Lesson 1.1: What is a Computer?

A **computer** is an electronic machine designed to carry out a set of instructions (called a program) to perform various tasks. It can:

- **Take Input**
- **Process Data**
- **Store Information**
- **Give Output**

Components of a Computer

Computers are made up of two main types of components:

Hardware – The Physical Components

- **Central Processing Unit (CPU)**: The brain of the computer.
- **Random Access Memory (RAM)**: Temporary memory that stores data while the computer is on.
- **Hard Disk / SSD**: Stores data permanently.
- **Motherboard**: Main circuit board connecting all components.
- **Input Devices**: Keyboard, Mouse, Microphone
- **Output Devices**: Monitor, Printer, Speakers

Software – The Intangible Components

Software is a collection of instructions. It tells the hardware what to do.

- **System Software**: Operating System (Windows, Linux, macOS)
- **Application Software**: MS Word, Browser, Games
- **Programming Software**: Tools used to write code (Python, IDEs, etc.)

Lesson 1.3: Functional Units of a Computer

1. **Input Unit**: Accepts data (keyboard, mouse)
2. **Storage Unit**:
    - **Primary Storage**: RAM, ROM (faster, temporary)
    - **Secondary Storage**: Hard disk, SSD, USB (permanent)
3. **Central Processing Unit (CPU)**:
    - **Control Unit (CU)**: Manages the operations
    - **Arithmetic Logic Unit (ALU)**: Performs calculations
4. **Output Unit**: Displays the result (monitor, printer)

Characteristics of a Computer

- **Speed**: Executes millions of operations in a second.
- **Accuracy**: Very precise, less chance of errors (unless input is wrong).
- **Automation**: Works automatically once instructions are given.
- **Storage**: Stores large amounts of data.
- **Versatility**: Can perform multiple types of tasks.
- **Diligence**: Never gets tired or bored.

Exercise for You:

1. Define a computer in your own words.
2. Name at least 4 hardware components and 2 software types.
3. What's the difference between RAM and Hard Disk?
4. Match the following:
    - Monitor →
    - Keyboard →
    - CPU →
    - MS Word →

A. Output device
B. Input device
C. Application software
D. Processing unit

Software and Operating Systems

Lesson 2.1: What is Software?

Software is a set of instructions that tells the computer what to do. Without software, your hardware (the physical parts) cannot function properly.

Relationship Between Hardware and Software:

Think of:

- **Hardware** = Body
- **Software** = Mind

Both must work together for the system to function.

Lesson 2.2: Types of Software

Software is mainly categorized into **two types**:

System Software

This controls the internal functioning of a computer.

- **Operating System (OS)**: Acts as a bridge between the user and hardware.
    - Examples: Windows, macOS, Linux, Android
- **Utilities**: Programs that help maintain and protect your system.
    - Examples: Antivirus, Disk Cleanup
- **Device Drivers**: Help the OS communicate with hardware.
    - Example: Printer driver, Graphics driver

Application Software

Used by users to perform specific tasks.

- **General Purpose Applications**
    - Examples: MS Word, Excel, Google Chrome, VLC
- **Custom Applications**
    - Built for specific organizations or users (like a college student portal)

Lesson 2.3: What is an Operating System (OS)?

An **Operating System** is the most important system software that:

- Manages **hardware** resources
- Controls **input/output devices**
- Manages **files and directories**
- Provides a **User Interface** (UI)
- Manages **memory** and **CPU time**

Functions of an Operating System

1. **Process Management**: Handles all running programs
2. **Memory Management**: Manages RAM usage
3. **File System Management**: Organizes and tracks files
4. **Device Management**: Controls hardware devices using drivers
5. **Security and Access Control**: User login, permissions
6. **User Interface**: CLI or GUI

Lesson 2.5: User Interfaces

There are two main types of interfaces that let you interact with an OS:

Graphical User Interface (GUI)

- Uses windows, icons, buttons
- Example: Windows 10, Ubuntu Desktop

Command Line Interface (CLI)

- Text-based; user types commands
- Example: Linux Terminal, Windows CMD

Exercise for You:

1. What is software? Explain its types.
2. List 3 system software and 3 application software examples.
3. What is the role of an operating system?
4. Compare GUI and CLI with examples.
5. Match the following:
    - Windows
    - MS Paint
    - Antivirus
    - Terminal

A. Application software
B. Operating system
C. Utility software
D. CLI

Programming Basics

Lesson 3.1: What is Programming?

Programming is the process of writing instructions for a computer to perform specific tasks.

These instructions are written in **programming languages** like:

- Python
- C/C++
- Java

- JavaScript

Lesson 3.2: Why Do We Program?

We program to:

- Automate tasks (e.g., calculator, billing system)
- Solve problems
- Build websites, apps, games, AI, and more

Lesson 3.3: What is a Programming Language?

A **programming language** is a set of rules used to write software instructions.

Tools You Can Use:

- **Python IDEs**: Thonny, VS Code, PyCharm
- **Online Compilers**: Replit.com, Programiz.com

Exercises:

1. Write a Python program to display your name.
2. Create a variable x = 10 and print it.
3. Take user input for age and print if they are eligible to vote (18+).
4. Write a loop that prints numbers from 1 to 5.

Data Structures (Basics)

Lesson 4.1: What Are Data Structures?

A **data structure** is a way to store and organize data so it can be used efficiently.

Imagine a **toolbox**—you use different tools for different tasks. Similarly, in programming, you use different data structures to solve different problems.

Lesson 4.2: Why Learn Data Structures?

- **Efficiency**: Helps store and access data faster
- **Organized**: Keeps data structured and manageable
- **Real-World Use**: Used in search engines, games, apps, databases, etc.

Algorithms (Theory Only)

Lesson 5.1: What is an Algorithm?

- An **algorithm** is a step-by-step method to solve a problem.
- It is like a cooking recipe: a set of clear instructions to complete a task.
- For example: To find the largest number in a list of numbers, an algorithm defines how to compare and track values step by step.

Lesson 5.3: Why Are Algorithms Important?

- Help us solve problems logically
- Make programs efficient and fast
- Enable us to compare different solutions

Complexity of Algorithms

To compare how fast or efficient an algorithm is, we analyze:

Time Complexity: How much time the algorithm takes relative to input size.

- Expressed using Big O notation, e.g., O(n), O(log n), $O(n^2)$

Space Complexity: How much memory the algorithm uses during execution.