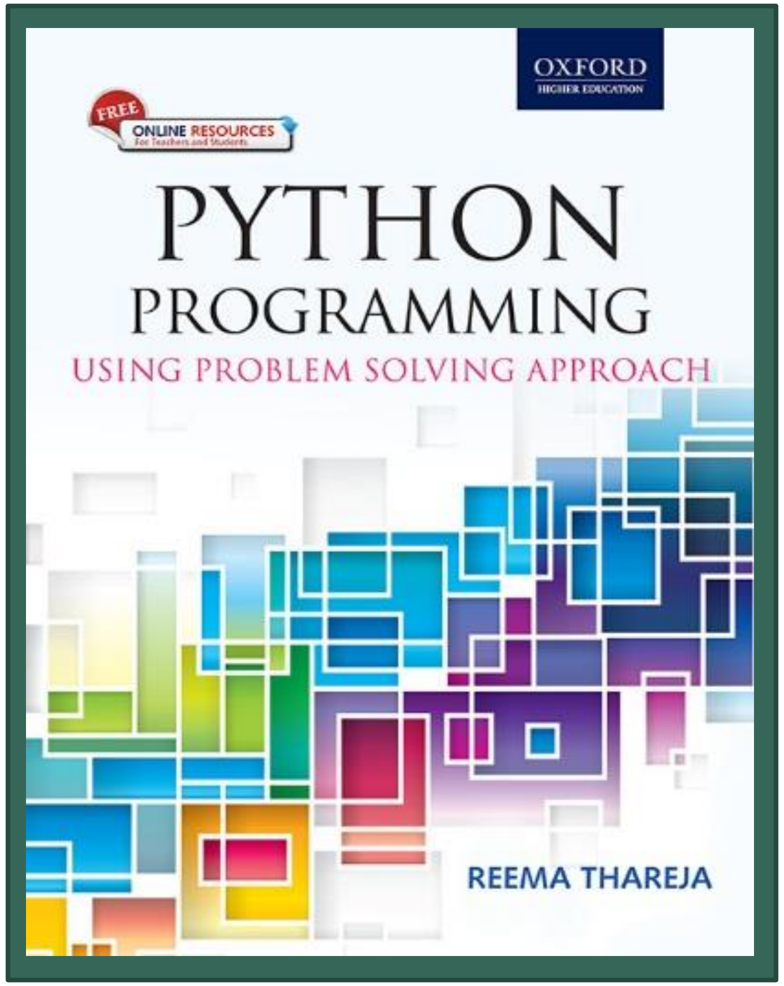
Python Programming

Using Problem Solving Approach

Reema Thareja



CHAPTER 1

Introduction to Computers and Problem Solving Strategies

What is a Computer?

A **computer** is an electronic machine that takes instructions and performs computations based on those instructions.

Data is a collection of raw facts or figures.

Information comprises processed data to provide answers to *'who'*, *'what'*, *'where'*, and *'when'* type of questions.

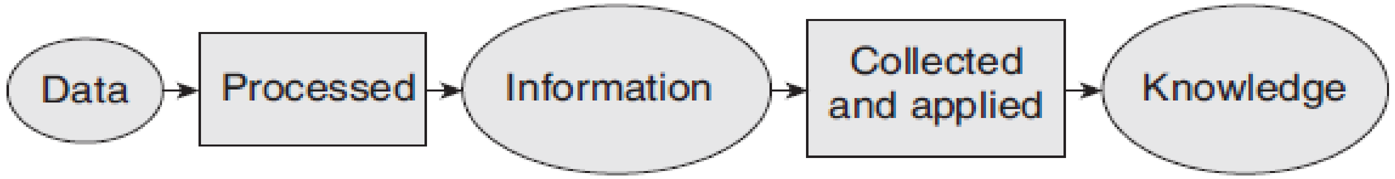
Knowledge is the application of data and information to answer *'how'* part of the question.

Instructions: Commands given to the computer that tells what it has to do are instructions.

Programs: A set of instructions in computer language is called a program.

Software: A set of programs is called software.

Hardware: A computer and all its physical parts are known as hardware.



First Generation of Computers (1942 – 1955)

Hardware Technology: Manufactured using thousands of vacuum tubes.

Software Technology: Programming was done in machine language or assembly language.

Used for: Scientific applications

Examples: ENIAC, EDVAC, EDSAC, UNIVAC I, IBM 701

Highlights: • They were the fastest calculating device of those times.

• Computers were too bulky and required a complete room for storage.

• Highly unreliable as vacuum tubes emitted a large amount of heat and burnt frequently.

• Required air-conditioned room for installation.

• Costly

• Difficult to use

• Required constant maintenance

Second Generation of Computers (1955 – 1964)

Hardware Technology: Manufactured using thousands of transistors.

Software Technology: Programming was done in high level language.

Used for: Scientific and commercial applications

Examples: Honeywell 400, IBM 7030, CDC 1604

Highlights: • Faster, smaller, cheaper, reliable, and easier to use than the first generation computers.

• Consumed 1/10th the power consumed by first generation computers.

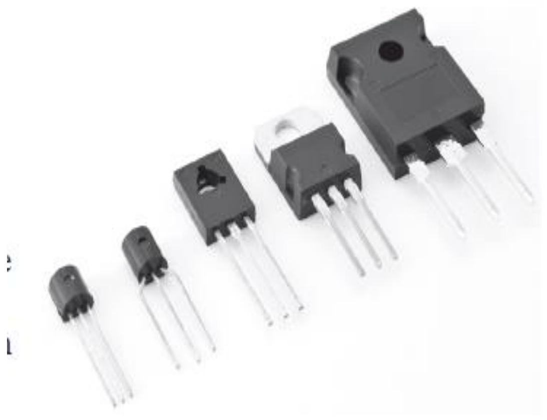
• Bulky in size and required a complete room for its installation.

• Dissipated less heat than first generation computers but still required Ac room

• Costly

• Difficult to use

• Required constant maintenance



Third Generation of Computers (1964 – 1975)

Hardware Technology: Manufactured using thousands of integrated circuits.

Software Technology: Programming was done in high level language.

Used for: Scientific, commercial and interactive applications

Examples: IBM 360/370, PDP-8, PADP-11, CDC6600

Highlights: • Faster, smaller, cheaper, reliable, and easier to use than the second generation computers.

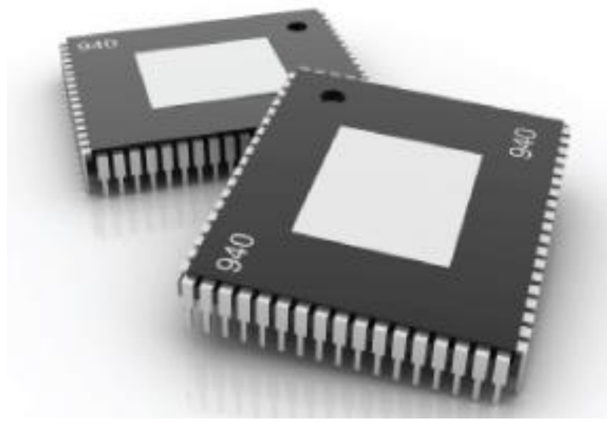
• They consumed less power than second generation computers.

• Bulky in size and required a complete room for its installation.

• Dissipated less heat than second generation computers but still required AC room.

• Costly

• Easier to use and upgrade.



Fourth Generation of Computers (1975 – 1989)

Hardware Technology: Manufactured using Large Scale and Very Large Scale integrated circuits.

Software Technology: Programming was done in high level language.

Used for: Scientific, commercial, interactive and network applications

Examples: IBM PC, Apple II, TRS-80, VAX 9000, CRAY-1, CRAY-2, CRAY-X/MP

Highlights: Faster, smaller, cheaper, powerful, reliable, and easier to use than the previous generation computers

Fifth Generation of Computers (1975 – 1989)

Hardware Technology: Manufactured using Ultra Large Scale integrated circuits. Use of Internet became widespread. Very powerful mainframes, desktops, portable laptops, smartphones are being used commonly. Super computers use parallel processing techniques.

Software Technology: Programming was done in high level language.

Used for: Scientific, commercial, interactive, network and multimedia applications

Examples: IBM notebooks, Pentium PCs, SUM workstations, IBM SP/2, Param supercomputer

Highlights: • Faster, smaller, cheaper, powerful, reliable, and easier to use.

• Speed of microprocessors and the size of memory are growing rapidly.

• High-end features available on mainframe computers in the fourth generation are now available on the microprocessors.

• Consume less power than computers of prior generations.

• Air-conditioned rooms required for mainframes and supercomputers but not for microprocessors.

Characteristics of Computer

Speed: The speed of the computer is usually given in *nano second* and *pico second*, where

1 nano second = 1 × 10^{-9} second and 1 pico second = 1 × 10^{-12} second

Accuracy: Computer never makes mistakes. It always gives accurate results provided that correct data and set of instructions are input to it.

Automatic: Computers are automatic devices that can perform without any user intervention

Diligence: Computer never gets tired. It can continually work for hours without creating any error.

Versatile: Versatile means flexible. Today, computers are being used in our daily lives in different fields.

Memory: Computers have internal memory (storage space) as well as external or secondary memory.

No IQ: Computers do not have any decision-making abilities of their own, i.e., their IQ level is zero.

Economical: Computers are considered as short-term investment for achieving long-term gain.

Classification of Computer

Supercomputers: Supercomputer is the fastest, most powerful, and most expensive computer. A single supercomputer can support thousands of users at the same time. They are mainly used for weather forecasting, nuclear energy research, aircraft design, automotive design, online banking, controlling industrial units, etc. Some examples of supercomputers are CRAY-1, CRAY-2, Control Data CYBER 205, and ETA A-10.

Mainframe Computers: Mainframe computers are large-scale computers. They are very expensive and need a very large clean room with air conditioning. Mainframes can also support multiple processors. They are typically used as servers on the World Wide Web. They are also used in organizations such as banks, airline companies, and universities.

Minicomputers: Minicomputers are smaller, cheaper, and slower than mainframes. Minicomputers are used in business, education, hospitals, government organizations, etc. While some minicomputers can be used only by a single user, others are specifically designed to handle multiple users simultaneously. Minicomputers can also be used as servers in a networked environment, and hundreds of PCs can be connected to it.

Classification of Computer

Microcomputers: Microcomputers or PCs are very small and cheap. They can be classified into the following categories:

- **Desktop** can be placed flat on a desk or table. It is widely used in homes and offices.
- **Laptops** are small microcomputers that can easily fit inside a briefcase. They are very handy and can easily be carried from one place to another.
- **Workstations** are single-user computers. Their processing speed matches that of a minicomputer or mainframe. They have advanced processors, more RAM and storage capacity than PCs. Therefore, they are more expensive and powerful than a normal desktop computer.
- **Network Computers** have less processing power, memory, and storage than a desktop. They are designed to be used as terminals in a networked environment. For example, to access data stored on a network.
- **Handheld Computers** include Smartphones, Tablet PCs and Phablets.

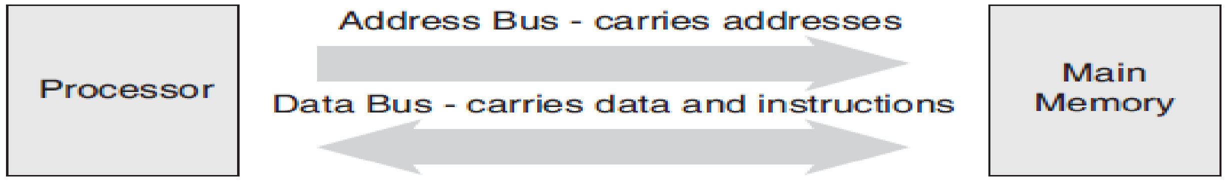
Basic Applications of Computer

- Communication
- Desktop Publishing
- Government
- Traffic Control
- Legal System
- Retail Business
- Sports
- Music
- Movies
- Travel and Tourism
- Business and Industry
- Hospitals
- Simulation
- Geology
- Astronomy
- Education
- Weather Forecasting
- Online Banking
- Industry and Engineering
- Robots
- Decision Support Systems
- Expert System
- Find jobs on the Internet
- Find a suitable match for a boy or girl
- Read news and articles online
- Find one's batchmates
- Send and receive greetings

Stored Program Concept

All digital computers are based on the principle of stored program concept, which was introduced by Sir John von Neumann in the late 1940s. The following are the key characteristic features of this concept:

- Before any data is processed, instructions are read into memory.
- Instructions are stored in the computer's memory for execution.
- Instructions are stored in binary form (using binary numbers—only 0s and 1s).
- Processing starts with the first instruction in the program, which is copied into a control unit circuit. The control unit executes the instructions.
- Instructions written by the users are performed sequentially until there is a break in the current flow.
- Input/ Output and processing operations are performed simultaneously. While data is being read/written, the central processing unit (CPU) executes another program in the memory that is ready for execution.


Processor
Address Bus - carries addresses
Data Bus - carries data and instructions
Main Memory

Components and Functions of Computer System

Input: process of entering data and instructions (also known as programs) into the computer system.

Storage: Process of saving data and instructions permanently in the computer so that it can be used for processing.

Primary Storage also known as the main memory is that storage area which is directly accessible by the **CPU** at a very fast speed. It is used to store the data and program, the intermediate results of processing and the recently generated results.

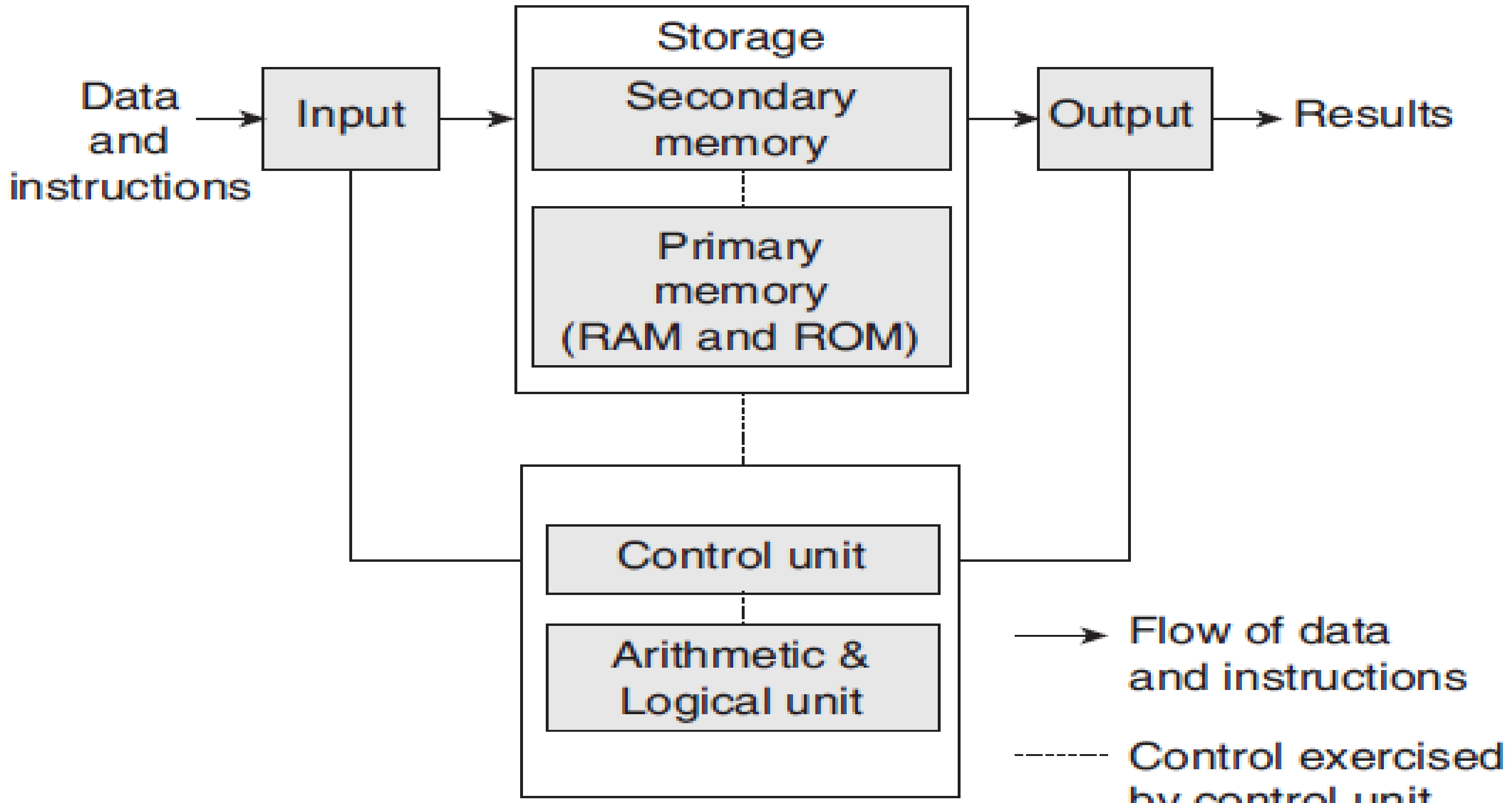
Secondary Storage also known as secondary memory is cheaper, non-volatile and used to permanently store data and programs of those jobs which are not being currently executed by the CPU.

Processing: The process of performing operations on the data as per the instructions specified by the user (program) is called processing.

Output: Process of giving the result of data processing to the outside world (external to the computer system).

Controlling: The function of managing, coordinating, and controlling all the components of the computer system is handled by the control unit, a part of CPU.

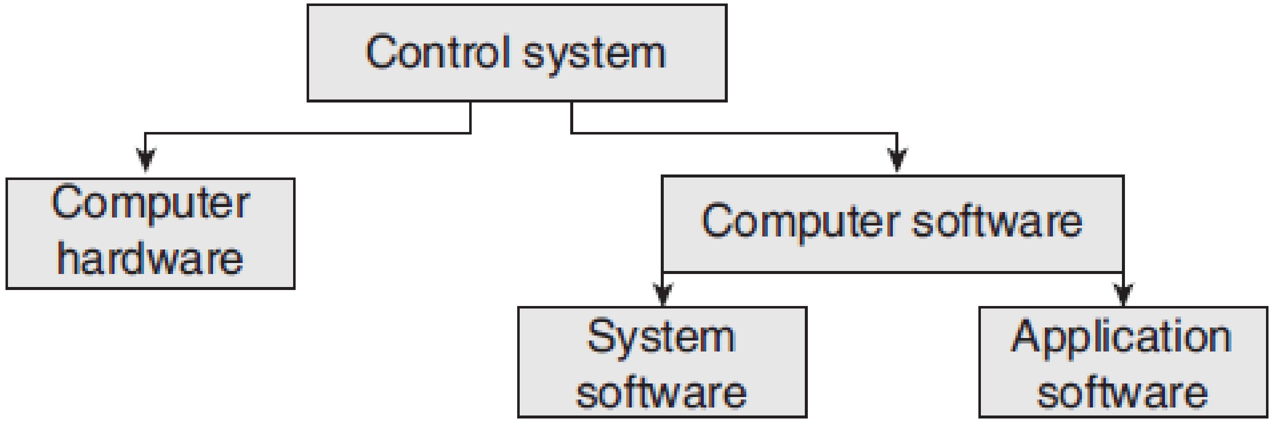
Components and Functions of Computer System


Data and instructions
Input
Storage
Secondary memory
Primary memory (RAM and ROM)
Output
Results
Control unit
Arithmetic & Logical unit
Flow of data and instructions
Control exercised by control unit

Concept of Hardware and Software

Hardware: All the physical parts that can be touched are called hardware.

Software: The hardware needs a software (a set of programs) to instruct what has to be done. A program is a set of instructions that is arranged in a sequence to guide a computer to find a solution for the given problem. The process of writing a program is called *programming*.



CPU Architecture

ALU: It performs all kinds of calculations, such as arithmetic, comparison and other operations.

Control Unit: It directs and coordinates the computer operations. It interprets the instructions (program) and initiates action to execute them.

Register: It is a computer memory that provides quick access to the data currently being used for processing. The ALU stores all temporary results and the final result in the processor registers

Accumulator and general-purpose registers: These are frequently used to store the data brought from the main memory and the intermediate results during program execution.

MAR stores the address of the data or instruction to be fetched from the main memory.

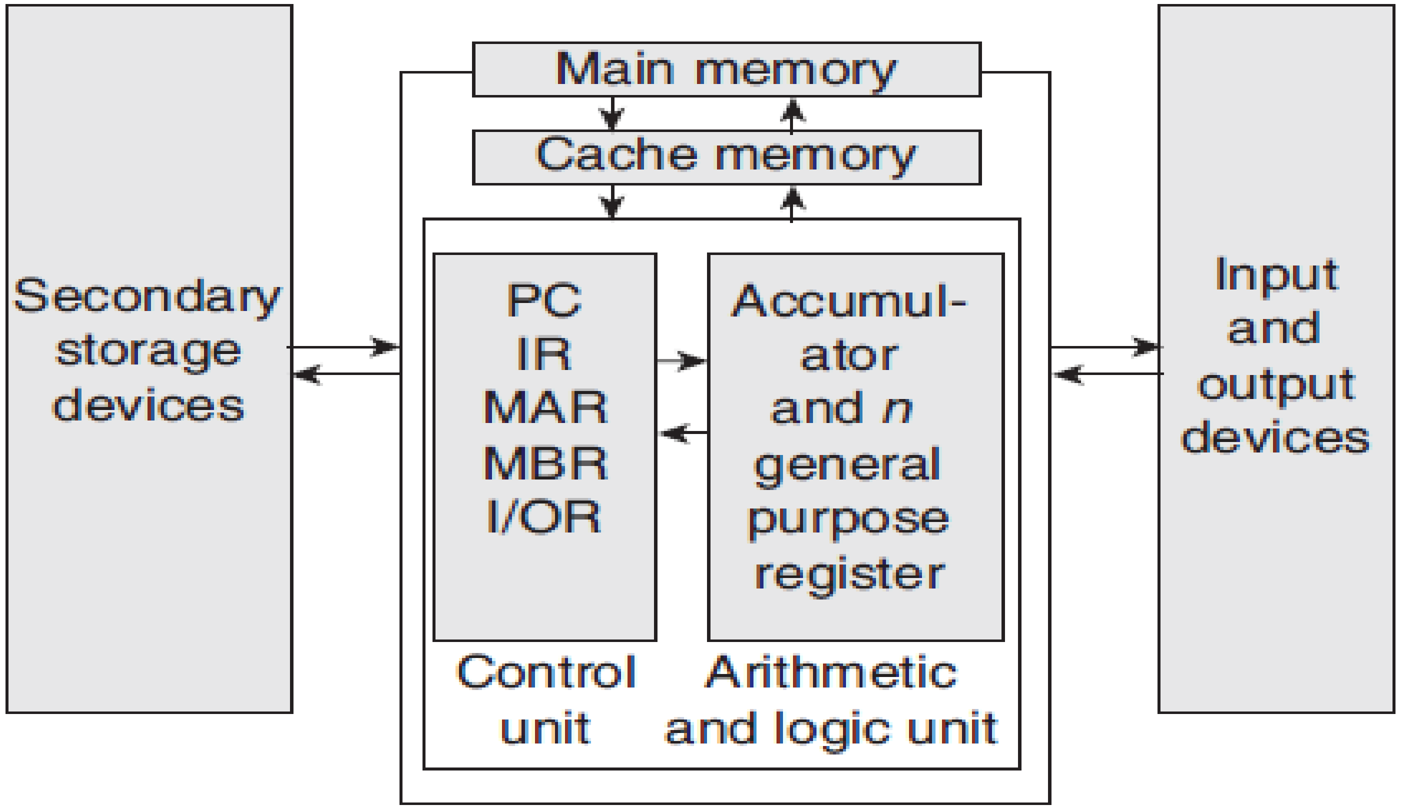
MBR stores the data or instruction fetched from the main memory

IR stores the instructions currently being executed.

I/O register is used to transfer data or instructions to or from an I/O device.

Program counter stores the address of the next instruction to be executed.

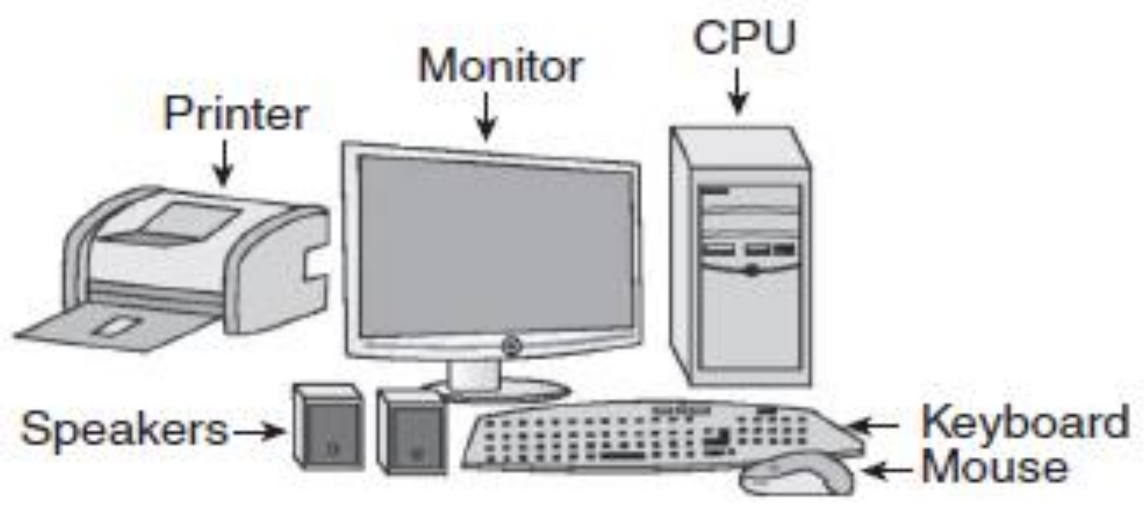
CPU Architecture


Main memory
Cache memory
Secondary storage devices
PC
IR
MAR
MBR
I/OR
Accumul-ator and *n* general purpose register
Input and output devices
Control unit
Arithmetic and logic unit

Input Devices

An input device is used to feed data and instructions into the computer.

Some of the input devices that are widely used by computer users to feed data or instruction to the computer are keyboard, mouse, trackball, joystick, stylus, touch screen, barcode reader, optical character recognition (OCR) device, optical mark recognition (OMR), MICR, web and digital cameras, etc.



Output Devices

Any device that outputs/gives information from a computer is called an *output device*.

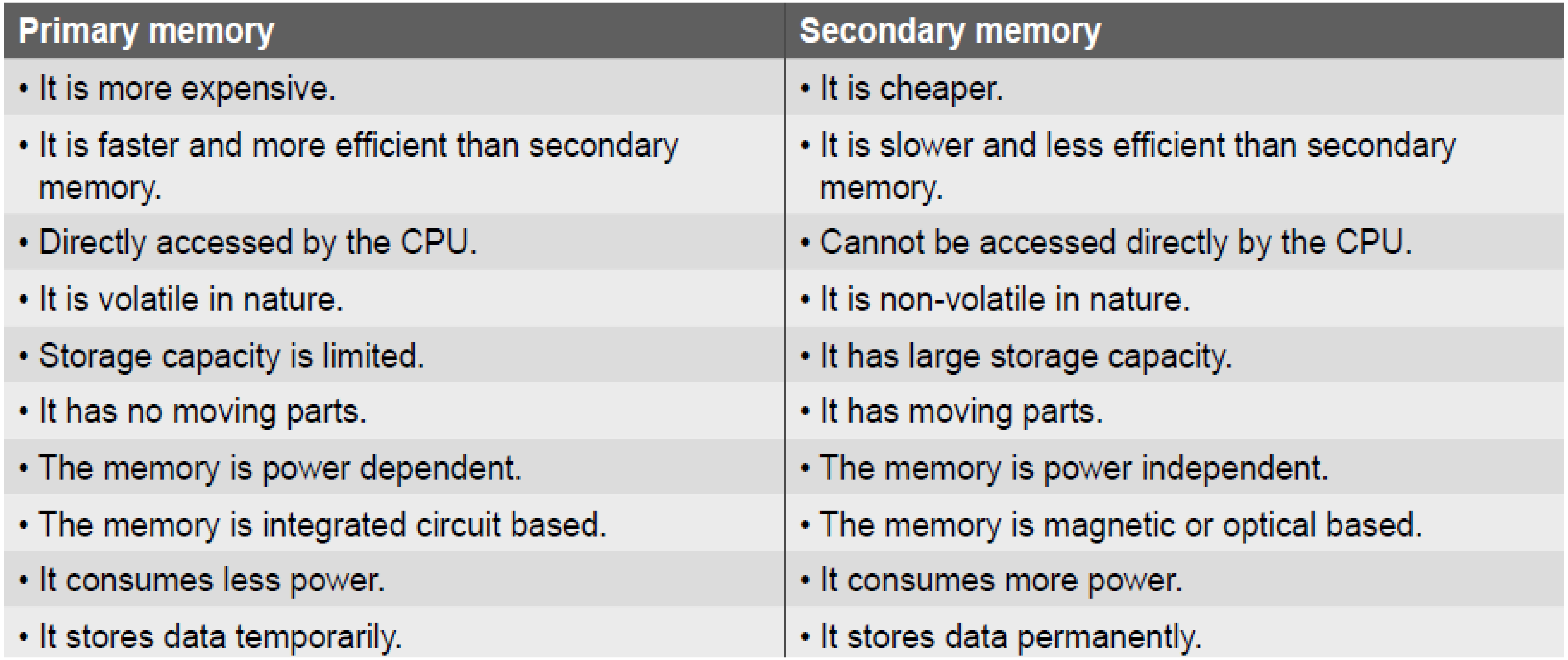
Soft copy output devices **are those O/P devices which produce an electronic version of an output. Features of a soft copy O/P are:**

- **The output can be viewed only when the computer is switched On.**
- **The user can easily edit the soft copy output.**
- **Soft copy cannot be used by people who do not have a computer.**
- **Searching data in a soft copy is easy and fast.**
- **Electronic distribution of a soft copy is cheaper. It can be done easily and quickly.**

Hard copy output devices **are those O/P devices which produce a physical form of output. Features of a hard copy • Computer is not needed to see the output.**

- **Editing and searching the hard copy is difficult.**
- **Hard copy output can be easily distributed to people who do not have a computer.**
- **Distribution of a hard copy is not only costly but also slower.**

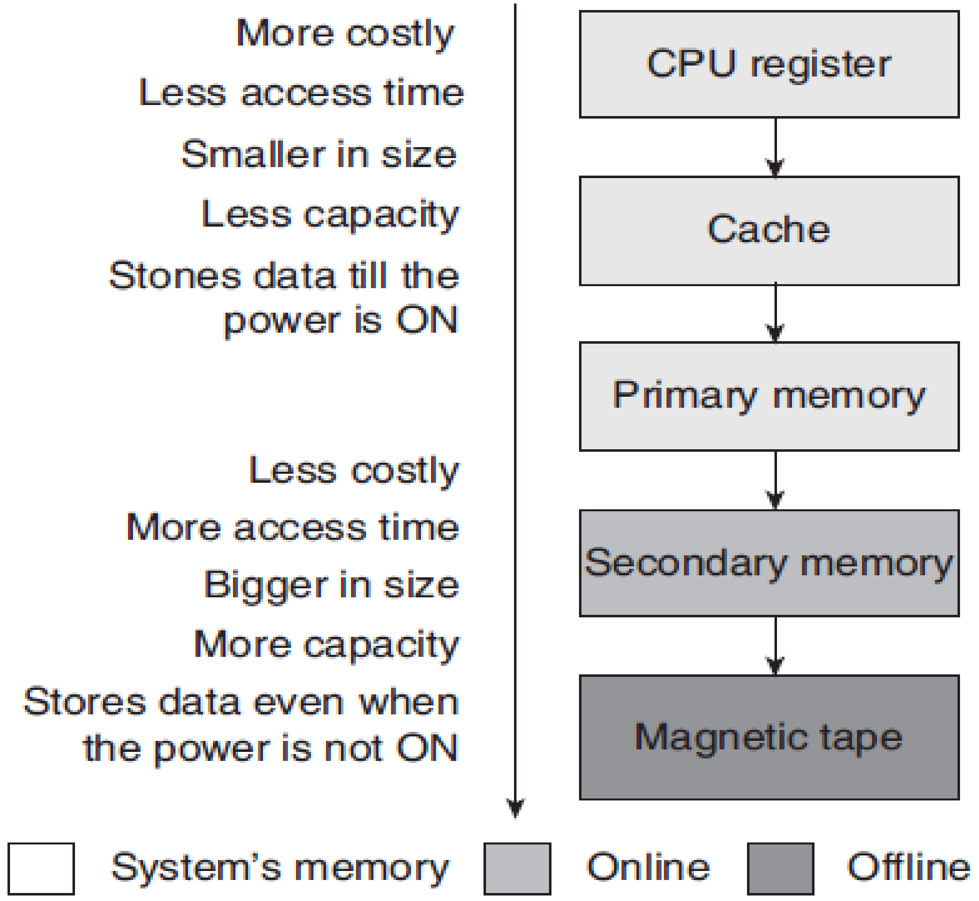
Memory



Primary memory	Secondary memory
• It is more expensive.	• It is cheaper.
• It is faster and more efficient than secondary memory.	• It is slower and less efficient than secondary memory.
• Directly accessed by the CPU.	• Cannot be accessed directly by the CPU.
• It is volatile in nature.	• It is non-volatile in nature.
• Storage capacity is limited.	• It has large storage capacity.
• It has no moving parts.	• It has moving parts.
• The memory is power dependent.	• The memory is power independent.
• The memory is integrated circuit based.	• The memory is magnetic or optical based.
• It consumes less power.	• It consumes more power.
• It stores data temporarily.	• It stores data permanently.

Memory Hierarchy

Cache memory is an intermediate form of storage between registers and the primary memory. It is used to store instructions and data that are repeatedly required to execute programs thereby improving the overall system speed and increase the performance of the computer. Keeping frequently accessed data and instructions in the cache avoids accessing the slower primary memory.



Primary Memory - RAM

RAM is a volatile storage area within the computer typically used to store data temporarily so that it can be accessed by the CPU. The information stored in RAM is loaded from the computer's hard disk, and includes data related to the operating system and applications that are currently being executed by the processor. There are two types of RAM

Static RAM: This is a type of RAM holds data without an external refresh as long as it is powered. SRAM is made of D flip-flops in which the memory cells flip-flop between 0 and 1 without the use of capacitors. Therefore, there is no need for an external refresh process to be carried out. SRAM occupies more space and is more expensive than DRAM

Dynamic RAM: This is the most common type of memory used in personal computers, workstations, and servers today. A DRAM chip contains millions of tiny memory cells. Each cell is made up of a transistor and a capacitor, and can contain 1 bit of information—0 or 1. To store a bit of information in a DRAM chip, a tiny amount of power is put into the cell to charge the capacitor. Hence, while reading a bit, the transistor checks for a charge in the capacitor. If a charge is present, then the reading is 1; if not, the reading is 0.

Primary Memory - ROM

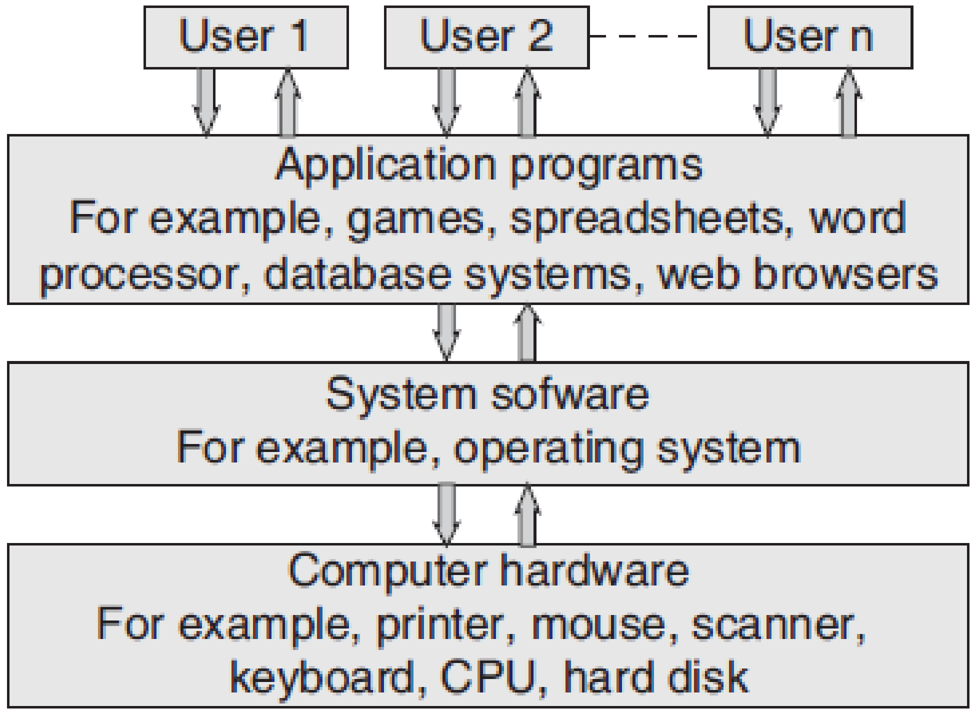
ROM is non-volatile, that is, the data is retained in it even when the computer is turned off.

Rewritable ROM chips include PROMs, EPROMs, and EEPROMs.

- ***Programmable read-only memory (PROM)*** also called one-time programmable ROM can be written to or programmed using a special device called a PROM programmer. The working of a PROM is similar to that of a CD-ROM recorder which enables the users to write programs just once but the recorded data can be read multiple times. Programming a PROM is also called ***burning***.
- ***Erasable programmable read-only memory (EPROM)*** is a type of ROM that can be erased and re-programmed. The EPROM can be erased by exposing the chip to strong ultraviolet light typically for 10 minutes or longer and then rewritten with a process that again needs higher than usual voltage applied.
- ***Electrically erasable programmable read-only memory (EEPROM)*** allows its entire or selected contents to be electrically erased, then rewritten electrically. The process of writing an EEPROM is also known as ***flashing***.

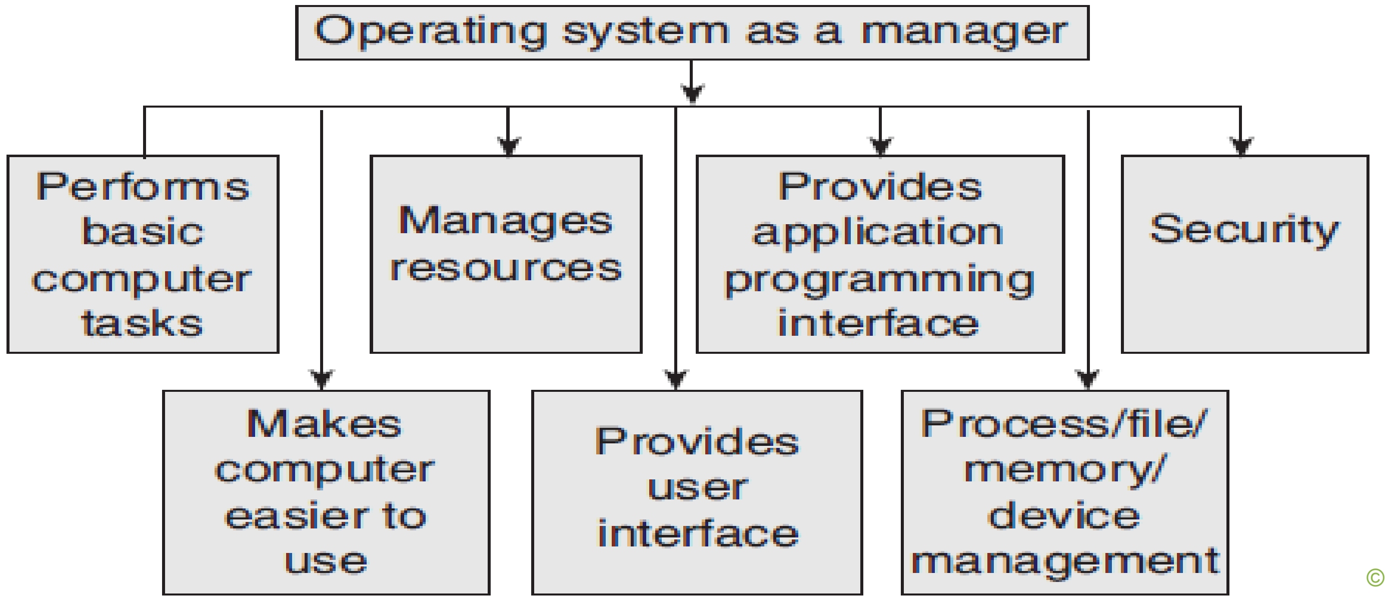
Classification Of Computer Software

- Driver software
- Educational software
- Media players and media development software
- Productivity software
- Operating systems software
- Computer games
- Application software
- System software


User 1
User 2
User n
Application programs
For example, games, spreadsheets, word processor, database systems, web browsers
System sofware
For example, operating system
Computer hardware
For example, printer, mouse, scanner, keyboard, CPU, hard disk

Operating System

An operating system is a group of computer programs that controls the computer's resources such as CPU, memory, I/O devices, etc. and provides the users with an interface that makes it easier to use. The primary goal of an operating system is to make the computer system (or any other device in which it is installed, such as a cell phone) convenient and efficient to use. It provides users an environment in which a user can execute programs conveniently and efficiently. It is the most important software in a computer system.


Operating system as a manager
Performs basic computer tasks
Manages resources
Provides application programming interface
Security
Makes computer easier to use
Provides user interface
Process/file/ memory/ device management

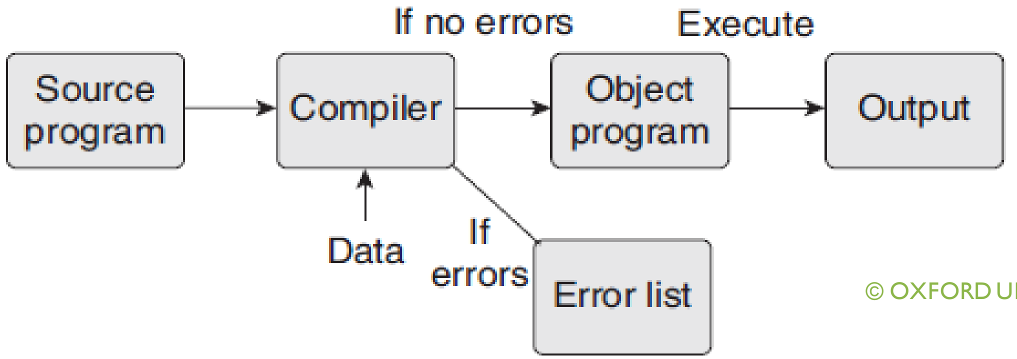
Translator

Compiler

A compiler is a special type of program that transforms the source code written in a programming language (the source language) into machine language, which uses only two digits—0 and 1 (the target language). The resultant code in 0s and 1s is known as the ***object code***. The object code is used to create an executable program.

If the source code contains errors, then the compiler will not be able to do its intended task. Errors that limit the compiler in understanding a program are called ***syntax errors***.

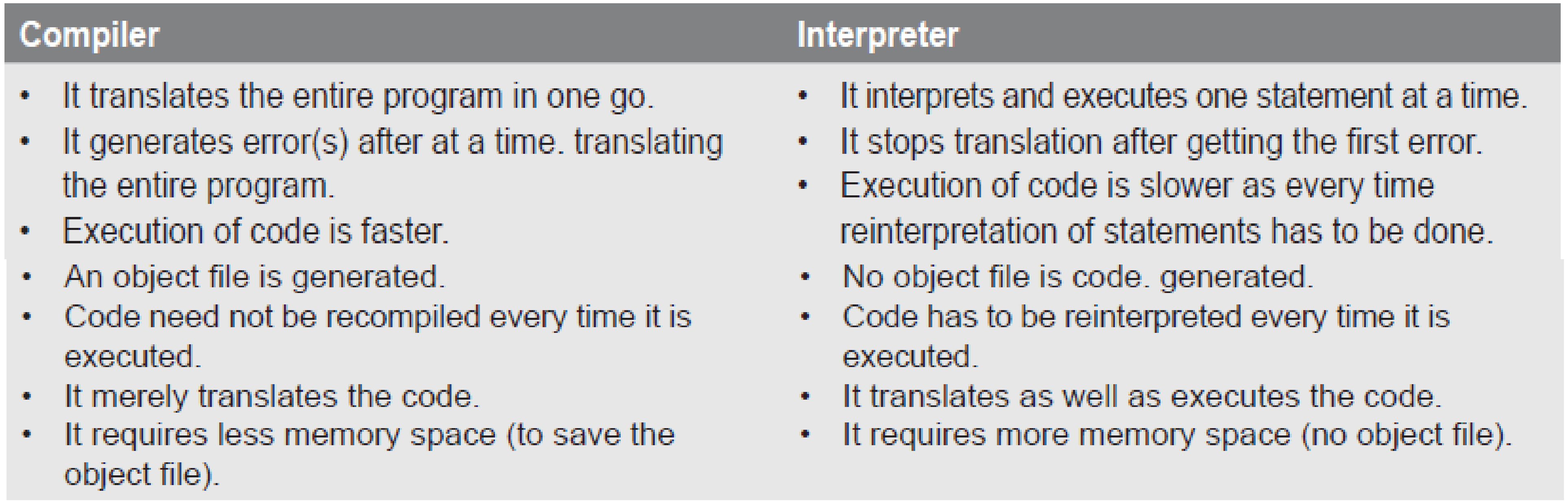
Each high-level language has a separate compiler. A compiler can translate a program in one particular high-level language into machine language.



Translator

Interpreter

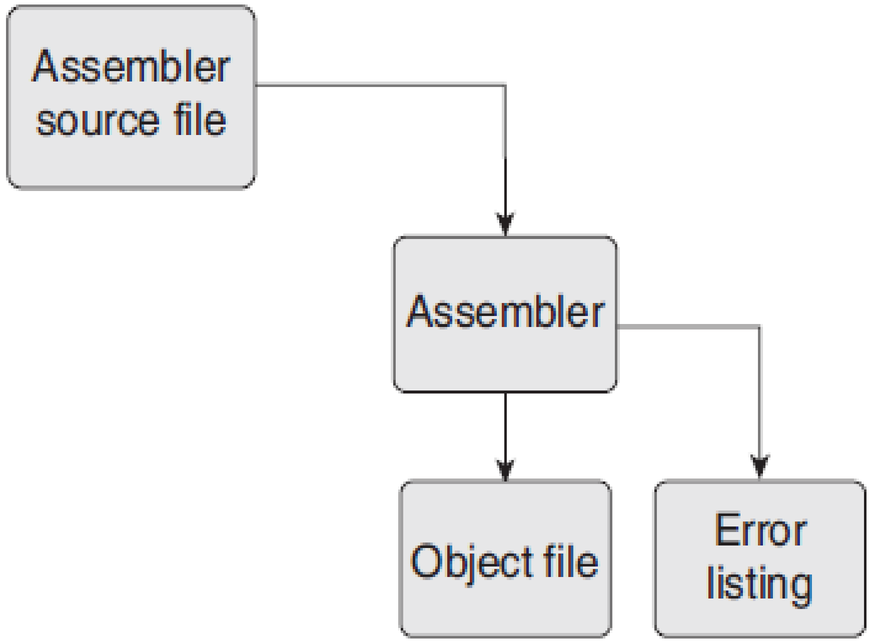
Interpreter translates the instructions into an intermediate form, which it then executes. The interpreter takes one statement of high-level code, translates it into the machine level code, executes it, and then takes the next statement and repeats the process until the entire program is translated.



Compiler	Interpreter
• It translates the entire program in one go.	• It interprets and executes one statement at a time.
• It generates error(s) after at a time. translating the entire program.	• It stops translation after getting the first error.
• Execution of code is faster.	• Execution of code is slower as every time reinterpretation of statements has to be done.
• An object file is generated.	• No object file is code. generated.
• Code need not be recompiled every time it is executed.	• Code has to be reinterpreted every time it is executed.
• It merely translates the code.	• It translates as well as executes the code.
• It requires less memory space (to save the object file).	• It requires more memory space (no object file).

Translator

Assembler Since computers can execute only codes written in machine language, a special program, called the assembler, is required to convert the code written in assembly language into an equivalent code in machine language, which contains only 0s and 1s. There is a one-to-one correspondence between the assembly language code and the machine language code. However, if there is an error, the assembler gives a list of errors. The object file is created only when the assembly language code is free from errors. The object file can be executed as and when required. For example, MASM, TASM, NASM, YASM, VASM, etc.

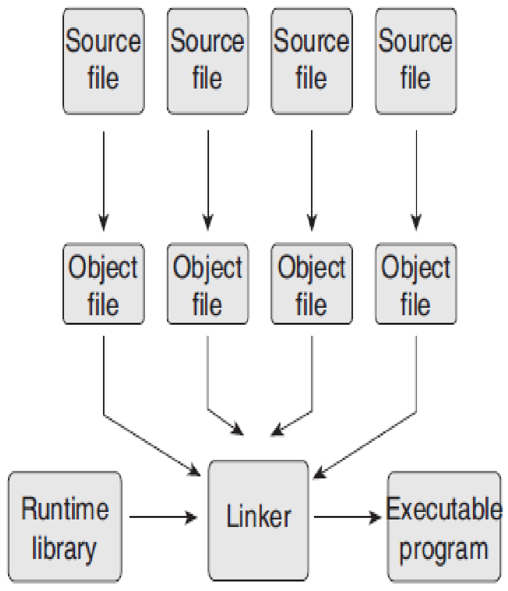


Translator

Linker

Software development in the real world usually follows a modular approach in which, a program is divided into various (smaller) modules as it is easy to code, edit, debug, test, document, and maintain them. Moreover, a module written for one program can also be used for another program. When a module is compiled, an object file of that module is generated.

Once the modules are coded and tested, the object files of all the modules are combined together to form the final executable file. Linker, also called a *link editor* or *binder*, is a program that combines the object modules to form an executable program.



Loader

A *loader* is a special type of program that is part of an operating system and which copies programs from a storage device to the main memory, where they can be executed.

Representation Of Data

Bit is a short form of binary digit. It is the smallest possible unit of data. In computerized data a bit can either be 0 or 1.

Nibble is a group of four binary digits.

Byte is a group of eight bits. A nibble is a half byte. Bits 0 through 3 are called the low order nibble, and bits 4 through 7 form the high order nibble

Word is group of two bytes is called a word. Bits 0 through 7 form the low order byte and bits 8 through 15 form the high order byte.

Unit	Abbreviation	Equal to	Bytes	Power of 2
Byte	Byte	8 Bits	1	2^0 bytes
Kilobyte	KB	1024 Bytes	1024	2^{10} bytes
Megabyte	MB	1024 KB	1048576	2^{20} bytes
Gigabyte	GB	1024 MB	1073741824	2^{30} bytes
Terabyte	TB	1024 GB	1099511627776	2^{40} bytes
Petabyte	PB	1024 TB	1 000 000 000 000 000	2^{50} bytes
Exabyte	EB	1024 PB	1 000 000 000 000 000 000	2^{60} bytes

Problem Solving Strategies

In **requirements analysis**, user's expectations are gathered to know why the software has to be built.

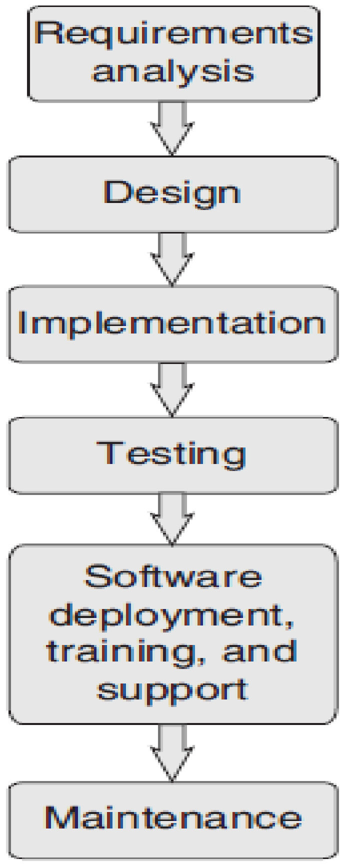
In the **design** phase, a plan of actions is made before the actual development process can start.

In **implementation** phase, the designed algorithms are converted into program code using any of the high-level languages.

During **testing**, all the modules are tested together to ensure that the overall system works well as a whole product.

In **software deployment, training, and support** phase, the software is installed or deployed in the production environment.

Maintenance and enhancements are ongoing activities that are done to cope with newly discovered problems or new requirements



Algorithms

An algorithm provides a blueprint to writing a program to solve a particular problem. It is considered to be an effective procedure for solving a problem in a finite number of steps. Algorithm should be:

- Be precise
- Be unambiguous
- Not even a single instruction must be repeated infinitely.
- After the algorithm gets terminated, the desired result must be obtained.

Different Approaches to Design an Algorithm

Top-down approach starts by dividing the complex algorithm into one or more modules.

Bottom-up approach is just the reverse of top-down approach. In the bottom-up design, we start with designing the most basic or concrete modules and then proceed towards designing higher level modules.

Control Structures Used in Algorithms

Sequence means that each step of the algorithm is executed in the specified order.

Decision statements are used when the outcome of the process depends on some condition

Repetition, which involves executing one or more steps for a number of times, can be implemented using constructs such as the while, do-while, and for loops.

Recursion is a technique of solving a problem by breaking it down into smaller and smaller sub-problems until you get to a small enough problem that it can be easily solved. Usually, recursion involves a function calling itself until a specified condition is met.

Examples:

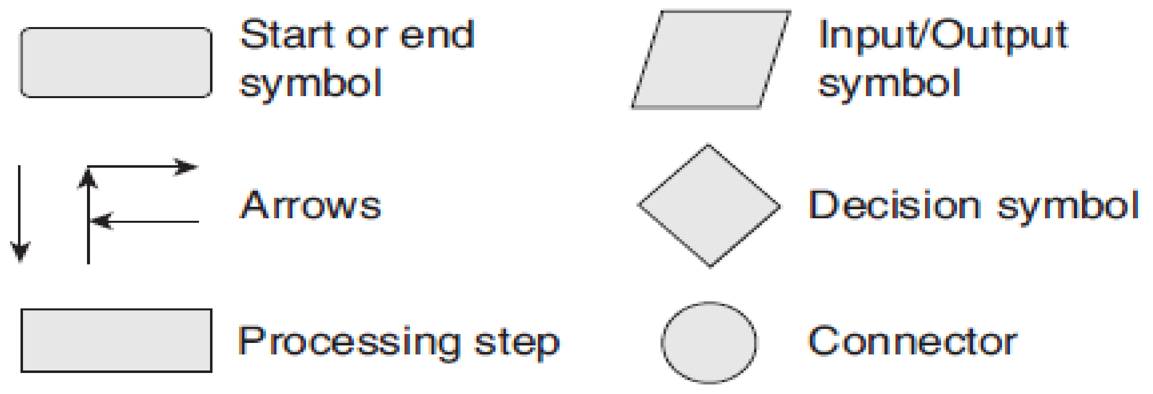
```
Step 1 : Input first number as A
Step 2 : Input second number as B
Step 3 : Set Sum = A + B
Step 4 : Print Sum
Step 5 : End
```

```
Step 1 : Input first number as A
Step 2 : Input second number as B
Step 3 : IF A = B
              Print "Equal"
         ELSE
              Print "Not equal"
         [END of IF]
Step 4 : End
```

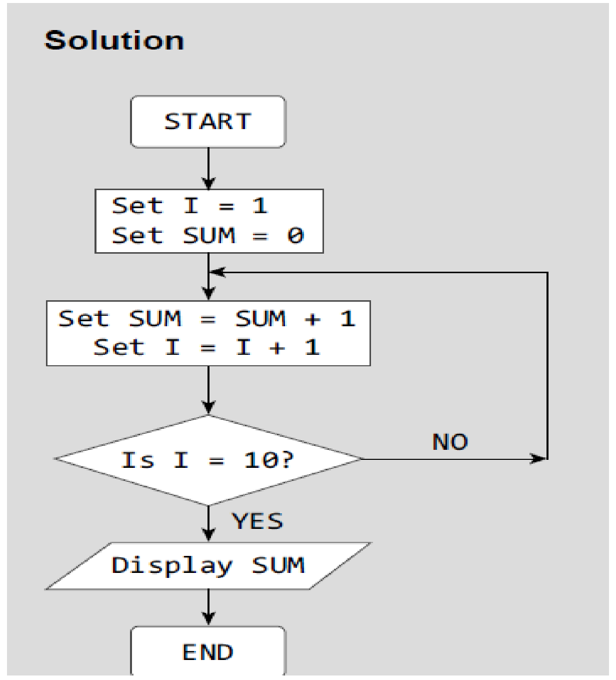
```
Step 1 : [initialize] Set I = 1, N = 10
Step 2 : Repeat Steps 3 and 4 while I <= N
Step 3 : Print I
Step 4 : SET I = I + 1
           [END OF LOOP]
Step 5 : End
```

Flowcharts

A flowchart is a graphical or symbolic representation of a process. It is basically used to design and document virtually complex processes to help the viewers to visualize the logic of the process.

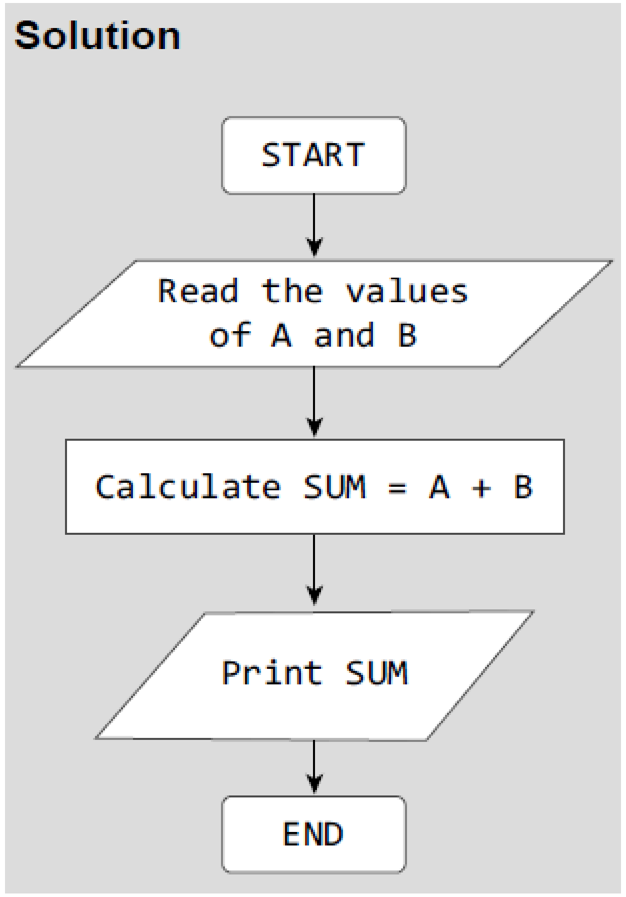


Example: Draw a flowchart to calculate the sum of the first 10 natural numbers.

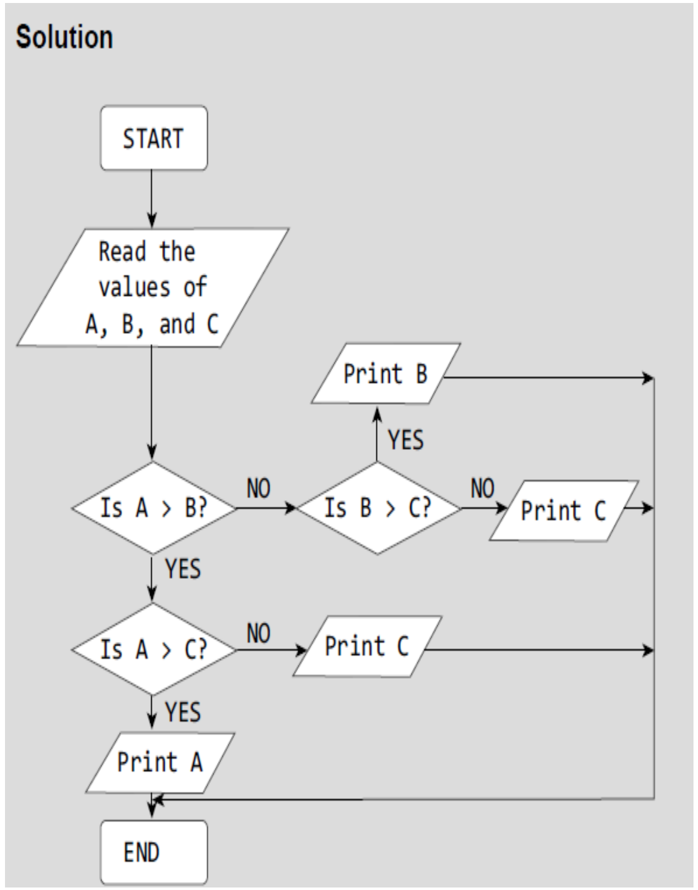


Flowcharts

Example: Draw a flowchart to add two numbers.



Example: Draw a flowchart to determine the largest of three numbers.



Pseudocodes

Pseudocode is a compact and informal high-level description of an algorithm that uses the structural conventions of a programming language. It facilitates designers to focus on the logic of the algorithm without getting bogged down by the details of language syntax. An ideal pseudocode must be complete, describing the entire logic of the algorithm, so that it can be translated straightaway into a programming language.

Example: Write a pseudocode for calculating the price of a product after adding the sales tax to its original price.

Solution

```
1. Read the price of the product
2. Read the sales tax rate
3. Calculate sales tax = price of the item x* sales tax rate
4. Calculate total price = price of the product + sales tax
5. Print total price
6. End
Variables: price of the item, sales tax rate, sales tax, total price
```

Types of Errors

Run-time errors occur when the program is being run executed. Such errors occur when the program performs some illegal operations like dividing a number by zero, opening a file that already exists, lack of free memory space, finding square or logarithm of negative numbers. Run-time errors may terminate program execution.

Syntax Errors are generated when rules of a programming language are violated. Python interprets (executes) each instruction in the program line by line. The moment interpreter encounters a syntactic error, it stops further execution of the program.

Semantic or Logical Errors are those errors which may comply with rules of the programming language but gives an unexpected and undesirable output which is obviously not correct. For example, if you write a program to add two numbers but instead of writing '+' symbol, you put the '-' symbol.

Linker Errors These errors occur when the linker is not able to find the function definition for a given prototype.

Testing

Unit Tests: Unit testing is applied only on a single unit or module to ensure whether it exhibits the expected behavior.

Integration Tests: These tests are a logical extension of unit tests. In this test, two units that have already been tested are combined into a component and the interface between them is tested.

System Tests: System testing checks the entire system.

Debugging

Debugging **is an activity that includes execution testing and code correction. It locates errors in the program code and fix them to produce an error-free code. Different approaches applied for debugging a code includes:**

Brute-Force Method: A printout of CPU registers and relevant memory locations is taken, studied, and documented. It is the least efficient way of debugging a program and is generally done when all the other methods fail.

Backtracking Method: It is used to debug small applications. It works by locating the first symptom of error and then tracing backward across the entire source code until the real cause of error is detected. With increase in number of source code lines, the possible backward paths become too large to manage.

Cause Elimination: A list of all possible causes of an error is developed. Then relevant tests are carried out to eliminate each of them.