# Chapter-01 Computer Organisation

## Introduction to computer

The word "computer" comes from the word "compute which means to calculate. Hence a computer is normally considered to be a calculating device, which can perform arithmetic operations at high speed.

More accurately, a computer may be defined as a device, which operates upon data.

Data is the raw material used as input to data processing and information is the processed data obtained as the output of data processing.

## Charcterstics of computers

1. Automatic -

A machine is said to be automatic, if it works by itself without human intervention. Compute are automatic machines

2. Speed :

A computer is a very fast devie. It can perform in a few seconds, ~~ear~~

The unit of speed are the microsecond ($10^{-6}$), nano second ($10^{-9}$), piccosecond ($10^{-12}$)

3. Accuracy

In addition to being very fast, compiters are very accurate. The accuracy of a computer is consistently high.

4. Diligence

A computer is free from tiedness, lack of concentration. It can work for hours without creating any error.

5. Versatility

A computer can perform completely different type of work at the same time.

Every piece of information can be retained as long as desired by the user, and can be recalled as and when required.

7. No I.Q.

8. No Feelings :- Computers are devoid of emotions.

## Evolution of computers

### Abacus

It is the first counting device which was developed in china more than 3000 years ago. This device basically consists of a rectangular wooden frame and beads. Counting was done by moving the beads from one end of the frame to the other.

### Napier's Bones

It is a device which contains a set of rods made of bones.

It was developed by John Napier, a Scottish Mathematician and hence the device was named as Napier's Bones.

The device was mainly developed for performing multiplication and division.

### Pascaline

Pascaline is a calculating machine developed by Blaise Pascal, a French Mathematician. It was the first device with an ability to perform additions and subtractions on whole numbers.

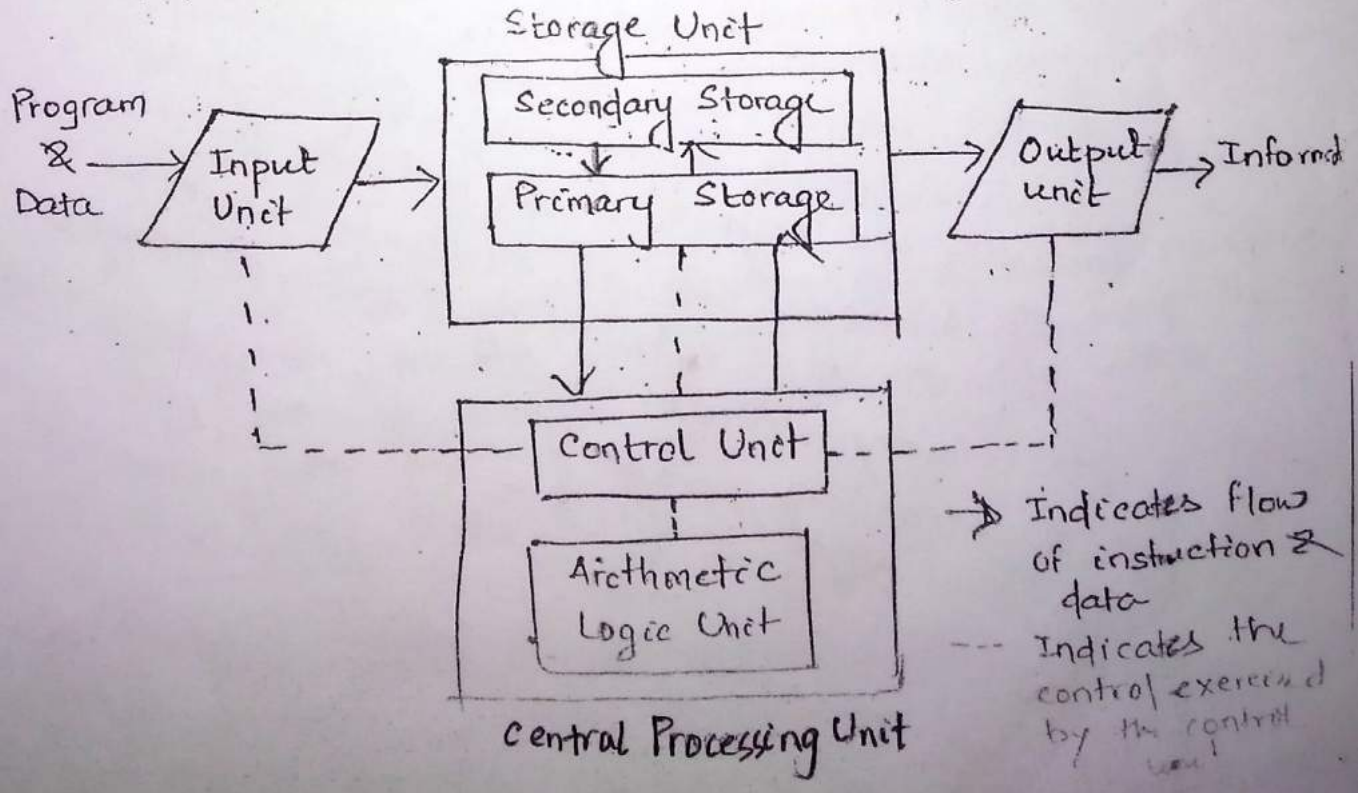
### Punched Card System

Punched Card System is used for storing and retrieving data. This was invented by Herman Hollerith, an American Statistician in US Census Bureau.

# Basic Computer Organization

All computer systems perform the following five basic operations, for converting raw input data into information, which is useful to their users :

1. Inputting : The process of entering data and instructions into the computer system
2. Storing :- Saving data and instructions to make them readily available for initial or additional processing, as and when required.
3. Processing : Performing arithmetic operations or logical operations on data, to convert them into useful information.
4. Outputting :- The process of producing useful information or results for the user
5. Controlling :- Directing the manner and sequence in which all of the above operations are performed.



## Input unit

The following functions are performed by an input unit :

1. It accepts (or reads) the instructions and data from the outside world.
2. It converts these instructions and data in computer acceptable form.
3. It supplies the converted instruction and data to the computer system for further processing.

## Output unit :-

The following functions are performed by an output unit

1. It accepts the results produced by the computer which are in coded form
2. It converts these coded result to human acceptable form.
3. It supplies the converted results to the outside world.

## Storage unit

The specific functions of the storage unit are to hold -

1. The data and instructions required for processing
2. The intermediate result of processing
3. Final result of processing before these results are released to an output device.

The storage unit of all computers is comprised of the following 2 types of storage.

1. Primary storage
2. Secondary storage

Primary storage :-

also known as main memory

It is used to hold pieces of program instructions and data, intermediate results of processing, and recently produced results of processing of the jobs.

As soon as the computer system is switched off or reset, the information held in the primary storage disappears.

Limited storage capacity because it is very expensive.

made up of semiconductor devices.

Secondary storage :-

also known as auxillary storage.

It is used to supplement the limited storage capacity and the volatile characterstic of primary storage.

Cheaper and non volatile.

Most commonly used is the magnetic disk.

## Arithmetic Logic Unit

All arithmetic calculations are performed and all comparisons are MADE in the ALU.

## Control Unit

The control unit acts as a central nervous system, for the other components of the computer system. It manages and coordinates the entire computer system.

## Central Processing Unit

It is the brain of the computer system.

Control Unit + Arithmetic Logic Unit = CPU

imp Generation of Computer :-

First generation (1940-1956)

→ In this generation of computer the technology used is vacuum tube.

→ Vacuum tube are fragile glass device that could control and amplify the electronic signal.

→ Instructions were written in machine level language which is the low level language consisting of 0 1 1.

e.g → ENIAC
EDVAC
UNIVAC

Advantages :-

→ Latest calculating device at that time.

Disadvantage :-

→ bulky in size, requiring a large room for installation.

→ Generated a lot of heat.

→ Consume a lot of electricity

→ Constant maintainance is required because of frequent hardware failure.

→ Very expensive

Second Generation (1956-1964)

Second generation of computers were manufactured using transistor.

Transistor were more reliable electronic switching device made up of the material called semiconductor

Symbolic or Assembly languages are used

e.g :- UNIVAC III
IBM 3070

Advantages-

Faster, cheaper, smaller & more reliable
Consumes less electricity

Disadvantage :-

Commercial production of these computers were difficult

## Third Generation (1964-1971)

Used technology is integrated circuit.
Several electronic components such as transistor, register and capacitor were placed on a silicon chip.

Integrated circuit contains only about 10-20 components named Small Scale Integration
Later it become possible to integrate about 100 components on a single chip named Medium scale integration (MSI).

Time sharing OS is used.

e.g → UNIVAC 1100
IBM 360/370

Advantages :-

Faster than second Generation of computer.
Smaller, cheaper & more reliable
Widely used for ~~specific~~ scientific and business application.
It has faster and larger primary & secondary memory

Disadvantages :-

Difficult to maintain.
Got heated very quickly.

## Fourth Generation (1971-1989)

It uses the technology called large scale Integration (LSI) which is integrated over 30,000 electronic circuit on a single chip and it was followed by very large scale integration (VLSI) about 1 million electronic components on a single chip.

This technology leads to the creation of microprocessor.

In this generation GUI, Mouse and handheld devices are used.

e.g → (a) IBM PC
(b) Apple II

Advantages:-

Smaller, cheaper, faster and more reliable than 3rd generation.

It has faster and larger primary and secondary memory.

Disadvantage :-

They were not an intellegent system.

## Fifth Generation (1989 onwards)

It uses the technology called Ultra Large scale integration.

These computers are completely based on the new concept of artificial intelligence.

It uses parallel processing OS and super conductor technology.

Artificial intelligence touches the area like gaming, expert system, natural languages, robotics etc.

ex: IBM notebook

imp

# Evolution of Computers

## Abacus

It is the first counting device which was developed in China more than 3000 years ago. This device basically consists of a rectangular wooden frame and beads. Counting was done by moving the beads from one end of the frame to the other.

## Napier's Bone

It is a device which contains a set of rods made of bones.

It was developed by John Napier, a scottish Mathematician and hence the device was named as Napier's Bones.

The device was mainly developed for performing multiplication and division.

## Pascaline :-

Pascaline is a calculating machine developed by Blaise Pascal, a French Mathematician in 1645.

It was the first device with an ability to perform additions and subtractions on whole numbers.

## 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 British Navy.

He upgraded this to, invent an Analytical engine which could store program instructions initially coded on punched cards and subsequently shared

Punched card equipment

It is used for storing and retriving data.

This is invented by Herman Hollerith, an American Statistician.

Atanasoff-Berry Computer

It was invented by J.V. Atanasoff and C. Berry.

It uses vacume tubes for both data storage and data computation.

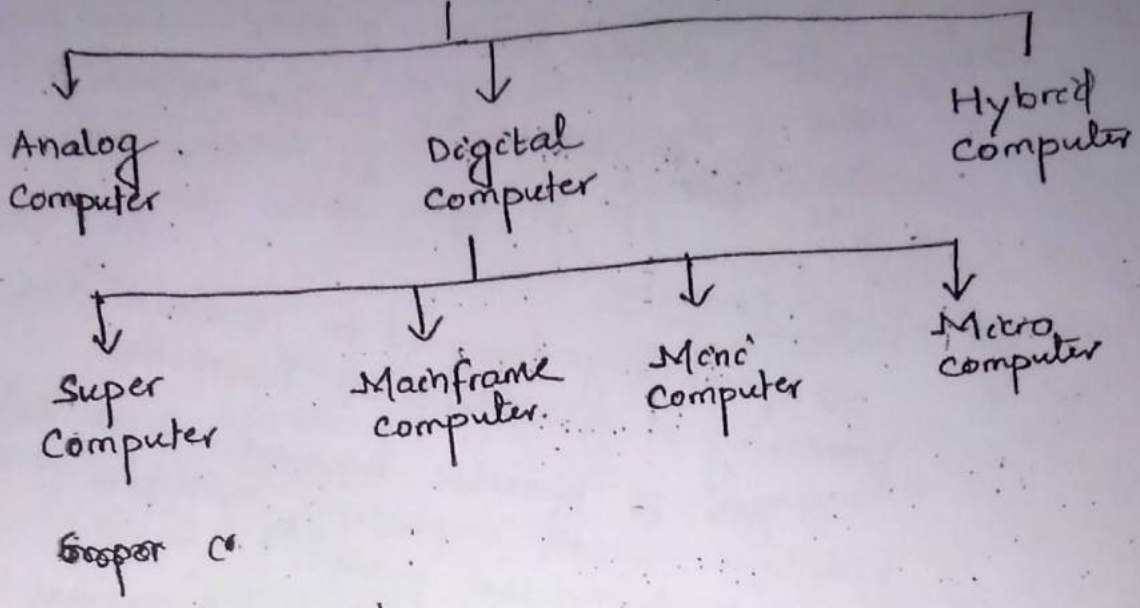
Subsequently ENIAC was designed and accepted as the general purpose computer.

UNIVAC-1

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.

imp

# Classification of Computers




## Analog computers

→ Mostly used in industries in process control activities
→ Work on analog data such as variation in temperature, pressure, speed, voltage etc.
→ They are specific to a particular application area. Therefore cost of such computer differ from application to application on the complexity.
→ Uses are very limited.

## Digital computer

→ General purpose computers, which work on digital / binary data.
→ Speed and accuracy are very high.

## Hybrid computer

Used to control the entire process
The analog feature of such computer enables it to measure the physical quantities such as 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 O/P may be taken in a paper as hard copy, may be seen on a display device or may be converted into analog form to automatically control various process

Digital computers are further classified into 4 types

Super Computer :-

→ 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.

→ Very expensive and used in very high end numerical processing, geographical information system etc.

e.g → Cray, Param, Anupam

→ Speed is measured in GFLOPs i.e Giga Floating Point Operations Per Second

→ uses their own OS and programming language so vary from computer to computer

Mainframe Computer

→ Intended for substantial high volume data processing

→ Large primary memory

→ Substantial processing capabilities (MIPS)

Processing speed = 30-100 MIPS

Word length - More than 64 bits

I/O device - wide range of peripheral devices

Internal storage :- More than 1 GB

Application

Space research, university connectivity, WAN implement

## Mini Computers

→ Fairly Large primary memory
→ Medium scale processing capability i.e lesser than mainframe but higher than personal computers
→ Can connect upto 500 terminals on LAN
→ Supports wide range of application areas

### Application

In the field of engineering and scientific organizations
Educational Institutes
Universities
Small / Medium business organization

### Specification

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

e.g → IBM
Burroughs

## Micro computer

Smallest and least expensive computers
PCs.

### Typical features

portable
require minm power
Processing power as appropriate.
Memory capacity as sufficient.
Ease of use and support to various kind of OS

Microcomputers
- PC
- PCXT Extended Technology
- PCAT Advanced Technology

# Typical specification of PC

Processor - I 8086 / I 8088 micro processor
Memory - 640 KB of RAM
Two 360K Floppy disk drive
Numerical Processor - I 8087
System bus - 8 bit databus & 16 bit address
Clock speed - 8 MHz

## PC AT

Processor - 80386 / 80486 / Pentium
Memory - 2 MB - 512 MB
- Floppy disk drive - 1.4 MB
Hard " " - 1.2 MB to 80 GB
System bus - 32 - 64 bit
OS - MSDOS, Windows, UNIX, Linux
Clock speed - Upto 3 GHz

imso

# Input Devices

→ Data and instructions are entered into a computer through input devices.

→ An input device converts an input data and instructions into binary form which can be accepted by the computer ~~form~~.

Different types of input devices are keyboard, mouse, scanner, touchscreen, trackball, joystick

## Keyboard

It is the most common input device.

The keyboard contains alphanumeric keys, special keys, fienctcon keys.

Alphanumeric keys are used to input most of the letters (A-Z), numbers (0-9) and other characters like space, . / ? < > " " ; : - ! @ # $ % ^ & * ( ) _ - + = \ | [ ] { }

The special keys are Enter or Return key, the Backspace key, the Delete key, the Insert key, the Shift key, the Caps Lock key, the Num Lock key, the Tab key, the Alt key.

The functional keys are used to ~~support~~ perform a set of operations.

## Mouse

It is the most widely used pointing device.

Mouse is designed to fit comfortably under the palm of our hand, so that ~~the movement of the pointer~~ we can control the movement of the pointer on the screen and make selection from the screen by pressing the button provided on the mouse.

Mouse

Now a days optical mouse are becoming popular which does not have a ball rather functions with a LED and a sensing mechanism to detect a location on the screen.

(iii) Trackball

→ It is a pointing device, similar to mouse.
→ The ball is rolled with finger to move the cursor around the screen.
→ It is used on the laptop where there is no space for conventional mouse.

(iv). Joystick :-

Joy stick is an input device used generally in computer games.
It is a handheld control stick that allows a player to control the movements of cursor on monotor.

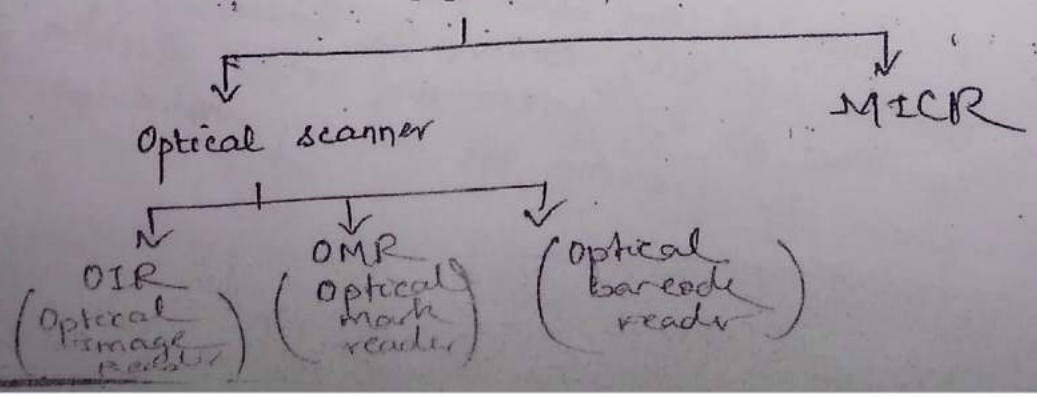
(v) Touch Screen :-

In this device user only touch the monitor screen to input data in to the computer.
e-g → ATM counter

(vi) Scanner

It is a kind of input device which converts printed text, graphics, pictures into a digital form.

2 types of scanners are
(a) Optical scanner
(b) Magnetic ink character reader



→ Optical Image Reader (OIR)

This scanner optically scan image, printed text, handwriting which is converted into a digital form.

By placing the picture on the flat transparent surface of the scanner, any hardcopy can be converted into the digital form.

→ Optical Mark Reader (OMR)

This scanner are capable of recognizing a specified type of mark made by pencil or pen on special designed OMR sheet

These marks are detected by an OMR and the corresponding signal are sent to the processor.

→ Optical Barcode Reader (OBR)

Data code is in the form of small lines of varying thickness and spacing between them.

An OBR can read such bars and converted them in to electrical pulse to be processed by computer.

(b) Magnetic Ink Character Reader (MICR) :-

It is widely used by banks to process large volumes of cheques and deposit forms written everyday.

A special ink is called magnetic ink (ink with iron oxide particles) is used to write character on the cheque and deposit forms which are to be processed by an MICR like bank identification code, account number etc.

(viii) Biometric Sensor

A biometric sensor is an input device that recognizes individuals based on their physical or behavioural tracts (fingerprint, voice, face, etc.) It converts these traits into electrical signals.

(ix) Web Camera

It is a video camera that captures images as data for a computer system & is normally connected to a computer through USB ports.

## Output Devices

The output device receives information from the computer and provide them to user

It converts machine readable information into people readable form

Ex:- Monitor, Printer, Speaker, Plotter etc.

### (a) Monitor

Also known as Visual Display Unit

For desktop computers, cathode ray monitors (CRT) are gradually being replaced by flat screen monitors such as the Liquid Crystal Displays (LCD) and Thin Film Transistor (TFT) displays used with Laptop computers because they use less power and take up less space

Softcopy

A softcopy is an output, which is not produced on a paper or some material; which can be touched or carried to another place in physical form.

These are temporary in nature and vanish after the use. Examples of soft copy are output on a monitor or sound produced by a voice response system.

Hard copy

It is an output on a paper or any other material which can be touched and carried for being shown to others. These are permanent in nature.

O/P's of printer and plotters are hard copy outputs

(b) Printer :-

It is an output device that produces a hard copy of data.

Printers can be divided into 3 categories by the way they print

Serial Printer - Also called character printer.
Print a single character at a time.
Inexpensive and slow

Line Printer - Print a line at a time
Expensive and fast

Page Printer - Also called a Laser printer.
Prints a page at a time.
Usually expensive.

Printers are classified into 2 forms according to the use of hammer

Impact Printers :- Printers which has a physical contact between print head and paper while printing.
Hammers hits ribbons, papers or print head.
e.g :- Dot matrix printer
Daisy wheel printer
Drum printer
Chain printer

Nonimpact printers: Printers which does not have a physical contact between the pri head and paper during printing don't have a hammer.
eg- inkjet printer, laser printer

(c) Plotter:-

It is a special purpose output device capable of printing sophisticated graphs, charts, maps and three dimensional graphics as well as high quality colored documents.

It can produce for large printout than normal printer.

Mainly used in many engineering applications, design like architectural plan, design for mechanical components of an aircraft or a car.

There are 2 main types of plotter.

→ Drum Plotter.
→ Flatbed Plotter

(d) Screen Image Projector (SIP) :-

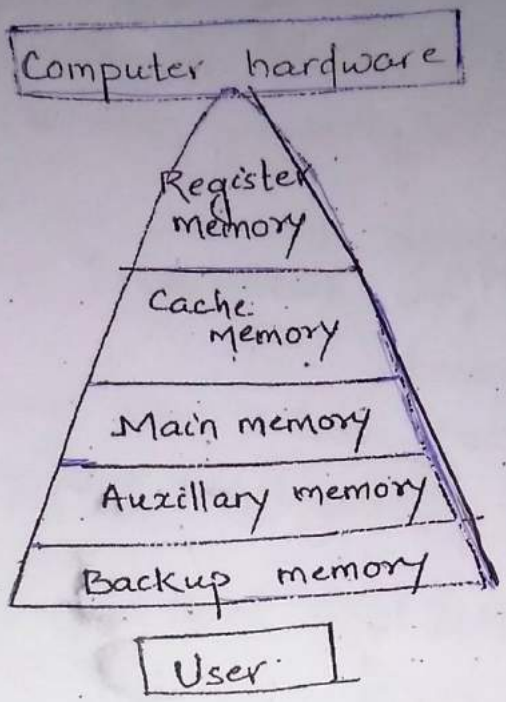
It is used to project information from a computer onto a large screen such as a cloth screen or a wall.

(e) Speaker

Also known as voice recognisation system. It produces sound.

They receive audio input from the computer's sound card and produce audio o/p in the form of sound waves.

# Classification of computer memory




## Register Memory

→ It is integrated inside CPU.
→ It consists of a number of flipflops arranged in a certain manner.
→ Small capacity
→ used for storing data & instruction temporarily during the execution of an instruction.
→ Special purpose registers :- MAR, MDR, IR, PC
→ Fastest memory

## Cache memory

It is a small memory situated between CPU and main memory.

The purpose of this memory is to hold/store frequently needed instruction or data from the main memory during the execution process.

This is a semiconductor memory which is having very low access time and hence is a fast memory.

## Main Memory

There are 2 types of main memory
i e Random access memory
Read only memory

RAM is volatile in nature i.e when the power goes off, the data stored in RAM automatically erased.
Used for both read & write operation
2 types - Static RAM & Dynamic RAM

ROM used to store small system programs permanently.
Nonvolatile in nature which means data stored in ROM remains even after the power goes off.
Various types of ROMs are there,

PROM - Programmable ROM
used once to write data

EPROM - Erasable Programmable ROM
used for both erasing and programming the ROM

EEPROM - Electrically erasable programmable ROM
UVEPROM - Ultra Violet erasable programmable ROM

## Auxillary Memory / Secondary Memory

The auxillary memory or secondary memory is a memory where data & program are stored permanently.

| | | capacity |
|---|---|---|
| e.g > | Floppy disk | 1.44 MB |
| | Hard disk | 20 MB - 80 GB |
| | CD - ROM | 680 MB - 700 MB |

Back up memory

Used by the user for keeping back up of important data for future reference.

Magnetic tapes and CD ROMs are mainly used.

Difference between RAM & ROM

| RAM | ROM |
|---|---|
| i) It refers to Random Access Memory. | i) It refers to Read Only Memory. |
| ii) Temporary | ii) Permanent |
| iii) We can read & write | iii) It is a read only memory. |
| iv) Volatile in nature | iv) Nonvolatile in nature |

# Software Concepts

A computer system mainly comprises 2 categories of resources i.e. computer hardware & computer software.

The computer hardware refers to all the physical components presents in a computer which we can touch i.e. all the tangible components.

→ Similarly computer software is the set of instructions which instructs the hardware what to do and how to do it.

→ Computer software can have various functions such as controlling the hardware, communication with other software etc.

→ Computer software can't be touched physically.

→ It can be classified into 2 types.

a) System software
b) Application software
c) Utility software
d) Device drivers

System software:

System software are designed to control the operation of computer hardware and support it for error free computation.

Some common system softwares are

→ Operating system
→ Language Processor
Device Drives
Utility Programs

The Operating System is a system software which is responsible for managing the various computer resources.

Language processors are the category of system software, which is responsible for translating and interpreting the program written by using programming languages. e.g → Compilers, Interpreters, Assemblers

Device Drivers are the system software which generally comes along with a peripheral device which is used to establish an error free and easy communication between device and computer.

Utility Programs are capable of interacting with the computer hardware for various purpose. Generally for system maintenance activity. e.g → antivirus software, data compression utility

## Application Software

It is a set of one/more program designed to solve a specific program or a specific task such as library, hospital management system

Difference between System software & application software

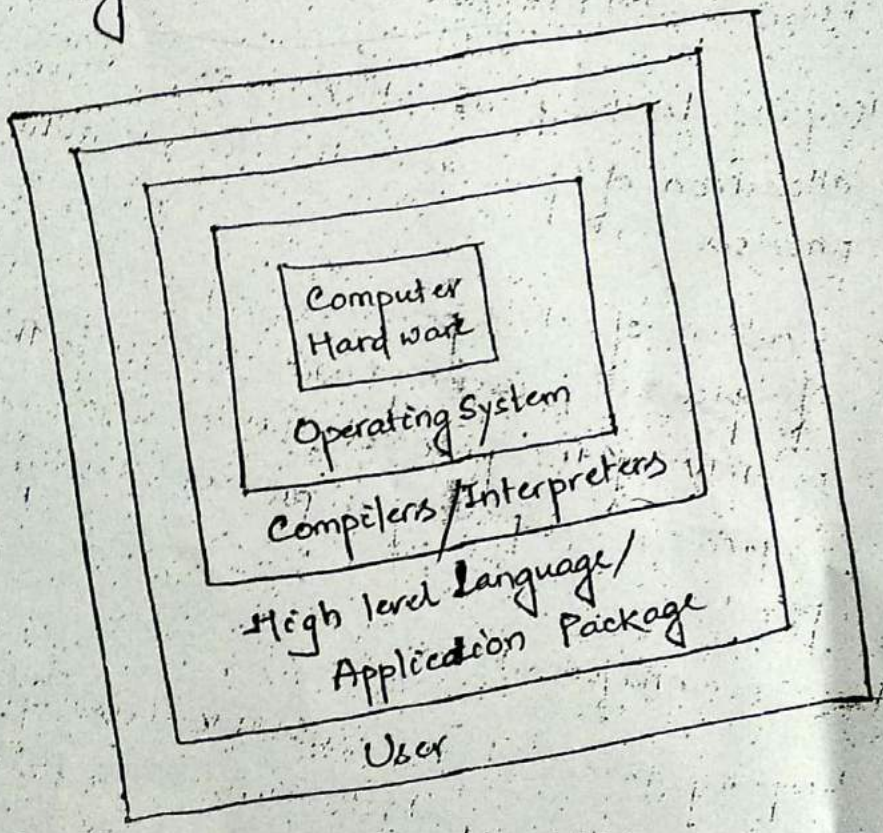
| System software | Application software |
|---|---|
| Enables the user to interact with the h/w components | Designed to solve a specific program / a specific task |
| Machine depended | Machine independent. |
| The programmer must understand the architecture of machine and the h/w details to develop a system s/w | Does not need to |

Imp

# Operating System

Operating system is the software, which is installed in a computer to act as an interface between computer and user.

Operating system is the program, after being initially loaded into the computer by a boot program manages all the other programs in a program.




## Functions of OS

→ Resource management
→ Processor management
→ Memory management
→ Device management
→ Information management

## Resource Management

→ Keeping track of all available resources

→ Allocation of resources to the various requesting jobs.

→ Deallocation of resources.

## Process Management

→ Keeping track of all active processes and allocation of processors to various active processes

→ Creation of child processes and termination of process, assigning and changing the priority of the processes, block and allowing a process, suspending a process, delaying a process

## Memory management

Keeping track of the available memory
Allocating the memory to different process
and deallocation of memory from the process

## Device management

Allocation of devices to process
and deallocation of devices

## Types of OS

→ OS are broadly classified into 2 groups

(a) Single user

(b) Multi user

(a) Single User :- This OS allows single user

This type of OS is designed to run either one job at a time or support multiple tasking of the same time.

e.g → Windows 98, XP

(b) Multi user :-

This type of OS allows different users to take advantage of the computer resources simultaneously

e.g → UNIX, LINUX

Some of the typical Operating System are multi user.

(a) Batch Operating System

Batch processing System supports processing of jobs one at a time. The jobs are submitted in a batch. It is the responsibility of the operating system to schedule the jobs in a queue and assign system resources one by one.

(b) Multiprogramming system

The drawback of batch processing systems is that it does not provide optimum CPU utilization.

Multiprogramming utilizes maximum CPU time by running multiple programs simultaneously

In order to increase the resource utilization system supporting multiprogramming approach allow more than one task to utilize the CPU time at any moment

The OS picks one of the program and starts executing. During execution of program it may need input, output operation to complete. In a sequential execution environment the CPU would sit idle. But in a multiprogramming system the OS will simply switch over the next program.

(c) Multiprocessing System

This system works with 2 or more CPU within a single computer system.

In this system, all CPUs may be equal or some may be reserved for special purposes.

Multiprocessing OS enables several programs to run concurrently.

e.g → UNIX

(d) Time sharing OS

This OS makes time slice of the CPU time and distribute among the multiple user sitting at various terminals.

Time sharing OS allows users to share the system simultaneously. Each user is allocated resources for a particular time slot.

The switching between users is so fast that each user thinks that he/she is getting the system resource continuously.

(e) Real time (RTOS) system

In this OS the total correctness of an operation depends not only upon its logical correctness, but also upon the time in which it is performed.

(f) Network Operating System

It is the software that includes special functions for connecting computers and devices into a local-area network.

It controls a network, and its message traffic and queues, controls access by multiple users to n/w resources such as files and provides for certain administrative functions including security.

e.g → Windows NT, Netware, Unix, Mac OS, Xenix

## Features of DOS, WINDOWS & UNIX

(Imp)

### Disk Operating System

The first widely installed OS for PCs.

The first version PC-DOS was developed for IBM by Bill Gates

DOS is the method

The main functions of DOS are to manage disk files, allocate system resources according to the requirement

# Features of DOS

→ Single User OS
→ provides a hierarchical file system and supports many application packages.
→ Command based OS.
  - → Internal command
  - → External command

Internal commands are included in command.com file whereas each external command reside in separate files.

## DOS internal commands

cd - change the current directory
md - make a new directory
cls - clear the screen
dir - To display the directory contents
rd - To remove/delete directory
ren - To rename a file/directory
date - display the current date
time - " " time
ver - " version of DOS
copy - copy one or more files to another location
exit - To quit the COMMAND.COM

## DOS external command

External commands reside in separate files on your hard drive and have an extension of .COM, .EXE, or .BAT.

diskcopy.com - Copies the content of one floppy disk to another

attrib.exe - sets the read-write-execute attribute of a file

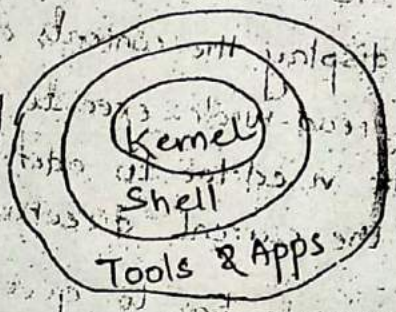
tree.com - Displays the entire file structure in a tree manner.

# Unix

→ Developed in 1969 by a group of AT & T employees at Bell labs including Ken Thompson, Dennis Ritchie, Douglas McIlroy and Joe Ossanna

→ It is functionally organized at 3 levels:

* The Kernel which schedules tasks and manage storage.
* The shell which connects and interprets users commands, calls programs from memory, and executes them
* The tools and applications that offer additional functionality to the OS




## Features of Unix

→ Portability permits it to move from one brand of computer to another with a minimum of code changes as it is written mostly in C language; therefore it is easy to modify the OS to make it suitable for any hardware.

→ Machine independent → hides the machine architecture from user, making it easier to write application programs that will run on any machine.

→ Multiuser operating system supports multiple users simultaneously

→ Hierarchical file system to store information

→ Unix shell provides the user required services
→ Utility modules
→ Pipes & fitters enables the user to give multiple commands in a single command.
→ Security mechanism which does not allow any program to enter into the core part

## Unix Commands

cp - Copy a file
lp - To print a file
cd - To change directory
mkdir - To create a directory
rmdir - To delete a directory
date - To print the current date
who - Display the current user
cat - create & display the contents of a file
chmod - Change read-write-execute permission
vi - invokes the vi editor to edit files
pwd - displays the current directory

All the unix commands has to given in lower case alphabets as Unix command interpreter is case sensitive

## Windows Operating System

It is the most widely used operating system on a personal computer

### Features of Windows operating system

GUI (Graphics User Interface)
It provides the command in the form of menu and icons. The GUI OSs are operated by using mouse.

- Desktop display area that represents the kind of objects one might find on a real desktop: documents

- Icon is a small graphical symbol which represents a particular executable program.

- Folder ≈ directory in DOS. We can create various folders on the storage device to keep relevant file in these folders.

- Taskbar normally present at the bottom of the screen, contains the active application program.

- Titlebar normally present at the top of the window contains the title of that window.

- Menubar present just below the titlebar of the window & contains pull down menu item.

- A window contains vertical scrollbar and horizontal scroll bar if the normal window size is larger than the opened window.

- A toolbar is generally displayed just below the menubar, contains buttons for various tools.

- At the top right corner, there are 3 buttons known as minimise, maximise and close buttons.
  (appln is closed, but still running) (appln is clouded) (appln window occupied full screen)

- Dialogue box provide an interface to enter text for setting different parameters.

- Accessory contain frequently needed tools & utilities.

- Control Panel contains various options for system maintenance, adding/removing program

- My computer contains the entire secondary storage

- Recycle bin contains all the deleted files

# Programming language

Computer can only understand binary data. But it is too difficult for a normal programmer to learn the language of 0 & 1 and write program by using them. Therefore, programming language is invented.

There are 3 categories of programming language.

(i) Machine level language
(ii) Assembly language
(iii) High level language

## Machine level language

→ Also known as low level language.
→ Programs written in binary format.
→ Does not require an interpreter for execution.

**Adv**

→ Executed very fast.

**Disadv**

(i) Machine dependant (programmer need to have an idea about the computer hardware).
(ii) The programs are more prone to error.
(iii) Difficult to understand and modify.
(iv) Not friendly to user/programmer.

## Assembly language

Programs are written using mnemonic codes
Minimises the probability of error

(i) An instruction is given by using mnemonic codes
(ii) Numeric address location can be addressed by using alphanumeric variable

Each instruction consist of 2 parts (ops)
- opcode and the operand
Opcode - tells the computer what to do
Operand - gives the address of operand on which the operation will be carried out

### Advantage

→ Easier to understand
→ Easier for locating errors and debugging

### Limitation

→ Machine dependant
→ Knowledge of h/w architecture is required

## High Level language

→ A high level language is a programming language, where a programmer can write program in English, like language.

→ It is user friendly
→ But A program written in HLL has to go through a translator, where it is interpreted into the corresponding binary format

## Advantage

→ Machine independent
→ Easier to learn and use
→ Gives minm no. of errors
→ documentation part os better
→ Easy to modify & maintain

## Compiler & interpreter

### Compoler

It is a program which translates a high level Language program into a machine level languag

A compiler works with 3rd generation Languages such as Java, C & higher level language

### Characterstics

- ~~Specify~~ Spends a lot of time analyzing and processing the program
- The resulting executable is some form of machine-specific binary code.
- The computer hardware interprets the resulting code.
- Program execution is fast.

### Interpreter

The language processor which converts a high level language into machine language line by line as well as execute it

If there is any error in the program, translation and execution stops and the error is reported for debugging. The conversion and execution resumes only after the error is rectified.

This is a slow process and consumes high memory as the interpreter is also required to execute the correct code along with reporting the errors.

Difference between compiler and interpreter

| Compiler | Interpreter |
| --- | --- |
| It saves the machine code permanently for future reference | The machine code produced by interpreter is not saved. |
| Occupies more memory | Occupies less memory |
| Faster | slower |

## Comparison between Dos & Windows

| Windows | DOS |
|---|---|
| 1. It is graphics user interface | 1. It is character user interface. |
| 2. Powerful & faster | 2. Less powerful & slower |
| 3. It provides in-built commands in various applications & accessories which can be used on a mouse click | 3. All the commands have to be typed at the DOS prompt |
| 4. Virtually all DOS commands are available in WINDOWS | 4. Not available |
| 5. It provides a multitasking environment. | 5. It does not provide multitasking environment |
| 6. It supports graphics | 6. It does not support graphics. |

## Difference between UNIX & DOS

| UNIX | DOS |
|---|---|
| 1. Powerful OS | 1. Less powerful OS |
| 2. Supports multitasking and multiprogramming environment and hierarchical file system | 2. Does not support multitasking and multiprogramming environment and hierarchical file system. |
| 3. Supports visual display and graphics | 3. Does not support visual display and graphics |
| 4. Supports networking of computers | 4. Does not supports networking of computers |
| 5. Secure | 5. Not safety |
| 6. Support concepts of shelling and kernel. | 6. Does not supports concept of shelling & kernel |
| 7. rich webhosting services | 7. It does not have any feature to host web pag |

## Computer Virus

It is a kind of malicious software, that when executed, replicates by inserting copies of itself 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.

A computer virus may corrupt or delete data on a computer.

Computer viruses are most easily spread by attachments in e-mail messages or by instant messaging messages or by using download on the internet

Some common types of viruses are

### Boot Viruses

It infect the master boot record in the hard disk.

They replace the boot record program (which is responsible for loading the operating system in memory) by copying it elsewhere on the disk or overwriting it. Therefore the boot virus load into memory when the computer boot.

e.g. * From * store virus
* Disk killer * Michael Angelo

The most dangerous. Often known as system virus

### Program Virus

It infects executable files having extension like .exe, .com, .bin, .drv, .ovl, .sys etc.

These viruses are loaded in memory during execution of the files, along with them.

The virus program remain active in the memory and multiply itself making the memory full.

Examples * Sunday
* Cascade

## Macro Virus

a series of commands that helps ↑ automating some tasks

infects the macros within a document or template

When we open a word processing or spread sheet document, the macro virus is activated and it infects the normal template.

This virus propagates from one computer to another through the infected document file.

eg: * DMV * Word concept
* Nuclear

## The Multipartite Virus

- A hybrid of boot and program virus.

They first infect the program files and when the infected program is executed, these virus infect the boot record.

When the system is booted next time, the virus from the boot record loads itself to the memory and infects other program.

Ex:- Invader
Flip
Tequela

## The Polymorphic Virus

It is capable of encrypting its code in different manner so that each appears different in each infection.

These virus are difficult to detect.

Ex:- Cascade
Evil
Proud
Virus 101

## The Stealth virus

It uses certain technique to avoid detection.

They usually direct the disk head to read a wrong sector instead of the one in which they reside, or they change the reading of infected file's size in the directory.

Examples

* Joshi
* Whale
* Frodo

## Detection and Prevention of Virus

### Virus Life cycle

Each virus goes through a life-cycle comprising of the following phases

→ Virus creation
→ Virus infection and replication
→ Virus activation
→ Virus detection
→ Virus eradication

A computer virus has to be created by a programmer who writes a program which later spread as the computer virus.

After this the virus infects certain computer by many means and replicates itself in the hard disk memory of the computer. This process of infection and replication is a continuous process through which the computer virus copies itself from one computer to other.

Once a computer virus infects a computer, it gets activated automatically. Depending on the nature of virus it activates itself on occurance of a particular event such as certain date, user executes a particular file, user opens an infected e-mail etc.

Virus is detected by some anti-virus program or other types of diagnostic programs.

Then the virus is eradicated through an appropriate antivirus software

## Symptoms of Virus

→ Computer is giving problem during booting or takes a lot of time for booting.
→ restarting automatically.
→ hanging when the user tries to execute a particular program
→ Displaying some unusual figures / signs
→ Performing some operations automatically though the user has not given any command for that operation.
→ Giving some message such as "insuffecient memory", "Diskfull", etc.
→ Harddisk is accessed usually many times.

A computer can be affected / infected by computer virus in either of the following ways

* By inserting a virus infected CD or pendrive to the system
* Through the LAN setup where one of the computer belonging to the n/w has been affected.
* Through internet and email.
* Through installation of illegal/pirated s/w's specially games.

Prevention!

* Don't allow outside CD/Floppy or Pen drive to be used without proper scanning.
* Always install and update a suitable antivirus s/w from an authorized source
* Protect the system by setting antivirus s/w and the firewall to the auto protection mode
* Don't visit websites which are not reputed
* Always protect your computer from unauthorized use by setting a password
* Don't open unknown emails received in your mailbox

* Check the size of the executable files at interval.

— o —

## Macro Viruses

Macro is an executable program (e.g. instructions opening a file, starting an application) embedded in a word processing document (e.g. MS Word)

A macro virus is attached to a word document

Document is loaded and opened on the host system.

When the macro executes, it copies itself to the global macro file.

The global macro can be activated/spread, when new documents are opened.

## Multipartite Virus

A multipartite virus is a computer virus that infects and spreads in multiple ways.

# Application of computer in different domains

Business

A computer has high speed of calculation, diligence, accuracy, reliability or versatility which has made it an integrated part in all business organizations.

Computer is used in business organizations for payroll calculations, budgeting, sales analysis, financial forecasting, managing employee database, maintenance of stocks, etc.

Buildings

Architects use computer animated graphics to experiments with possible exteriors and to give clients a plan of their proposed buildings.

Education :-

Computer based education enhances the knowledge of the student at a much faster pace than the old traditional methods.

Energy

Energy companies use computers to locate oil, coal, natural gas and uranium. These companies can figure out the site of a natural resources, its concentration and other related figures.

Law enforcement :-

Recent innovation in computerised law enforcement include national fingerprint files and computer modeling of DNA, which can be used to match traces from a criminal's body such as blood at a crime scene.

Transportations :-

Computers are used in cars to monitor fluid levels, temperatures and electrical

Money

Computers have helped to fuel the cashless economy enabling the widespread use of credit cards, debit cards and instantaneous credit checks by banks and retailers.

Health and Medicine.

Computers are helping to monitor the extremely ill in the intensive care unit and provide cross sectional views of the body.

Doctors use computers to assist them in diagnosing certain diseases of the sort.

Scientific Research :-

Because of high speed characteristics of computer system, researchers can simulate environments, emulate physical characteristics and allow scientists to proof of their theories in a cost effective manner.

Communication with the world :-

The computers are most popular for their uses to connect with others on the world wide web. Therefore, communication between 2 or more parties is possible which is relatively cheap. Emailing, teleconferencing and the use of voice messages are very fast, effective and surprisingly cheaper as well.

Training

It is much more cheaper and effective to teach pilots how to fly in a computerized cockpit or simulators, than is real airplanes.

Paperwork :

Computer systems will increasingly cut down the paperwork. All the necessary data and information is transferred to the memory of the computer. This makes managing various tasks easier, faster, more effective than the manual system.

CHAPTER-3

# Computer Network and Internet

Network :-

→ A computer network is a system of interconnected computers and peripheral devices to exchange data.

→ Each device on the network is known as node.

→ Each node has a unique address.

## Network goals

- Resource sharing

By keeping a number of computer in a network, it becomes possible to share the various types of resources in terms of computer hardware, computer software and peripheral devices to be shared between the individual system. This breaks the geographical barrier of resource usage.

- Sharing the job load.

A huge task is divided into small modules and distributed to various computers present in a network. This technique reduces the load on individual computer, but finishes the whole task in time. This technique is often referred to as parallel processing of jobs

- Achieving reliability

By making a system available at any instant of time, we achieve reliability. In a networked system, if a system fails, then we can use another system present in a network to attend an ongoing job. This increases the reliability of the system.

# Protocol

A protocol is a set of rules that governs the communication between computers on a network.

These rules include guidelines that regulate the following characterstics of a network:

- access method
- allowed physical topologies
- types of cabling
- Speed of data transfer

## Types of Network Protocols

### Ethernet :-

It uses an access method called CSMA/CD (Carrier Sense Multiple Access/Collision Detection)

Here each computer listens to the cable before sending anything through the network. If the n/w is clear, the computer will send. If some other node is already transmitting on the cable, the computer will wait and try again when the line is clear.

Sometimes, two computers attempt to transmit at the same time, collision may occur. Each computer then backs off and waits a random amount of time.

Allowed physical topologies - linear bus, star, or tree

Types of cabling - twisted pair, coaxial or fibre optic cable

Speed of data transfer - 10-1000 Mbps

## Local talk :-

Developed by Apple Computer
Method - CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance)
Allowed physical topologies - Linear bus, star or tree
Types of cabling - Twisted pair cable
Speed of transmission - 230 Kbps

## Token ring

Developed by IBM
Access method - Token passing.

In Token ring, the computers are connected so that the signal travels around the network from one computer to another in a logical ring.

A single electronic token moves around the ring from one computer to the next. If a computer does not have information to transmit, it simply passes the token to the next workstation. If a computer wishes to transmit and receives an empty token, it attaches data to token. At this point, the data is captured by the receiving computer.

Types of cabling - Twisted pair / fibre optic cable
Speed of transmission - 4 - 16 Mbps
Allowed physical topology - Star wired ring

## FDDI

(Fiber Distributed Data Interface)
It is used primarily to interconnect 2 or more LANs often over Large distances.
Access method - Token passing.
Allowed physical topology - dual ring
Speed of transmission - 100 Mbps

## ATM (Asynchronous Transfer Mode)

It transmits data in small packets of fixed size
Allowed physical topology - star
Types of cabling - fibre optic, twisted pair cable
Speed of data transfer - 155 - 2488 Mbps

# Data transmission mode

The way in which data is transmitted from one place to another is called data transmission mode. It is also called data communication mode.

It indicates the direction of flow of information. Also called directional modes.

## Types of data transmission mode.

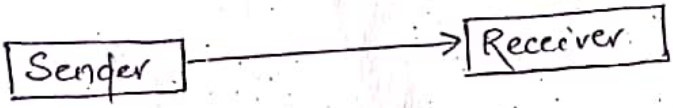
1. Simplex
2. Half duplex mode
3. Full duplex mode

## 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~~ receive data but cannot send it.

e.g → Data sent from computer to printer
Radio and T.V transmissions

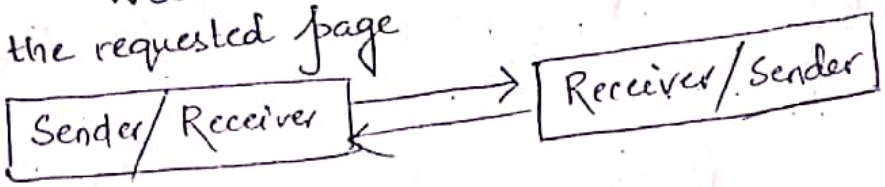



## Half-Duplex Mode

Data can flow in both directions but only in one direction at a time. In this mode, data is sent and received alternatively.

e.g :- A one lane bridge

Internet browsing :-
User sends request to web server for a web page.
Webserver receives request and sends data of the requested page



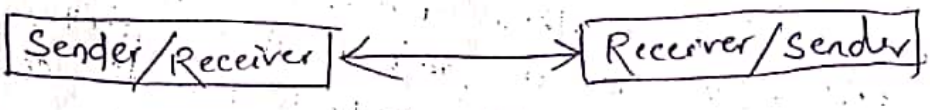
## Full-Duplex Mode

Data can flow in both direction at the same time

Fastest directional mode of data communication

eg → Telephone communication system

Automobile traffic on a two lane road




## What is a topology?

The topology of a network is the geometric representation of the relationship of all the links and linking devices (usually called nodes) to one another.

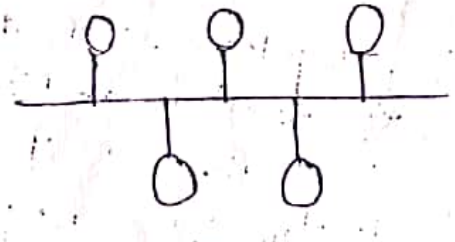
There are 5 basic topologies

1. Bus
2. Star
3. Ring
4. Tree
5. Mesh

### Bus

Bus networks use a common backbone to connect all devices.

A single cable, the backbone functions as a shared communication medium that devices attach or tap into with an interface connector.



Bus

Advantage :-

Easy installation

Disadvantage :-

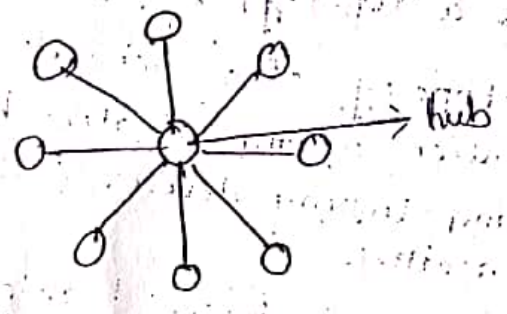
Difficult reconnection and fault isolation.

A fault in the bus cable stops all transmission.

## Star

In a star topology, each device has a dedicated point to point link only to a central controller, called a hub. The devices are not directly connected to one another.

If one device wants to communicate to another, it sends the data to the controller, which then relays the data to the other connected device.




Star

### Advantage

- Easy to install and reconfigure
- Less cabling
- If one link fails, only that link is affected. All other links remain active. So, easy to fault identification and fault isolation.

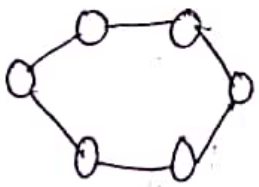
### Disadvantage

- If the hub/central controller fails, nodes attached are disabled.

## Ring

In a ring topology, each device has a dedicated point to point connection only with the two devices on either side of it.

All messages travel through a ring in the same direction (ie either clockwise or anticlockwise)



Advantage

Easy to install and reconfigure

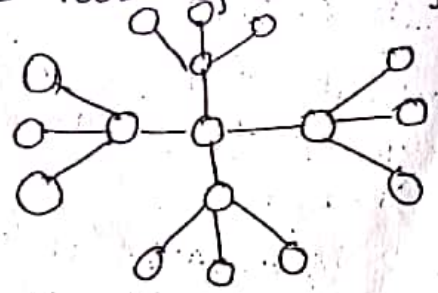
Disadvantage

Unidirectional traffic. In a simple ring, a break in the ring can disable the entire network.

## Tree / Expanded Star

Tree topologies integrate multiple star topologies together onto a bus.

In its simplest form, only hub devices connect directly to the tree bus and each hub functions as the "root" of a tree of devices.



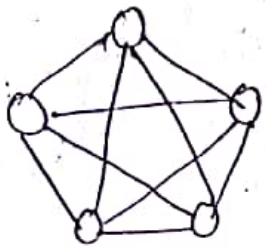
Disadvantage

- More difficult to configure and wired.
- If the backbone line breaks, the entire segment goes down

## Mesh Topology

In a mesh topology, every device has a dedicated point to point link to every other device

A fully connected mesh network therefore has $n(n-1)/2$ physical channels to link n devices.



Advantage

A mesh topology is robust. If one link becomes unusable, it does not incapacitate the entire system.

finally point to point links make fault identification and fault isolation easy.

## Disadvantage

- Installation & reconnection are difficult.
- Very expensive.

## Types of networks

Mostly networks are classified on the basis of geographical spread. There can be 3 types of networks

- Local Area Network
- Metropolitan Area network
- Wide Area Network

### Local Area Network (LAN)

Small computer networks that are confined to a localised area (e.g. an office, a building or a factory) are known as Local area network.

The key purpose of a LAN is to serve its users in resource sharing. The hardware as well as software resources are shared through LANs.

It is limited in size, typically spanning a few hundred meters and no more than a mile.

It is fast, with speeds from 10 Mbps - 10 Gbps.

### Metropolitan Area Network (MAN)

Metropolitan Area Networks are the networks spread over a city

e.g :- Cable TV networks

A MAN typically covers an area of between 5 and 50 km. diameter

Ex :- Telephone company

## Wide Area Network

The networks spread across countries are known as WANs.

A wide area network is a group of computers that are separated by large distances and tied together

The WAN link computers to facilitate fast and efficient exchange of information at-

| LAN | WAN |
| --- | --- |
| Limited to a small Geographical location | A WAN is spread over a very large geographical location |
| Rate of data transmission is higher | Rate of data transmission is lower |
| A LAN is established by wired media. | A WAN is always established by wireless media. |
| Cost of communication is less. | Cost of communication is higher. |

## Networking devices

### Hub

A hub is a hardware device used to connect several computers together.

A concentrator is, a device that provides a central connection point for cables from workstations, servers and peripherals.

Hubs can be either passive or active.

Active hubs electrically amplify the signal as it moves from one connected device to another.

Passive hubs allow the signal to pass from one computer to another without any change

## Repeater

A repeater is a device that amplifies a signal being transmitted on the network. It is used on long network lines, which exceed the maximum rated distance for a single run.

Over distance, the cables connecting a network lose the signal transmitted. If the signal degrades too much, it fails to reach the destination Or if it does arrive, the degradation of the message makes it useless.

Repeaters can be installed along the way to ensure that data packets reach their destination.

## Switches

A switch is a device that is used to segment networks into different sub networks called subnets or LAN segments.

Segmenting the network into smaller subnets prevents traffic overloading in a network.

A switch is responsible for filtering i.e transforming data in a specific way and for forwarding packets between LAN segments.

## Bridge

A bridge is a device that lets you link two n/w's together.

Bridges are smart enough to know which computers are on which side of the bridge, so they only allow those messages that need to get to the other side to cross the bridge. This improves performance on both sides of the bridge.

As a packet arrives at the bridge, the bridge examines the physical destination address of the packet. The bridge then decides whether or not to let the packet cross

## Router

A device that works like a bridge, but can handle different protocols, is known as a router.

(For example, a router can link Ethernet to a mainframe.)

If the destination is unknown to a router, it sends the traffic to another router which knows the destination.

A bridge uses physical addresses whereas the router uses logical addresses.

## Gateways

A gateway is a device that connects dissimilar networks.

It expands the functionality of routers by performing data translation and protocol conversion.

A gateway is actually a node on a network that serves as an entrance to another network.

## NICs (Network Interface Card)

It is a hardware card installed in a computer so it can communicate on a network

The n/w adapter provides one or more ports for the n/w cable to connect to, and it transmits and receives data onto the n/w cable

# Internet Services

## Electronic mail

→ Often abbreviated as e-mail.

→ E mail is an electronic way of sending and receiving digital messages.

→ Modern e-mail can consists of text, picture, audio, video and some other files.

→ The e-mail is cheaper and faster.

→ Ray Tomilson is considered as the inventor of e-mail.

## Advantage of e-mail

→ The e-mail is very fast.

→ E-mail does not depend on geographic location of the recipient.

→ Files can be sent as attachment.

→ Easy to use

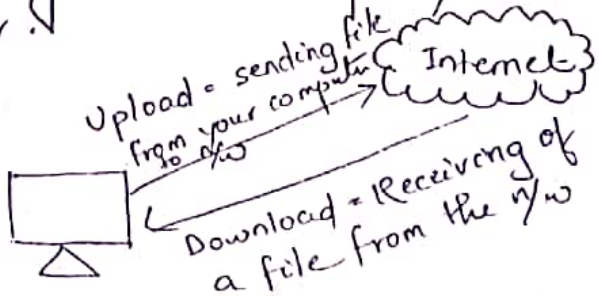
→ Saves a lot of time.

## File Transfer Protocol

∴ Used to transfer files from one computer to another.

File upload is the process of transferring the file from your computer to the other network computer ~~[illegible]~~ or server

File download is the process of transferring file from any network computer/server to your computer.



# World wide web

The world wide web is a set of protocols that allows you to access any document on the net through a naming system based on URLs.

Sir Tim Berners Lee invented the world wide web.

## Chatting

Chatting is the most fantastic thing on the internet. It is like a text phone. In chatting you type a message on your screen, which is immediately received by the recipient, then the recipient can type a message in response to your message, which is received by you instantly.

## Internet Relay Chat (IRC)

IRC is a multiuser, multichannel chatting system. It works on client/server technology.

## Internet (Web) conferencing

Web conferencing is a form of real time communication in which multiple computer users, all connected to the Internet, see the same screen at all times in their web browsers.

Web conferencing allows users to carry on business meetings and seminars, make presentations, conduct demonstrations, provide online education and offer direct customer support.

## Electronic newspaper

It is the online version of a newspaper.

Going online created more opportunities for newspaper such as competing with broadcast journalism in presenting breaking news in a more timely manner.

## Online shopping

Online shopping involves purchasing products or services over the internet.

All the products in online stores are described through text, with photos and multimedia files.

# Different types of Internet Connectivity and ISP

### 1. Dial up connections

→ The most basic type of internet connection is called a dial up connection.

→ This connection is made through a modem that uses a 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.

### Modem - Dial-Up connection

The fastest modem that you can use for this type of Internet connection is called a 56K modem.

When a regular analog telephone line is used for an Internet connection the modem must convert the analog signal that it receives from the telephone line into digital signals that the computer can comprehend. To send information from your computer the modem must take the computer's digital signal and convert them into analog signal to be sent over the telephone line. All of these conversions take time; so this is a relatively slow connection.

# ISDN Dial up connection

ISDN (Integrated Services Digital Network) This connection is a high speed dial up and requires digital telephone line.

Improves speed because signals are in digital form. But it is costly.

## Advantage

Very economic
Widely available.
H/w costs are minimal.

## Disadvantage

Very slow

When connected to the internet, the same phoneline cannot be used for phone calls, so if anyone phones you when you are connected, they will get a busy signal.

Analog to digital & digital to analog conversion adds a performance overhead which affects the speed of the connection.

## ADSL connections

Asymmetric Digital Subscribers Line.

→ Widely available
→ can provide an excellent internet connection.
→ The connection work by splitting your phone line into 2 separate channels, one for data (internet) and one for voice (phonecalls)

Ex:- 256 Kbps / 128 Kbps
downloading speed / Uploading speed

## ADSL connection advantages

→ ADSL technology eliminates the need for a second phone line by allowing voice and data transfer at the same time.

→ ADSL does not need to convert the data from digital to analogue and back again.

→ ADSL connections are Always on, which makes the usual long wait to connect a thing of the past.

ADSL connection disadvantages

→ ADSL connections are not available to everyone.

→ The hardware costs can be quite significant.

→ ADSL connections are Always on, so you will need a firewall to protect your PC.

## Cable connections

Offer very fast and reliable connections with a fixed monthly fee.

Cable companies usually offer different packages to suit different internet subscribers,

The different packages will offer different speed specifications and bandwidth limits

Because a cable connection uses a totally separate medium to transfer data it does not affect your ability to make receive phone calls

### Advantages

Speed is very high.

transfer data digitally, eliminating any digital/analogue conversion overhead.

Cable connections are always on, eliminating long waits to make a connection.

### Disadvantages

Cable connections are not available in every area

Because cable connections are always on,

~~eliminating long waits to~~

you will need a firewall to protect your PC

## ISP:-

(Internet Service Provider)

An ISP is a company that provides individuals and other companies access to the Internet and other related services such as website building.

There are 2 types of ISPs:

local and national level ISP.

A local ISP provides internet service in a particular city or area whereas a national ISP provides services throughout the country.

e.g → Satyam Infoway
MTNL Mahanagar Telephone Nigam Limited
VSNL Videsh Sanchar Nigam Limited

# CHAPTER:-4 FILE MANAGMENT AND DATA PROCESSING

## WHAT IS A FILE ?

- File is an electronic document.
- It contain of the file can be ordinary text and executable
- Each file is given by a file name to identify it.
- The file name is in the form file name extension
- File name consist of alphabets special character etc

  Example : ABC.doc
- Here ABC is a file name. .doc is a extension name is indicate the document file.

## WHAT IS A FOLDER ?

- A folder is the collection of multiple file it is otherwise known as directories

A folder can also store other folder called sub folder. folder helps organising file.

Example: A person can store all photos in a folder name while he can store vedio in another similarly named folder, then he can place all such folder in a folder called my document.

# DIFFERENCE BETWEEN FILE AND FOLDER

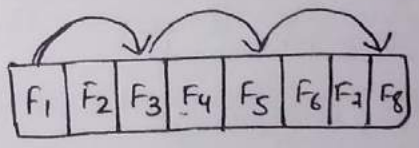
| FILE | FOLDER |
|---|---|
| ① File store data | ① folders store file |
| ② file size ranging from a few bit to 2 killo bit as in world file 2 giga file. | ② folder usually takes no space in the hard drive |
| ③ Storage capacity of data is len compare to file | ③ Storage capacity data is more as compare to file |
| ④ File have some extension name | ④ Folder does'nt have extension name. |
| ⑤ like - Doc, text, etc | |

## FILE ACCESS METHOD

There are three types of file accen method

- Random (Direct) Accen file
- Sequential Accen file
- 9ndex Accen file

## RANDOM ACCESS FILE




- 9n Random Access or direct Accen method of file organisation each record has its own addren on the file
- with the help physical addren the record can be directly accen for reading and writting.
- There are created only on magnetic disk since every record is independently accen and every transaction can be manipulated individually.
- Rased suited for online procesing system.

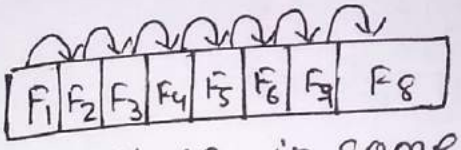
## ADVANTAGES

- Immidiate Access to record is possible
- Uptodate information will be ~~available~~ avaliable on the file
- Addition and deletion is not very complex.

## DISADVANTAGES

- Less effiient in use of storage space.
- Data ~~see~~ security is less due to direct Access facility.

## SEQUENTIAL ACCESS FILE




- It is a file in which records are store in some order
- There are ~~prefor~~ preferable when they are store in sequential access devices such as magnetic tapes.

## ADVANTAGES

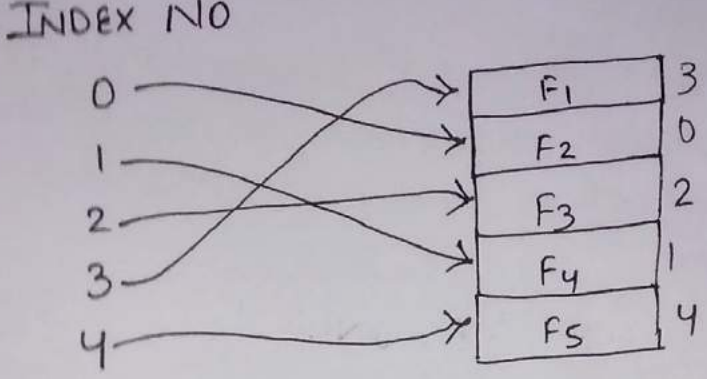
- File design is simple
- Location or Record required only the record key.
- Magnetic tapes are used for storing data.

## DISADVANTAGES

- Addition and deletion of data is not simple
- Updating requires all transaction ~~file~~ records are shorted in record key sequence.

# INDEX SEQUENTIAL FILE

- It combine the advantages of sequential and direct Access file organization.
- It is basically organize serially on key fields.
- An ed addition or and inden is maintain with split of Access of isolated records.




- It is also known as inden sequential method (ISAM)
- Here within each block the record is searched sequentially

## ADVANTAGES

- Suitable for both seauencial and online or direct access procoming.

## DISADVANTAGES

- Less efficient in use of storage space
- Addition and deletion of records are more complex.

# DATA CAPTURE

- It is the process of identification and extraction of data from a scan document.
- methods of data capture from documents in electronic format are as follows:-

## (a) SINGLE CLICK

It is an OCR (optical character Recognition) tool used to capture machine produced characters in low volume adhoc. Capture application and populating a line ~~cap~~ of business application

## OCR ( OPTICAL CHARACTER RECOGNITION)

It has the ability to succesfully capture machine produce character in full page. It is used to capture low to high volume of data where the information is inconsistant location on the document.

## ICR (INTELLIGENT CHARACTER RECOGNITION)

- IE is a computer translation of hand printed and written character.
- Data is entered from hand printed forms through a scanner & the image of capture data is a len and translated by sofisticated ICR software.

## BARCODE RECOGNITION

It is independent upon the type of bar code that is used amount of data can included is high as

# IDR (INTELLIGENT DOCUMENT RECOGNITION)

- Here the level of capability is depending upon the indivisual product.
- There application are used to capture meta data from documents i.e. Rules based.
- Example :- Product will identify post codes, logo's, key board, vat registration Number, on going running, capture information from multiple document file.

## DATA STORAGE

Data storage is the holding of data in an electro magnetic form for access by a computer procesor.

- Data Storage is two types
  - → Primary storage
  - → Secondary storage

## PRIMARY STORAGE

- Data is hold in RAM (Random Accen memory) and other memory devices that are built into computer.

## SECONDARY DSTORAGE

- It is the data that is stored on external storage devices such as Hard disk, CD's, tapes, following are stem of the devices for data storage.

# Ch-5: Problem Solving Methodology

A the programmer has to go through various steps while writing a program, as mentioned below:

⇒ The programmer gets the problem and understands it.

⇒ Then he frames a logic for solving the problem and represents it in the from of a flowchart or algorithm.

⇒ Then he chooses a particular programming language to encode the algorithm and convert it to a computer program.

⇒ Then he loads the program in the computer, compiles it by using the language compiler and executes the program.

⇒ After the program is executed, we get the result of the problem.

## *Algorithm:

⇒ An algorithm can be defined as a step by step method for writing the various steps of the solution to a problem.

⇒ Before writing any program it is always advisable to have the algorithm for the problem.

### Algorithm characteristics:

⇒ Algorithm should be definite

⇒ Algorithm should have definite number of steps.

⇒ Algorithm should mention the input required for the program clearly

⇒ Algorithm should give an idea the output that will obtained.

⇒ Example:

To print all the 2-digit odd numbers.

step 1: Initialize a variable NUM with 11.
step 2: Print this variable NUM.
step 3: Add 2 to the variable NUM.
step 4: Go on repeating step 2 and step 3 until NUM becomes more than 99.

Algorithm Types:

⇒ In an algorithm we define the logic of a problem solution, normally there are three types of logic i.e.,

- sequence logic
- selection logic
- iteration logic.

⇒ The sequence logic is the simplest one where the problem solution is achieved by executing the steps in a linear sequence one after another from top to bottom.

⇒ In the selection logic, the flow of the problem logic is not linear, rather it different paths basic on the result of certain conditions.

⇒ Similarly in iteration logic, a number of steps are repeated certain number of times depending on the iteration counter. This is also known as looping logic.

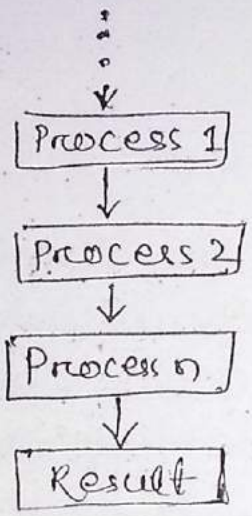
Pseudocode:

When we write the logic of a problem solution in a step manner in English and follow certain programming construction, then it is known as Pseudocode.

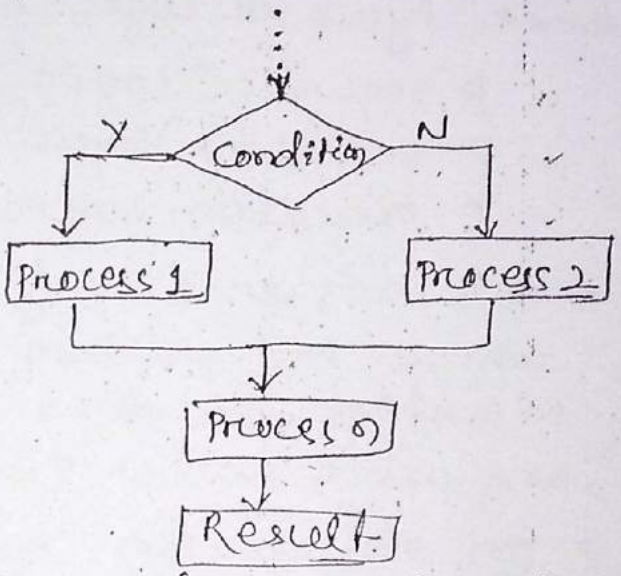
⇒ Pseudocode is a set of codes which may not be written by following the correct syntax of the code..

⇒ Like algorithm, a pseudocode can be written in a 3 basic ways i.e.
- Sequence logic
- Selection logic and
- Iteration logic

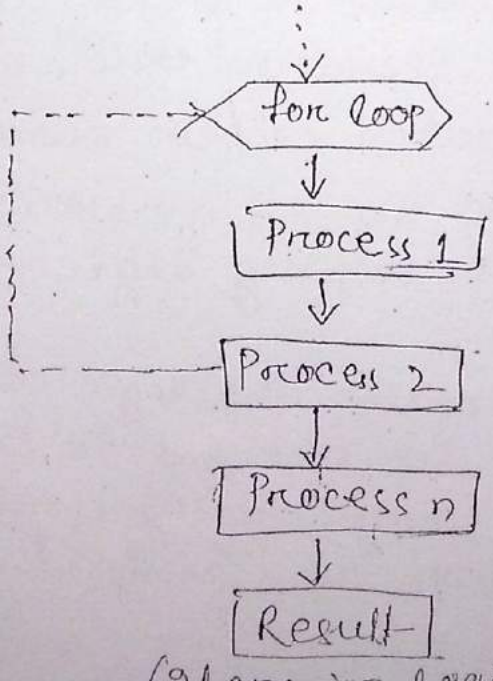
⇒ A pseudocode is also known as program design language (PDL) as it emphasises



(Sequence logic)



(Selection logic)



(Iteration logic)

Pseud

Ex: To print all 2 digit odd numbers.

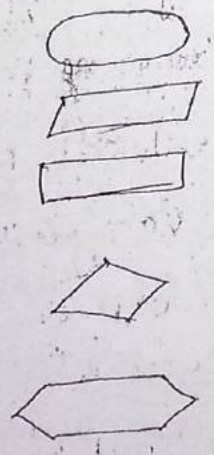
1. Begin
2. set NUM = 11
3. print NUM
4. set NUM = NUM + 2
5. if NUM <= 99 then repeat step 3 & 4.
6. End.

## Flowchart:

⇒ A flowchart is another program planning tool where we represent the logic of a problem in a pictorial manner by using a set of predefined symbols.

⇒ These symbols are joined by solid lines with arrows and contain the various ~~tasks~~ tasks to be performed at different steps.

⇒ While drawing a flowchart, we uses different types of symbols to contain different types of statements of the problem solving logic. They are shown below:



⇒ contain START or STOP

⇒ contains input statement

⇒ contains any calculation or processing statement

⇒ Contains condition for selection logic.

⇒ contain number of iterations in the for of a starting and ending condition

⇒ continuation symbol

⇒ shows the direction of the flow of logic and connect different symbols of the flowchart

Advantages:

⇒ Program logic represented in a graphical manner is easy to create.
⇒ The logic in a flowchart is easy to interprete.
⇒ It can be used as a program planning document even by non computer professionals.
⇒ It is easy to modify the logic of a problem solution if it is in the form of a flowchart.
⇒ Long and complicated problem solution can be represented by small and simple flowcharts.

Disadvantages:

⇒ The problem solution represented in a flowchart is difficult to convert into a program.
⇒ Sometimes it become difficult to represent problem solution in flowchart if it contains certain specific type of structure.

## * Generation of Programming language:

### First Generation of Programming language:

⇒ The first generation of programming language is machine language.
⇒ Machine language is set of instructions and data that a computer's central processing unit can execute directly.
⇒ Machine language statements are written in binary code.

## Second Generation of Programming Language:

⇒ The second generation of programming language is assembly language.

⇒ Assembly language is the human-readable notation for the machine language used to control specific computer operations.

⇒ An assembly language programmer writes instructions using symbolic instruction codes that are meaningful abbreviations or mnemonics.

⇒ An assembler is a program that translates assembly language into machine language.

## Third generation of Programming Language:

⇒ The third generation of programming language or procedural language uses a series of English-like words, that are closer to human language, to write instructions.

⇒ High-level programming languages make complex programming simpler and easier to read, write and maintain.

⇒ Programs written in a high-level programming language by a compiler or interpreter.

⇒ Ex: C, C++.

## Fourth Generation of Programming language:

⇒ The fourth generation of programming language or non procedural language enables users to access data in a database.

⇒ A very high-level programming language is often referred to as goal oriented programming language because it is usually limited to a very specific application and it might use syntax that is never used in other programming languages.

⇒ Ex: SQL

## Fifth Generation of Programming Language:

⇒ The fifth generation programming language or visual programming language, is also known as natural language.

⇒ Provides a visual or graphical interface, called a visual programming environment, for creating source codes.

⇒ It allows people to interact with computers without needing any specialized knowledge.

⇒ People can talk to computers and the voice recognition systems can convert spoken sounds into written words.

⇒ Ex: Prolog.

## * Structured Programming Language:

⇒ Structured programming is a subset of procedural programming that enforces a logical structure on the program being written to make it more efficient and easier to understand and modify.

⇒ Certain languages such as Ada, Pascal, and dBASE are designed with features that encourage or enforce a logical program structure.

⇒ Structured programming frequently employs a top-down design model, in which developers map out the overall program structure into separate ~~module or submodule~~ subsections.

⇒ Program flow follows a simple hierarchical model that employs looping constructs such as for, while.

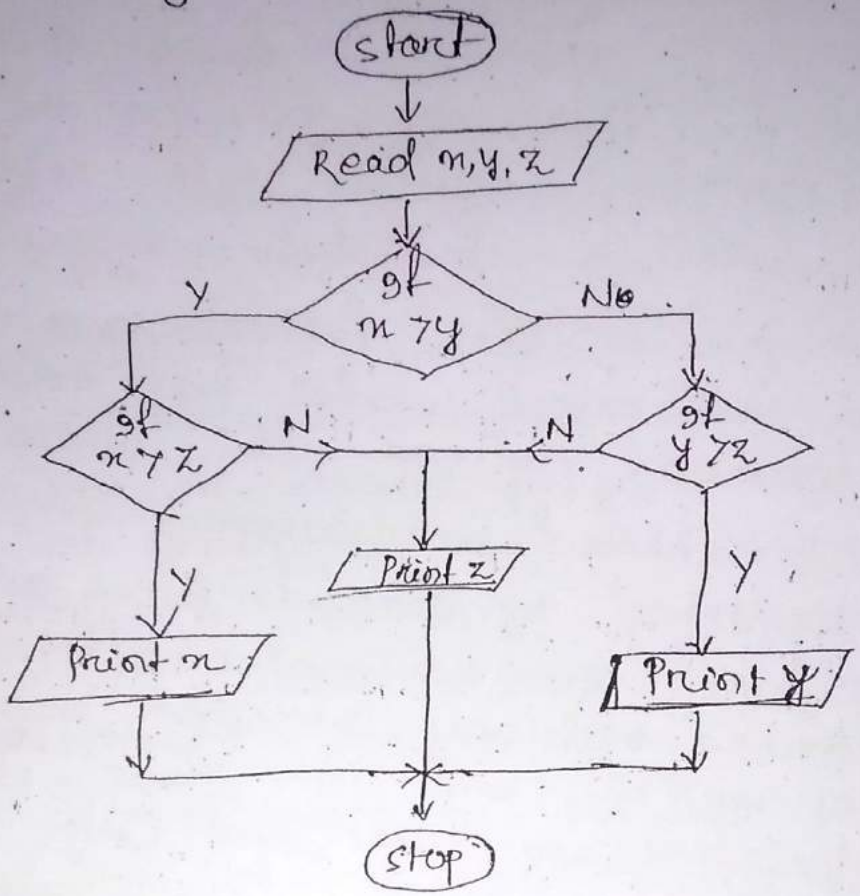
⇒ Structured programming was first suggested by Corrado Bohm and Guiseppe Jacopini. The two mathematicians demonstrated that any computer program can be written with just three structures: decisions, sequences, and loops.

⇒ Most modern procedural languages include features that encourage structured programming.

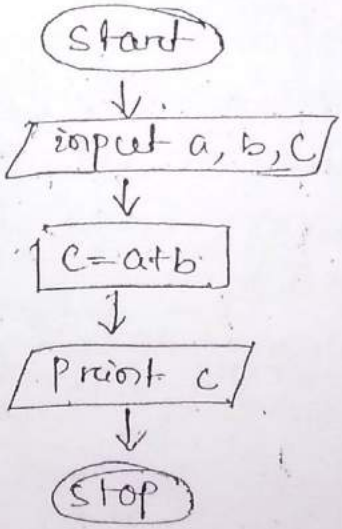
⇒ Object-oriented programming (OOP) can be thought of as a type of structured programming, uses structured programming techniques for program flow, and adds more structure for data to the model.

## * Examples of problem solving through Flowchart :

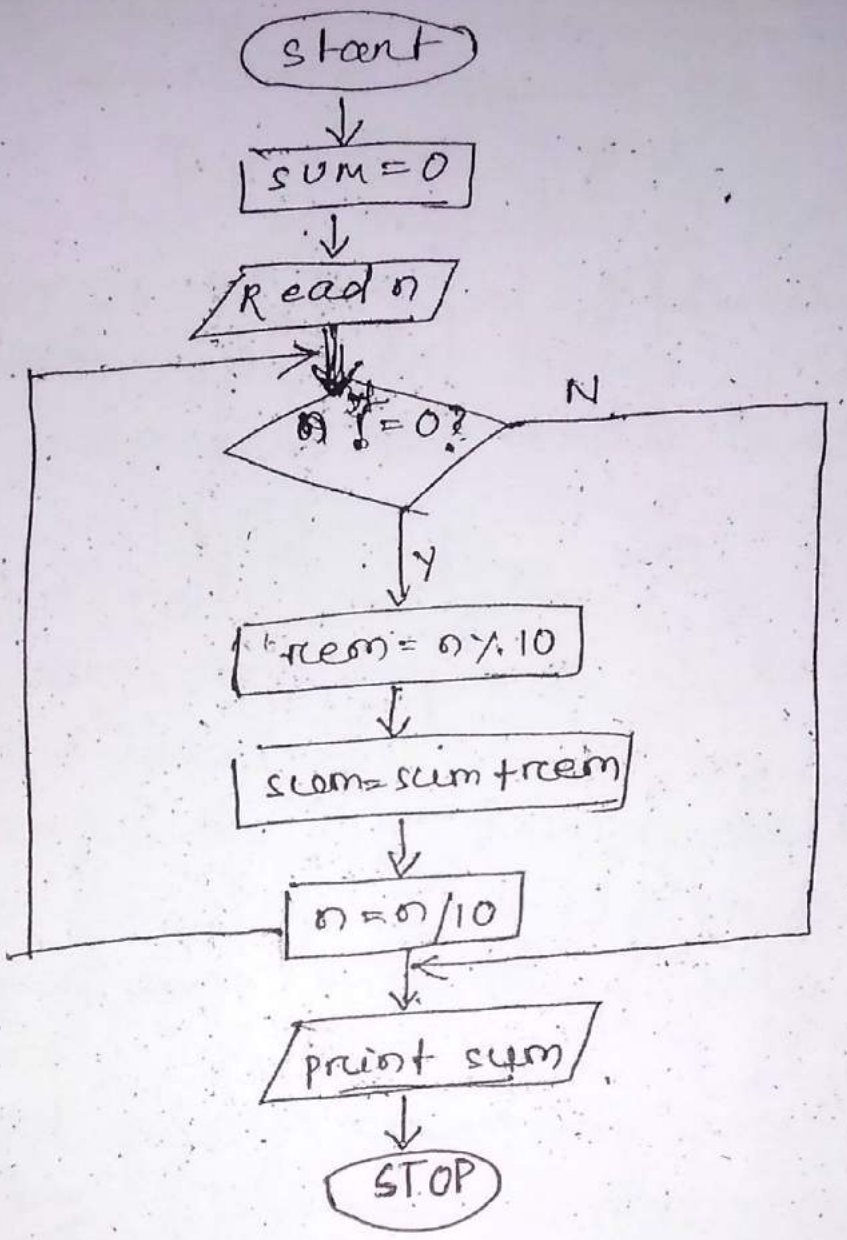
**Ex-1:** Draw a flowchart to find the largest among three numbers x, y, z.




**Ex-2:** Draw a flowchart to add of 2 integers.



Ex-3: Addition of all digits of a given num.



# Structured Programming Language

It is sometimes known as modular programming.

→ It is a subset of procedural programming that enforces a logical structure on the program, written to make it more efficient, easier to understand and modify

→ ADA, Pascal, database are designed to encourage or enforce a logical program structure

→ It employs a top down design model in which developers map out the overall program structure into separate subsections.

→ Structured programming was first suggested by Corrado Bohm and Guiseppe Jacopini

→ The two mathematicians demonstrated that any computer program can be written using three structures such as decision, sequence and loops.

→ The most common methodology employed was developed by Dijkstra In this model the developer separates programs into subsections that each have only one point of access and one point of exist.

→ OOPs can be thought of as a type of structured programming, uses Structured programming language for program flow and adds more structure for data to the model.

Q Write an Algorithm to check whether number is even or odd.

Step 1 : Read the value of x
Step 2 : If (x % 2 = 0) then
Step 3 : Print number is even
Step 4 : else
Step 5 : Print number is odd.

2. Write an algorithm of swapping two numbers.

Step 1 : Start
Step 2 : Read a and b value
Step 3 : Interchange the values
temp = a
a = b
b = temp
Step 4 : Write a and b value
Step 5 : Stop

3. Write an algorithm to find the largest among three numbers x, y, z.

Step 1 : Read three numbers x, y, z
Step 2 : Compare x & y
Step 3 : If x is larger, compare it with z
Step 4 : If x is larger than z then x is the largest otherwise z is the largest
Step 5 : If x is smaller than or equal to y in the first step then y is compared with z
Step 6 : If y is larger than z then y is the largest number otherwise z is largest.
Step 7 : Stop

Q write an algorithm to find out the area of triangle where three sides are a, b, c.

step 1: Read the value of a, b, c

step 2: $s = \frac{a+b+c}{2}$

step 3: area $= \sqrt{s(s-a)(s-b)(s-c)}$

step 4: Display the area of the triangle

step 5: stop

Write an algorithm to find the roots of Quadratic equation: $ax+bx+c=0$

step 1: Read the value of a, b, c

step 2: $d = b*b - 4*a*c$

step 3: If $d<0$ then display the roots are imaginary else if $d=0$ then display roots are equal

$r_0 = -b/2 * a$, display r

else $r_1 = -b + \frac{\sqrt{d}}{2} \times a$

$r_2 = -b - \frac{\sqrt{d}}{2} \times a$

Display roots are real and $r_1$ & $r_2$

step 4: Stop

6. Write an algorithm to find factorial of an Integear

step 1: Read a value of n

step 2: set variable fact as 1

step 3: Fact <= Fact * n

decrease n

Step 4 : Check if n is equal to zero
If n is equal to zero go to step 5
else go to step 3
Step 5: Print the result fact.

7. Write an algorithm to convert temperature to Fahrenheit.

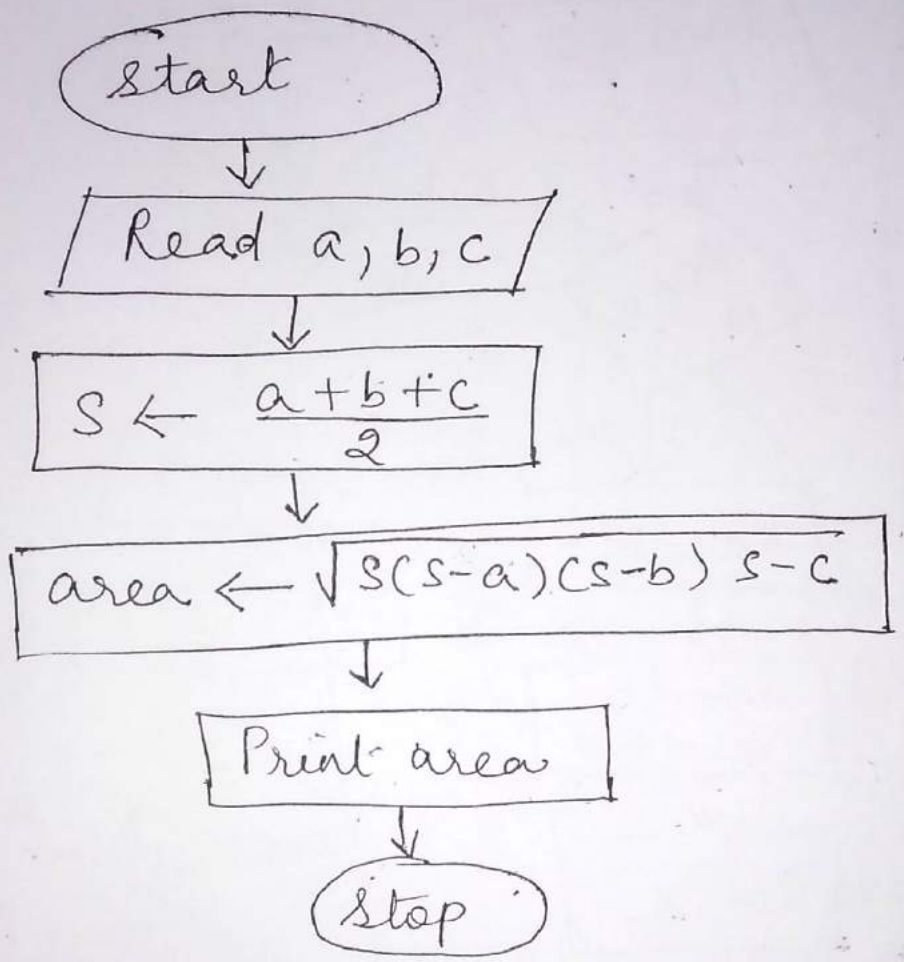
Step 1 : Start
Step 2 : Read the temperature in Centigrade
Step 3: Store the value in C
Step 4: Set F to $32 + \frac{9}{5} * C$
Step 5: Print value C & F
Step 6 : Stop.

Sequence Logic:

* Create a flowchart to calculate the area of a triangle whose sides are a, b, c

Soln. Area of a triangle

$$\sqrt{s(s-a)(s-b)(s-c)}$$

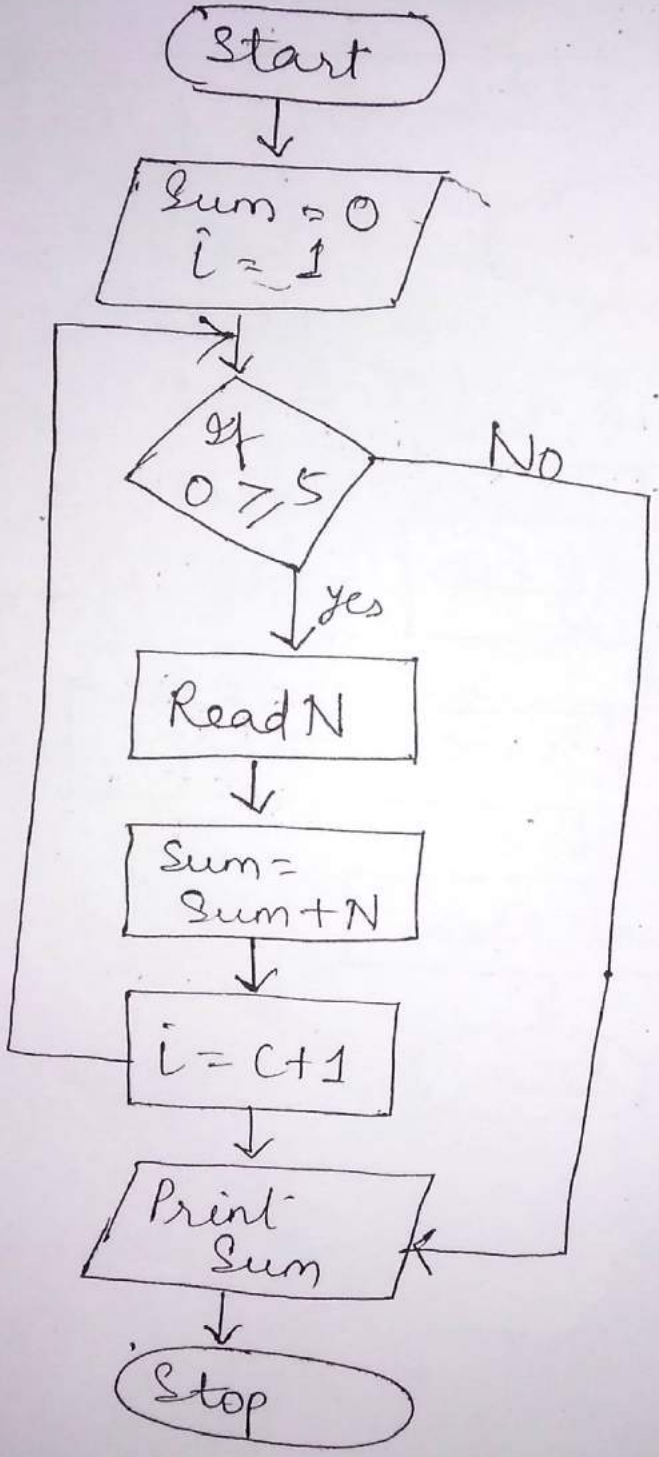


1. ... logic

* Draw a flowchart of five random numbers.

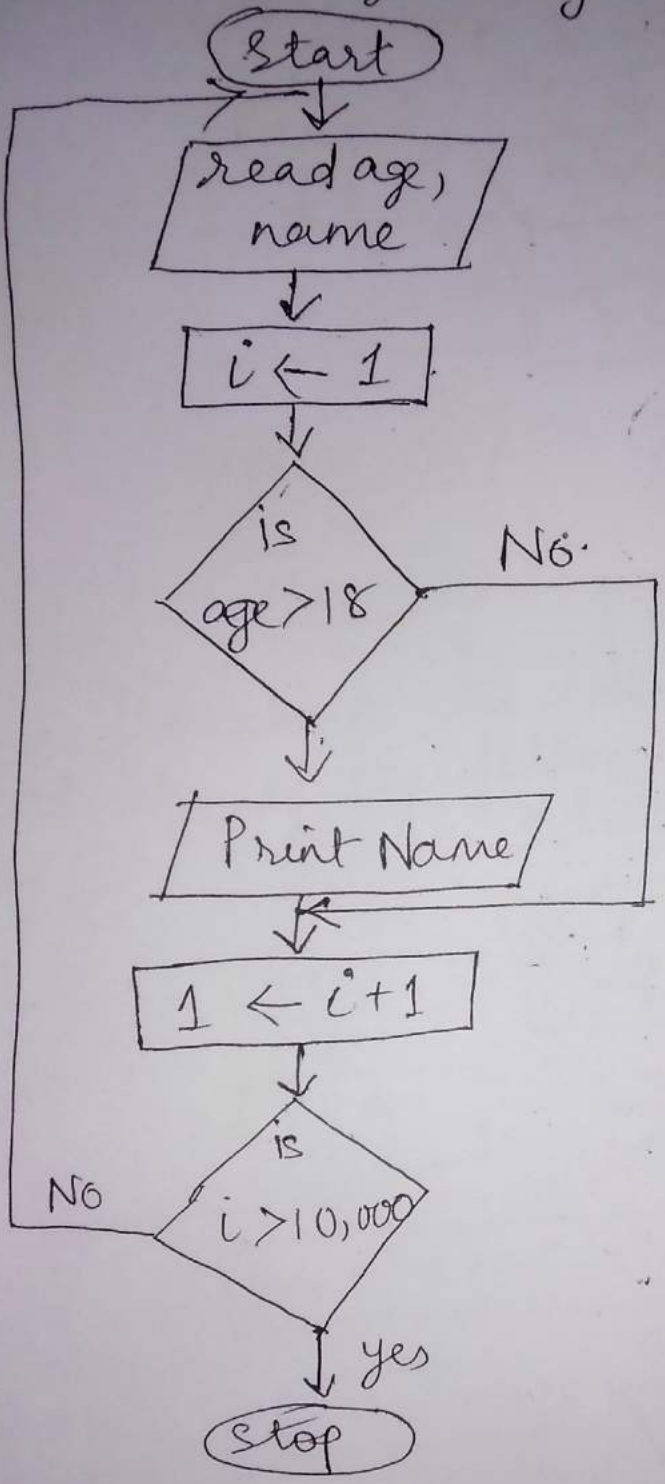
Soln Let numbers be 1, 2, 3, 4, 5.

Sum = 1 + 2 + 3 + 4 + 5 = 15

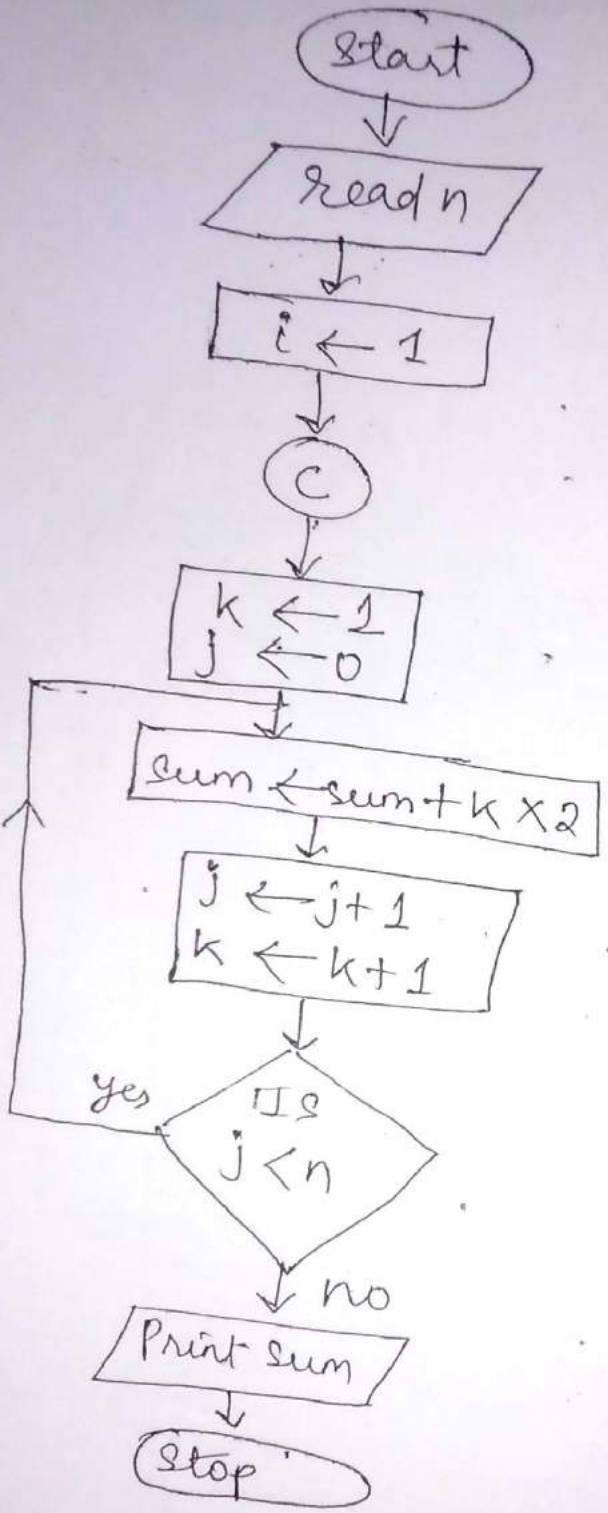


(b)

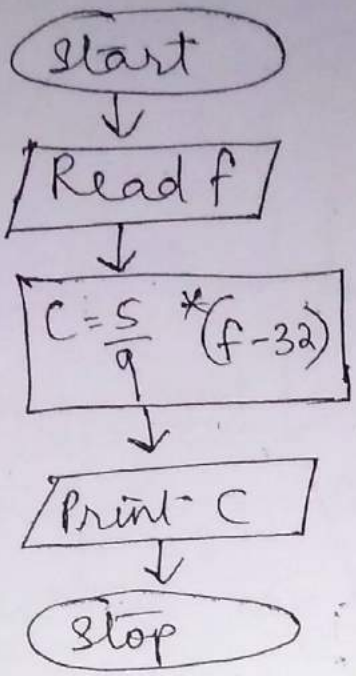
s a flow chart which prents the names
tizen eligible for voting in a city of 10,000
pulation, and the eligibility of the person should
be more than 18 years of age



11 Draw a flowchart to find the sum of square of first n integers

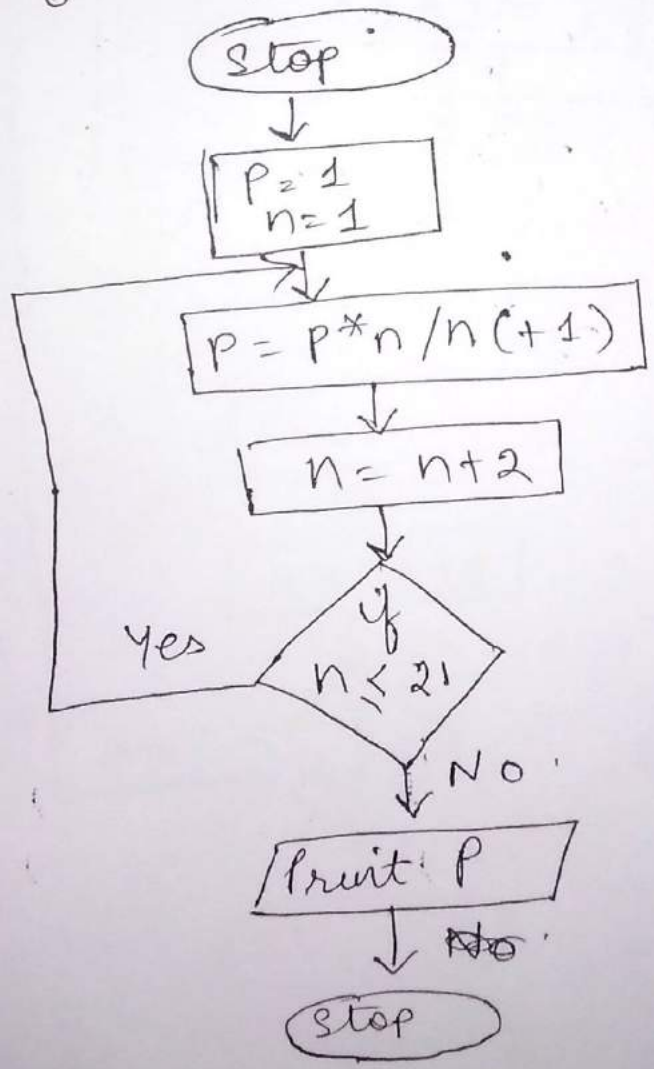


5 a flow chart to convert temperature °f to °C

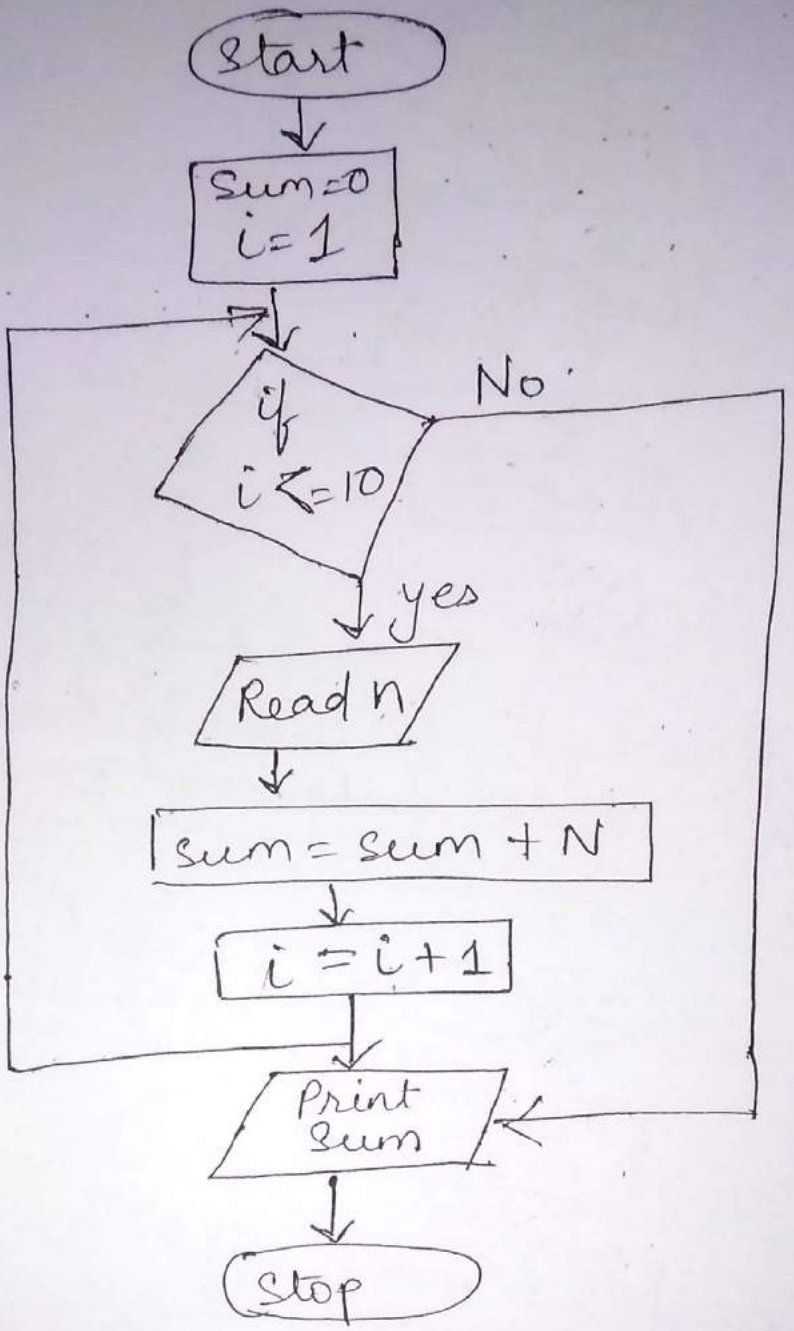



Draw a flowchart to calculate and print the product of the following series

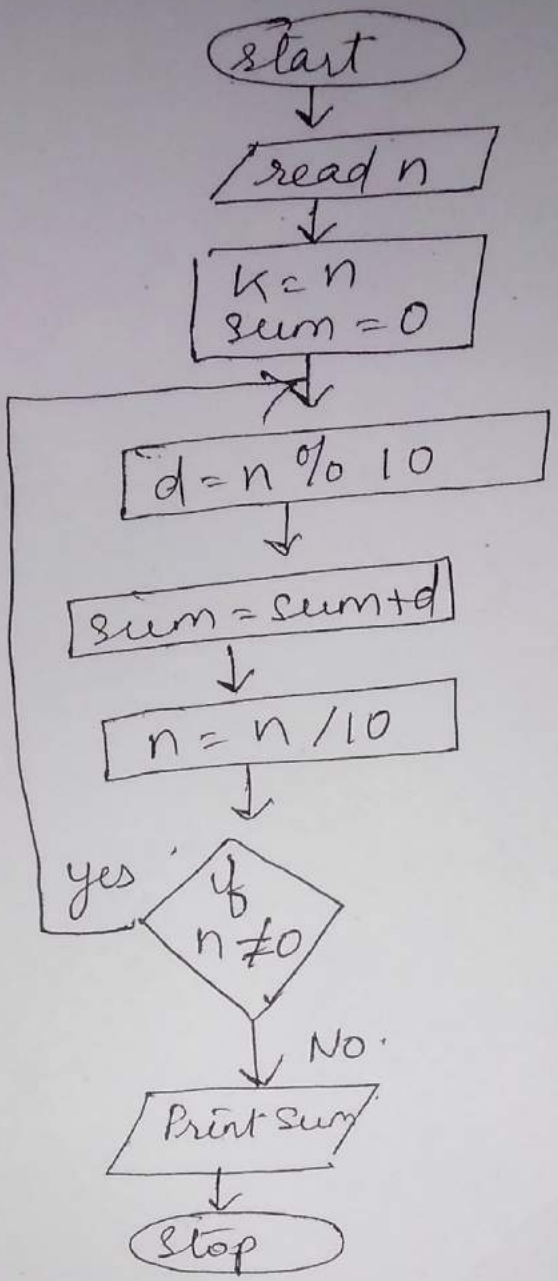
$\frac{1}{2}, \frac{3}{4}, \frac{5}{6} \cdots \frac{21}{22}$



Draw a flow chart to find sum of 10 random numbers



even Draw a flowchart to calculate the sum digits of a given number.



# OVERVIEW OF C PROGRAMMING LANGUAGE

CHAPTER-6



## Introduction to C :-

'C' is one of the most widely used procedural language which has been closely associated with the UNIX Operating System for which it was developed. It was originally developed by Dennis Ritchie at Bell laboratory.
It is easy to use and Implement.
It is also fully based on coding.
It is written in high level language and also use procidural language.

## Types of C :-

### (1) Common C

Until recently there was one dominant form of the C language. This was the native UNIX form, which for historical reasons is known as either Bell labs C, after the most popular compiler, or K. & R. C, after the authors of the most popular textbook on the language. It is now often called "Classic C".

### (2) ANSI-C

The American National Standards Institute defined a Standard for C, eliminating much uncertainty about the exact Syntax of the language. This newcomer, called ANSI C, proclaims itself the standard version of the language. As such it will inevitably overtake, and eventually replace Common C.
ANSI C does incorporate a few improvements over the old Common C. The main difference is in the grammer of the language.

(3) C 90

In 1990, the ANSI C standard was adopted by the International Organization for Standardization as ISO/IEC 9899:1990. This version is sometimes called C 90. Therefore, the terms "C89" and "C90" refers to essentially the same language.

(4) C 99

In March 2000, ANSI adopted the ISO/IEC 9899:1999 standard. This standard is commonly referred to as C99, and it is the current standard for C programming language.

## C Program Structure :-

A C program basically has the following form :

(a) preprocessor commands.
(b) type definitions
(c) Function prototypes
(d) Variables
(e) Functions

## Pre-Processor

It directives are instructions for the compiler. pre-processor directives are prefixed by the '#' character. In C, two different pre-processor commands are needed.

(1) # include <filename.h> : The include directive is used to link C source files, object files, and library files together. For this discussion we will use this to link the library files. This directive must appear before global variable

declarations and function declarations.

(2) # define NAME VALUE : The define directive is used to set definitions.

Example : # define PI 3.14.

## Character Set

The character set in C language can be grouped into the following catagories.

1. Letters (Alphabet from A to Z and a to Z.)
2. Digits (Numbers, from 0 to 9)
3. Special characters
4. White Spaces.

White Spaces are ignored by the compiler until they are a part of string constant. White Space may be used to separate words, but are strictly prohibited while using between characters of key to keywords or identifiers.

Character - Set Table

| Letters | Digits |
|---|---|
| upper case A to Z | 0 to 9 |
| lower case a to z | |

Special characters

| | | |
|---|---|---|
| Comma - , | Slash - / | opening angle - < |
| period - . | Backslash - \ | closing angle - > |
| Semicolon - ; | percentage sign - % | left parenthesis - ( |
| Colon - : | Ampersand - & | Right Parantheses - ) |
| Question mark - ? | Caret - ^ | left Bracket - [ |
| Apostrophe - ' | Asterisk - * | Right Bracket - ] |
| Quotion mark - " | Minus sign - - | left Brace - { |
| Exclamation mark - ! | plus sign - + | Right Brace - } |
| | | Number sign - # |

White Space
1) Blank space
2) Horizontal Tab
3) Carriage Return
4) Newline
5) Form Feed.

C tokens
The tokens of a language are basic building blocks which can be put together to construct programs.

## CONSTANTS, VARIABLES AND DATA TYPES IN C

### CONSTANTS

A constant can be defined as "a quantity that does not change during the execution of a program".
C supports several types of constants.
1) Integer constants
2) Real constants
3) Single character constants
4) String constants

### INTEGER CONSTANTS

An Integer constants must have at least one digit and should not have decimal point. It could either be positive or negative. There are 3 types of integers namely decimal integer, Octal integers and hexadecimal integer.

### REAL CONSTANTS

A real constant must have at least one digit and must have a decimal point. Real constants consists of a fractional part in their representation.

Real numbers can also be represented by exponential notation.

## SINGLE CHARACTER CONSTANTS

A single character Constants represent a single character which is enclosed in a pair of quotation symbols.

## STRING CONSTANTS

A string constants is a set of characters enclosed in double quotation marks.

## BACKSLASH CHARACTER CONSTANTS

Backslash character constants are special characters used in output functions.

| Constant | Meaning |
|---|---|
| '\a' | Audible Alert |
| '\f' | Form feed |
| '\r' | Carriage return |
| '\v' | Vertical Tab |
| '\"' | Double Quote |
| '\\' | Back slash |
| '\b' | Back Space |
| '\n' | New line |
| '\t' | Horizontal tab |
| '\'' | single Quote |
| '\?' | Question mark |
| '\0' | Null. |

# VARIABLES

A Variables is names give to the space in the memory for holding data such as Integers, Characters, floating point numbers, Strings, etc.
A Variable is a Value that can change anytime.

## Declaration of Variables

~~Cost Datato~~

| Constant | Datatype | Variable name | = | Value |
|---|---|---|---|---|
| ↓ | ↓ | ↓ | | ↓ |
| Constant | Float | Pi | | 3.14 |

## DATA TYPES IN C

When programming, we store the Variable in our Computer's memory, but the Computer has to know what kind of data we want to store in them, Since it is not going to occupy the same amount of memory to store a simple number than to store a single letter or a large number, and they are not going to be interpreted the same way.

The memory in our Computers is organized in bytes. A byte is the minium amount of memory. A byte can store a relatively small amount of data: One single character or a small integer. In addition, the Computer can manipulate more complex data types that come from grouping several bytes, such as long numbers or non-integer numbers.

PRIMARY DATA TYPE :-

| NAME | DESCRIPTION | Size * | RANGE * |
|---|---|---|---|
| ① Char | use for alphabet only. | 1 byte | Signed = -128 to 127<br>Consigned = 0 to 255 |
| ② Short (int) | use for Numeric only. | 2 byte | Signed = -32768 to 32767 |
| ③ Int. | use for Numeric only. | 4 byte | Signed = -2147483648 to +2147483647<br>Consigned = 0 to 4294967295 |
| ④ Long int. | use for numeric only. | 4 byte | signed = -2147483648 to +2147483647<br>Consigned = 0 to 4294967295 |
| ⑤ Float | decimal point | 4 byte | + or - 3.4e to + or - 38 (7 digit). |
| ⑥ Double | decimal Point | 8 byte | + or - 1.7e to + or - 308 (15 digit). |
| ⑦ Long double | decimal Point | 8 byte | + or - 1.7e to + or - 308 (15 digit). |

Secondary data type / user defined data type

Syntax

type def. data type Identifier

Eg
① type def. Int. Salary
② type def. & Float. average
③ type def. Char. adal.

PRIMARY DATA TYPE :-

| NAME | DESCRIPTION | Size* | RANGE* |
|---|---|---|---|
| ① char | use for alphabet only. | 1 byte | Signed = -128 to 127<br>Consigned = 0 to 255 |
| ② Short (int) | use for Numeric only. | 2 byte | Signed = -32768 to 32767 |
| ③ Int. | use for Numeric only. | 4 byte | Signed = -2147483647 to +2147483647<br>Consigned = 0 to 4294967295 |
| ④ Long int. | use for numeric only. | 4 byte | signed = -2147483648 to +2147483647<br>Consigned = 0 to 4294967295 |
| ⑤ Float | decimal point | 4 byte | + or - 3.4e to + or - 3% (7 digit). |
| ⑥ Double. | decimal Point | 8 byte | + or - 1.7e to for -308 (15 digit). |
| ⑦ Long double | decimal Point | 8 byte | + or - 1.7e to or -308 (15 digit). |

Secondary data type / user defined data type

Syntax

type def. datatype identifier

Eg
① type def. Int. Salary
② type def. Float. average
③ type def. char. adal.

# Managing Input - Output Operations

In any programming language, the interface forms a very important part. It deals with taking data from the user and displaying back output. For this purpose, we require the Input Output operations.

Reading, processing and writing of data are the three essential functions of a Complex program. Most programs takes some data as input and display the processed data, often known as information or result, on a suitable medium.

Each program that uses a standard Input/output function must contain the statement

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

at beginning.

## Single Character Input Output :

The simplest of all Input/output operations is reading a character from the standard input unit and writting it to the standard output unit. Reading a single character can be done by using the function getchar. The getchar takes the following form

```
Variable name = getchar();
```

Variable name is a valid C name that has been declared as char type.

# OPERATORS & EXPRESSIONS, TYPE CONVERSION AND TYPECASTING



An operators is a symbol which helps the user to command the computer to do a certain mathematic or logical manipulations. Operators are used in C language program to operate on data and variables. C has a rich set of operators which can be classified as

1. Arithmetic Operators
2. Relational operators
3. logical operators
4. Assignment Operators
5. Increments and Decrements operators
6. Conditional operators
7. Bitwise operators
8. Special operators.

Operands are variable or expressions which are used in operators to evaluate the expression.

An expression is a combination of operands and operators. For instance. a = b + c ; denote an expression in which there are 3 operands namely a, b, c and one operator namely +.

The association of expressions and keywords is called statements. For instance int a = b + c ; denote a statement.

## 1. Arithmetic Operators

Arithmetic operaters are used for arithmetic operation like addition, substraction, multiplication, Division etc.

| Operator | Meaning |
|---|---|
| + | Addition |
| - | Subtraction |
| * | Multiplication |
| / | division |
| % | Modulus operator |

2. Relational operators

Often it is required to compare the relationship between operands and bring out a decision and program accordingly. This is when the relational operator come into picture. C supports the following relational operators.

| Operator | Description | Example |
|---|---|---|
| > | greater than | 5 > 4 |
| >= | greater than or equal to | mark >= Score |
| < | less than | height < 75 |
| <= | less than or equal to | height <= input |
| == | equal to | Score == mark |
| != | not equal to | 5 != 4 |

3. Logical operators

The term logical refers to the ways these relationships can be connected together using the rules of formal logic. C has the following logical operators, they compare or evaluate logic and relational expressions.

| Operator | meaning |
|---|---|
| && | Logical AND |
| \|\| | logical OR |
| ! | logical NOT |

## Assignment Operators



The Assignment operators evaluates an expression on the right of the expression and substitutes it to the value or variable on the left of the expression.

| Statement with simple assignment operator | Statement with shorthand operator |
|---|---|
| a = a+1 | a+=1 |
| a = a-1 | a-=1 |
| a = a*1 | a*=1 |
| a = a/1 | a./=1 |
| a = a %1 | a%=1 |
| a = a % (n+5+6/2) | a %= n+5+6/2 |
| a = a* (n+1)(n+2) | a *= (n+1)(n+2) |

5. Increment and Decrement Operators

The increment and decrement operators are one of the unary operators which are very useful in C language.

| Syntax for Increment operator | Syntax for Increment operator |
|---|---|
| Variable name ++ or ++ Variable name | Variable name -- or -- Variable name |
| Example a ++ or ++a | Example a-- or --a |

6. Conditional or Ternary operator
The conditional operator consists of 2 symbols the question mark (?) and the colon (:).

Syntax
exp1 ? exp2 : exp3

x = (a>b) ? a : b

7. Bitwise operators
C has a distinction of supporting special operator known as bitwise operators for manipulation data at bit level.

| operator | meaning |
|---|---|
| & | Bitwise AND |
| \| | Bitwise OR |
| ^ | Bitwise exclusive |
| << | Shift left |
| >> | Shift to right. |

8. Special operators
C supports some special operators of interest such as comma operator, size of operator, pointer operator (& and *) and member selection operators (. and ->).

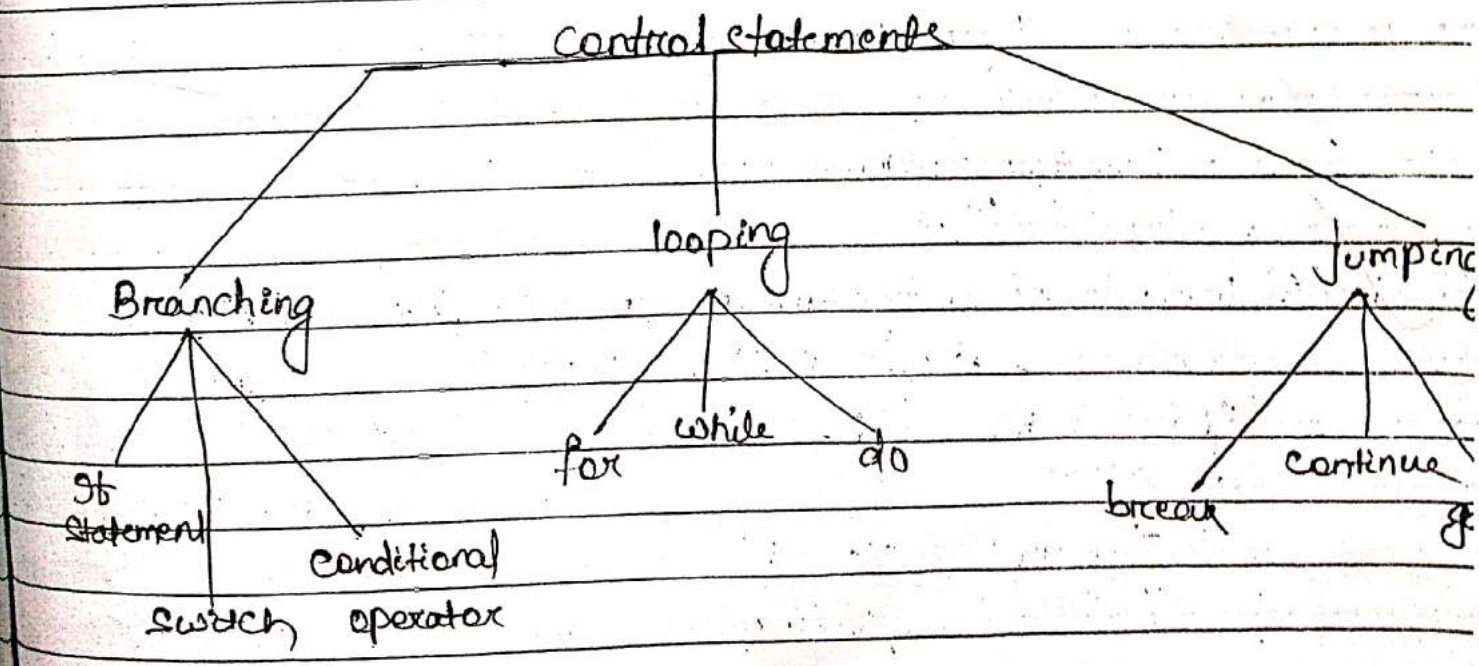
## EXPRESSIONS

An expression is a combination of variable constants and operators written according to the syntax of C language.

| Algebraic Expression | C Expression |
|---|---|
| $a \times b - c$ | a * b - c |
| $(m+n)(x+y)$ | (m+n) * (x+y) |
| $(ab/c)$ | a * b / c |
| $3x^2 + 2x + 1$ | 3 * x * x + 2 * x + 1 |
| $(x/y) + c$ | x / y + c |

## Decision Control and looping statements

The control statements are used to control the cursors in a program according to the condition or according to the requirement in a loop. Further we can say, changing the order or flow controls; these are required. There are mainly three types of control statement or flow controls. These are mainly three types of control statement or flow controls. These are illustrated below:



1. Branching Statement :

C supports many branching statement depending upon their flow of control and according their decision making policy. So it is also called decision making statements. The various branching statements used are as.

(a) If statement
(b) Switch statement
(c) Conditional operator statement.

(a) If statement :

There are mainly four types of if statements used in the C programming as:

(i) Simple if statement
(ii) if - else statement
(iii) nested if statement
(iv) else - if ~~or~~ ladder if or multi-condition if statement.

(i) Simple if statement

Syntax

```
if (condition)
{
true statement
}
```

(P-1) Write a C program to your computer mark if your mark is greater than >= 90 they display you are the best.

```
#include <stdio.h>
#include <conio.h>
main()
{
int mark;
printf("\n enter your mark\n");
```

```
scanf ("%d", &mark);
if (mark > 90)
  {
printf ("you are the best \n");
  }
getch ();
  }
```

(ii) If else statement

Syntax

```
if (condition)
  {
true statement;
  }
else
  {
false statement;
  }
```

4(P-2) Write a C program to enter the mark if the mark is greater than 90°. then display you are the best other wise you are the worst.

```
# include <stdio.h>
# include <conio.h>
main()
  {
int mark;
Printf ("enter your mark \n");
scanf ("%d", &mark);
if (mark > 90)
  {
print ("your are the best")
  }
```

```
else
{
printf("you are the worst \n");
}
getch();
}
```

Output

entre your mark
85
you are the worst.

(P-3) Write a program in C, find the greatest between the two number using if else condition.

Syntax

```
if (condition)
{
true statement
}
#include<stdio.h>
#include<conio.h>
main()
{
int a,b;
printf("\n entre the two no.\n");
scanf("%d %d",&a,&b);
if(a>b)
{
printf("a is greater \n");
}
else
{
printf("b is greater \n");
}
getch();
}
```

(P-4) Write a programming in C to check, int number is even or odd.

```
#include <stdio.h>
#include <conio.h>
main()
{
int number;
printf("enter any no.\n");
scanf("%d", &number);
if(number % 2 == 0);
{
print("even number\n");
}
else
{
printf("odd number\n");
}
getch();
}
```

(P-5) write a program C to findout greatest number among 3 number.

```
#include <stdio.h>
#include <conio.h>
main()
{
int a,b,c
printf("enter any three number\n");
scanf("%d %d %d", &a, &b, &c);
if(a>b && a>c);
{
printf("a is greater\n");
}
```

```
else if (b>a && b>c)
{
printf("b is greater.\n");
}
else
{
printf("c is greater.\n");
}
getch();
}
```

(iii) Nested if statement :

Syntax :

```
if (condition 1)
{
if (condition 2)
{
Statement - 1;
}
else
{
Statement - 2;
}
}
else
{
if (condition 3)
{
Statement 3;
}
else
{
Statement 4;
}
}
```

(Q-6) Write a program C to find out greatest among 3 number using nested if condition.

```
# include <stdio.h>
# include <conio.h>
main()
{
int a,b,c;
printf("enter any three number \n");
scanf("%d %d %d", &a, &b, &c);
else
{
if (b>c)
{
printf("b is greater \n");
}
else
{
printf("c is greater \n");
}
getch();
}
```

(b) Switch Statement

Syntax

```
Switch (Variable)
{
case Value 1:
block 1;
break;
case Value 2:
block 2;
break;
case Value n:
```

```
block 0 ;
break ;
default :
block n+1 ;
    }
```

(P-7) Write a program in C to print the colour according to the code. that is 1 for red, 2 for green, 3 for white, 4 for yellow.

```
# include <stdio.h>
# Include <conio.h>
Void main ( )
    {
    int code
print f ("main menu \n");
printf ("1 for red \n");
Printf ("2 for green \n");
Print f ("3 for white \n");
Print f ("4 for yellow \n");
Print ("enter any code \n");
scan f ("%d", & code);
switch (code)
    {
Case 1 :
Print f ("Colour is Red \n");
    break ;
Case 2 :
print f ("Colour is green \n");
    break ;
Case 3 :
printf ("colour is white \n");
    break ;
Case 4 :
```

```
Printf (" colour is yellow \n ") ;
    break ;
    default ;
printf (" out of choice. \n ") ;
        }
    getch ( ) ;
        }
```

Output

Main menu
1 for red
2 for green
3 for white
4 for yellow

enter any code
1
Colour is Red.

(c) Conditional Control Statement :

Syntax

Exp 1 ? Exp 2 : Exp 3 ;

Example

x = (a>b) ? a : b ;

If a = 5
b = 10
then (5>10) ? 5 : 10 ;
No.

If a = 7
b = 3
then (7>3) ? 7 : 3 ;
Yes

## 2. Looping Statements :-

When a single statement or a group of statements will be executed again and again in a program then such type processing is called loop. The looping statements used in C-language are:

(a) while statement or while loop
(b) do statement or do loop
(c) for statement or for loop
(d) Nested for loop statement.

### (c) For loop

Syntax

```
For (initialization ; condition ; increment/decrement)
    {
    statement ;
    }
```

(P-8) Write a program of C sum of n random number using for loop.

```
# include <stdio.h>
# include <conio.h>
main()
{
int i, sum = 0, n ;
printf ("enter the value of n \n");
scanf ("%d", &n) ;
For (i = 1 ; i <= n ; ++) .
{
sum = sum + i ;
}
printf ("sum = %d", sum) ;
getch () ;
}
```

output

0+1 = 1
1+2 = 3
3+3 = 6
6+4 = 10
10+5 = 15

(a) While loop

Syntax

```
While (test Condition)
  {
block of statements ;
  }
Statement - x ;
```

(P-9) Write a program 5 natural numbers using while loop.

```
# Include <stdio.h>
# Include <conio.h>
 Void main()
   {
 Int i, sum = 0, n ;
 printf ("enter value of n:\n");
  Scanf ("%d", &n);
        i = 1 ;
  while (i <= n)
        {
  ~~scan = scanf i ;~~ sum = sum + i ;
    i ++
        }
 printf ("Sum = %d", sum);
      getch ();
        }
```

(b) do while

Syntax

```
do
{
Statement
}
while (condition) ;
```

(P-10) Write a program of C using 5 random number using do while.

```
# include <stdio.h>
# include <conio.h>
  Void main()
    {
  int i, sum = 0, n;
  printf("enter value of n\n");
  scanf("%d", &n\n");
   do ~~{i<=n);~~
     {
  ~~scan = scanf c=; sum sum+i;~~
     i++
     } while(i<=n)
  printf("sum = %d", &sum);
     getch();
     }
```

(d) Nested for statement :-

Syntax

```
for (initialization; condition; increment/decrement)
  {
```

(P-11) write a program in C using the concept of nested for loop.

```
# include <stdio.h>
# include <conio.h>
  main()
    {
  int n=2, m=3, i, j;
  printf("Out put is as\n");
  for (i=1, i<n, i++)
    {
  for (j=1, k<=m, j++)
    {
```

```
Printf("Hello \n");
printf("OK \n");
    }
getch();
    }
```

Q-12) Write a program in C to display the result of addition substraction, multiplication & division of two number by taking different variables.

```
# include <stdio.h>
# include <conio.h>
 Void main()
    {
 int a=5, b=6;
  a = a+b;
 printf("Sum is = %d \n", a);
  int c = 6, d = 5;
    c = c-d;
 printf("Sub is = %d \n", c)
  int  e = 7, f = 5;
     e = e * f;
 printf("mul is = %d \n", e);
  int g = 8, h = 2
     g = g / h;

 Printf("div is = %d \n", g);
    int i = 5, j = 6
     i = i % j
 printf("mod is = %d \n", i);
      getch();
    }
```

Output

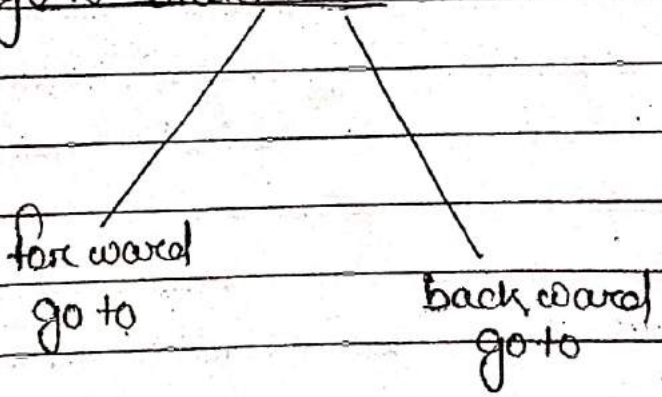
add = 11
sub = 1
mul = 35
div = 4
mod = 5.

## 3. Jumping Statement :-

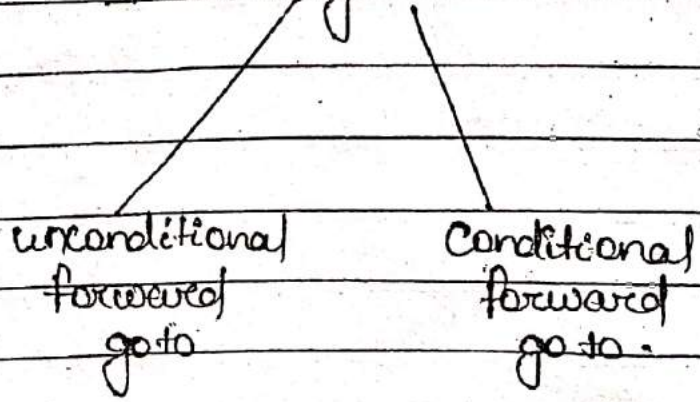
There are three different controls used to jump from one C program statement to another and make the execution of the programming procedure fast. These three jumping controls are

(a) goto statement
(b) break statement
(c) Continue statement

### (a) goto statement




### forward go to



## unconditional forward go to

Syntax

```
Statement 1 ;
Statement 2 ;
go to label ;
Statement 3 ;
Statement 4 ;
label ;
Statement 5 ;
Statement 6 ;
```

(P-13) Write a program C to add & sub two number by using unconditional go to statement.

```
# Include <stdio.h>
# Include <conio.h>
  Void main ()
      {
   int a, b, c, d
  printf ("enter the value of a & b \n");
  Scanf ("%d %d", &a, &b);
      c = a + b ;
      d = a - b ;
  printf ("Sum is = %d \n", c);
        go to mm ;
  printf ("Sub is = %d \n", d);
        mm ;
      getch ();
      }
```

Output

enter the value of a & b.

6

7

Sum is = 13.

## Conditional forward go to:

Syntax

```
Statement 1 ;
Statement 2 ;
If (condition)
go to label ;
Statement 3 ;
Statement 4 ;
label :
Statement 5 ;
Statement 6 ;.
```

## Conditional backward go to:

Syntax

```
Statement 1 ;
Statement 2 ;
If (condition)
label :
Statement 3 ;
Statement 4 ;
go to label ;
Statement 5 ;
Statement 6 ;.
```

(b) Break statement :

Syntax

```
for (initial value ; condition, increment / decrement)
{
if (condition -1)
{
break ;
}
}
```

(c) Continue Statement

Syntax

```
while (condition -1)
{
Statement 1 ;
Statement 2 ;
if (condition -2)
{
continue ;
}
Statement 3 ;
Statement 4 ;
}
```

(P-15) Program to Interchange value of two variable without using third variable.

```
#Include <stdio.h>
# Include <conio.h>
main()
{
int a=10, b=20;
printf("\n Before Interchange the value is : a= %d,
                                    b= %d", a, b);
a = a+b;
b = a-b;
a = a-b;
printf("\n after Interchange the value is : a= %d,
                                    b= %d", a, b);
getch();
}
```

Output

Before Interchange the value is : a=10 b=20

After Interchange the value is : a=20 b=10.

(P-16) program to interchange value of two numbers using third variable.

```
#include <stdio.h>
#include <conio.h>
main()
{
int a=10, b=20, temp;
printf("\n Before interchange the value is :
                a=%d, b=%d", a,b);
temp=a;
a=b;
b=temp;
printf("\n After interchange the value is :
                a=%d, b=%d", a,b);
getch();
}
```

Output

Before interchange the value is: a=10 b=20

After interchange the value is : a=20 b=10.

(P-17) Program to find factorial of a number. 

```
# Include <stdio.h>
# Include <dos.h>
# Include <conio.h>
main()
  {
  int n, i, m;
  clrscr();
  printf("\n Enter the number : ");
  scanf("%d", &n);
  m = 1;
  for (i = 1; i <= n; i++)
    {
    m = m * i;
    }
  printf("\n factorial of %d is = %d", n, m);
  getch();
  }
```

Output

Enter the number : 5

factorial of 5 is = 120.

(P-18) program to find the sum of the series.
$1 + 1/x + 1/x^2 + 1/x^3 + \ldots\ldots 1/x^n$.

```
# include <stdio.h>
# include <dos.h>
# include <math.h>
main()
  {
int i, x, n;
 float sum, P;
printf("\n Enter the base value :");
 scanf("%d", &x);
 printf("\n Enter the power value:");
 scanf("%d", &n);
    i=1;
  sum = 1;
   while (i <= n)
    {
  P = pow(x, i)
   sum = sum + (1/P);

    i = i+1;
    }
printf("\n sum is = %f", sum);
  getch();
    }
```

Output is :

Enter the base Value : 2
Enter the power Value : 3
Sum is = 1.87.

(P-19) program to illustrate the concept of puts() and gets() function.

```
#include <stdio.h>
main()
{
char name[20];
puts("enter any name");
gets(name);
puts("The entered name is :");
puts(name);
}
```

Output is

```
enter any name
yashmika
The entered name is :
Yashmika.
```

# CHAPTER - 7

## Complex Data Types

C has the usual facilities for grouping things together to form composite types—arrays and records (which are called "structures"). The following definition declares a type called "struct fraction" that has two integer sub fields named "numerator" and "denominator". If you forget the semicolon it tends to produce a syntax error in whatever thing follows the struct declaration.

```
struct fraction {
int numerator;
int denominator;
}; // Don't forget the semicolon!
```

This declaration introduces the type struct fraction (both words are required) as a new type. C uses the period (.) to access the fields in a record. You can copy two records of the same type using a single assignment statement, however == does not work on structs.

```
struct fraction f1, f2; // declare two fractions
f1.numerator = 22;
f1.denominator = 7;
f2 = f1; // this copies over the whole struct
```

## Arrays

The simplest type of array in C is one which is declared and used in one place. There are more complex uses of arrays which I will address later along with pointers. The following declares an array called scores to hold 100 integers and sets the first

and last elements. C arrays are always indexed from 0. So the first int in scores array is scores[0] and the last is scores[99].

int scores[100];

scores[0] = 13; // set first element

scores[99] = 42; // set last element

0

scores

Index 1 2 99

Someone else's memory off either end of the array — do not read or write this memory. There is space for each int element in the scores array — this element is referred to as scores[0].

-5673 22541 42

These elements have random values because the code has not yet initialized them to anything. The name of the array refers to the whole array. (implementation) it works by representing a pointer to the start of the array.

It's a very common error to try to refer to non-existent scores[100] element. C does not do any run time or compile time bounds checking in arrays. At run time the code will just access or mangle whatever memory it happens to hit and crash or misbehave in some unpredictable way thereafter. "Professional programmer's language." The convention of numbering things 0..(number of things - 1) pervades the language. To best integrate with C and other C programmers, you should use that sort of numbering in your own data structures as well.

## Multidimensional Arrays

The following declares a two-dimensional 10 by 10 array of integers and sets the first and last elements to be 13.

```
int board [10][10];

board[0][0] = 13;

board[9][9] = 13;
```

The implementation of the array stores all the elements in a single contiguous block of memory. The other possible implementation would be a combination of several distinct one dimensional arrays -- that's not how C does it. In memory, the array is arranged with the elements of the rightmost index next to each other. In other words, board[1][8] comes right before board[1][9] in memory. (highly optional efficiency point) It's typically efficient to access memory which is near other recently accessed memory. This means that the most efficient way to read through a chunk of the array is to vary the rightmost index the most frequently since that will access elements that are near each other in memory.

## Array of Structs

The following declares an array named "numbers" which holds 1000 struct fraction's.

```
struct fraction numbers[1000];

numbers[0].numerator = 22; /* set the 0th struct fraction */

numbers[0].denominator = 7;
```

Here's a general trick for unraveling C variable declarations: look at the right hand side and imagine that it is an expression. The type of that expression is the left hand side. For the above declarations, an expression which looks like the right hand side (numbers[1000], or really anything of the form numbers[...]) will be the type on the left hand side (struct fraction).

## Pointers

We'll see shortly how a pointer is set to point to something -- for now just assume the pointer points to memory of the appropriate type. In an expression, the unary * to the left of a pointer dereferences it to retrieve the value it points to.

There's an alternate, more readable syntax available for dereferencing a pointer to a struct. A "->" at the right of the pointer can access any of the fields in the struct. So the reference to the numerator field could be written f1->numerator.

Here are some more complex declarations...

```
struct fraction** fp; // a pointer to a pointer to a struct fraction

struct fraction fract_array[20]; // an array of 20 struct fractions

struct fraction* fract_ptr_array[20]; // an array of 20 pointers to

// struct fractions
```

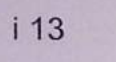
One nice thing about the C type syntax is that it avoids the circular definition problems which come up when a pointer structure needs to refer to itself. The following definition defines a node in a linked list. Note that no preparatory declaration of the node pointer type is necessary.

```
struct node {

int data;

struct node* next;

};
```

**The & Operator**

The & operator is one of the ways that pointers are set to point to things. The & operator computes a pointer to the argument to its right. The argument can be any variable which takes up space in the stack or heap (known as an "LValue" technically). So &i and &(f1->numerator) are ok, but &6 is not. Use & when you have some memory, and you want a pointer to that memory.

```
void foo() {
int* p; // p is a pointer to an integer
int i; // i is an integer
p = &i; // Set p to point to i
*p = 13; // Change what p points to -- in this case i -- to 13
// At this point i is 13. So is *p. In fact *p is i.
}
```




When using a pointer to an object created with &, it is important to only use the pointer so long as the object exists. A local variable exists only as long as the function where it is declared is still executing (we'll see functions shortly). In the above example, i exists only as long as foo() is executing. Therefore any pointers which were initialized with &i are valid only as long as foo() is executing. This "lifetime" constraint of local memory is standard in many languages, and is something you need to take into account when using the & operator.

**NULL**

A pointer can be assigned the value 0 to explicitly represent that it does not currently have a pointee. Having a standard representation for "no current pointee" turns out to be very handy when using pointers. The constant NULL is defined to be 0 and is typically used when setting a pointer to NULL. Since it is just 0, a NULL pointer will behave like a boolean false when used in a boolean context. Dereferencing a NULL pointer is an error which, if you are lucky, the computer will detect at runtime -- whether the computer detects this depends on the operating system.

**Pitfall -- Uninitialized Pointers**

When using pointers, there are two entities to keep track of. The pointer and the memory it is pointing to, sometimes called the "pointee". There are three things which must be done for a pointer/pointee relationship to work...

(1) The pointer must be declared and allocated

(2) The pointee must be declared and allocated

(3) The pointer (1) must be initialized so that it points to the pointee (2)

The most common pointer related error of all time is the following: Declare and allocate the pointer (step 1). Forget step 2 and/or 3. Start using the pointer as if it has been setup to point to something. Code with this error frequently compiles fine, but the runtime results are disastrous. Unfortunately the pointer does not point anywhere good unless (2) and (3) are done, so the run time dereference operations on the pointer with * will misuse and trample memory leading to a random crash at some point.

```
{
int* p;
*p = 13; // NO NO NO p does not point to an int yet
// this just overwrites a random area in memory
}
```

-14346

p

i

Of course your code won't be so trivial, but the bug has the same basic form: declare a pointer, but forget to set it up to point to a particular pointee.

**Using Pointers**

Declaring a pointer allocates space for the pointer itself, **but it does not allocate space for the pointee**. The pointer must be set to point to something before you can dereference it.

Here's some code which doesn't do anything useful, but which does demonstrate (1) (2)

(3) for pointer use correctly...

```
int* p;                 // (1) allocate the pointer
int i;                  // (2) allocate pointee
struct fraction f1;     // (2) allocate pointee
p = &i;                 // (3) setup p to point to i
*p = 42;                // ok to use p since it's setup
p = &(f1.numerator);    // (3) setup p to point to a different int
*p = 22;
p = &(f1.denominator);  // (3)
*p = 7;
```

So far we have just used the & operator to create pointers to simple variables such as i. Later, we'll see other ways of getting pointers with arrays and other techniques.

**C Strings**

C has minimal support of character strings. For the most part, strings operate as ordinary arrays of characters. Their maintenance is up to the programmer using the standard facilities available for arrays and pointers. C does include a standard library of functions which perform common string operations, but the programmer is responsible for the managing the string memory and calling the right functions. Unfortunately computations involving strings are very common, so becoming a good C programmer often requires becoming adept at writing code which manages strings which means managing pointers and arrays.

A C string is just an array of char with the one additional convention that a "null" character ('\0') is stored after the last real character in the array to mark the end of

the string. The compiler represents string constants in the source code such as "binky" as arrays which follow this convention. The string library functions (see the appendix for a partial list) operate on strings stored in this way. The most useful library function is strcpy(char dest[], const char source[]); which copies the bytes of one string over to another. The order of the arguments to strcpy() mimics the arguments in of '=' -- the right is assigned to the left. Another useful string function is strlen(const char string[]); which returns the number of characters in C string not counting the trailing '\0'.

Note that the regular assignment operator (=) does **not** do string copying which is why strcpy() is necessary. See Section 6, Advanced Pointers and Arrays, for more detail on how arrays and pointers work.

The following code allocates a 10 char array and uses strcpy() to copy the bytes of the string constant "binky" into that local array.

```
{
char localString[10];
strcpy(localString, "binky");
}
```

b i n k y 0 x x x x x

0 1 2 ...

localString

The memory drawing shows the local variable localString with the string "binky" copied into it. The letters take up the first 5 characters and the '\0' char marks the end of the string after the 'y'. The x's represent characters which have not been set to any particular value. If the code instead tried to store the string "I enjoy languages whichh have good string support" into localString, the code would just crash at run time since the 10 character array can contain at most a 9 character string. The large

string will be written passed the right hand side of localString, overwriting whatever was stored there.

**String Code Example**

Here's a moderately complex for loop which reverses a string stored in a local array. It demonstrates calling the standard library functions strcpy() and strlen() and demonstrates that a string really is just an array of characters with a '\0' to mark the effective end of the string. Test your C knowledge of arrays and for loops by making a drawing of the memory for this code and tracing through its execution to see how it works.

```
{
char string[1000]; // string is a local 1000 char array
int len;
strcpy(string, "binky");
len = strlen(string);
/*
Reverse the chars in the string:
i starts at the beginning and goes up
j starts at the end and goes down
i/j exchange their chars as they go until they meet
*/
int i, j;
char temp;
for (i = 0, j = len - 1; i < j; i++, j--) {
temp = string[i];
```

```
string[i] = string[j];
string[j] = temp;
}
// at this point the local string should be "yknib"
}
```

### "Large Enough" Strings

The convention with C strings is that the owner of the string is responsible for allocating array space which is "large enough" to store whatever the string will need to store. Most routines do not check that size of the string memory they operate on, they just assume its big enough and blast away. Many, many programs contain declarations like the following...

```
{
char localString[1000];
...
}
```

The program works fine so long as the strings stored are 999 characters or shorter. Someday when the program needs to store a string which is 1000 characters or longer, then it crashes. Such array-not-quite-big-enough problems are a common source of bugs, and are also the source of so called "buffer overflow" security problems. This scheme has the additional disadvantage that most of the time when the array is storing short strings, 95% of the memory reserved is actually being wasted. A better solution allocates the string dynamically in the heap, so it has just the right size. To avoid buffer overflow attacks, production code should check the size of the data first, to make sure it fits in the destination string. See the strlcpy() function in Appendix A.

**char***

Because of the way C handles the types of arrays, the type of the variable localString above is essentially char*. C programs very often manipulate strings using variables of type char* which point to arrays of characters. Manipulating the actual chars in a string requires code which manipulates the underlying array, or the use of library functions such as strcpy() which manipulate the array for you. See Section 6 for more detail on pointers and arrays.

**TypeDef**

A typedef statement introduces a shorthand name for a type. The syntax is...

typedef <type><name>;

The following defines Fraction type to be the type (struct fraction). C is case sensitive, so fraction is different from Fraction. It's convenient to use typedef to create types with upper case names and use the lower-case version of the same word as a variable.

```
typedef struct fraction Fraction;
Fraction fraction; // Declare the variable "fraction" of type "Fraction"
// which is really just a synonym for "struct fraction".
```

The following typedef defines the name Tree as a standard pointer to a binary tree node where each node contains some data and "smaller" and "larger" subtree pointers.

```
typedef struct treenode* Tree;
struct treenode {
int data;
Tree smaller, larger; // equivalently, this line could say
```

```
}; // "struct treenode *smaller, *larger"
```

## Functions

All languages have a construct to separate and package blocks of code. C uses the "function" to package blocks of code. This article concentrates on the syntax and peculiarities of C functions. The motivation and design for dividing a computation into separate blocks is an entire discipline in its own.

A function has a name, a list of arguments which it takes when called, and the block of code it executes when called. C functions are defined in a text file and the names of all the functions in a C program are lumped together in a single, flat namespace. The special function called "main" is where program execution begins. Some programmers like to begin their function names with Upper case, using lower case for variables and parameters, Here is a simple C function declaration. This deciares a function named Twice which takes a single int argument named num. The body of the function computes the value which is twice the num argument and returns that value to the caller.

```
/*
Computes double of a number.
Works by tripling the number, and then subtracting to get back to double.
*/
static int Twice(int num) {
int result = num * 3;
result = result - num;
return(result);
}
```

**Syntax**

The keyword "static" defines that the function will only be available to callers in the file where it is declared. If a function needs to be called from another file, the function cannot be static and will require a prototype -- see prototypes below. The static form is convenient for utility functions which will only be used in the file where they are declared. Next , the "int" in the function above is the type of its return value. Next comes name of the function and its list of parameters. When referring to a function by name in documentation or other prose, it's a convention to keep the parenthesis () suffix, so in this case I refer to the function as "Twice()". The parameters are listed with their types and names, just like variables. Inside the function, the parameter num and the local variable result are "local" to the function -- they get their own memory and exist only so long as the function is executing. This independence of "local" memory is a standard feature of most languages.

The "caller" code which calls Twice() looks like...

```
int num = 13;

int a = 1;

int b = 2;

a = Twice(a); // call Twice() passing the value of a

b = Twice(b + num); // call Twice() passing the value b+num

// a == 2

// b == 30

// num == 13 (this num is totally independent of the "num" local to Twice()
```

**Call by Value vs. Call by Reference**

C passes parameters "by value" which means that the actual parameter values are copied into local storage. The caller and callee functions do not share any memory --

they each have their own copy. This scheme is fine for many purposes, but it has twodisadvantages.

1) Because the callee has its own copy, modifications to that memory are not communicated back to the caller. Therefore, value parameters do not allow the callee to communicate back to the caller. The function's return value can communicate some information back to the caller, but not all problems can be solved with the single return value.

2) Sometimes it is undesirable to copy the value from the caller to the callee because the value is large and so copying it is expensive, or because at a conceptual level copying the value is undesirable.

The alternative is to pass the arguments "by reference". Instead of passing a copy of a value from the caller to the callee, pass a pointer to the value. In this way there is only one copy of the value at any time, and the caller and callee both access that one value through pointers.

Some languages support reference parameters automatically. C does not do this – the programmer must implement reference parameters manually using the existing pointer constructs in the language.

**Swap Example**

The classic example of wanting to modify the caller's memory is a swap() function which exchanges two values. Because C uses call by value, the following version of Swap will not work...

```
void Swap(int x, int y) { // NO does not work
int temp;
temp = x;
x = y; // these operations just change the local x,y,temp
```

```
y = temp; // -- nothing connects them back to the caller's a,b
}
// Some caller code which calls Swap()...
int a = 1;
int b = 2;
Swap(a, b);
```

Swap() does not affect the arguments a and b in the caller. The function above only operates on the copies of a and b local to Swap() itself. This is a good example of how "local" memory such as ( x, y, temp) behaves -- it exists independent of everything else only while its owning function is running. When the owning function exits, its local memory disappears.

**Reference Parameter Technique**

To pass an object X as a reference parameter, the programmer must pass a pointer to X instead of X itself. The formal parameter will be a pointer to the value of interest. The caller will need to use & or other operators to compute the correct pointer actual parameter. The callee will need to dereference the pointer with * where appropriate to access the value of interest. Here is an example of a correct Swap() function.

```
static void Swap(int* x, int* y) { // params are int* instead of int
int temp;
temp = *x; // use * to follow the pointer back to the caller's memory
*x = *y;
*y = temp;
}
```

```
// Some caller code which calls Swap()...
```

```
int a = 1;
int b = 2;
Swap(&a, &b);
```

Things to notice...

- The formal parameters are int* instead of int.
- The caller uses & to compute pointers to its local memory (a,b).
- The callee uses * to dereference the formal parameter pointers back to get the caller's memory.

Since the operator & produces the address of a variable -- &a is a pointer to a. In Swap() itself, the formal parameters are declared to be pointers, and the values of interest (a,b) are accessed through them. There is no special relationship between the **names** used for the actual and formal parameters. The function call matches up the actual and formal parameters by their order -- the first actual parameter is assigned to the first formal parameter, and so on. I deliberately used different names (a,b vs x,y) to emphasize that the names do not matter.

**const**

The qualifier const can be added to the left of a variable or parameter type to declare that the code using the variable will not change the variable. As a practical matter, use of const is very sporadic in the C programming community. It does have one very handy use, which is to clarify the role of a parameter in a function prototype...

```
void foo(const struct fraction* fract);
```

In the foo() prototype, the const declares that foo() does not intend to change the struct fraction pointee which is passed to it. Since the fraction is passed by pointer, we could not know otherwise if foo() intended to change our memory or not. Using the const, foo() makes its intentions clear. Declaring this extra bit of information helps to clarify the role of the function to its implementor and caller.

**Bigger Pointer Example**

The following code is a large example of using reference parameters. There are several common features of C programs in this example...Reference parameters are used to allow the functions Swap() and IncrementAndSwap() to affect the memory of their callers.

There's a tricky case inside of IncrementAndSwap() where it calls Swap() -- no additional use of & is necessary in this case since the parameters x, y inside InrementAndSwap() are already pointers to the values of interest. The names of the variables through the program(a, b, x, y, alice, bob) do not need to match up in any particular way for the parameters to work. The parameter mechanism only depends on the types of the parameters and their order in the parameter list -- not their names. Finally this is an example of what multiple functions look like in a file and how they are called from the main() function.

```
static void Swap(int* a, int* b) {
int temp;
temp = *a;
*a = *b;
*b = temp;
}
static void IncrementAndSwap(int* x, int* y) {
(*x)++;
(*y)++;
Swap(x, y); // don't need & here since a and b are already
// int*'s.
}
int main()
{
```

```
int alice = 10;
int bob = 20;
Swap(&alice, &bob);
// at this point alice==20 and bob==10
IncrementAndSwap(&alice, &bob);
// at this point alice==11 and bob==21
return 0;
}
```