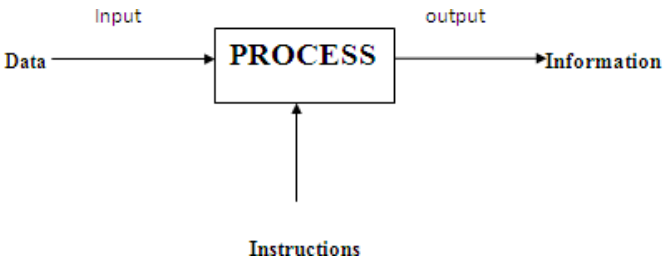
UNIT-I
HISTORY AND HARDWARE

1. **Computer:** A Computer is an electronic device that takes data and instructions as an ***input*** from the user, ***process*** data, and provides useful information known as ***output***. This cycle of operation of a computer is known as the input-process-output cycle.



Charles Babbage (26 December 1791 – 18 October 1871) is a father of computers. He was invented first analytical computer in year 1822.

1.1 Classification of Computers:-

Computers can be broadly classified into three types:-

i. **Analog Computers**:- These computers use analog signals for its operations and work on basic principle of measuring of signals.

ii. **Digital Computers**: - These computers work with digital signal, works on 0's and 1's. The basic principle is counting of digits. These computers can perform a variety of operations.

iii. **Hybrid Computers: -** it is the combination of both digital and analog computers. These are used for special purposes such as industrial automation etc.

Types of digital computers: -

S. No	Type	Specifications
1	PC (Personal Computer)	It is a single user computer system having moderately powerful microprocessor. It is also called as micro computers (smallest computers). These are inexpensive and popular.
2	Workstation	It is also a single user computer system which is similar to personal computer but have more powerful microprocessor.
3	Mini Computer	It is 16 times faster than micro computers. It is a multi-user computer system which is capable of supporting hundreds of users simultaneously. At a time 12 users can interact with these computers.
4	Main Frame	It is large computer having the capability of around 16bmini computers. At a time 64 users can interact with these. It is a multi-user computer system which is capable of supporting hundreds of users simultaneously. Software technology is different from minicomputer.

5	Super Computer	It is an extremely fast computer which can execute hundreds of millions of instructions per second. It is having the capability of around 64 mainframe computers. At a time 120 users can interact. Another name is maxi computers. Examples: weather forecasting, scientific simulations, satellite launching

1.2 Characteristics of Computer:

The characteristics of computers that have made them so powerful and universally useful are speed, accuracy, diligence, versatility, reliability and storage capacity.

- Speed

  Computers work at an incredible speed.

  It is capable of performing calculation of very large amount of data.

  A powerful computer is capable of performing about 3-4 million simple instructions per second.

- Accuracy

  In addition to being fast, computers are also accurate.

  The calculations are 100% error free.

  Computers perform all jobs with 100% accuracy provided that correct input has been given.

- Diligence

  Computers can perform repetitive calculations any no. Times with same level of accuracy.

- Versatility

  Computers are versatile machines and are capable of performing more no. Of varies tasks and can be used for many different purposes, ex: - Railway/Air reservation, Banks, Hotels, Weather forecasting and many more.

- Reliability

  Computer provides results with no error. Computer generated error are human error at specification.

- Storage Capacity

  Computers can store huge amount of data in multiple formats. The storage area mainly divided into two categories: i. Main Memory ii. Secondary Memory

Drawbacks of a computer:

No I.Q

Lack of Decision making power

No feelings

Applications of computers:

Following list demonstrates various applications of computers in today's arena.

1. Business
2. Banking
6. Engineering design
7. Military

3. Education
4. Marketing
5. Health Care
8. Communications
9. Government
10. Scientific research

1.3 Computer Generations:

Generations	Zero	First	Second	Third	Fourth	Fifth
Year	Pre 1946	1946-1956	1957-1963	1964-1981	1982-1989	1990- -
Representative Computers	Abacus, Difference Engine, Analytical Engine	Enaiac, Edvac, Univac, IBM 650	IBM 7094, CDC 6600, Honey well 400.	IBM 360/370, Honey Well 2000, cyber ZD5	CRAY-XMP, IBM 305	Distributed computing, India, CRAY-III
Hardware	Counter wheels with mechanical components	Vacuum tubes, magnetic drum, CRT screen	Transistor, magnetic core memory.	Semiconductor memory, magnetic disk, micro and mini computers.	Distributed computing system, VLSI microprocessor based.	Artificial intelligence
Software	-	Stored program, machine code	HLL, FORTRAN, COBOL, ALGOL	Pascal, C Computer Graphics, OS	C++, OOPS	Java
Performance	Numerical operations at slow speed	2 KB Memory, 10 KIPS	32KB, 200 KIPS	2MB, 5 MIPS	8MB, 30 MIPS	1 GIGA, 1 TERA (IPS)

2. **Computer system:** A computer system consists of hardware and software components.

2.1 Hardware

The physical parts of computer system are referred to as hardware. All electronic and mechanical items that constitute computer system fall into the category of hardware. Hardware includes following components:

i. Input devices ii. Output devices iii. Processor iv. Memory

i. Input devices

An input device sends information to a computer system for processing. An **input** device can send data to another device, but it cannot receive data from another device. Examples of an input device include:

- ✔ Keyboard
- ✔ mouse

- ✔ Scanner
- ✔ Joystick
- ✔ light pen and
- ✔ track ball

Which can send data (input) to the computer, but they cannot receive or reproduce information (output) from the computer.

ii. Output devices

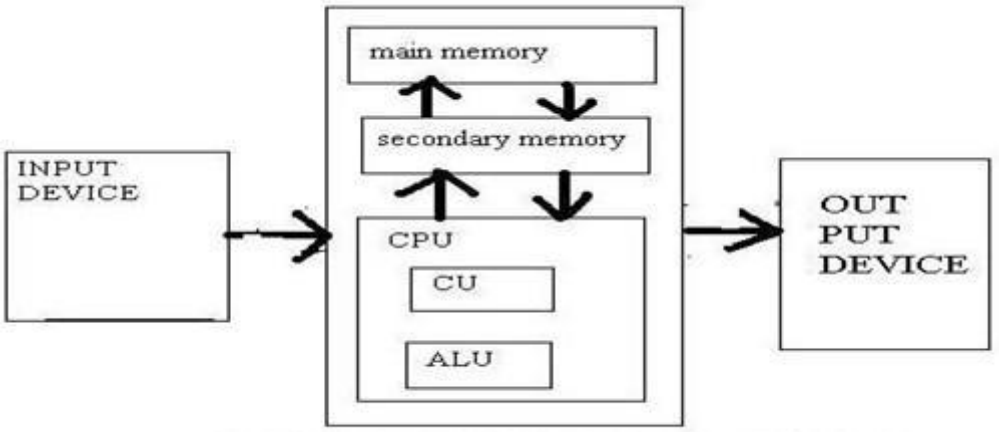
An output device reproduces or displays the results of that processing. An **output** device can receive data from another device, but it cannot send data to another device. Examples of an output device include a

- ✔ Monitor
- ✔ Speakers
- ✔ Printer and
- ✔ Projector

Which can receive data (output) from the computer, but they cannot send information (input) to the computer.

iii. Processor

It is also called as CPU (Central Processing Unit). It is the brain of the computer that is responsible for controlling and executing program instructions entered through the input devices like keyboard and mouse.



BLOCK DIAGRAM OF A DIGITAL COMPUTER

- ✔ **Control unit (CU):** stores the instruction set, which specifies the operations. CU transfers data and instructions to ALU for arithmetic operations.
- ✔ **Arithmetic & Logical Unit (ALU):** it performs both arithmetic and logical operations on the received data.

iv. Memory

The *computer memory* is a temporary storage area in a computer system. It is used to store data and instructions. The memory is divided into large number of small parts called cells. Computer Memory can be divided into mainly two types.

1) Primary memories which includes RAM and ROM
2) Secondary memories which includes hard disks, floppy discs, USB storages and CD-ROM, Magnetic tapes and drives.

1) **Primary memory:** It is also called as ***main memory*** of the computer. It holds program instructions and data temporarily. Each bit in memory is identified by a unique address. Data is stored in Machine-Understandable binary form.

There are three types of primary memory:

i.Random Access Memory ii. Read-Only Memory iii. Cache Memory

i. Random Access Memory(RAM)

It is the internal memory of the CPU for storing data, program and program result. It is read/write memory which stores data until the machine is working. As soon as the machine is switched off, data is erased.RAM is divided into two types

- ✔ **Static RAM (SRAM):** The word **static** indicates that the memory retains its contents as long as power is being supplied. However, data is lost when the power gets down due to volatile nature.
- ✔ **Dynamic RAM (DRAM):** DRAM, unlike SRAM, must be continually **refreshed** in order to maintain the data. This is done by placing the memory on a refresh circuit that rewrites the data several hundred times per second.

ii. Read-Only Memory(ROM)

ROM stands for Read-only Memory. The memory which can be used only read but cannot write on it. This type of memory is non-volatile. The information is stored permanently in such memories during manufacture.ROM is divided into different types

- ✔ MROM (Maskable Read-only Memory)
- ✔ PROM (Programmable Read-only Memory)
- ✔ EPROM (Erasable Programmable Read-only Memory)
- ✔ EEPROM (Electrically Erasable Programmable Read-only Memory)

Advantages of ROM

The advantages of ROM are as follows:

- ✔ Non-volatile in nature
- ✔ These cannot be accidentally changed
- ✔ Cheaper than RAMs

- ✔ Easy to test
- ✔ More reliable than RAMs
- ✔ These are static and do not require refreshing
- ✔ Its contents are always known and can be verified

Differences between RAM & ROM:-

RAM	ROM
It is read/write memory	It is read only memory
It is volatile storage device	It is non-volatile (permanent) storage device
Data is erased as soon as power is turned off	Data remains stored even after power supply is turned off
It is used as main memory for instruction execution	It is used to store Basic Input Output System(BIOS)

iii. Cache Memory

It is used to store the data related to application that was last processed by ALU. It always placed between processor and main memory of the system.

When the processor is processing, it checks for the cache memory and then RAM, if not available, it goes to hard disk.

Characteristics of Main Memory

- ✔ These are semiconductor memories
- ✔ It is known as main memory.
- ✔ Usually volatile memory.
- ✔ Data is lost in case power is switched off.
- ✔ It is working memory of the computer.
- ✔ Faster than secondary memories.
- ✔ A computer cannot run without primary memory.

2) Secondary Memory

This type of memory is also known as *external memory* or *non-volatile*. It is slower than main memory. These are used for storing large data/information permanently. CPU directly does not access these memories instead they are accessed via input-output routines. Contents of secondary memories are first transferred to main memory, and then CPU can access it. These devices are classified as:

i. **Magnetic storage device**: - Stores information that can be read, erased, re-written no. of times. Ex- floppy disk, hard disk.

ii. **Optical storage device**: - Uses laser beams to read the stored data. Ex- CD-ROM, CD-RW (Rewritable CD) and digital video disks (DVD-ROM).

iii. **Magneto-optical storage device**: - Generally used to store large programs, files, and backup data.

iv. **Universal serial bus (USB) drive**: - it is removable storage device known as "pen drive".

Characteristic of Secondary Memory

- ✔ These are magnetic and optical memories
- ✔ It is known as backup memory.
- ✔ It is non-volatile memory.
- ✔ Data is permanently stored even if power is switched off.
- ✔ It is used for storage of data in a computer.
- ✔ Computer may run without secondary memory.
- ✔ Slower than primary memories.

2.2 Memory units

- ✔ The amount of data that can be stored in the storage unit that in which storage capacity is expressed in terms of Bytes.

1 Word = 4 Bytes
1 byte = 8 Bits
1 Binary digit = 1 Bit
8 Bits (2 Nibbles) = 1 Byte
4 Bits= 1 Nibble
16 Bits (2 Bytes) = 1 Half Word
32 Bits (4 Bytes) = 1 full word
64 Bits (8 Bytes) = 1 Double Word

Following are the main memory storage units:

S. No	Unit	Description
1	Bit(Binary digit)	A binary digit is logical 0 and 1 representing a passive or an active state of a component in an electric circuit.
2	Nibble	A group of 4 bits is called nibble.
3	Byte	A group of 8 bits is called byte. A byte is the smallest unit which can represent a data item or a character.
4	Word	A computer word, like a byte, is a group of fixed number of bits processed as a unit which varies from computer to computer but is fixed for each computer. The length of a computer word is called word-size or word length and it may be as small as 8 bits or may be as long as 96 bits. A computer stores the information in the form of computer words.
5	Kilobyte (KB)	1 KB = 1024 Bytes
6	Megabyte (MB)	1 MB = 1024 KB
7	Gigabyte (GB)	1GB = 1024 MB
8	Terabyte (TB)	1 TB = 1024 GB
9	Petabyte(PB)	1 PB = 1024 TB

2.3 Software

Software means a **set** of computer instructions or programs. Basically computer software divided into two types.

1. System software
2. Application software

1. **System software:** It is a collection of programs that directly interfaces with the hardware that provides the basic non-task-specific functions of the **computer**. Some common categories of system software are as follows

- ✔ **Language translators:** It is system software that transforms a computer program written by a programmer into a form that can be understood by the machine.
- ✔ **Operating System:** The Operating System (OS) is an interface between the computer software and hardware. The most popular and latest operating systems include UNIX, Linux, Windows 10, Mac, Sun Solaris etc.
- ✔ **Utilities**: Utility programs are those that may be requested by application programs many times during the execution phase. Some examples are as follows: sort/merge techniques and Transfer programs for transforming data content from one medium to another. Ex: disk to tape, tape to disk, etc.
- ✔ **Special purpose software**: It extends the capability of operating systems to provide specialized services to application programs. Ex: Loader, linker etc.

Features of system software are as follows:

- ✔ Close to system
- ✔ Fast in speed
- ✔ Difficult to design
- ✔ Difficult to understand
- ✔ Less interactive
- ✔ Smaller in size
- ✔ Difficult to manipulate
- ✔ Generally written in low-level language

2. **Application software:** Application software which is used by users to accomplish specific tasks. The application software includes printing documents, and permitting access to internet for web and video conferencing activities. Ex: MS-word, Excel, powerpoint etc.

Examples of Application software are following

- ✔ Payroll Software
- ✔ Student Record Software
- ✔ Inventory Management Software
- ✔ Income Tax Software
- ✔ Railways Reservation Software
- ✔ Microsoft Office Suite Software
- ✔ Microsoft Word
- ✔ Microsoft Excel
- ✔ Microsoft Powerpoint

Features of application software are as follows

- ✔ Close to user
- ✔ Easy to design

- ✔ More interactive
- ✔ Slow in speed
- ✔ Generally written in high-level language
- ✔ Easy to understand
- ✔ Easy to manipulate and use
- ✔ Bigger in size and requires large storage space

3. Computer Languages

3.1 **Programming language**: The language used in the communication of computer instructions is known as the programming language.

Programming languages can be used to create programs to control the behaviour of a machine. They are written to tell:

- ✔ What operation to perform
- ✔ Where to locate data
- ✔ How to give output
- ✔ When to make decisions

There are three levels of programming languages are available. They are:

1. Machine languages (Low-level Languages)
2. Assembly languages (Symbolic Languages)
3. High level languages (Procedure oriented)

1. Machine Languages

A language that is directly understood by the computer without any translation is called machine languages. A machine language is set of 0s and 1s. Machine language is usually referred to as the *first generation* languages. It is also referred to as machine code or object code. Therefore, all instructions and data should be written using *binary codes* i.e., 0's and 1's.

Advantages:

- ✔ Execution speed is very fast
- ✔ Efficient use of primary memory
- ✔ It does not require any translation because machine code is directly understood by the computer.

Disadvantages

- ✔ **Machine dependent:** The machine language is different for different types of computers.
- ✔ **Difficult to write program:** because it requires memorizing dozens of *opcodes* for different commands.
- ✔ Error prone
- ✔ Difficult to modify

2. **Assembly Languages:** Assembly languages are usually referred to as the *second-generation* languages. It is also lowest-level programming language. These are introduced in 1950's.

- ✔ Assembly languages are just one level higher than machine language.
- ✔ Assembly language consists of special codes called *mnemonics* to represent the language instructions instead of 0's and 1's. The mnemonic code is also called as operation code or *opcode*.
- ✔ During execution, the Assembly language program is converted into machine code through software called "Assembler".
- ✔ For example, ADD, SUB, MUL, JMP, LOAD

Advantages

- ✔ **Easier to memorize and use:** Assembly language program is easy to use, understand and memorize because it uses mnemonic codes in place of binary codes
- ✔ **Easy to write input data:** In assembly language programs the input data can be written in a decimal number system, later they are converted into binary.
- ✔ It is easier to correct errors and modify program instructions.
- ✔ The Assembly Language has the same efficiency of execution as the machine level language. Because this is a one-to-one translator between the assembly language program and its corresponding machine language program.
- ✔ Easy print out.
- ✔ Good library facility.

Disadvantages

- ✔ Machine dependent. A program written for one computer might not run on other computers with different hardware configurations.
- ✔ Knowledge of hardware is required.
- ✔ Time consuming
- ✔ Translators required (i.e. **Assembler**)

3. **High Level Languages (HLL):** It is designed for a specific job, and is easier to understand. It is more like human language and less like machine language. However, for a computer to understand and run a program created with a high-level language, it must be compiled into machine language.

The first high-level languages were introduced in the 1950's. Today, there are many high-level languages in use, including BASIC, C, C++, COBOL, FORTRAN, Java, Pascal, Perl, PHP, Python, Ruby and Visual Basic.

High-level programming languages can be classified into the following three categories:

i. Procedure-oriented languages (Third generation)
ii. Problem-oriented languages (Fourth generation)
iii. Natural languages (Fifth generation)

i. Procedure-oriented languages (Third generation)

HLL is designed to solve general-purpose problems. These are also called as "third generation languages". It uses English words to denote variables, programming structures and commands. 3GLs are FORTRAN, BASIC, Pascal and the C-family (C, C+, C++, C#, Objective-C) of languages. 3GLs are used to develop software applications for specific field.

ii. Problem-oriented languages (Fourth generation)

It is also called as "fourth generation" languages. 4GLs are designed to reduce the overall time, effort and cost of software development. These languages use either visual environment or textual environment for program development. Graphical user interface components such as icons, buttons, textboxes etc, used for program development.

iii. Natural Languages (Fifth Generation)

These languages make a computer to behave like an expert and solve problems. These are also known as "fifth generation languages". These languages are mainly focussed on constrain programming.

Advantages:

- ✔ High-level languages are user-friendly
- ✔ They are easier to learn
- ✔ They are easier to maintain
- ✔ They are problem-oriented rather than 'machine'-based
- ✔ A program written in a high-level language can be translated into many machine languages and can run on any computer for which there exists an appropriate translator
- ✔ The language is independent of the machine on which it is used i.e. programs developed in a high-level language can be run on any computer text
- ✔ They are similar to English and use English vocabulary and well-known symbols

Disadvantages:

- ✔ A high-level language has to be translated into the machine language by a translator, which takes up time
- ✔ The object code generated by a translator might be inefficient compared to an equivalent assembly language program

3.2 Language Translators

Editor

In general, an **editor** refers to any program capable of editing files. The term **editor** is commonly used to refer to a **text editor**, which is a software program that allows users to create or manipulate plain text computer files or C source code. For example gedit, notepad, WordPad etc.,

Debugger

This program helps us identify syntax errors in the source code.

Pre Processer

There are certain special instructions within the source code identified by the "#" symbol that are carried on by a special program called a pre-processor.

Compiler

It is a program which translates a high level language program into a machine language program.

Interpreter

An Interpreter is a program which translates statements of a program into machine code. It translates only one statement of the program at a time.

Linker

The machine code relating to the source code you have written is combined with some other machine code to derive the complete program in an executable file. This is done by a program called the linker.

Loader

Loader is a program that loads machine codes of a program into the system memory. In Computing, a **loader** is the part of an Operating System that is responsible for loading programs.

Difference between Compiler and Interpreter

S. No	Compiler	Interpreter
1.	Compiler Takes Entire program as input	Interpreter takes Single instruction as input.
2.	Intermediate Object code is generated	No intermediate Object code is generated
3.	Conditional control statements are executes faster	Conditional control statements are executes slower
4.	Memory requirement: more (since object code is generated)	Memory requirement: less
5.	Program need not be compiled every time	Every time higher level program is converted into lower level program
6.	Error are displayed after entire program is checked	Error are displayed for every instruction interpreted (if any)
7.	Example : C Compiler	Example : BASIC

4. Problem Solving or Program Development Steps

4.1 **Problem Solving:** is the process of solving a problem in a computer system by following a sequence of steps.

Step1: Developing an Algorithm**:** Algorithm is a sequence of steps written in English language to specify the tasks for solving the problem.

Step2: Drawing the flowchart: it is a pictorial representation of the flow control and logic in the solution. It is also called graphical representation.

Step3: Writing Pseudo code: It is similar to algorithms. It uses generic syntax for describing steps to be performed for solving problem.

Algorithm:-

An algorithm is a finite set of step-by-step instructions to solve a problem. It helps programmer in breaking down the solution into no. of sequential steps. The essential properties of an algorithm are:

1. **Finiteness:** The algorithm must always terminate after a finite number of steps.
2. **Definiteness:** Each and every instruction should be precise and unambiguous i.e. each and every instruction should be clear and should have only one meaning.
3. **Effectiveness:** Each instruction should be performed in finite amount of time.
4. **Input and Output**: An algorithm must take zero or more inputs, and produce one or more outputs.
5. **Generality:** An algorithm applies to different sets of same input type.

Advantages:

1. It is step-by-step solution to a given problem which is very easy to understand.
2. It has got a definite procedure.
3. It is easy to first develop an algorithm, then convert into a flowchart and then into a computer program.
4. It is easy to debug as every step has got its own logical sequence.
5. It is independent of programming languages.

Disadvantages:

1. It is time consuming and cumbersome as an algorithm is developed first which is converted into flowchart and then into a computer program
2. Algorithm lacks visual representation hence understanding the logic becomes difficult.

Examples:-

Write an algorithm to add two numbers

Step 1: Start
Step 2: Declare variables num1, num2 and sum
Step 3: Read num1 and num2
Step 4: Add num1 and num2 and assign the result to sum. sum←num1+num2
Step 5: Display sum
Step 6: Stop

Write an algorithm to find the largest among three different numbers

Step 1: Start
Step 2: Declare variables A, B and C.
Step 3: Read variables A, B and C.
Step 4: If A>B
If A>C {Display A is the largest number.}
else
Display C is the largest number.
Else If B>C

else

Step 5: Stop

Display the number

Display the number

Flowchart:-

Flowchart is a pictorial or symbolic representation of an algorithm. It indicates process of solution. They are constructed by using special geometrical symbols. Each symbol represents an activity. Flowcharts are read from top to bottom unless a branch alters the normal flow.

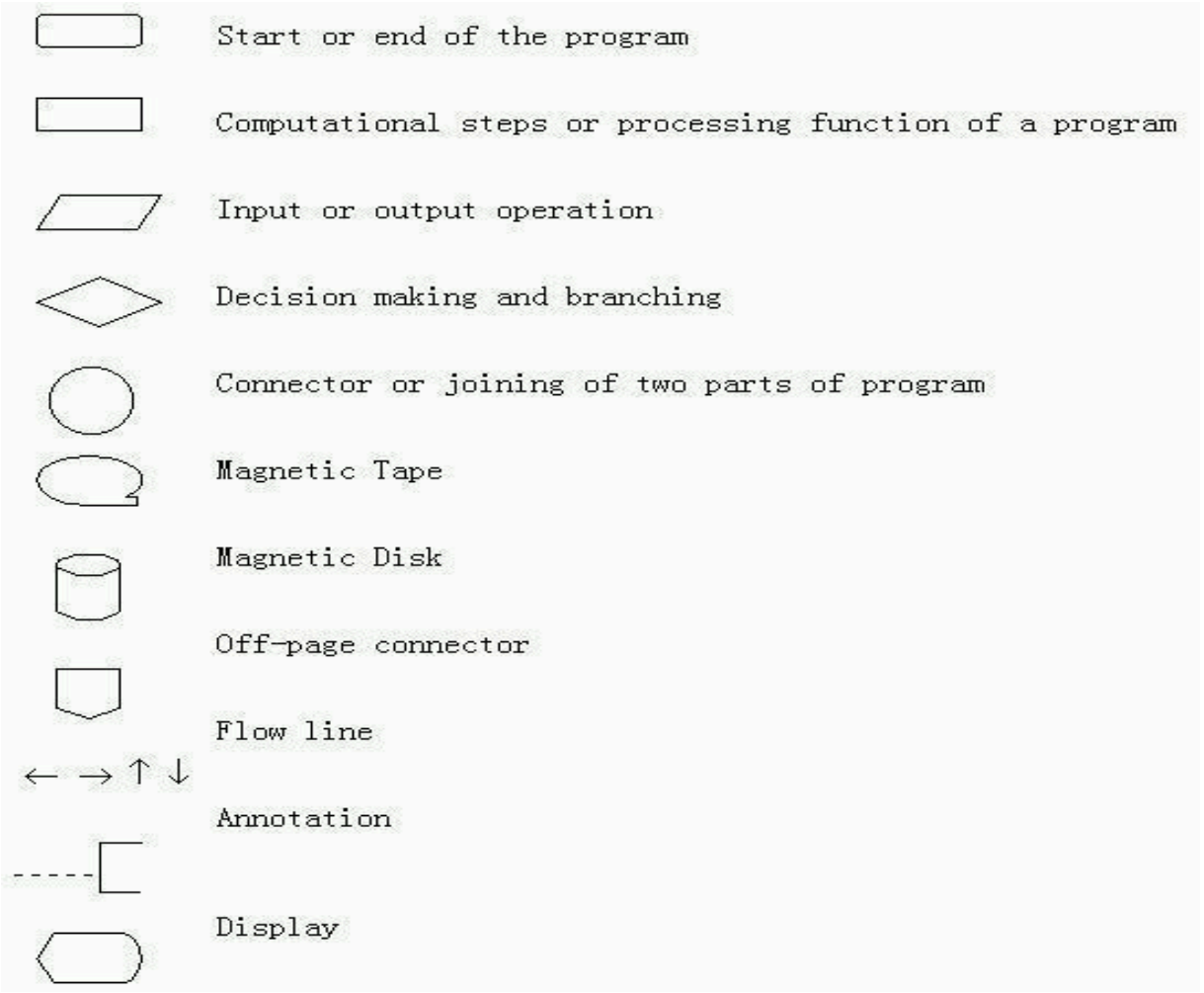
Advantages

1. It clarifies the program logic.
2. Before coding begins the flowchart assists the programmer in determining the type of logic control to be used in a program
3. The flowchart gives pictorial representation
4. Serves as documentation
5. Serves as a guide for program writing.
6. Ensure that all possible conditions are accounted for.
7. Help to detect deficiencies in the problem statement.

Disadvantages:

1. When the program logic is complex the flowchart quickly becomes complex and clumsy and lacks the clarity of decision table.
2. It alterations and modifications are required, the flowchart may require re-drawing completely.
3. As the flowchart symbols cannot be typed reproduction of flowcharts often a problem.
4. It is sometimes difficult for a business person or user to understand the logic depicted in a flowchart.

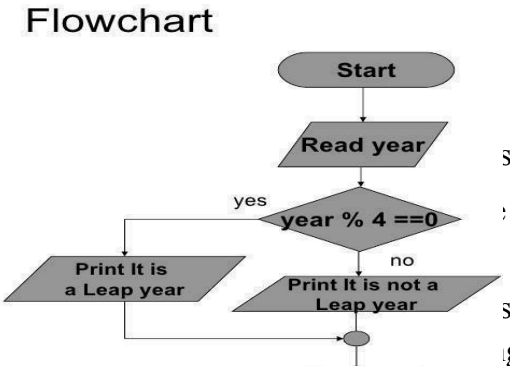
Symbols used to draw flowcharts:



Flowchart design rules:-

- ✔ It must begin with "start" and end with "stop" symbol.
- ✔ Standard process flow should be from top to bottom or from left to right.
- ✔ Instructions specified in symbols must be crisp and concise.
- ✔ Arrows to be aligned properly to depict program.
- ✔ Use of connectors is generally avoided.
- ✔ "Process or Action" symbol must have only one input arrow and one output arrow.
- ✔ Two arrows should never intersect each other if such a case arises, "bridge or crossover" symbols are used.

Example for a flowchart



Pseudo cod

It is code in English. It is a combination of generic syn seudo code is not actual programming language. It basic logic of the program.

Pseudo c

- ✔ st be kept all capitalized.
- ✔ g constructs must be labelled

Comparison between flow chart and algorithm

S. No.	Algorithm	Flow chart
1	An *algorithm* is a finite set of step-by-step instructions that solve a problem	*Flowchart* is a pictorial or symbolic representation of an algorithm
2	Algorithm gives verbal which is almost similar to English language.	Gives pictorial representation.
3	Suitable for large and modular programs	Suitable for small programs.
4	Easier to understand	To understand flowcharts one has to familiar with symbols.
5	Drawing tools are not required.	Required.
6	Algorithm can be typed so reproduction is easy.	Flowcharts cannot be typed. So reproduction is a problem.
7	It is independent of programming languages.	We have to use predefined standard symbols only.

appropriately with “end” keywords.

- ✔ Complex instructions must be avoided by writing simple and clear instructions.

Example: Add two numbers and display the result

```
DEFINE: Integer num1, num2, result
READ: Integer num1
READ: Integer num2
SET: result=num1+num2
Display: result
```

Advantages

- ✔ Developing program code using *Pseudo code*is easier.
- ✔ The program code instructions are easier to modify in comparison to a flowchart.
- ✔ It is well suited for large program design.

Disadvantages

- ✔ It is difficult to understand the program logic since it does not use pictorial representation.
- ✔ There is no standard format for developing a *Pseudo code.*

4.2 **Program development steps: -** the steps involved in the program development are:

1. Defining the problem.
2. Designing the algorithm.
3. Coding the program.
4. Testing and debugging the program.
5. Implementing the program.
6. Maintaining and updating the program.

Defining the problem

It involves recognizing the problem; identifying exactly what the problem is; determining the available inputs and desired output; deciding whether problem can be solved by using computer.

Designing the algorithm

An algorithm is finite set of step-by-step instructions that solve a problem. After defining the problem, algorithm can designed.

Coding the program

It involves actually writing the instructions in a particular language that tell the computer how to operate.

Testing and debugging the program

After coding, the program must be tested to ensure that is correct and contains no errors. Three types of errors or bugs can be found which are

i. Syntax error
ii. Logical error
iii. Runtime error

i. Syntax Error

Syntax is a set of rules by which a programming language is governed. Syntax error occurs when these rules are violated while coding of the program.

ii. Logical Error

Improper coding of individual statements either or sequence of statements causes this type of computer program. It does not stop the program execution but gives wrong results.

iii. Runtime Error

This error in a computer program stops its execution. It may be caused by an entry of invalid data.

Implementing the Program

After a program has been listed and debugged, it can be installed and implemented.

Maintaining and Upgrading the Program

After a program is developed and implemented can be upgraded as per the requirements of the user. Maintaining and upgrading the program is an ongoing process

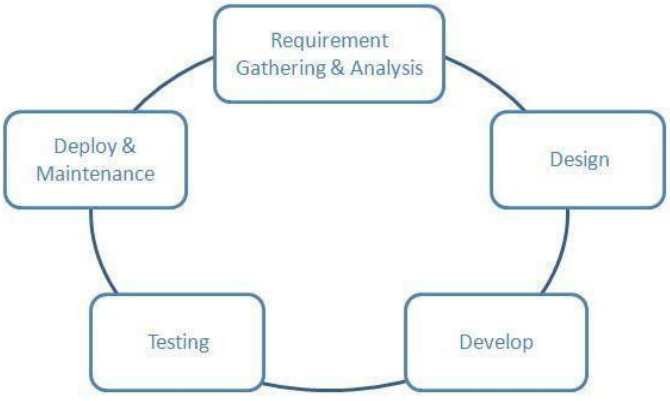
5. Software Development Life Cycle (SDLC)

Software Development Life Cycle is a process used by software industry to design, develop and test high quality software's. It is also called as Software development process. The software development life cycle (SDLC) is a framework defining tasks performed at each step in the software development process.

5.1 What is SDLC?

SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process.

The following figure is a graphical representation of the various stages of a typical SDLC.



There are following six phases in every Software development life cycle model:

1. Requirement gathering and analysis
2. Design
3. Implementation or coding
4. Testing
5. Deployment
6. Maintenance

Stage1: Requirement gathering and analysis

Business requirements are gathered in this phase. This phase is the main focus of the project managers and stake holders. Meetings with managers, stake holders and users are held in order to determine the requirements like;

- ✔ Who is going to use the system?
- ✔ How will they use the system?
- ✔ What data should be input into the system?
- ✔ What data should be output by the system?

After requirement gathering these requirements are analyzed for their validity and the possibility of incorporating the requirements in the system to be development is also studied.

Finally, a Requirement Specification document is created which serves the purpose of guideline for the next phase of the model.

Stage 2: Design

In this phase the system and software design is prepared from the requirement specifications which were studied in the first phase. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture. The system design specifications serve as input for the next phase of the model.

Stage 3: Implementation / Coding

On receiving system design documents, the work is divided in modules/units and actual coding is started. Since, in this phase the code is produced so it is the main focus for the developer. This is the longest phase of the software development life cycle.

Stage 4: Testing

After the code is developed it is tested against the requirements to make sure that the product is actually solving the needs addressed and gathered during the requirements phase. During this phase unit testing, integration testing, system testing, acceptance testing are done.

Stage 5: Deployment

After successful testing the product is delivered / deployed to the customer for their use.

Stage 6: Maintenance

Once when the customers starts using the developed system then the actual problems comes up and needs to be solved from time to time. This process where the care is taken for the developed product is known as maintenance.

5.2 SDLC Models

There are various software development life cycle models defined and designed which are followed during software development process. These models are also referred as "Software Development Process Models". Each process model follows a Series of steps unique to its type, in order to ensure success in process of software development.

Following are the most important and popular SDLC models followed in the industry:

1. Waterfall Model
2. Iterative Model
3. Spiral Model
4. Rapid Application Development and
5. Prototyping Model