or equal to second value otherwise returns FALSE	10 <= 5 is FALSE
>=	Returns TRUE if the first value is larger than or equal to second value otherwise returns FALSE	10 >= 5 is TRUE
==	Returns TRUE if both values are equal otherwise returns FALSE	10 == 5 is FALSE
!=	Returns TRUE if both values are not equal otherwise returns FALSE	10 != 5 is TRUE

Logical Operators (&&, ||, !)

The logical operators are the symbols that are used to combine multiple conditions into one condition. The following table provides information about logical operators.

Operator	Meaning	Example
&&	Logical AND - Returns TRUE if all conditions are TRUE otherwise returns FALSE	10 < 5 && 12 > 10 is FALSE
||	Logical OR - Returns FALSE if all conditions are FALSE otherwise returns TRUE	10 < 5 || 12 > 10 is TRUE
!	Logical NOT - Returns TRUE if condition is FLASE and returns FALSE if it is TRUE	!(10 < 5 && 12 > 10) is TRUE

- Logical AND - Returns TRUE only if all conditions are TRUE, if any of the conditions is FALSE then complete condition becomes FALSE.
- Logical OR - Returns FALSE only if all conditions are FALSE, if any of the conditions is TRUE then complete condition becomes TRUE.

Increment & Decrement Operators (++ & --)

The increment and decrement operators are called unary operators because both need only one operand. The increment operators add one to the existing value of the operand and the decrement operator subtracts one from the existing value of the operand. The following table provides information about increment and decrement operators.

Operator	Meaning	Example	
++	Increment	Adds one to existing value	int a = 5;a++; ⇒ a = 6
--	Decrement	Subtracts one from existing value	int a = 5;a--; ⇒ a = 4

The increment and decrement operators are used in front of the operand (++a) or after the operand (a++). If it is used in front of the operand, we call it as pre-increment or pre-decrement and if it is used after the operand, we call it as post-increment or post-decrement.

Pre-Increment or Pre-Decrement

In the case of pre-increment, the value of the variable is increased by one before the expression evaluation. In the case of pre-decrement, the value of the variable is decreased by one before the expression evaluation. That means, when we use pre-increment or pre-decrement, first the value of the variable is incremented or decremented by one, then the modified value is used in the expression evaluation.

Example Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i = 5,j;
j = ++i; // Pre-Increment
printf("i = %d, j = %d",i,j);
}
```

Post-Increment or Post-Decrement

In the case of post-increment, the value of the variable is increased by one after the expression evaluation. In the case of post-decrement, the value of the variable is decreased by one after the expression evaluation. That means, when we use post-increment or post-decrement, first the expression is evaluated with existing value, then the value of the variable is incremented or decremented by one.

Example Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i = 5,j;
j = i++; // Post-Increment
printf("i = %d, j = %d",i,j);
  }
```

Assignment Operators (=, +=, -=, *=, /=, %=)

The assignment operators are used to assign right-hand side value (Rvalue) to the left-hand side variable (Lvalue). The assignment operator is used in different variants along with arithmetic operators. The following table describes all the assignment operators in the C programming language.

Operator	Meaning	Example
=	Assign the right-hand side value to left-hand side variable	A = 15
+=	Add both left and right-hand side values and store the result into left-hand side variable	A += 10⇒ A = A+10
-=	Subtract right-hand side value from left-hand side variable value and store the result into left-hand side variable	A -= B⇒ A = A-B
*=	Multiply right-hand side value with left-hand side variable value and store the result into left-hand side variable	A *= B⇒ A = A*B
/=	Divide left-hand side variable value with right-hand side variable value and store the result into the left-hand side variable	A /= B⇒ A = A/B
%=	Divide left-hand side variable value with right-hand side variable value and store the remainder into the left-hand side variable	A %= B⇒ A = A%B

Bitwise Operators (&, |, ^, ~, >>, <<)

The bitwise operators are used to perform bit-level operations in the c programming language. When we use the bitwise operators, the operations are performed based on the binary values. The following table describes all the bitwise operators in the C programming language.

Let us consider two variables A and B as A = 25 (11001) and B = 20 (10100).

Operator	Meaning	Example
&	the result of Bitwise AND is 1 if all the bits are 1 otherwise it is 0	A & B⇒ 16 (10000)
|	the result of Bitwise OR is 0 if all the bits are 0 otherwise it is 1	A | B⇒ 29 (11101)
^	the result of Bitwise XOR is 0 if all the bits are same otherwise it is 1	A ^ B⇒ 13 (01101)
~	the result of Bitwise once complement is negation of the bit (Flipping)	~A⇒ 6 (00110)
<<	the Bitwise left shift operator shifts all the bits to the left by the specified number of positions	A << 2⇒ 100 (1100100)
>>	the Bitwise right shift operator shifts all the bits to the right by the specified number of positions	A >> 2⇒ 6 (00110)

Conditional Operator (?:)

The conditional operator is also called a ternary operator because it requires three operands. This operator is used for decision making. In this operator, first we verify a condition, then we perform one operation out of the two operations based on the condition result. If the condition is TRUE the first option is performed, if the condition is FALSE the second option is performed. The conditional operator is used with the following syntax.

Condition ? TRUE Part : FALSE Part;

Example

A = (10<15)?100:200; ⇒ A value is 100

Special Operators (sizeof, pointer, comma, dot, etc.)

The following are the special operators in c programming language.

sizeof operator

This operator is used to find the size of the memory (in bytes) allocated for a variable. This operator is used with the following syntax.

sizeof(variable Name);

Example

sizeof(A); ⇒ the result is 2 if A is an integer

Pointer operator (*)

This operator is used to define pointer variables in c programming language.

Comma operator (,)

This operator is used to separate variables while they are declaring, separate the expressions in function calls, etc.

Dot operator (.)

This operator is used to access members of structure or union.

Operator Precedence and Associativity

What is Operator Precedence?

Operator precedence is used to determine the order of operators evaluated in an expression. In c programming language every operator has precedence (priority). When there is more than one operator in an expression the operator with higher precedence is evaluated first and the operator with the least precedence is evaluated last.

What is Operator Associativity?

Operator associativity is used to determine the order of operators with equal precedence evaluated in an expression. In the c programming language, when an expression contains multiple operators with equal precedence, we use associativity to determine the order of evaluation of those operators.

In c programming language the operator precedence and associativity are as shown in the following table.

Precedence	Operator	Operator Meaning	Associativity
1	() [] -> .	function call array reference structure member access structure member access	Left to Right
2	! ~ + - ++ -- & * sizeof (type)	negation 1's complement Unary plus Unary minus increment operator decrement operator address of operator pointer returns size of a variable type conversion	Right to Left
3	* / %	multiplication division remainder	Left to Right
4	+ -	addition subtraction	Left to Right
5	<< >>	left shift right shift	Left to Right
6	< <= > >=	less than less than or equal to greater than greater than or equal to	Left to Right
7	== !=	equal to not equal to	Left to Right
8	&	bitwise AND	Left to Right
9	^	bitwise EXCLUSIVE OR	Left to Right
10	|	bitwise OR	Left to Right
11	&&	logical AND	Left to Right
12	||	logical OR	Left to Right
13	?:	conditional operator	Left to Right
14	= *= /= %= += -= &= ^= |= <<= >>=	assignment assign multiplication assign division assign remainder assign addition assign subtraction assign bitwise AND assign bitwise XOR assign bitwise OR assign left shift assign right shift	Right to Left
15	,	separator	Left to Right

In the above table, the operator precedence decreases from top to bottom and increases from bottom to top.

6.3.2 Expressions

What is an expression?

In any programming language, if we want to perform any calculation or to frame any condition etc., we use a set of symbols to perform the task. These set of symbols makes an expression.

In the C programming language, an expression is defined as follows.

An expression is a collection of operators and operands that represents a specific value.

In the above definition, an operator is a symbol that performs tasks like arithmetic operations, logical operations, and conditional operations, etc.

Operands are the values on which the operators perform the task. Here operand can be a direct value or variable or address of memory location.

Expression Types in C

In the C programming language, expressions are divided into THREE types. They are as follows...

- Infix Expression
- Postfix Expression
- Prefix Expression

The above classification is based on the operator position in the expression.

Infix Expression

- The expression in which the operator is used between operands is called infix expression.
- The infix expression has the following general structure.

Operand1 Operator Operand2

Postfix Expression

- The expression in which the operator is used after operands is called postfix expression.
- The postfix expression has the following general structure.

Operand1 Operand2 Operator

Prefix Expression

- The expression in which the operator is used before operands is called a prefix expression.
- The prefix expression has the following general structure.

Operator Operand1 Operand2

6.3.3 Type Conversion and Type Casting

In a programming language, the expression contains data values of the same datatype or different data types. When the expression contains similar datatype values then it is evaluated without any problem. But if the expression contains two or more different datatype values then they must be converted to the single datatype of destination datatype. Here, the destination is the location where the final result of that expression is stored. For example, the multiplication of an integer data value with the float data value and storing the result into a float variable. In this case, the integer value must be converted to float value so that the final result is a float datatype value.

In a c programming language, the data conversion is performed in two different methods as follows...

- Type Conversion
- Type Casting

Type Conversion

The type conversion is the process of converting a data value from one data type to another data type automatically by the compiler. Sometimes type conversion is also called implicit type conversion. The implicit type conversion is automatically performed by the compiler.

For example, in c programming language, when we assign an integer value to a float variable the integer value automatically gets converted to float value by adding decimal value 0. And when a float value is assigned to an integer variable the float value automatically gets converted to an integer value by removing the decimal value. To understand more about type conversion observe the following...

```
int i = 10 ;
float x = 15.5 ;
char ch = 'A' ;
i = x ; =======> x value 15.5 is converted as 15 and assigned to variable i
x = i ; =======> Here i value 10 is converted as 10.000000 and assigned to variable x
i = ch ; =======> Here the ASCII value of A (65) is assigned to i
```

Example Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i = 95 ;
float x = 90.99 ;
char ch = 'A' ;
i = x ;
printf("i value is %d\n",i);
        x = i ;
printf("x value is %f\n",x);
        i = ch ;
printf("i value is %d\n",i);
}
```

In the above program, we assign i = x, i.e., float variable value is assigned to the integer variable. Here, the compiler automatically converts the float value (90.99) into integer value (90) by removing the decimal part of the float value (90.99) and then it is assigned to variable i. Similarly, when we assign x = i, the integer value (90) gets converted to float value (90.000000) by adding zero as the decimal part.

Typecasting

Typecasting is also called an explicit type conversion. Compiler converts data from one data type to another data type implicitly. When compiler converts implicitly, there may be a data loss. In such a case, we convert the data from one data type to another data type using explicit type conversion. To perform this we use the unary cast operator. To convert data from one type to another type we specify the target data type in parenthesis as a prefix to the data value that has to be converted.

The general syntax of typecasting is as follows.

(TargetDatatype) DataValue

Example

```
int totalMarks = 450, maxMarks = 600 ;
float average ;
average = (float) totalMarks / maxMarks * 100 ;
```

In the above example code, both totalMarks and maxMarks are integer data values. When we perform totalMarks / maxMarks the result is a float value, but the destination (average) datatype is a float. So we use type casting to convert totalMarks and maxMarks into float data type.

Example Program

```
#include<stdio.h>
#include<conio.h>
int main()
{
int a, b, c ;
float avg ;
printf( "Enter any three integer values : ");
scanf("%d%d%d",a,b, c);
avg = (a + b + c) / 3 ;
printf("avg before casting = &f", avg \n);
avg = (float)(a + b + c) / 3 ;
printf("avg after casting = %f",avg\n);
return 0;
}
```

Comments

Comments in C are enclosed by slash/star pairs: /* .. comments .. */ which may cross multiple lines. C++ introduced a form of comment started by two slashes and extending to the end of the line:

```
// comment until the line end
```

The // comment form is so handy that many C compilers now also support it, although it is not technically part of the C language.

Along with well-chosen function names, comments are an important part of well written code. Comments should not just repeat what the code says. Comments should describe what the code accomplishes which much more is interesting than a translation of what each statement does. Comments should also narrate what is tricky or non-obvious about a section of code.

6.4 Decision Control and Looping Statements (If, If-else, If-else-if, Switch, While, Do- while, For, Break, Continue & Goto)

Control Structures

C uses curly braces ({}) to group multiple statements together. The statements execute in order. Some languages let you declare variables on any line (C++). Other languages insist that variables are declared only at the beginning of functions (Pascal). C takes the middle road -- variables may be declared within the body of a function, but they must follow a '{'. More modern languages like Java and C++ allow you to declare variables on any line, which is handy.

What is Decision Making Statement?

In the C programming language, the program execution flow is line by line from top to bottom. That means the c program is executed line by line from the main method. But this type of execution flow may not be suitable for all the program solutions. Sometimes, we make some decisions or we may skip the execution of one or more lines of code. Consider a situation, where we write a program to check whether a student has passed or failed in a particular subject. Here, we need to check whether the marks are greater than the pass marks or not. If marks are greater, then we decide that the student has passed otherwise failed. To solve such kind of problems in c we use the statements called decision making statements.

Decision-making statements are the statements that are used to verify a given condition and decide whether a block of statements gets executed or not based on the condition result.

In the c programming language, there are two decision-making statements they are as follows.

1. if statement
2. switch statement

if statement in c

In c, if statement is used to make decisions based on a condition. The if statement verifies the given condition and decides whether a block of statements are executed or not based on the condition result. In c, if statement is classified into four types as follows...

1. Simple if statement
2. if-else statement
3. Nested if statement
4. if-else-if statement (if-else ladder)

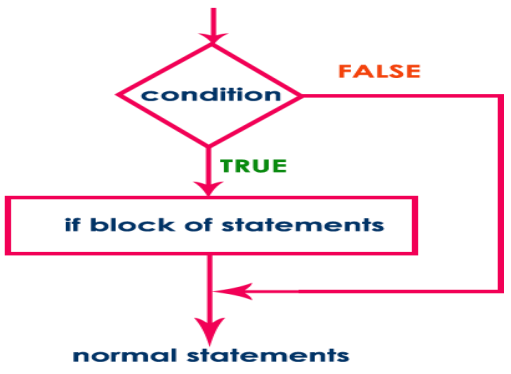
Simple if statement

Simple if statement is used to verify the given condition and executes the block of statements based on the condition result. The simple if statement evaluates specified condition. If it is TRUE, it executes the next statement or block of statements. If the condition is FALSE, it skips the execution of the next statement or block of statements. The general syntax and execution flow of the simple if statement is as follows.

Syntax

```
if ( condition )
{
    ....
    block of statements;
    ....
}
```

Execution flow diagram



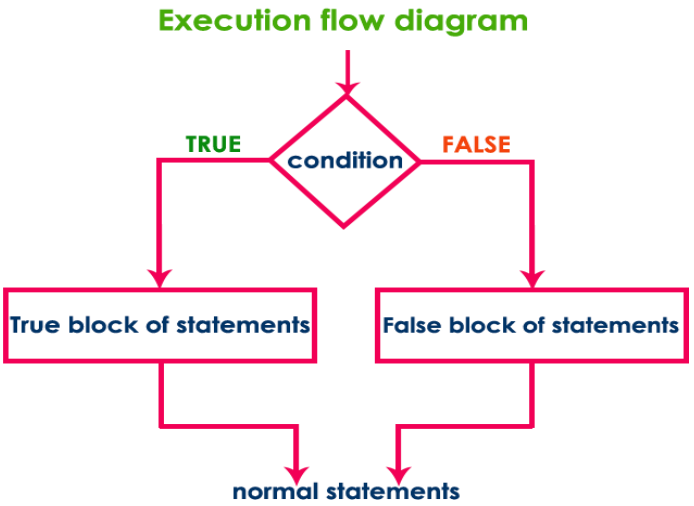
Simple if statement is used when we have only one option that is executed or skipped based on a condition.

if-else statement

The if-else statement is used to verify the given condition and executes only one out of the

Syntax

```
if ( condition )
{
   ....
   True block of statements;
   ....
}
else
{
   ....
   False block of statements;
   ....
}
```



two blocks of statements based on the condition result. The if-else statement evaluates the specified condition. If it is TRUE, it executes a block of statements (True block). If the condition is FALSE, it executes another block of statements (False block). The general syntax and execution flow of the if-else statement is as follows.

The if-else statement is used when we have two options and only one option has to be executed based on a condition result (TRUE or FALSE).

Nested if statement

Writing a if statement inside another if statement is called nested if statement. The general syntax of the nested if statement is as follows.

Syntax

```
if ( condition1 )
{
   if ( condition2 )
   {
      ....
      True block of statements 1;
   }
   ....
}
else
{
    False block of condition1;
}
```

The nested if statement can be defined using any combination of simple if & if-else statements.

---

if-else-if statement (if-else ladder)

Writing a if statement inside else of an if statement is called if-else-if statement. The general syntax of the if-else-if statement is as follows...

Syntax

```
if ( condition1 )
{
    ....
    True block of statements1;
    ....
}
else if ( condition2 )
{
    False block of condition1;
    &
    True block of condition2
}
```

The if-else-if statement can be defined using any combination of simple if & if-else statements.

Examples:

Program 1:(Simple if statement)

```
#include <stdio.h>
int main()
{
int x = 20;
int y = 22;     if (x<y)
{
printf("Variable x is less than y");
}
return 0;
}
```

Output:
Variable x is less than y

Program 2:(Simple if statement)

```
#include <stdio.h>
int main()
{
int i = 10;
if (i> 15)
{
printf("10 is less than 15");
}
printf("I am Not in if");
```

```
}
```

Output: I am Not in if

Program 3:(if-else statement)

```
#include <stdio.h>
int main()
{
int i = 20;
if (i< 15)
printf("i is smaller than 15");
else
printf("i is greater than 15");
return 0;
}
```

Output: i is greater than 15

Program 4:(if-else statement)

```
#include <stdio.h>
int main()
{
int age;
printf("Enter your age:");
scanf("%d",&age);
  if(age >=18)
{
/* This statement will only execute if the
 * above condition (age>=18) returns true
 */
printf("You are eligible for voting");
}
else
{
/* This statement will only execute if the
* condition specified in the "if" returns false. */
printf("You are not eligible for voting");
}
return 0;
}
```

Output: Enter your age:14 You are not eligible for voting

Program 5:(Nested-if statement)

```
#include <stdio.h>
int main()
{
 int var1, var2;
printf("Input the value of var1:");
scanf("%d", &var1);
printf("Input the value of var2:");
```

```
scanf("%d",&var2);
if (var1 != var2)
{
printf("var1 is not equal to var2\n");

//Nested if else
  if (var1 > var2)
{
printf("var1 is greater than var2\n");
}
else
{
printf("var2 is greater than var1\n");
}
}
```

Output:

Input the value of var1:12
Input the value of var2:21
var1 is not equal to var2
var2 is greater than var1

Program 6:(Nested-if statement)

```
#include <stdio.h>
int main()
{
int i = 10;
if (i == 10)
{
// First if statement
if (i< 15)
printf("i is smaller than 15\n");
// Nested - if statement  will only be executed if statement above  is true
if (i< 12)
printf("i is smaller than 12 too\n");
else
printf("i is greater than 15");
}
return 0;
}
```

Output:

i is smaller than 15
i is smaller than 12 too

Program 7:(if-else-if statement)

```
#include <stdio.h>
void main()
{
int i = 20;
if (i == 10)
```

```
printf("i is 10");
else if (i == 15)
printf("i is 15");
else if (i == 20)
printf("i is 20");
else
printf("i is not present");
}
```

Output:

```
i is 20
```

Program 8:(if-else-if statement)

```
#include <stdio.h>
int main()
{    int var1, var2;
printf("Input the value of var1:");
scanf("%d", &var1);
printf("Input the value of var2:");
scanf("%d",&var2);
if (var1 !=var2)
{
printf("var1 is not equal to var2\n");
}
else
if (var1 > var2)
{
printf("var1 is greater than var2\n");
}
else
if (var2 > var1)
{
printf("var2 is greater than var1\n");
}
else
{
printf("var1 is equal to var2\n");
}
return 0;
}
```

Output:

```
Input the value of var1:12
Input the value of var2:21
var1 is not equal to var2
```

'switch' statement in C

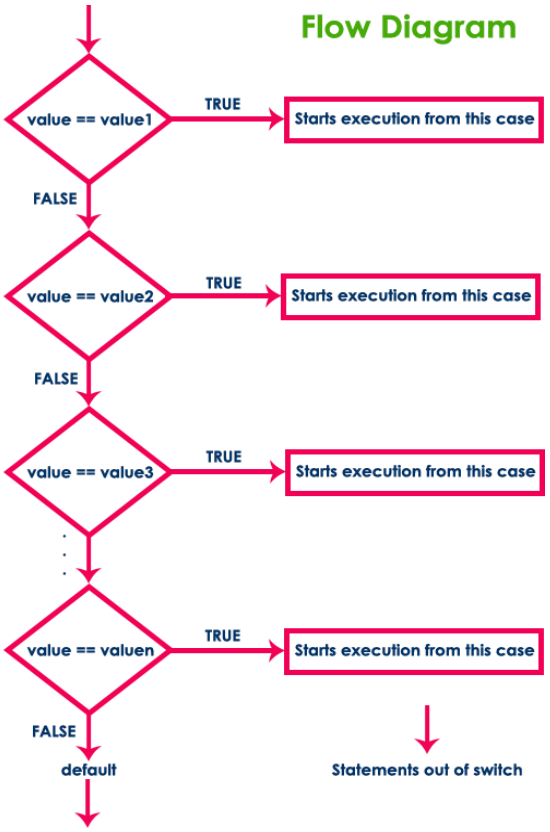
Consider a situation in which we have many options out of which we need to select only one option that is to be executed. Such kind of problems can be solved using nested if statement. But as the number of options increases, the complexity of the program also gets increased. This type of problem can be solved very easily using a switch statement. Using the switch statement, one can select only one option from more number of options very easily. In the switch statement, we provide a value that is to be compared with a value associated with

each option. Whenever the given value matches the value associated with an option, the execution starts from that option. In the switch statement, every option is defined as a case.

The switch statement has the following syntax and execution flow diagram.

Syntax

```
switch ( expression or value )
{
        case  value1:  set of statements;
                       ....
        case  value2:  set of statements;
                       ....
        case  value3:  set of statements;
                       ....
        case  value4:  set of statements;
                       ....
        case  value5:  set of statements;
                       ....
        .
        .
        default: set of statements;
}
```



The switch statement contains one or more cases and each case has a value associated with it. At first switch statement compares the first case value with the switchValue, if it gets matched the execution starts from the first case. If it doesn't match the switch statement compares the second case value with the switch Value and if it is matched the execution starts from the second case. This process continues until it finds a match. If no case value matches with the switchValue specified in the switch statement, then a special case called default is executed. When a case value matches with the switch Value, the execution starts from that particular case. This execution flow continues with the next case statements also. To avoid this, we use the "break" statement at the end of each case. That means the break statement is used to terminate the switch statement. However, it is optional.

Examples:

Program 1:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n ;
clrscr() ;
printf("Enter any digit: ") ;
scanf("%d", &n) ;
switch( n )
{
case 0: printf("ZERO") ;
```

```
break ;
case 1:printf("ONE") ;
break ;
case 2:printf("TWO") ;
```

```
break ;
case 3: printf("THREE") ;
break ;
case 4: printf("FOUR") ;
break ;
case 5: printf("FIVE") ;
break ;
case 6: printf("SIX") ;
break ;
case 7: printf("SEVEN") ;
break ;
case 8: printf("EIGHT") ;
break ;
case 9: printf("NINE") ;
break ;  default: printf("Not a Digit") ;
}
getch() ;
}
```

Output:

Enter any digit: 5
FIVE

Program 2:

```
#include <stdio.h>
int main()
{
int num=2;
switch(num+2)
{
case 1:printf("Case1: Value is: %d", num);
case 2:printf("Case2: Value is: %d", num);
case 3:printf("Case3: Value is: %d", num);
default:printf("Default: Value is: %d", num);
}
return 0;
}
```

Output: Default: value is: 2

Looping statements

Consider a situation in which we execute a single statement or block of statements repeatedly for the required number of times. Such kind of problems can be solved using looping statements in C. For example, assume a situation where we print a message 100 times. If we want to perform that task without using looping statements, we have to either write 100 printf statements or we have to write the same message 100 times in a single printf statement. Both are complex methods. The same task can be performed very easily using looping statements.

The looping statements are used to execute a single statement or block of

statements repeatedly until the given condition is FALSE.

C language provides three looping statements...

- while statement
- do-while statement
- for statement

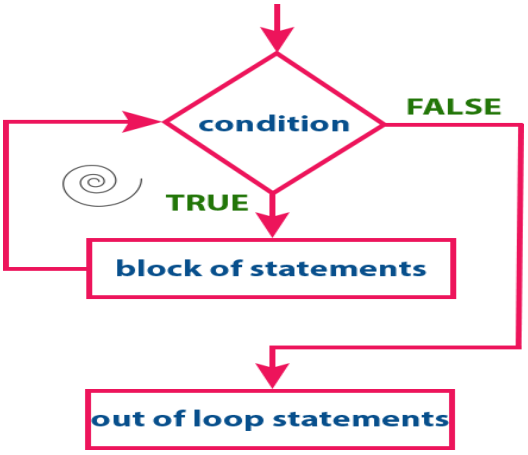
while Statement

The while statement is used to execute a single statement or block of statements repeatedly as long as the given condition is TRUE. The while statement is also known as Entry control looping statement. The while statement has the following syntax... The while statement has the following execution flow diagram...

At first, the given condition is evaluated. If the condition is TRUE, the single statement or block of statements gets executed. Once the execution gets completed the condition is evaluated

Syntax:

```
while( condition )
{
    ...
    block of statements;
    ...
}
```



again. If it is TRUE, again the same statements get executed. The same process is repeated until the condition is evaluated to FALSE. Whenever the condition is evaluated to FALSE, the execution control moves out of the while block.

Examples:
Program 1:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int n = 0;
clrscr() ;
printf("Even numbers upto 10 are");
while( n<= 10 )
{
if( n%2 == 0)
printf("%d\t", n) ;
n++ ;
}
```

```
getch() ;
}
```

Output:

Even numbers upto10 are0 2 4 6 8 10

Program 2:

```
#include <stdio.h>
int main()
{
int count=1;
while (count <= 4)
{
printf("%d ", count);
count++;
}
return 0;
}
```

Output:

1 2 3 4

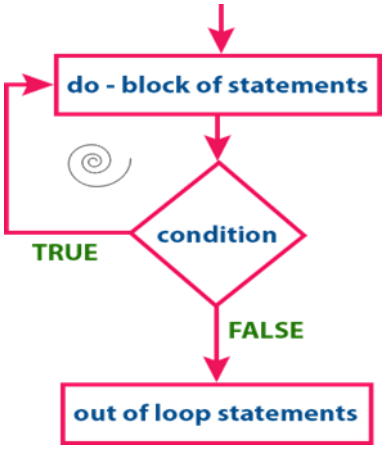
'do-while' statement

The do-while statement is used to execute a single statement or block of statements repeatedly as long as given the condition is TRUE. The do-while statement is also known as the Exit control looping statement. The do-while statement has the following syntax...

```
Syntax:

do
{
    ...
    block of statements;
    ...
} while( condition ) ;
```

The do-while statement has the following execution flow diagram...



At first, the single statement or block of statements which are defined in do block are executed. After the execution of the do block, the given condition gets evaluated. If the condition is evaluated to TRUE, the single statement or block of statements of do block are executed again.

Once the execution gets completed again the condition is evaluated. If it is TRUE, again the same statements are executed. The same process is repeated until the condition is evaluated to FALSE. Whenever the condition is evaluated to FALSE, the execution control moves out of the while block.

Examples:
Program 1:

```
#include <stdio.h>
int main()
{
int j=0;
do
{
printf("Value of variable j is: %d\n", j);
j++;
}
while (j<=3);
return 0;
}
```

Output:

```
Value of variable j is: 0
Value of variable j is: 1
Value of variable j is: 2
Value of variable j is: 3
```

Program 2:

```
#include <stdio.h>
int main()
{
int i=0;
do
{
printf("while vs do-while\n");
}
while(i==1);
printf("Out of loop");
}
```

Output:

```
while vs do-while
Out of loop
```

'for' statement

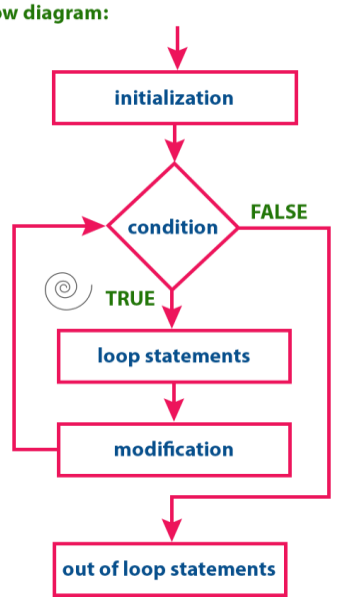
Thefor statement is used to execute a single statement or a block of statements repeatedly as long as the given condition is TRUE.The for statement has the following syntax and execution flow diagram...

At first, the for statement executes initialization followed by condition evaluation. If the condition is evaluated to TRUE, the single statement or block of statements of for statement are executed. Once the execution gets completed, the modification statement is executed and again the condition is evaluated. If it is TRUE, again the same statements are executed. The same process is repeated until the condition is evaluated to FALSE. Whenever the condition is evaluated to FALSE, the execution control moves out of the for block.

Syntax:

```
for( initialization ; condition ; modification )
{
    ...
    block of statements;
    ...
}
```

Execution flow diagram:



Examples:

Program 1:

// Print the naturalnumbers from 1 to 10

```
#include <stdio.h>
int main()
{
int i;
for (i = 1; i< 11; ++i)
{
printf("%d ", i);
}
return 0;
}
```

Output:

1 2 3 4 5 6 7 8 9 10

Program 2:

```
#include <stdio.h>
int main()
{
for (int i=0; i<2; i++)
{
for (int j=0; j<4; j++)
{
printf("%d, %d\n",i ,j);
}
}
return 0;
}
```

Output:

0, 0 0, 1 0, 2 0, 3 1, 0 1, 1 1, 2 1, 3

break, continue and goto in C

In c, there are control statements that do not need any condition to control the program execution flow. These control statements are called as unconditional control statements. C programming language provides the following unconditional control statements...

- break
- continue
- goto

The above three statements do not need any condition to control the program execution flow.

"break" statement

In C, the break statement is used to perform the following two things...

- break statement is used to terminate the switch case statement
- break statement is also used to terminate looping statements like while, do-while and for.

When a break statement is encountered inside the switch case statement, the execution control moves out of the switch statement directly. For example, consider the following program.

Examples:
Program 1:

```
#include<stdio.h>
#include<stdlib.h>
void main ()
{
int i;
for(i = 0; i<10; i++)
{
printf("%d ",i);
if(i == 5)
break;
}
printf("\ncame outside of loop i = %d",i);
}
```

Output:

```
0 1 2 3 4 5
came outside of loop i = 5
```

Program 2:

```
#include <stdio.h>
int main ()
{
        /* local variable definition */
int a = 10;
        /* while loop execution */
while( a< 20 )
{
```

```
printf("value of a: %d\n", a);
a++;
if( a> 15)
{
break;
        /* terminate the loop using break statement */
        }
        }
        return 0;
        }
```

Output:

```
value of a: 10
value of a: 11
value of a: 12
value of a: 13
value of a: 14
value of a: 15
```

continue statement

The continue statement is used to move the program execution control to the beginning of the looping statement. When the continue statement is encountered in a looping statement, the execution control skips the rest of the statements in the looping block and directly jumps to the beginning of the loop. The continue statement can be used with looping statements like while, dowhile and for.

When we use continue statement with while and do-while statements the execution control directly jumps to the condition. When we use continue statement with for statement the execution control directly jumps to the modification portion (increment/decrement/any modification) of the for loop.

Examples:

Program 1:

```
#include<stdio.h>
void main ()
{
int i = 0;
while(i!=10)
{
printf("%d", i);
continue;
i++;
}
}
```

Output:

Infinite loop

Program 2:

```
#include<stdio.h>
int main()
{
```

```
int i=1;//initializing a local variable
        //starting a loop from 1 to 10
for(i=1;i<=10;i++)
{
if(i==5)
{       //if value of i is equal to 5, it will continue the loop
continue;
}
printf("%d \n",i);
}       //end of for loop
return 0;
}
```

Output:

```
1
2
3
4
6
7
8
9
10
```

goto statement

The goto statement is used to jump from one line to another line in the program. Using goto statement we can jump from top to bottom or bottom to top. To jump from one line to another line, the goto statement requires a label. Label is a name given to the instruction or line in the program. When we use a goto statement in the program, the execution control directly jumps to the line with the specified label.

Examples:

Program 1:

```
#include <stdio.h>
void main()
{
int num,i=1;
printf("Enter the number whose table you want to print:");
scanf("%d",&num);
table:   printf("%d x %d = %d\n",num,i,num*i);
i++;
if(i<=10)
goto table;
}
```

Output:

```
Enter the number whose table you want to print:10
10 x 1 = 10
10 x 2 = 20
10 x 3 = 30
10 x 4 = 40
10 x 5 = 50
10 x 6 = 60
10 x 7 = 70
10 x 8 = 80
```

10 x 9 = 90
10 x 10 = 100

Program 2:

```
#include <stdio.h>
void main()
{
int i, j, k;
for(i=0;i<10;i++)
{
 for(j=0;j<5;j++)
{
for(k=0;k<3;k++)
{
printf("%d %d %d\n",i,j,k);
if(j == 3)
{
goto out;
}
}

}
}
out:
printf("came out of the loop");
}
```

Output:

```
0 0 0
0 0 1
0 0 2
0 1 0
0 1 1
0 1 2
0 2 0
0 2 1
0 2 2
0 3 0
came out of the loop
```

6.5 Programming Assignments using the above features.

Some more Examples:

Program to Check Even or Odd

```
#include <stdio.h>
int main()
{
int num;
printf("Enter an integer: ");
scanf("%d", &num);
if(num % 2 == 0)// True if num is perfectly divisible by 2
printf("%d is even.", num);
else
printf("%d is odd.", num);
return 0;
}
```

Output

```
Enter an integer: 7
7 is odd.
```

Program to Check Vowel or consonant

```
#include <stdio.h>
int main()
{
char c;
 int lowercase_vowel, uppercase_vowel;
printf("Enter an alphabet: ");
scanf("%c", &c);
// evaluates to 1 if variable c is a lowercase vowel
lowercase_vowel = (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u');
// evaluates to 1 if variable c is a uppercase vowel
uppercase_vowel = (c == 'A' || c == 'E' || c == 'I' || c == 'O' || c == 'U');
// evaluates to 1 (true) if c is a vowel
if (lowercase_vowel || uppercase_vowel)
printf("%c is a vowel.", c);
 else
printf("%c is a consonant.", c);
return 0;
}
```

Output

```
Enter an alphabet: G
G is a consonant.
```

Program to Check Leap Year

```
#include <stdio.h>
int main()
{
int year;
```

```
        printf("Enter a year: ");

        scanf("%d", &year);
        // leap year if perfectly visible by 400
        if (year % 400 == 0)
        {
        printf("%d is a leap year.", year);
        }
        // not a leap year if visible by 100
        // but not divisible by 400
        else if (year % 100 == 0)
        {
        printf("%d is not a leap year.", year);
        }
         // leap year if not divisible by 100
        // but divisible by 4
        else if (year % 4 == 0)
        {
        printf("%d is a leap year.", year);
        }
// all other years are not leap year else
        {
        printf("%d is not a leap year.", year);
        }
        return 0;
        }
```

Output

```
Enter a year: 1900
1900 is not a leap year
```

Program to Check Alphabet

```
#include <stdio.h>
int main()
{
char c;
printf("Enter a character: ");
scanf("%c", &c);
if ((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z'))
printf("%c is an alphabet.", c);
 else
 printf("%c is not an alphabet.", c);
return 0;
}
```

Output

```
Enter a character: *
* is not an alphabet
```

To find the Factorial of a Number

```
#include <stdio.h>
int main()
{
int n, i;
unsigned long long fact = 1;
```

```
printf("Enter an integer: ");
scanf("%d", &n);
        // shows error if the user enters a negative integer
if (n < 0)
printf("Error! Factorial of a negative number doesn't exist.");
else
{
for (i = 1; i<= n; ++i)
{
fact *= i;
}
printf("Factorial of %d = %llu", n, fact);
}
return 0;
}
```

Output

Enter an integer: 5
Factorial of 10 = 120

Multiplication Table Up to 10

```
#include <stdio.h>
int main()
{
   int n, i;
   printf("Enter an integer: ");
   scanf("%d", &n);
   for (i = 1; i<= 10; ++i)
    {
      printf("%d * %d = %d \n", n, i, n * i);
    }
   return 0;
}
```

Output

```
Enter an integer: 9
                9 * 1 = 9
                9 * 2 = 18
                9 * 3 = 27
                9 * 4 = 36
                9 * 5 = 45
                9 * 6 = 54
                9 * 7 = 63
                9 * 8 = 72
                9 * 9 = 81
                9 * 10 = 90
```

To Reverse an Integer number

```
#include <stdio.h>
int main() {
   int n, rev = 0, remainder;
   printf("Enter an integer: ");
   scanf("%d", &n);
```

```
        while (n != 0) {
            remainder = n % 10;
            rev = rev * 10 + remainder;
            n /= 10;
        }
        printf("Reversed number = %d", rev);
        return 0;
    }
```

Output

Enter an integer: 2345
Reversed number = 5432

Program to Check Palindrome

```
#include <stdio.h>
int main()
{
    int n, reversedN = 0, remainder, originalN;
    printf("Enter an integer: ");
    scanf("%d", &n);
    originalN = n;
    // reversed integer is stored in reversedN
    while (n != 0) {
        remainder = n % 10;
        reversedN = reversedN * 10 + remainder;
        n /= 10;
    }
    // palindrome if orignalN and reversedN are equal
    if (originalN == reversedN)
        printf("%d is a palindrome.", originalN);
    else
        printf("%d is not a palindrome.", originalN);
    return 0;
}
```

Output

Enter an integer: 1001
1001 is a palindrome.

Simple Calculator using switch Statement

```
#include <stdio.h>
int main()
{
    char operator;
    double first, second;
    printf("Enter an operator (+, -, *,): ");
    scanf("%c", &operator);
    printf("Enter two operands: ");
    scanf("%lf %lf", &first, &second);
    switch (operator)
    {
    case '+':printf("%.1lf + %.1lf = %.1lf", first, second, first + second);
        break;
```

```
        case '-':printf("%.1lf - %.1lf = %.1lf", first, second, first - second);
            break;
        case '*':printf("%.1lf * %.1lf = %.1lf", first, second, first * second);
            break;
        case '/':printf("%.1lf / %.1lf = %.1lf", first, second, first / second);
            break;
        default:printf("Error! operator is not correct");

        }
        return 0;
    }
```

Output

Enter an operator (+, -, *,): *
Enter two operands: 1.54.5
1.5 * 4.5 = 6.75

Solved Questions

Short Answer Type Questions.

Q.1 Name of the first developer of C programming languages.

Ans:- Dennis Ritchie

Q.2 What are the various types of statement available in C program?

Ans:- C has three types of statement.

(i) Assignment =

(ii) Selection (branching) if (expression) else switch.

(iii) Iteration (looping) while (expression)for(expression;expression;expression) do{block}

Q.3 What is the purpose of i/o statement in 'C'?
Ans:-

- It is used to display a string inputted by gets() function. It isalso used to display an text (message) on the screen for program simplicity.
- Input Output Statement.

Q.4 Explain the importance of C-language?

Ans:-

C is highly portable and is used for scripting system applications which form a major part of Windows, UNIX, and Linux operating system. C is a general-purpose programming language and can efficiently work on enterprise applications, games, graphics, and applications requiring calculations, etc.

Long Answer Type Questions.

Q.1 Give the general structure of a 'C' program, and discuss about each of the lines. ***(2017-Winter)***

Ans:-

Programming in C is a difficult task for someone who is completely oblivious to the basic structure of a C program. After completing this tutorial, you would learn how the Structure of C

Program looks like and soon you would be comfortable writing your own programs with ease!

Part of C program

1. # include <stdio.h> – This command is a preprocessor directive in C that includes all standard input-output files before compiling any C program so as to make use of all those functions in our C program.

2. int main() – This is the line from where the execution of the program starts. The main() function starts the execution of any Program.

3. { (Opening bracket) – This indicates the beginning of any function in the program (Here it indicates the beginning of the main function).

4. /* some comments */ – Whatever is inside /*——-*/ are not compiled and executed; they are only written for user understanding or for making the program interactive by inserting a comment line. These are known as multiline comments. Single line comments are represented with the help of 2 forward slashes "//——".

5. printf("Hello World") –The printf() command is included in the C stdio.h library, which helps to display the message on the output screen.

6. getch() – This command helps to hold the screen.

7. return 0 – This command terminates the C program and returns a null value, that is, 0.

8. } (Closing brackets)- This indicates the end of the function. (Here it indicates the end of the main function)

Q.2 WAP in C to find the real roots of a quadratic equation. *(2017-Summer)*

Ans:-

```
#include <stdio.h>
#include <math.h>
int main()
{
  int a, b, c, d;
  double root1, root2;
  printf("Enter a, b and c where a*x*x + b*x + c = 0\n");
  scanf("%d%d%d", &a, &b, &c);
  d = b*b - 4*a*c;
  if (d < 0)
{
// complex roots, i is for iota (√-1, square root of -1)
printf("First root = %.2lf + i%.2lf\n", -b/(double)(2*a), sqrt(-d)/(2*a));
printf("Second root = %.2lf - i%.2lf\n", -b/(double)(2*a), sqrt(-d)/(2*a));
  }
  else
{
// real roots
root1 = (-b + sqrt(d))/(2*a);
root2 = (-b - sqrt(d))/(2*a);
printf("First root = %.2lf\n", root1);
printf("Second root = %.2lf\n", root2);
}
  return 0;
```

}

EXERCISE

Short Answer Type Questions.

Q.1 Differentiate between Numeric and Character Constant?

Q.2 How can you use a symbolic statement?

Q.3 What do you mean by operator and operand?

Q.4 What are the various types of Operator used in the C programming? **(2016-Summer)**

Q.5 Differentiate between logical and bitwise operator?

Q.6 Differentiate between increment and decrement operator?

Q.7 Differentiate between pre and post increment/decrement operator? **(2013-Summer)**

Q.8 Differentiate between unary plus and unary minus operator?

Q.9 Why conditional operator is called ternary operator?

Q.10 Define Operator?

Q.11 What is the relation of arithmetic operator with relational operator? Explain with an appropriate expression?

Q.12 How can expression with increment and decrement operators will be solved?

Q.13 Differentiate between if-else and else-if statement?

Q.14 Differentiate between ladder if and switch statement? (2017-Summer)

Q.15 What do you mean by conditional control statement?

Q.16 Define an iterative statement in a C program.

Q.17 Give the general syntax of switch…… case statement in C.

Q.18 Differentiate between do-while and while….. do statement in C.

Long Answer Type Questions

Q.1 WAP in C to print all 2-Digit Odd Numbers.

Q.2 WAP in C to Calculate and print the factorial of a given number.

Q.3 WAP in C to Compute and print the sum of the following series. ***(2017-Winter)***

$S= 1 + 1/x + 1/x^2 + 1/x^3 + 1/x^4 \ldots\ldots\ldots\ldots 1/x^n$

Q.4 WAP in C to calculate the sum of the digits of a given number.

Q.5 WAP in C to Compute and print the simple interest and compound interest.

Q.6 WAP in C to compute $(a+b)^2$.

Q.7 WAP in C to interchange value of two variables without using third variable.

Q.8 WAP in to find the sum of the given series. **(2015-Winter)**

$1^n + 2^n + 3^n + 4^n + \ldots\ldots\ldots\ldots m^n$.

Q.9 WAP in C to print.

*

**

***

Q.10 WAP in C to find the prime number. (2017-Summer)

Q.11 WAP in C to find greatest number among three integer numbers. ***(2015-Winter)***

Q.12 WAP in C to compute and print the sum of the following series. ***(2017-Winter)***

$$\boldsymbol{S = 1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{6} + \cdots \frac{1}{20}}$$

Q.13 WAP in C to find whether a number is Armstrong number or not.

****

CHAPTER –7: ADVANCED FEATURES OF C

7.1 Functions and Passing Parameters to the Function (Call by Value and Call by Reference)

Functions

- A function is a group of statements that together perform a task. Every C program has at least one function, which is main (), and all the most trivial programs can define additional functions.
- You can divide up your code into separate functions. How you divide up your code among different functions is up to you, but logically the division is such that each function performs a specific task.
- A function declaration tells the compiler about a function's name, return type, and parameters. A function definition provides the actual body of the function.
- A function can also be referred as a method or a sub-routine or a procedure, etc.

Defining a Function

- The general form of a function definition in C programming language is as follows −
    - return type function name(parameter list) {
    - body of the function
    - }
- A function definition in C programming consists of a function header and a function body. Here are all the parts of a function −
    - **Return Type −** A function may return a value. The return type is the data type of the value the function returns. Some functions perform the desired operations without returning a value. In this case, the return type is the keyword void.
    - **Function Name −** This is the actual name of the function. The function name and the parameter list together constitute the function signature.
    - **Parameters −** A parameter is like a placeholder. When a function is invoked, you pass a value to the parameter. This value is referred to as actual parameter or argument. The parameter list refers to the type, order, and number of the parameters of a function. Parameters are optional; that is, a function may contain no parameters.
    - **Function Body −** The function body contains a collection of statements that define what the function does.

Parameters in C functions

- A Parameter is the symbolic name for "data" that goes into a function. There are two ways to pass parameters in C: Pass by Value, Pass by Reference.

Call by Value

- Pass by Value, means that a copy of the data is made and stored by way of the name of the parameter. Any changes to the parameter have NO effect on data in

the calling function.

- In call by value method, the value of the actual parameters is copied into the formal parameters. In other words, we can say that the value of the variable is used in the function call in the call by value method.

- In call by value method, we cannot modify the value of the actual parameter by the formal parameter.
- In call by value, different memory is allocated for actual and formal parameters since the value of the actual parameter is copied into the formal parameter.
- The actual parameter is the argument which is used in the function call whereas formal parameter is the argument which is used in the function definition.

Call by Reference

- A reference parameter "refers" to the original data in the calling function. Thus, any changes made to the parameter are also made to the original variable.
- In call by reference, the address of the variable is passed into the function call as the actual parameter.
- The value of the actual parameters can be modified by changing the formal parameters since the address of the actual parameters is passed.
- In call by reference, the memory allocation is similar for both formal parameters and actual parameters. All the operations in the function are performed on the value stored at the address of the actual parameters, and the modified value gets stored at the same address.

- There are two ways to make a pass by reference parameter:

ARRAYS

- Arrays are always pass by reference in C. Any change made to the parameter containing the array will change the value of the original array.

The ampersand (&) used in the function prototype. Function (& parameter name)

- To make a normal parameter into a pass by reference parameter, we use the "& param" notation. The ampersand (&) is the syntax to tell C that any changes made to the parameter also modify the original variable containing the data.

Call by Value Example: Swapping the values of the two variables (Swapping not Possible)

```
#include <stdio.h>
void swap(int , int); //prototype of the function
int main()
{
   int a = 10;
   int b = 20;
   printf("Before swapping the values in main a = %d, b = %d\n",a,b);
   swap(a,b);
   printf("After swapping values in main a = %d, b = %d\n",a,b);
// The value of actual parameters do not change by changing the formal parameters in call by value, a = 10, b = 20
}

void swap (int a1, int b1)
{
   int temp;
   temp = a;
```

```
    a=b;
    b=temp;
printf("After swapping values in function a = %d, b = %d\n",a,b); // Formal parameters, a
= 20, b = 10
}
```

Output

Before swapping the values in main a = 10, b = 20
After swapping values in function a = 20, b = 10
After swapping values in main a = 10, b = 20

Call by reference Example:
Swapping the values of the two variables

```
#include <stdio.h>
void swap(int *, int *); //prototype of the function
int main()
{
   int a = 10;
   int b = 20;
   printf("Before swapping the values in main a = %d, b = %d\n",a,b); // printing the value
of a and b in main
   swap(&a,&b);
   printf("After swapping values in main a = %d, b = %d\n",a,b); // The values of actual
parameters do change in call by reference, a = 10, b = 20
}
    void swap (int *a, int *b)
{
   int temp;
   temp = *a;
   *a=*b;
   *b=temp;
printf("After swapping values in function a = %d, b = %d\n",*a,*b); // Formal parameters,
a = 20, b = 10
}
```

Output:

Before swapping the values in main a = 10, b = 20
After swapping values in function a = 20, b = 10
After swapping values in main a = 20, b = 10

7.2 Scope of Variables and Storage Classes, Recursion, Function and Types of Recursion

7.2.1 Scope of variables

- When we declare a variable in a program, it cannot be accessed against the scope rules. Variables can be accessed based on their scope. The scope of a variable decides the portion of a program in which the variable can be accessed. The scope of the variable is defined as follows...
- Scope of a variable is the portion of the program where a defined variable can be accessed.

- The variable scope defines the visibility of variable in the program. Scope of a variable depends on the position of variable declaration.
- In C programming language, a variable can be declared in three different positions and they are as follows...
    - Before the function definition (Global Declaration)
    - Inside the function or block (Local Declaration)
    - In the function definition parameters (Formal Parameters)

Before the function definition (Global Declaration)

Example Program

```
#include<stdio.h>
#include<conio.h>
int num1, num2 ;
void main()
{
void addition() ;
void subtraction() ;
void multiplication() ;
clrscr() ;
num1 = 10 ;
num2 = 20 ;
printf("num1 = %d, num2 = %d", num1, num2) ;
addition() ;
subtraction() ;
multiplication() ;
getch() ;
}
void addition()
{
int result ;
result = num1 + num2 ;
printf("\naddition = %d", result) ;
}

void subtraction()
{
int result ;
result = num1 - num2 ;
printf("\nsubtraction = %d", result) ;
}
void multiplication()
{
int result ;
result = num1 * num2 ;
printf("\nmultiplication = %d", result) ;
}
```

Output:

Inside the function or block (Local Declaration)

Example Program

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

```
#include<conio.h>
void main()
{
void addition() ;
int num1, num2 ;
clrscr() ;
num1 = 10 ;
num2 = 20 ;
printf("num1 = %d, num2 = %d", num1, num2) ;
addition() ;
getch() ;
}
void addition()
{
int sumResult ;
sumResult = num1 + num2 ;
printf("\naddition = %d", sumResult) ;
}
```

Output:

In the function definition parameters (Formal Parameters)

Example Program

```
#include<stdio.h>
#include<conio.h>
void main()
{
void addition(int, int) ;
int num1, num2 ;
clrscr() ;
num1 = 10 ;
num2 = 20 ;
addition(num1, num2) ;
getch() ;
}
void addition(int a, int b)
{
int sumResult ;
sumResult = a + b ;
printf("\naddition = %d", sumResult) ;
}
```

7.2.2 Storage Classes

- Storage classes in C are used to determine the lifetime, visibility, memory location, and initial value of a variable. There are four types of storage classes in C
    - Automatic
    - External
    - Static

- Register

Automatic

- Automatic variables are allocated memory automatically at runtime.
- The visibility of the automatic variables is limited to the block in which they are defined.
- The scope of the automatic variables is limited to the block in which they are defined.
- The automatic variables are initialized to garbage by default.
- The memory assigned to automatic variables gets freed upon exiting from the block.
- The keyword used for defining automatic variables is auto.
- Every local variable is automatic in C by default.

Example

```
#include <stdio.h>
int main()
{
int a; //auto
char b;
float c;
printf("%d %c %f",a,b,c); // printing initial default value of automatic variables a, b, and c.
return 0;
}
```

Output:

```
garbage garbagegarbage
```

Static

- The variables defined as static specifier can hold their value between the multiple function calls.
- Static local variables are visible only to the function or the block in which they are defined.
- A same static variable can be declared many times but can be assigned at only one time.
- Default initial value of the static integral variable is 0 otherwise null.
- The visibility of the static global variable is limited to the file in which it has declared.
- The keyword used to define static variable is static.

Example

```
#include<stdio.h>
static char c;
static int i;
static float f;
static char s[100];
void main ()
{
printf("%d %d %f %s",c,i,f); // the initial default value of c, i, and f will be printed.
}
```

Output:

```
0 0 0.000000 (null)
```

Register

- The variables defined as the register is allocated the memory into the CPU registers depending upon the size of the memory remaining in the CPU.
- We cannot dereference the register variables, i.e., we cannot use &operator for the register variable.
- The access time of the register variables is faster than the automatic variables.
- The initial default value of the register local variables is 0.
- The register keyword is used for the variable which should be stored in the CPU register. However, it is compiler's choice whether or not; the variables can be stored in the register.
- We can store pointers into the register, i.e., a register can store the address of a variable.
- Static variables cannot be stored into the register since we cannot use more than one storage specifier for the same variable.

Example

```
#include <stdio.h>
int main()
{
 register int a; // variable a is allocated memory in the CPU register. The initial default value of a is 0.
printf("%d",a);
}
```

Output:

0

External

- The external storage class is used to tell the compiler that the variable defined as extern is declared with an external linkage elsewhere in the program.
- The variables declared as extern are not allocated any memory. It is only declaration and intended to specify that the variable is declared elsewhere in the program.
- The default initial value of external integral type is 0 otherwise null.
- We can only initialize the extern variable globally, i.e., we can not initialize the external variable within any block or method.
- An external variable can be declared many times but can be initialized at only once.
- If a variable is declared as external then the compiler searches for that variable to be initialized somewhere in the program which may be extern or static. If it is not, then the compiler will show an error.

Example

```
#include <stdio.h>
int main()
{
  extern int a;
  printf("%d",a);
}
```

Output

```
main.c:(.text+0x6): undefined reference to `a'
collect2: error: ld returned 1 exit status
```

7.2.3 Recursion Function

- Recursion is the process of repeating items in a self-similar way. In programming languages, if a program allows you to call a function inside the same function, then it is called a recursive call of the function.

```
void recursion()
{
recursion (); /* function calls itself */
}
int main() {
recursion ();
}
```

- The C programming language supports recursion, i.e., a function to call itself. But while using recursion, programmers need to be careful to define an exit condition from the function, otherwise it will go into an infinite loop.
- Recursive functions are very useful to solve many mathematical problems, such as calculating the factorial of a number, generating Fibonacci series, etc.

7.2.4 Types of Recursion

Recursion are mainly of two types depending on weather a function calls itself from within itself weather two function call one another mutually.

Thus, the two types of recursion are:

- Direct recursion
- Indirect recursion

Recursion may be further categorized as:

- Linear recursion
- Binary recursion
- Multiple recursion

7.3 One Dimensional Array and Multidimensional Array, String Operations and Pointers

7.3.1 One-dimensional array

- Conceptually you can think of a one-dimensional array as a row, where elements are stored one after another.
- Syntax: data type array_name[size];
- datatype: It denotes the type of the elements in the array.
- array_name: Name of the array. It must be a valid identifier.
- size: Number of elements an array can hold. here are some example of array declarations:

```
int num[100];
float temp[20];
char ch[50];
```

7.3.2 Multidimensional Arrays

- The simplest form of multidimensional array is the two-dimensional array. A two-

dimensional array is, in essence, a list of one-dimensional arrays. To declare a two-dimensional integer array of size [x][y], you would write something as follows −

type array Name [x][y];

- Where type can be any valid C data type and array Name will be a valid C identifier.
- A two-dimensional array can be considered as a table which will have x number of rows and y number of columns.
- A two-dimensional array a, which contains three rows and four columns can be shown as follows –

	Column 0	Column 1	Column 2	Column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

- Thus, every element in the array a is identified by an element name of the form a[i][j], where 'a' is the name of the array, and 'i' and 'j' are the subscripts that uniquely identify each element in 'a'.

7.3.3 String operations

- Strings are actually one-dimensional array of characters terminated by a null character '\0'. Thus a null-terminated string contains the characters that comprise the string followed by a null.
- The following declaration and initialization create a string consisting of the word "Hello". To hold the null character at the end of the array, the size of the character array containing the string is one more than the number of characters in the word "Hello."

```
char greeting[6] = {'H', 'e', 'l', 'l', 'o', '\0'};
```

- If you follow the rule of array initialization then you can write the above statement as follows −

```
char greeting[] = "Hello";
```

- Following is the memory presentation of the above defined string in C/C++ −
- Actually, you do not place the null character at the end of a string constant. The C compiler automatically places the '\0' at the end of the string when it initializes the array. Let us try to print the above mentioned string − Live Demo

```
#include <stdio.h>
int main ()
{
   char greeting[6] = {'H', 'e', 'l', 'l', 'o', '\0'};
   printf("Greeting message: %s\n", greeting );
   return 0;
}
```

7.3.4 Pointers

- A pointer is a variable whose value is the address of another variable, i.e., direct address of the memory location. Like any variable or constant, you must declare a pointer before using it to store any variable address. The general form of a pointer variable declaration is −

type *var-name;

- Here, type is the pointer's base type; it must be a valid C data type and var-name is the name of the pointer variable. The asterisk * used to declare a pointer is the same asterisk used for multiplication.

How to Use Pointers?

- There are a few important operations, which we will do with the help of pointers very frequently.
- We define a pointer variable,
- assign the address of a variable to a pointer and
- Finally access the value at the address available in the pointer variable.
- This is done by using unary operator * that returns the value of the variable located at the address specified by its operand. The following example makes use of these operations –

```
#include <stdio.h>
int main () {
int var = 20; /* actual variable declaration */
int *ip; /* pointer variable declaration */
ip = &var; /* store address of var in pointer variable*/
printf("Address of var variable: %x\n", &var );
/* address stored in pointer variable */
printf("Address stored in ip variable: %x\n", ip );
/* access the value using the pointer */
printf("Value of *ip variable: %d\n", *ip );
return 0;
}
```

7.4 Pointer Expression and Pointer Arithmetic Programming, Assignments using the above features.

- Pointers are used to point to address the location of a variable. A pointer is declared by preceding the name of the pointer by an asterisk (*).
- Syntax:

```
datatype *pointer_name;
```

- When we need to initialize a pointer with variable's location, we use ampersand sign(&) before the variable name.

```
// Declaration of integer variable
int var=10;
// Initialization of pointer variable
int *pointer=&var;
```

- The ampersand (&) is used to get the address of a variable. We can directly find the location of any identifier by just preceding it with an ampersand (&) sign.

Example:

```
// This code prints the address of x
#include <stdio.h>
```

```
int main()
{
  int x = 10;
// Prints address of x
  printf("Address of variable x = %p", &x);
  return 0;
}
```

Pointer Arithmetic Programming

Pointer in c is an address, which is a numeric value. Therefore, you can perform arithmetic operations on a pointer just as you can on a numeric value. There are four arithmetic operators that can be used on pointers: ++, --, +, and -

To understand pointer arithmetic, let us consider that ptr is an integer pointer which points to the address 1000. Assuming 32-bit integers, let us perform the following arithmetic operation on the pointer −ptr++

After the above operation, the ptr will point to the location 1004 because each time ptr is incremented, it will point to the next integer location which is 4 bytes next to the current location. This operation will move the pointer to the next memory location without impacting the actual value at the memory location. If ptr points to a character whose address is 1000, then the above operation will point to the location 1001 because the next character will be available at 1001.

Incrementing a Pointer

We prefer using a pointer in our program instead of an array because the variable pointer can be incremented, unlike the array name which cannot be incremented because it is a constant pointer. The following program increments the variable pointer to access each succeeding element of the array −

```
#include <stdio.h>
const int MAX = 3;
void main ()
{
     int  var[] = {10, 100, 200};
     int  i, *ptr;
     ptr = var;   /* let us have array address in pointer */
     for ( i = 0; i< MAX; i++)
     {
        printf("Address of var[%d] = %x\n", i, ptr );
        printf("Value of var[%d] = %d\n", i, *ptr );
        ptr++;      /* move to the next location */
     }
}
```

Output

```
Address of var[0] = bf882b30
Value of var[0] = 10
Address of var[1] = bf882b34
Value of var[1] = 100
Address of var[2] = bf882b38
Value of var[2] = 200
```

Decrementing a Pointer

The same considerations apply to decrementing a pointer, which decreases its value by the number of bytes of its data type as shown below −

```
#include <stdio.h>
const int MAX = 3;
int main ()
{
   int  var[] = {10, 100, 200};
  int  i, *ptr;
  ptr = &var[MAX-1];/* let us have array address in pointer */
  for ( i = MAX; i> 0; i--)
  {
     printf("Address of var[%d] = %x\n", i-1, ptr );
     printf("Value of var[%d] = %d\n", i-1, *ptr );
     ptr--;/* move to the previous location */
   }
   return 0;
}
```

Output

```
Address of var[2] = bfedbcd8
Value of var[2] = 200
Address of var[1] = bfedbcd4
Value of var[1] = 100
Address of var[0] = bfedbcd0
Value of var[0] = 10
```

Pointer Comparisons

Pointers may be compared by using relational operators, such as ==, <, and >. If p1 and p2 point to variables that are related to each other, such as elements of the same array, then p1 and p2 can be meaningfully compared.

The following program modifies the previous example − one by incrementing the variable pointer so long as the address to which it points is either less than or equal to the address of the last element of the array, which is &var[MAX - 1] −

Example

```
#include <stdio.h>
const int MAX = 3;
int main ()
{
  int  var[] = {10, 100, 200};
  int  i, *ptr;
   /* let us have address of the first element in pointer */
  ptr = var;
  i = 0;
   while ( ptr<= &var[MAX - 1] )
   {
      printf("Address of var[%d] = %x\n", i, ptr );
```

```
      printf("Value of var[%d] = %d\n", i, *ptr );
      /* point to the next location */
      ptr++;
      i++;
    }
    return 0;
}
```

Output

```
Address of var[0] = bfdbcb20
Value of var[0] = 10
Address of var[1] = bfdbcb24

Value of var[1] = 100
Address of var[2] = bfdbcb28
Value of var[2] = 200
```

7.5 Structure and Union (Only concepts, No Programming)

What is a structure?

A structure is a user defined data type in C. A structure creates a data type that can be used to group items of possibly different types into a single type.

How to create a structure?

"**struct**" keyword is used to create a structure.

Following is an example

```
struct address
{
  char name[50];

  char street[100];
  char city[50];
  char state[20];
  int pin;
};
```

How to declare structure variables?

- A structure variable can either be declared with structure declaration or as a separate declaration like basic types.
- A variable declaration with structure declaration.

```
struct Point
{
   int x, y;
} p1;
```

The variable p1 is declared with 'Point'.

- A variable declaration like basic data types

```
struct Point
{
   int x, y;
};
int main()
{
   struct Point p1; // The variable p1 is declared like a normal variable
}
```

Union

- A union is a special data type available in C that allows storing different data types in the same memory location.
- You can define a union with many members, but only one member can contain a value at any given time.
- Unions provide an efficient way of using the same memory location for multiple purposes.

Defining a Union:

- To define a union, you must use the union statement in the same way as you did while defining a structure.
- The union statement defines a new data type with more than one member for your program.
- The format of the union statement is as follows:

```
union [union name]
{
member definition;
member definition;
...
member definition;
        };
```

	STRUCTURE	UNION
Keyword	The keyword **struct** is used to define a structure	The keyword **union** is used to define a union.
Size	When a variable is associated with a structure, the compiler allocates the memory for each member. The size of structure is **greater than or equal to the sum of sizes of its members.**	when a variable is associated with a union, the compiler allocates the memory by considering the size of the largest memory. So, size of **union is equal to the size of largest member.**
Memory	Each member within a structure is assigned unique storage area of location.	Memory allocated is shared by individual members of union.
Value Altering	Altering the value of a member will not affect other members of the structure.	Altering the value of any of the member will alter other member values.
Accessing members	Individual member can be accessed at a time.	Only one member can be accessed at a time.
Initialization of Members	Several members of a structure can initialize at once.	Only the first member of a union can be initialized.

Standard Library Functions

Many basic housekeeping functions are available to a C program in form of standard library functions. To call these, a program must #include the appropriate .h file. Most compilers link in the standard library code by default. The functions listed in the next section are

the most commonly used ones, but there are many more which are not listed here.

stdio.h	file input and output
ctype.h	charactertests
string.h	stringoperations
math.h	mathematical functions such as sin() and cos()
stdlib.h	utility functions such as malloc() andrand()
assert.h	the assert() debugging macro
stdarg.h	support for functions with variable numbers of arguments
setjmp.h	support for non-local flow controljumps
signal.h	support for exceptional conditionsignals
time.h	date andtime
limits.h,	float.h constants which define type range values such asINT_MAX

Examples of advance programming in C

Example of Array In C programming to find out the average of 4 integers

```
#include <stdio.h>
int main()
{
   int avg = 0;
   int sum =0;
   int x=0;
   int num[4];/* Array- declaration – length 4*/
   for (x=0; x<4;x++)/* We are using for loop to traverse through the */
   {
      printf("Enter number %d \n", (x+1));
      scanf("%d", &num[x]);
   }
   for (x=0; x<4;x++)
   {
     sum = sum+num[x];
   }
   avg = sum/4;
   printf("Average of entered number is: %d", avg);
   return 0;
}
```

Output:

```
Enter number 1
10
Enter number 2
10
Enter number 3
20
Enter number 4
40
Average of entered number is: 20
```

Example to print the address of array elements

```
#include <stdio.h>
int main( )
{
  int val[7] = { 11, 22, 33, 44, 55, 66, 77 } ;
  /* for loop to print value and address of each element of array*/
  for ( inti = 0 ; i< 7 ; i++ )
  {
    printf("val[%d]: value is %d and address is %d\n", i, val[i], &val[i]);
  }
  return 0;
}
```

Output:

```
val[0]: value is 11 and address is 1423453232
val[1]: value is 22 and address is 1423453236
val[2]: value is 33 and address is 1423453240
val[3]: value is 44 and address is 1423453244
val[4]: value is 55 and address is 1423453248
val[5]: value is 66 and address is 1423453252
val[6]: value is 77 and address is 1423453256
```

Example: Passing Pointer to a Function in C

```
#include <stdio.h>
void salaryhike(int  *var, int b)
{
    *var = *var+b;
}
int main()
{
   int salary=0, bonus=0;
   printf("Enter the employee current salary:");
   scanf("%d", &salary);
   printf("Enter bonus:");
   scanf("%d", &bonus);
   salaryhike(&salary, bonus);
   printf("Final salary: %d", salary);
   return 0;
}
```

Output:

```
Enter the employee current salary:10000
Enter bonus:2000
Final salary: 12000
```

Example for Swapping two numbers using Pointers

```
#include <stdio.h>
void swapnum(int *num1, int *num2)
```

```
{
  int tempnum;
tempnum = *num1;
  *num1 = *num2;
  *num2 = tempnum;
}
int main( )
{
 int v1 = 11, v2 = 77 ;
  printf("Before swapping:");
  printf("\nValue of v1 is: %d", v1);
  printf("\nValue of v2 is: %d", v2);
  /*calling swap function*/
  swapnum( &v1, &v2 );
  printf("\nAfter swapping:");
  printf("\nValue of v1 is: %d", v1);
  printf("\nValue of v2 is: %d", v2);
}
```

Output:

```
Before swapping:
Value of v1 is: 11
Value of v2 is: 77
After swapping:
Value of v1 is: 77
Value of v2 is: 11
```

Example: Program to find the size of an array

```
#include <stdio.h>
int main()
{
    double arr[] = {11, 22, 33, 44, 55, 66};
    int n;
    /* Calculating the size of the array with this formula.
     * n = sizeof(array_name) / sizeof(array_name[0])
     * This is a universal formula to find number of elements in
     * an array, which means it will work for arrays of all data
     * types such as int, char, float etc.
     */
    n = sizeof(arr) / sizeof(arr[0]);
    printf("Size of the array is: %d\n", n);
    return 0;
}
```

Output:

```
Size of the array is: 6
```

Example: Program to Calculate Standard Deviation

```
#include <math.h>
#include <stdio.h>
float calculateSD(float data[]);
int main()
{
    int i;
    float data[10];
    printf("Enter 10 elements: ");
    for (i = 0; i< 10; ++i)
      scanf("%f", &data[i]);
    printf("\nStandard Deviation = %.6f", calculateSD(data));
    return 0;
}
float calculateSD(float data[])
{
    float sum = 0.0, mean, SD = 0.0;
    int i;
    for (i = 0; i< 10; ++i)
    {
        sum += data[i];
    }
    mean = sum / 10;
    for (i = 0; i< 10; ++i)
        SD += pow(data[i] - mean, 2);
    return sqrt(SD / 10);
}
```

Output Enter 10 elements: 1
2
3
4
5
6
7
8
9
10
Standard Deviation = 2.872281

Example: Program to Add Two Matrices

```
#include <stdio.h>
int main()
{
    int r, c, a[100][100], b[100][100], sum[100][100], i, j;
    printf("Enter the number of rows (between 1 and 100): ");
    scanf("%d", &r);
    printf("Enter the number of columns (between 1 and 100): ");
```

```
        scanf("%d", &c);
        printf("\nEnter elements of 1st matrix:\n");
        for (i = 0; i< r; ++i)
            for (j = 0; j < c; ++j)
            {
                printf("Enter element a%d%d: ", i + 1, j + 1);
                scanf("%d", &a[i][j]);
            }

        printf("Enter elements of 2nd matrix:\n");
            for (i = 0; i< r; ++i)
                for (j = 0; j < c; ++j)
                {
                    printf("Enter element a%d%d: ", i + 1, j + 1);
                    scanf("%d", &b[i][j]);
                }
               for (i = 0; i< r; ++i)// adding two matrices
      for (j = 0; j < c; ++j)
        sum[i][j] = a[i][j] + b[i][j];
      printf("\nSum of two matrices: \n");    // printing the result
    for (i = 0; i< r; ++i)
      for (j = 0; j < c; ++j)
      {
                    printf("%d   ", sum[i][j]);
                    if (j == c - 1)
                    {
                        printf("\n\n");
                    }
                }
            return 0;
        }
```

Output

```
Enter the number of rows (between 1 and 100): 2
Enter the number of columns (between 1 and 100): 3
Enter elements of 1st matrix:
Enter element a11: 2
Enter element a12: 3
Enter element a13: 4
Enter element a21: 5
Enter element a22: 2
Enter element a23: 3
Enter elements of 2nd matrix:
Enter element a11: -4
Enter element a12: 5
Enter element a13: 3
Enter element a21: 5
Enter element a22: 6
Enter element a23: 3
Sum of two matrices:
```

```
-2   8   7
10   8   6
```

Example: Multiply Matrices by Passing it to a Function

```
#include <stdio.h>
// function to get matrix elements entered by the user
void getMatrixElements(int matrix[][10], int row, int column)
{
  printf("\nEnter elements: \n");
  for (int i = 0; i< row; ++i)
  {
    for (int j = 0; j < column; ++j)
    {
      printf("Enter a%d%d: ", i + 1, j + 1);
      scanf("%d", &matrix[i][j]);
    }
  }
}

// function to multiply two matrices
void multiplyMatrices(int first[][10],int second[][10], int result[][10], int r1, int c1, int
r2, int c2)
{
  // Initializing elements of matrix mult to 0.
  for (int i = 0; i< r1; ++i)
  {
    for (int j = 0; j < c2; ++j)
   {
      result[i][j] = 0;
    }
  }

  // Multiplying first and second matrices and storing it in result
  for (int i = 0; i< r1; ++i) {
    for (int j = 0; j < c2; ++j) {
      for (int k = 0; k < c1; ++k) {
        result[i][j] += first[i][k] * second[k][j];
      }
    }
  }
}
// function to display the matrix
void display(int result[][10], int row, int column) {
  printf("\nOutput Matrix:\n");
  for (int i = 0; i< row; ++i) {
    for (int j = 0; j < column; ++j) {
      printf("%d  ", result[i][j]);
```

```
            if (j == column - 1)
              printf("\n");
          }
        }
      }

      int main()
      {

      int first[10][10], second[10][10], result[10][10], r1, c1, r2, c2;
        printf("Enter rows and column for the first matrix: ");
        scanf("%d %d", &r1, &c1);
        printf("Enter rows and column for the second matrix: ");
        scanf("%d %d", &r2, &c2);
        // Taking input until
        // 1st matrix columns is not equal to 2nd matrix row
        while (c1 != r2)
        {
          printf("Error! Enter rows and columns again.\n");
          printf("Enter rows and columns for the first matrix: ");
          scanf("%d%d", &r1, &c1);
          printf("Enter rows and columns for the second matrix: ");
          scanf("%d%d", &r2, &c2);
        }
        getMatrixElements(first, r1, c1);// get elements of the first matrix
        getMatrixElements(second, r2, c2);   // get elements of the second matrix
        multiplyMatrices(first, second, result, r1, c1, r2, c2);   // multiply two matrices.
        display  (result, r1, c2);// display the result
        return 0;
      }
```

Output

```
Enter rows and column for the first matrix: 2
3
Enter rows and column for the second matrix: 3
2

Enter elements:
Enter a11: 2
Enter a12: -3
Enter a13: 4
Enter a21: 53
Enter a22: 3
Enter a23: 5

Enter elements:
Enter a11: 3
Enter a12: 3
Enter a21: 5
Enter a22: 0
```

```
Enter a31: -3
Enter a32: 4

Output Matrix:
-21  22

159  179
```

Example: Program to Find the Transpose of a Matrix

```
#include <stdio.h>
int main()
{
    int a[10][10], transpose[10][10], r, c, i, j;
    printf("Enter rows and columns: ");
    scanf("%d %d", &r, &c);
    // Assigning elements to the matrix
    printf("\nEnter matrix elements:\n");
    for (i = 0; i< r; ++i)
        for (j = 0; j < c; ++j)
        {
          printf("Enter element a%d%d: ", i + 1, j + 1);
          scanf("%d", &a[i][j]);
        }

  // Displaying the matrix a[][]

      printf("\nEntered matrix: \n");
      for (i = 0; i< r; ++i)
        for (j = 0; j < c; ++j)
        {
            printf("%d  ", a[i][j]);
            if (j == c - 1)
              printf("\n");
        }
// Finding the transpose of matrix A
    for (i = 0; i< r; ++i)
        for (j = 0; j < c; ++j)
        {
            transpose[j][i] = a[i][j];
        }
    // Displaying the transpose of matrix a
    printf("\nTranspose of the matrix:\n");
    for (i = 0; i< c; ++i)
        for (j = 0; j < r; ++j)
```

```
            {
              printf("%d  ", transpose[i][j]);
              if (j == r - 1)
                 printf("\n");
              }
          return 0;
       }
```

Output

```
Enter rows and columns: 2
3
Enter matrix elements:
Enter element a11: 1
Enter element a12: 4
Enter element a13: 0
Enter element a21: -5
Enter element a22: 2
Enter element a23: 7

Entered matrix:
1  4  0
-5  2  7

Transpose of the matrix:
1  -5
4  2
0   7
```

Using Call by Reference

```
#include <stdio.h>
void cyclicSwap(int *a, int *b, int *c);
int main()
{
   int a, b, c;
   printf("Enter a, b and c respectively: ");
   scanf("%d %d %d", &a, &b, &c);
   printf("Value before swapping:\n");
   printf("a = %d \nb = %d \nc = %d\n", a, b, c);
   cyclicSwap(&a, &b, &c);
   printf("Value after swapping:\n");
   printf("a = %d \nb = %d \nc = %d", a, b, c);
   return 0;
}
void cyclicSwap(int *n1, int *n2, int *n3)
{
   int temp;
   // swapping in cyclic order
```

```
    temp = *n2;
    *n2 = *n1;
    *n1 = *n3;
    *n3 = temp;
}
```

Output

```
Enter a, b and c respectively: 1
2
3
Value before swapping:
a = 1
b = 2
c = 3
Value after swapping:
a = 3
b = 1
c = 2
```

Solved Questions

Short Answer Type Questions.

Q.1 What is an array? What is its importance in C? *(2015-Summer)*

Ans:-

Arrays a kind of data structure that can store a fixed-size sequential collection of elements of the same type. An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type.

Q.2 What is a single dimensional array? How can it be initialized?

Ans:-

1D Arrays in C language – A list of items can be given one variable name using only one subscript and such a variable is called single sub-scripted variable or one dimensional array.

Q.3 What are the various string operations?

Ans:-

strcat - concatenate two strings. strchr -string scanning operation. strcmp - compare two strings. strcpy -copy a string.

Q.4 How can you reverse a string?

Ans:- C program to reverse a string

```
int main()
{
char s[100];
printf("Enter a string to reverse\n"); gets(s);
strrev(s);
printf("Reverse of the string: %s\n", s);
return 0;
}
```

Long Answer Type Questions.

Q.1 Write a program to find the sum of even numbers of an array? *(2017-Summer)*

Ans:

```
#include<stdio.h>
main()
{
 int a[10],i,sum=0;
```

```
printf("Enter upto 5 Values: ");
for(i=0; i<5; i++)
scanf("%d",&a[i]);
for(i=0; i<5; i++)
{
if(a[i]%2==0)
sum=sum+a[i];
 }
printf("Total Sum of Even values is: %d ",sum);
getch();
}
```

Output :

```
Enter upto 5 Values: 2 3 5 4 7
Total Sum of Even values is: 6
```

Q.2 Write a program which prints the smallest element?

Ans:

```
#include<stdio.h>

int main()
{
   int a[30], i, num, smallest;
   printf("\nEnter no of elements :");
   scanf("%d", &num);
   //Read n elements in an array
   for (i = 0; i< num; i++)
      scanf("%d", &a[i]);
   //Consider first element as smallest
   smallest = a[0];
   for (i = 0; i< num; i++)
  {
      if (a[i] < smallest)
      {
         smallest = a[i];
      }
   }
   // Print out the Result
  printf("\nSmallest Element : %d", smallest);
  return (0);
}
```

Output:

```
Enter no of elements : 5
11 44 22 55 99
Smallest Element : 11
```

EXERCISE

Short Answer Type Questions.

Q.1 What is a multi-dimensional array? How is it initialized? ***(2017-Summer)***

Q.2 How can a two-dimensional array be declared in C program?

Q.3 What is built-in function? Give some example of these functions?

Q.4 Give some examples of Function Prototypes?

Q.5 Differentiate between strcat( ) and strcpy( )?

Q.6 How can you apply arithmetic operations on strings?

Q.7 How is a structure different or similar from/to an array?

Q.8 How can you create structure variables? ***(2015-Summer)***

Q.9 How can you define array within structures?

Q.10 Define a union? ***(2014-Winter)***

Q.11 How can you declare a union?

Q.12 Define a pointer? ***(2015-Winter) (2016-Winter)***

Q.13 What is the relation between pointers and arrays? ***(2015-Summer)***

Q.14 What is the function of malloc() library function?

Q.15 What is the meaning of reference?

Q.16 What is address operator?

Q.17 Give an example of pointer initialization?

Q.18 Briefly explain the association between one dimensional arrays and pointers?

Q.19 Explain the relation between pointers and functions?

Long Answer Type Questions

Q.1 Write a program which replaces every blank space in a string with underscore?

Q.2 How can you calculate the size of an array? Illustrate with the help of example. ***(2014-Summer)***

Q.3 How we can call a function by

i. Reference

ii. Value

Q.4 Write a C program which illustrates passing of arguments by reference? ***(2016-Summer)***

Q.5 Write a program which reads a string and prints the number of words init?

Q.6 What is the different between putc() and puts() and putchar() functions? **(2016-Winter)**

****

Chapter-wise Multiple Choice Questions

CHAPTER-1: MCQ

Q.1 The "0" and "1" in the binary numbering system are called Binary digits or known as:

a) Bytes
b) Kilobytes
c) Bits
d) Kilobits

Ans: c) Bits

Q.2 The generation based on VLSI microprocessor:

a) 1st
b) 2nd
c) 3rd
d) 4th

Ans: d) 4th

Q.3 ULSI stands for?

a) Ultra Large Scale Integration
b) Under Lower Scale Integration
c) Ultra Lower Scale Integration
d) Under Large Scale Integration

Ans: a) Ultra Large Scale Integration

Q.4 The period of _____ generation was 1952-1964.

a) 1st
b) 2nd
c) 3rd
d) 4th

Ans: b) 2nd

Q.5 The brain of any computer system is:

a) ALU
b) Memory

c) CPU
d) Control Unit

Ans: c) CPU

Q.6 CD-ROM is a:

a) Semiconductor memory
b) Memory register
c) Magnetic memory
d) None of above

Ans: d) None of above

Q.7 A hybrid computer:

a) Resembles digital computer
b) Resembles analogue computer
c) Resembles both a digital and analogue computer
d) None of the above

Ans: c) Resembles both a digital and analogue computer

Q.8 What was the computer invented by Attanasoff and Clifford?

a) Mark I
b) ABC
c) Z3
d) None of above

Ans: b) ABC

Q.9 Which of the following is not an input device?

a) OCR
b) Optical scanners
c) Voice recognition device
d) COM (Computer Output to Microfilm)

Ans: d) COM (Computer Output to Microfilm)

Q.10 When was vacuum tube invented?

a) 1900
b) 1906
c) 1910
d) 1880

Ans: b) 1906

CHAPTER-2 : MCQ

Q.1 What does part number, part description and number of parts ordered belong to?

a) Output
b) Input
c) Feedback
d) Control

Ans: b) Input

Q.2 CPU is mainly responsible for:

a) Calculations
b) Processing the data
c) Both 1 and 2
d) Neither 1 nor 2

Ans: c) Both 1 and 2

Q.3 What is the use of control unit of a microprocessor?

a) To accept input data from keyboard
b) To perform arithmetic and logic functions
c) To store data in memory
d) All of the above

Ans: b) To perform arithmetic and logic functions

Q.4 In the CPU of a computer, the logical unit is mainly responsible for :

a) Control flow of information
b) Comparing numbers
c) Producing result
d) Mathematical operation

Ans: b) Comparing numbers

Q.5 CPU is made up of two main components, and they are :

a) Registers and main memory
b) Control unit and registers
c) ALU and bus

d) Control unit and ALU

Ans: d) Control unit and ALU

Q.6 A bit is the measuring unit of the width of a processor's data path. Commonly used data path is of:

a) 24 bits
b) 32 bits
c) bits
d) 16 bits

Ans: c) 8 bits

Q.7 What is the combination with which central processing unit is made ?

a) Arithmetic logic and control unit
b) Control and storage
c) Control and output unit
d) All of the above

Ans: a) Arithmetic logic and control unit

Q.8 Which language is the set of rules that tells the computer what operation to perform?

a) Programming language
b) Command language
c) Procedural language
d) Structures

Ans: a) Programming language

Q.9 Which one of the following statement is true for the machine language ?

a) Programs were first written in this language
b) Computer understands only this language
c) It differs from computer to computer
d) All of the above

Ans: d) All of the above

Q.10 The language which is understood by the computer is:

a) Machine language
b) Low level language
c) High level language
d) All of the above

Ans: a) Machine language

CHAPTER-3 : MCQ

Q.1 Junk e-mail is also called:

a) spam
b) spoof
c) sniffer script
d) spool

Ans: a) spam

Q.2 Office LANS, which are scattered geographically on large scale, can be connected by the use of corporate:

a) CAN
b) DAN
c) LAN
d) WAN

Ans: d) WAN

Q.3 The device used to carry digital data on analogue lines is called as:

a) Modem
b) Multiplexer
c) Modulator
d) Demodulator

Ans: a) Modem

Q.4 Malicious software is known as:

a) Badware
b) Malware
c) Maliciousware
d) Illegalware

Ans: b) Malware

Q.5 A program that performs a useful task while simultaneously allowing destructive acts is:

a) Worm
b) Trojan horse

c) Virus
d) Macro virus
Ans: b) Trojan horse

Q.6 What is the name of an application program that gathers user information and sends it to someone through the Internet ?

a) A virus
b) Spybot
c) Logic bomb
d) Security patch
Ans: b) Spybot

Q.7 ISDN stands for:

a) Integrated Services Digital Network
b) Integrated Subscriber Digital Network
c) Internet Services Digital Network
d) Integrated Several Digital Network
Ans: a) Integrated Services Digital Network

Q.8 Which of the following is a network topology?

a) LAN
b) WAN
c) MAN
d) BUS
Ans: d) BUS

Q.9 The first web browser is:

a) Mosaic
b) Netscape
c) Internet explorer
d) Collabra
Ans: a) Mosaic

Q.10 Which one of following are set of rules and procedures to control the data transmission over the internet:

a) IP address
b) Domains
c) Protocol
d) Gateway
Ans: c) Protocol

CHAPTER-4 : MCQ

Q.1 Which one of following is collection of related fields that can be treated as a unit by some application program:

a) field
b) record
c) file
d) database

Ans: b) record

Q.2 Which of the following is not a part of the usage information ?

a) data created
b) identity of creator
c) owner
d) last date modified

Ans: c) owner

Q.3 Several instructions execution simultaneously in:

a) processing
b) parallel processing
c) serial processing
d) multitasking

Ans: b) parallel processing

Q.4 A term that refers to the way in which the nodes of a network are linked together:

a) network
b) topology
c) connection
d) interconnectivity

Ans: b) topology

Q.5 The Details view show all of the following about a file EXCEPT:

a) name
b) size
c) type
d) password

Ans: d) password

Q.6 An easy way to sort files is to:

a) right click on a file in Details view
b) click on the column header in Details view
c) click the sort icon in Details view
d) alphabetize them

Ans: b) click on the column header in Details view

Q.7 After creating a file management system on your computer, you should do all of the following EXCEPT:

a) delete files that are no longer needed
b) move files to appropriate folders
c) rename folders to be more meaningful
d) run the Task Management

Ans: a) delete files that are no longer needed

Q.8 When you right-click on a folder on the hard drive and choose Delete, the files:

a) are erased
b) go into the Recycle Bin
c) are moved into the header section of the hard drive
d) go into the Old Documents folder

Ans: b) go into the Recycle Bin

Q.9 One of the first steps when creating a file management system is to :

a) create new folders
b) delete files that will be moved
c) change to Details view
d) select multiple files

Ans: a) create new folders

Q.10 Folder names should :

a) use numbers only
b) be as short as possible
c) not contain spaces
d) be meaningful and recognizable

Ans: d) be meaningful and recognizable

CHAPTER-5 : MCQ

Q.1 An Algorithm represented in the form of programming languages is _____

a) Flowchart
b) Pseudo code
c) Program
d) None

Ans: c) Program

Q.2 Keep the statement language _____ while writing a pseudo code.

a) Dependent
b) Independent
c) Case sensitive
d) Capitalized

Ans: b) Independent

Q.3 Flowcharts and Algorithms are used for

a) Better Programming
b) Easy testing and Debugging
c) Efficient Coding
d) All

Ans: d) All

Q.4 Which of the following is not a keyword?

a) Read
b) Write
c) Start
d) Endif

Ans: c) Start

Q.5 _____ is used to show hierarchy in a pseudo code.

a) Indentation
b) Curly Braces
c) Round Brackets
d) Semicolon

Ans: a) Indentation

Q.6 _____ are identified by their addresses, we give them names (field names

/ variable names) using words.

a) Memory variables
b) Memory Locations
c) Memory Addresses
d) Data variables

Ans: b) Memory Locations

Q.7 _____ begins with lower case letters.

a) Keywords
b) Variables
c) Tokens
d) Functions

Ans: b) Variables

Q.8 A symbol used for grouping.

a) ()
b) {}
c) []
d) “”

Ans: a) ()

Q.9 A statement used to close the IF block.

a) ELSE
b) ELSEIF
c) END
d) ENDIF

Ans: d) ENDIF

Q.10 In structural language, we can't add a new sort of

a) Loop
b) Function
c) Variable
d) Constant

Ans: a) Loop

CHAPTER-6 : MCQ

Q.1 Who invented C Language?

a) Charles Babbage
b) Grahambel
c) Dennis Ritchie
d) Steve Jobs

Ans: c) Dennis Ritchie

Q.2 C is _____ type of programming language?

a) Object Oriented
b) Procedural
c) Bit level language
d) Functional

Ans: b) Procedural

Q.3 A C program is a combination of?

a) Statements
b) Functions
c) Variables
d) All of the above

Ans: d) All of the above

Q.4 Types of Integers are?

a) short
b) int
c) long
d) All the above

Ans: d) All the above

Q.5 Choose a correct statement about C break; statement?

a) break; statement can be used inside switch block
b) break; statement can be used with loops like for, while and do while.
c) break; statement causes only the same or inner loop where break; is present to quit suddenly.
d) All the above.

Ans: d) All the above.

Q.6 What is the output of the C Program?

```
int main()
{
   if( 4 < 5 )
      printf("Hurray..\n");
      printf("Yes");
   else
      printf("England")

   return 0;
}
```

a) Hurray..Yes
b) Hurray..
Yes
c) Compiler error
d) None of the above

Ans: c) Compiler error

Q.7 Operator % in C Language is called?

a) Percentage Operator
b) Quotient Operator
c) Modulus
d) Division

Ans: c) Modulus

Q.8 Can you use C Modulo Division operator % with float and int?

a) Only int variables = Okay
b) Only float variables = Okay
c) int or float combination = Okay
d) Numerator int variable, Denominator any variable = Okay

Ans: a) Only int variables = Okay

Q.9 Which loop is faster in C Language, for, while or Do While?

a) for
b) while
c) do while
d) All work at same speed

Ans: d) All work at same speed

Q.10 What is the output of C program with switch statement or block?

```
int main()
{
int a;

switch(a);
{

printf("DEER ");
}

printf("LION");
}
```

a) LION
b) DEER LION
c) Compiler error
d) None of the above

Ans: b) DEER LION

CHAPTER-7 : MCQ

Q.1 Choose correct statement about Functions in C Language.

a) A Function is a group of c statements which can be reused any number of times.
b) Every Function has a return type.
c) Every Function may no may not return a value.
d) All the above.

Ans: d) All the above.

Q.2 A function which calls itself is called a ___ function.

a) Self Function
b) Auto Function
c) Recursive Function
d) Static Function

Ans: c) Recursive Function

Q.3 How many values can a C Function return at a time?

a) Only One Value
b) Maximum of two values
c) Maximum of three values
d) Maximum of 8 values

Ans: a) Only One Value

Q.4 What are types of Functions in C Language?

a) Library Functions
b) User Defined Functions
c) Both Library and User Defined
d) None of the above

Ans: c) Both Library and User Defined

Q.5 Every C Program should contain which function?

a) printf()
b) show()
c) scanf()
d) main()

Ans: d) main()

Q.6 What is the maximum number of statements that can present in a C function?

a) 64
b) 128
c) 256
d) None of the above

Ans: d) None of the above

Q.7 Arguments passed to a function in C language are called ___ arguments.

a) Formal arguments
b) Actual Arguments
c) Definite Arguments
d) Ideal Arguments

Ans: b) Actual Arguments

Q.8 An array Index starts with?

a) -1
b) 0
c) 1
d) 2

Ans: b) 0

Q.9 What is the output of C program with arrays and pointers?

```
int main()
{
int a[3] = {20,30,40};
int *p[3];
p=&a;
printf("%d", *p[0]);
}
```

a) 20
b) address of element 20
c) Garbage value
d) Compiler error

Ans: d) Compiler error

Q.10 What is actually passed to PRINTF or SCANF functions?

a) Value of String
b) Address of String
c) End address of String
d) Integer equivalent value of String

Ans: b) Address of String

REFERENCES

1. https://www.tutorialspoint.com/
2. https://www.geeksforgeeks.org/
3. https://www.codecademy.com/
4. https://en.wikipedia.org/