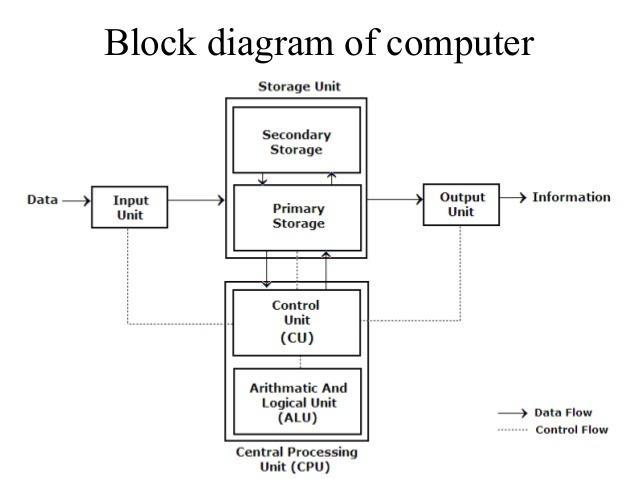
**1.3. Block diagram of computer**



The Basic components & parts of computer system are given below:

* Input Devices
* Output Devices
* CPU (Central Processing Unit)
  + CU
  + ALU
* Storage Unit
  + Primary storage unit
  + Secondary storage unit

# Input Unit:

* Input Devices accepts or receives the data or instruction from outside the world such as input devices like keyboard and mouse
* It converts the data or instruction or information into binary form for further processing.
* They act like a connection between outside the world and the computer system
* Keyboard and mouse are examples of input devices.

# Output Unit:

* The output unit display the results of the processing.
* The output unit converts the binary codes into human readable form for better understanding.
* They acts as a connection or link between outside world and computer system.
* Printers, monitors and projectors are the prime examples of output devices.

# Central Processing Unit (CPU)

The Central Processing Unit is also called the brain of a computer. It performs all the operation such as arithmetic and logic operation.

The different component of CPU is

* Arithmetic logical unit :
* Control Unit:
* Control Unit
  + control unit of a CPU controls the entire operation of a computer. It also controls all devices such as memory, input/output devices connected to the CPU.
  + CU fetches instructions from memory, decodes the instruction, interprets the instruction and sends suitable control signals to the other components to execute the instruction.
* Arithmetic Logic Unit (ALU)
  + Here arithmetic logic unit performs all arithmetic operations such as addition, subtraction, multiplication and division. It also uses logic operation for comparison.

# Storage Unit

Data and instruction enters into a computer system through input device have to stored inside the computer before actual processing start. It stores programs, data as well as intermediate results and output results.

Two types of storage unit are

* primary storage unit
* secondary storage unit.
* Primary Storage Unit

They are also called as Main memory or in other words as RAM (Random Access Memory).The Data or set of instruction is stored in primary storage before processing and later the data is transferred to ALU where further processing is done. The data store in this memory is temporary. They are very expensive.

* Secondary Storage Unit

The Secondary storage is also called a permanent storage unit as when the data store in this memory is stored permanently user can recall the data whenever they need. They are much cheaper than primary memory.

**1.4 Generation of computer**

The history of computer development characterized by a major technological development that fundamentally changed the way computers operate, resulting in smaller, cheaper, powerful and efficient computing devices.

**First generation (1940 - 1956)**

The first generation of computers used vacuum tubes as a major piece of technology. Vacuum tubes were larger components and resulted in first generation computers being quite large in size, taking up a lot of space in a room or even a house.

Example:

The ENIAC, EDSAC, IBM 701, and Manchester Mark 1.

**Second generation (1956 - 1963)**

The second generation of computers saw the use of transistors instead of vacuum tubes. Transistors were smaller than vacuum tubes and allowed computers to be smaller in size, faster in speed, and cheaper to build.

Example:

IBM 7070, RCA 501.

**Third generation (1964 - 1971)**

The third generation of computers introduced the use of IC (integrated circuits) in computers. Using IC's in computers helped reduce the size of computers even more compared to second-generation computers, and make them faster.

IC's are still used in computers today. Over 45 years later, today's computers have deep roots going back to the third generation.

Example:

IBM-360 series, Honeywell-6000 series

**Fourth generation (1972 - 2010)**

The fourth generation of computers took advantage of the invention of the microprocessor, more commonly known as a CPU. Microprocessors, with integrated circuits, helped make it possible for computers to fit easily on a desk and for the introduction of the laptop.

Example:

Altair 8800, IBM 5100.

**Fifth generation (2010 to present)**

The fifth generation of computers is beginning to use AI (artificial intelligence), an exciting technology with many potential applications around the world.

Example:

Apple's Siri, Microsoft's. The Google search engine also utilizes AI to process user searches.

**1.5. Types of Computer**

Classify on the basis of principle of operation:

* + **Analog Computer**
  + **Digital Computer**
  + **Hybrid Computer**

Classify on the basis of size:

* + **Micro Computer**
  + **Mini Computer**
  + **Mainframe Computer**
  + **Super Computer**

**On the basis of principle of operation:**

**Analog Computer**

Analog Computer is a computing device that works on continuous range of values. It deals with physical variables such as voltage, pressure, temperature, speed, etc. Speedometer and mercury thermometer are examples of analog computers.

**Digital Computer**

The digital computer is designed using digital circuits in which there are two levels for an

input or output signal. These two levels are known as logic 0 and logic 1.

Digital computer are used in the field of design, research and data processing. All modern computers like laptops, desktops including smartphones that we use at home or office are digital computers.

**Hybrid Computer**

Hybrid computer has features of both analog and digital computer. It is fast like an analog computer and has memory and accuracy like digital computers. A processor is used in petrol pumps that converts the measurements of fuel flow into quantity and price. Similarly, they are used in airplanes, hospitals, and scientific applications.

**On the basis of size**

**Mainframe computer**

Mainframe computers are designed to support multiple programs at the same time. It means they can execute different processes simultaneously. These features of mainframe computers make them ideal for big organizations like banking and telecom sectors, which need to manage and process a high volume of data.

**Minicomputer**

It is a midsize multiprocessing computer. Mini computers are used for tasks billing, accounting and inventory management. A minicomputer lies between the mainframe and microcomputer as it is smaller than mainframe but larger than a microcomputer.

**Microcomputer**

Microcomputer is also known as a personal computer. It is a general-purpose computer that is designed for individual use. It has a microprocessor as a central processing unit, memory, storage area, input unit and output unit. Laptops and desktop computers are examples of microcomputers.

**Supercomputer:**

Supercomputers are the biggest and fastest computers. They are designed to process huge amount of data. Supercomputers are used in scientific and engineering applications such as weather forecasting, scientific simulations and nuclear energy research.

**1.6. Types of Software**

Various types of computer software are used to simplify the operations and applications of

computer programs.

Basically, there are only 2 main types of Software.

* + **System Software**
  + **Application Software**

Besides these, software can also be categorized depending on the nature of use.

* + **Ready-made (of-the-shelf) Software**
  + **Tailored (custom) Software**

Also, we can categorize software as:

* + **Open Source Software**
  + **Closed Source (Proprietary) Software**

**System Software**

Offers a protective shield to all software applications, physical components of computers. System software sits directly on top of computer's hardware components . This includes the operating system, drivers for your hardware devices, linkers and debuggers.

**Application Software**

Application software is used for commercial purpose. Applications software, sits on top of system software, as it is unable to run without the operating system and other utilities.

Computer games, Industrial automation, databases, business software and medical software are examples of application software.

**Ready-made (off-the-shelf) Software**

These types of software are made and distributed by large enterprise companies. They are

made with a general purpose requirement in mind. These kind of software may

need to adjust in few features and usability.

Advantages:

* Lower up-front cost
* Support is often included or can be added with a maintenance contract

Disadvantages:

• Slow to adapt or change to industry needs

• May require you to change your process to fit the software

• Higher customization fees

**Tailored (custom) Software**

These types of software are made with a specific customer in mind.

Advantages:

• You can start with the minimum necessary requirements and add on later

• Can be tailored to your exact business needs and processes

• Changes can be made quickly

Disadvantages:

• Very high initial cost

• May occur additional costs for hiring new developers

**Open Source Software**

These types of software provide have their source code available publicly. They are

maintained and contributed by people all over the world. They are generally free of cost as well, but not all of the open source software are free of cost.

Examples:

Linux Kernel, Firefox Web Browser, Apache Web Server etc.

**Closed Source (Proprietary) Software**

These types of software do not provide their source code to the general public.

They are generally controlled by a particular enterprise company. These software generally charge money for license or subscription. But, there can be closed source software that are free of cost, they are called Freewares.

Examples:

Microsoft Windows, Internet Explorer, Skype, etc.

**1.7. Types of Programming Languages**

Programming language is classified into three categories.

* **Machine level Language**
* **Assembly level Language**
* **High-level Language**

**Machine level languages:**

Machine level language consist only two condition i.e. either true (1) or false (0);

Advantages:

* Machine level languages are directly interacting with computer system.
* There is no requirement of software of conversion like compiler or interpreters.
* It takes very less time to execute a program, because there is no conversion take place.

Disadvantages:

* Its machine dependent language i.e. individual program required for each machine.
* Its time consuming to develop new programs.
* Debugging process is very hard .

**Assembly level languages:**

It contains the same instruction as machine level language, but the instructions and the variables have specific name or called commands instead of being just binary numbers.

Advantages:

* It is easily understood by human because it is uses statements instead of binary digits.
* To develop a program it takes less time.
* Debugging and troubleshoot is easy due to easily find error.

Disadvantages:

* It’s a machine dependent language.
* Sometime it’s hard to understand the statement or command use.

**High-level language:**

High level languages are understandable by humans. C, C++, JAVA etc are example of high level language.

Advantages:

* In this instructions and commands much easier to remember by programmer.
* Its logic and structure are much easier to understand.
* Debugging is easier compare to other languages.
* Less time consuming to writing new programs.

Disadvantages:

* It takes more space compare to machine level language and Assembly level language.
* This programming language execute slowly.

**1.8 Language translator or language processor:**

Assembly language is machine dependent yet mnemonics that are being used to represent instructions in it are not directly understandable by machine and high Level language is machine independent. A computer understands instructions in machine code, i.e. in the form of 0s and 1s. It is a tedious task to write a computer program directly in machine code. The programs are written mostly in high level languages like Java, C++, Python etc. and are called **source code**. These source code cannot be executed directly by the computer and must be converted into **machine code** to be executed. Hence, a special translator system software is used to translate the program written in high-level language into machine code is called Language Processor and the program after translated into machine code (object program / object code).

The language processors can be any of the following three types:

**Compiler**

The language processor that reads the complete source program written in high level language as a whole in one go and translates it into an equivalent program in machine language is called as a Compiler. In a compiler, the source code is translated to object code successfully if it is free of errors. The compiler specifies the errors at the end of compilation with line numbers when there are any errors in the source code. The errors must be removed before the compiler can successfully recompile the source code again.

**Example: C, C++, C#, Java**

Lightbox

**Assembler**

The Assembler is used to translate the program written in Assembly language into machine code. The source program is a input of assembler that contains assembly language instructions. Assembler converts source code to an object code first then it converts the object code to the machine language with the help of linker programs. The output generated by assembler is the object code or machine code understandable by the computer.

Lightbox

**Interpreter**

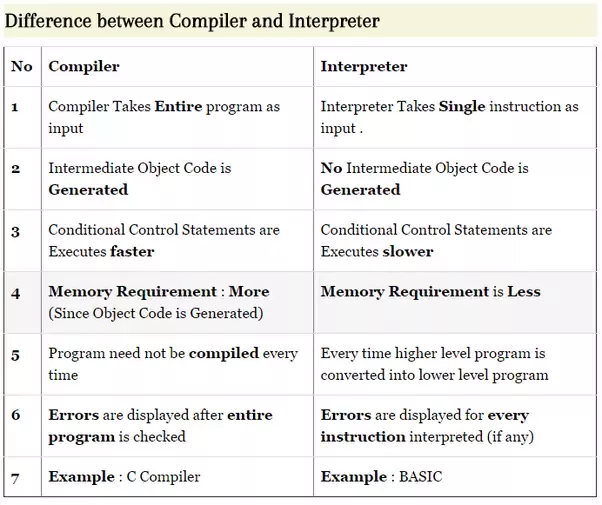
The translation of single statement of source program into machine code is done by language processor and executes it immediately before moving on to the next line is called an interpreter. If there is an error in the statement, the interpreter terminates its translating process at that statement and displays an error message. The interpreter moves on to the next line for execution only after removal of the error. An Interpreter directly executes instructions written in a programming or scripting language without previously converting them to an object code or machine code.

**Example: Perl, BASIC and Matlab.**

Source Code(High Level Language) one line at a time

Object Code(Machine Language)

Interpreter



**1.9. Traditional (Structured) Programming Concepts procedure oriented programming (POP)**

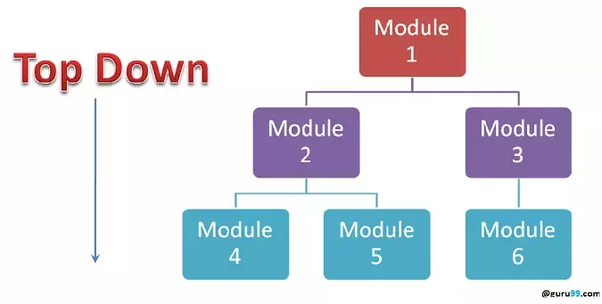
In pop large program is divided into smaller programs. Programs are written as a sequence of procedures(functions).

**Features of Structured Programming:**

* each procedure contains a series of instructions for performing a specific task.
* During the program execution, each procedure can be called by other procedures.
* To call the procedure, we have to write procedure name only.
* Major emphasis of these languages is on the procedures and not on the data.
* Pop languages allow data to move freely around the system.
* Top down program approach is used by POP languages
* Eg: COBOL, FORTRAN, C, ALGOL, BASIC.

**The Top-Down Approach**

In the top-down approach, a complex algorithm is broken down into smaller fragments, better known as ‘modules.’ These modules are then further broken down into more smaller fragments until they can no longer be fragmented. This process is called ‘modularization.’ During the modularization process, you must always maintain the originality of the algorithm. In this approach, each function in a code is unique and works independently of other function.



2. PROGRAMMING LOGIC

Stages involved in Problem solving or Software Development Life Cycle(SDLC)

1. Problem Definition

2. Requirement analysis

3. design

a. algorithm

b. flowchart

4.Coding

5. compilation and execution

6. debugging and testing

7. delivery and maintenance

8. Documentation

a.documentation for programmer

b. user manual

# Problem definition:

It is also called problem analysis. In this step, we should define problem in understandable way without ambiguity or confusion.

# Requirement analysis:

In this phase we should specify following requirements:

* Objectives
* Input requirement
* Output requirement
* Processing requirement
* Feasibility evaluation

# Design

This can be done through:

1. **Algorithm**

Set of instruction written in simple English language to solve a particular problem.

1. **Flowchart**

The pictorial/ graphical representation of an algorithm is called flowchart.

# Coding

Each programming language has its own syntax. So we should follow rules and syntax properly while coding. The statements should be arranged and comments should be added whenever necessary.

# Compilation and execution:

The program needs to be translated using some translator in a machine code which is called compilation. Compiler checks syntax error in the program. The compilation figure is shown below.

Source code

Example.c

compiler

Object code

Example.obj

Other library files

linker

Executable code

Example.exe

run

Fig: compilation process

# **Debugging and testing**

Debugging is the process of detecting or finding the errors and removing it.

Testing is the process of checking the desired output for number of inputs.

# Delivery and maintenance

After the desired product has been developed, it needs to be delivered to concerned authority.

The software needs frequent maintenance or update to address problem arising in the future.

# Documentation

It starts with very beginning of the project and ends with delivery of the product.

1. **Documentation for programmer**

It is required for other developers working on the project for future.

1. **User manual**

A user may need a manual to operate a software system.

## Algorithm and flowchart

**Algorithm:**

It is a set of instructions or steps written in simple **English language** to perform a particular task.

**Features of algorithm:**

* Should be written in simple English language.
* Should be understandable and clear to any user
* Should be language independent (should not depend on any particular programming language)
* It should consist of finite number of steps.
* It should not contain any ambiguous instructions or any confusing statements.
* It should terminate after finite number of steps.

Example: The algorithm to add two numbers can be stated as:

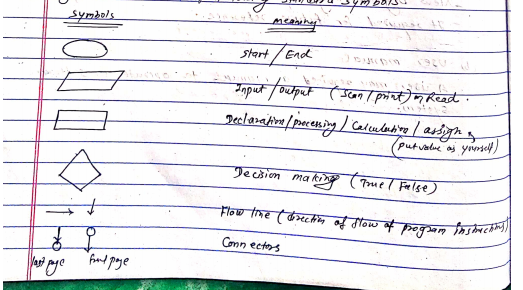
Step1: Input numbers as a and b

Step2: Sum = x + y

Step3: Print the sum

**Flowchart:**

It is a pictorial representation of an algorithm. The flowcharts are helpful in understanding the logic of complicated and lengthy problems. Once the flowchart is drawn it becomes easy to write the program in any high-level language. It contains following standard symbols.



Questions on algo and flowchart. Refer class note.