Unit 3

Computer Software

Prepared By: Ananda Adhikari

Introduction

- ✓ Software is a **collection of set of programs,** which are used to execute all kinds of specific instruction.
- ✓ It consists of a number of machine instructions, array in a specific order to perform a particular task.
- ✓ Software is used to describe all the programs and its associated documents which run on a computer.
- ✓ So, a computer needs both software and hardware for its proper functioning.

Introduction

- ✓ Software means **computer instructions or data**.
- ✓ Anything that can be stored electronically is software.
- ✓ **Firmware** are software (programs or data) that has been permanently written onto read-only memory (ROM).

All software falls into two general types or categories:

1. **System Software**
2. **Application Software**

System Software

- ✓ Basic software required for operating a computer.
- ✓ Used to control and coordinate computer hardware and software.
- ✓ Responsible for resource management, memory management, file and task management, software repairing, etc.
- ✓ Examples include **Operating System, Device Drivers, anti-virus softwares and printer drivers**.

Classification of System Software:

1. Operating System
2. Language Processor/ Translator
3. Utility Software
4. Device Driver

1. Operating System

- ✓ Interface between user and computer hardware.
- ✓ Set of system programs that control the execution of application programs.
- ✓ The device is not usable in the absence of Operating system.
- ✓ Every computer and computer like devices including tablets, mobile phones, smart-watch, laptops require operating system too.

For example, Windows Operating system, Linux, Andriod, iOS, etc.

2. Language Processor/ Translator

- ✓ System software that converts language from high level to low or machine level language.
- ✓ Perfoms tasks like translating or interpreting a specified programming language lie C, C++, etc.

For example, Compiler, Interpreter, Assembler, etc.

3. Utility Software

- ✓ Programs designed to help manage, maintain, and optimize computer hardware, software, and the overall system environment.
- ✓ These tools are usually focused on **system maintenance, troubleshooting, performance enhancement, and resource management**.
- ✓ For example, Backup Utility, File Manager, Disk Cleaner, etc.

4. Device Driver

- ✓ A device driver is specialized software that allows an operating system (OS) to communicate with hardware devices.
- ✓ It acts as a translator between the hardware and the system, ensuring that commands from the OS or software are correctly executed by the hardware.

For example, Realtek HD Audio, Wifi Adapter Driver, Printer drivers, etc.

Application Software

- ✓ Application software refers to programs designed to **perform specific tasks** for users.
- ✓ Unlike system software, which runs the hardware and system services, application software directly assists users in accomplishing tasks, such as **creating documents, managing data, communicating, or entertaining**.

For example,

Writing a report: Microsoft Word

Designing a logo: Adobe Illustrator

Watching a movie: Netflix

Sending an email: Gmail and many more.

Application Software (Types)

1. **Tailored Software or Custom Software**

- ✓ Software developed as per the need of a particular user or organization.
- ✓ Software developed for specific purpose like Hotel reservation software, e-payment software, etc.
- ✓ Usually high level language like C, C++, Java, PHP, etc are used for developing tailored software.

2. **Packaged Software**

- ✓ Generalized set of programs designed for general purpose.
- ✓ All word processing software like Office 365, web designing software, database software are packaged software.

Differences

Aspect	System Software	Application Software
Purpose	Manages hardware and system resources (e.g., booting the computer).	Performs specific tasks for the user (e.g., writing a document).
Examples	Operating systems (e.g., **Windows 10**, **Linux**) or device drivers (e.g., **NVIDIA graphics driver**).	Microsoft Word, Google Chrome, VLC Media Player.
Dependency	Works independently of application software (e.g., an OS can run without apps).	Requires system software to function (e.g., Word requires an OS like Windows).
User Interaction	Minimal user interaction; runs in the background (e.g., file management tasks).	Requires direct user interaction (e.g., creating a presentation in PowerPoint).

Software Acquisition

- ✓ Software acquisition refers to the process of obtaining software for individual or organizational use.
- ✓ It involves **selecting, purchasing, or acquiring software** that meets specific requirements.

Types of Software Acquisition:

1. Commercial Software:

- ✓ Pre-built software available for immediate purchase or download. Examples: Microsoft Office, Adobe Photoshop.

2. Custom Software

- ✓ Software developed specifically to meet unique organizational needs.
- ✓ Examples: Hospital management systems, supply chain management tools.

Software Acquisition

Types of Software Acquisition:

3. Open-Source Software

- ✓ Software with publicly available source code, allowing customization and redistribution.
- ✓ Examples: Linux, LibreOffice.

4. Freeware and Shareware

- ✓ **Freeware:** Free to use with no cost (e.g., VLC Media Player).
- ✓ **Shareware:** Free for a trial period, with payment required for continued use (e.g., WinRAR).

Software Acquisition

Types of Software Acquisition:

5. Subscription-Based Software

- ✓ Paid on a recurring basis, often as a Software-as-a-Service (SaaS) model.
- ✓ Examples: Microsoft 365, Zoom, etc.

6. Cloud-Based Software

- ✓ Hosted on the internet and accessed via a browser.
- ✓ Examples: Google Workspace, Salesforce.

Software Acquisition

Types of Software Acquisition:

7. In-House Development

- ✓ Software developed internally by an organization’s IT team.
- ✓ Example: A custom CRM built specifically for a company.

Classwork:

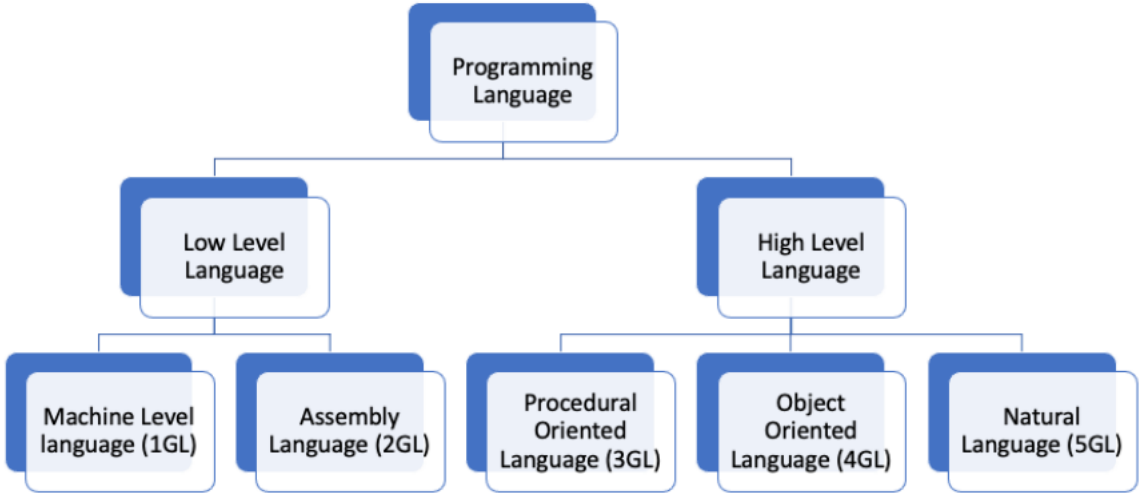
1. Differentiate between System Software and Application Software.
2. Briefly explain Software Acquisition Types.

Programming Language

- ✓ Computer language used for developing programs and software.
- ✓ Provides a set of symbols, characters, operators that instruct the computer to perform some specific task.

Types of Programming language are :-

1. Low Level language
2. High Level Langauge



Programming Language

1. **Low Level language**

a. **Machine level language (1GL)**

- ✓ In this language, program is developed using the combinations of binary numbers (i.e. 0s and 1s).
- ✓ It is the only one language the computer directly understands.

Merits:

- ✓ Language translator is not required which enhances the execution of program.
- ✓ Efficient use of hardware components like memory and processor.

Programming Language

1. **Low Level language**

a. **Machine level language (1GL)**

Demerits:

- ✓ Tedious, difficult and time consuming to develop and debug program.
- ✓ All operation codes and memory address has to be remembered.
- ✓ Incompatibility issues.

Programming Language

1. **Low Level language**

b. **Assembly Level Language (2GL)**

- ✓ The language that uses mnemonics (like ADD, SUB, DIV) instead of 0s and 1s.
- ✓ The program is converted into machine code by using assembler.
- ✓ It is second generation language.

Merits:

- ✓ Easier debugging and development of program in comparision to machine level language.
- ✓ Faster execution speed than high level language.

Programming Language

1. **Low Level language**

b. Assembly Level Language (2GL)

Demerits:

- ✓ Machine dependent language.
- ✓ Programmers require detailed knowledge of hardware architecture.
- ✓ Additional software is required to convert it into machine code to understand by computer.

Programming Language

2. **<u>High Level language</u>**

a. **Procedure Oriented Language (3GL)**

- ✓ Uses a simple English and mathematical statement for developing a program.
- ✓ Machine independent language.
- ✓ It is called procedure oriented language because a programmer has to specify the working procedure in detail.

For example, C, C++, etc.

Programming Language

2. **High Level language**

a. **Procedure Oriented Language (3GL)**

Merits:

- ✓ Easier to develop and debug program in comparision to low level language.
- ✓ More advanced and user friendly software can be developed using this language.
- ✓ Programmers does not require detail knowledge of hardware architecture.

Demerits

- ✓ Slower program execution.
- ✓ Programmer need to specify working process of program in detail.

Programming Language

2. High Level language

b. Object Oriented Language (4GL)

- ✓ An object-oriented language is a type of programming paradigm that focuses on objects, which represent real-world entities.
- ✓ Objects encapsulate both data (attributes) and behavior (methods), making programs modular and easier to maintain.
- ✓ It is based on classes and objects.
- ✓ Encourages reusability via inheritance.

Examples: Java, PHP, C++, C#, Ruby.

Programming Language

2. High Level language

b. Object Oriented Language (4GL)

Merits:

- ✓ Easier debugging and development of programs than procedure oriented language.
- ✓ Web based software can be developed using this language.
- ✓ Machine independent language.

Demerits:

- ✓ Program exection will be even slower than procedure oriented language.

Programming Language

2. High Level language

c. Natural Language (4GL)

- ✓ Normal statements and graphical tools are used to develop a programs.
- ✓ It is still in developing stage, however a programming language named PROLOG is in used.
- ✓ Proposed language for fifth generation of computer.

Merits:

- ✓ Easier debugging and development of programs.
- ✓ User friendly software can be developed.

Demerits:

- ✓ Requirement of language translator.

Language Translator (Processor)

- ✓ System software that translates program developed in either **assembly language or high level language** into **machine language** and vice versa.
- ✓ It converts **source code** (Human readable instruction) into **object code** (machine readable instruction).

Types of language translator are:

a. Assembler
b. Compiler
c. Interpreter

Language Translator (Processor)

a. Assembler

- ✓ Translates program developed in assembly language into machine language and vice versa.
- ✓ It translates complete program at once, if the program is error free.

b. Compiler

- ✓ Translates a high level program (source code) into machine instruction (object code) at once.
- ✓ Programming language like C, C++, Java, etc use the compiler.

c. Interpreter

- ✓ Similar to compiler while it tanslates one line at a time and executes later after complete translation of source code.
- ✓ For example, C#, PHP use an interpreter.

Classwork:

1. Write the difference between compiler and interpreter.

Operating System

- ✓ An **Operating System (OS)** is a system software that acts as an intermediary between computer hardware and the end user.
- ✓ OS coordinates the function performed by the computer hardware, including the CPU, input/output devices, secondary storage devices, and communication and network equipment.
- ✓ OS keep tracks of all the files and directories on the disks, and control peripheral devices such as disk drivers and printers.
- ✓ It hides the complexities of the hardware from the user.

Operating System

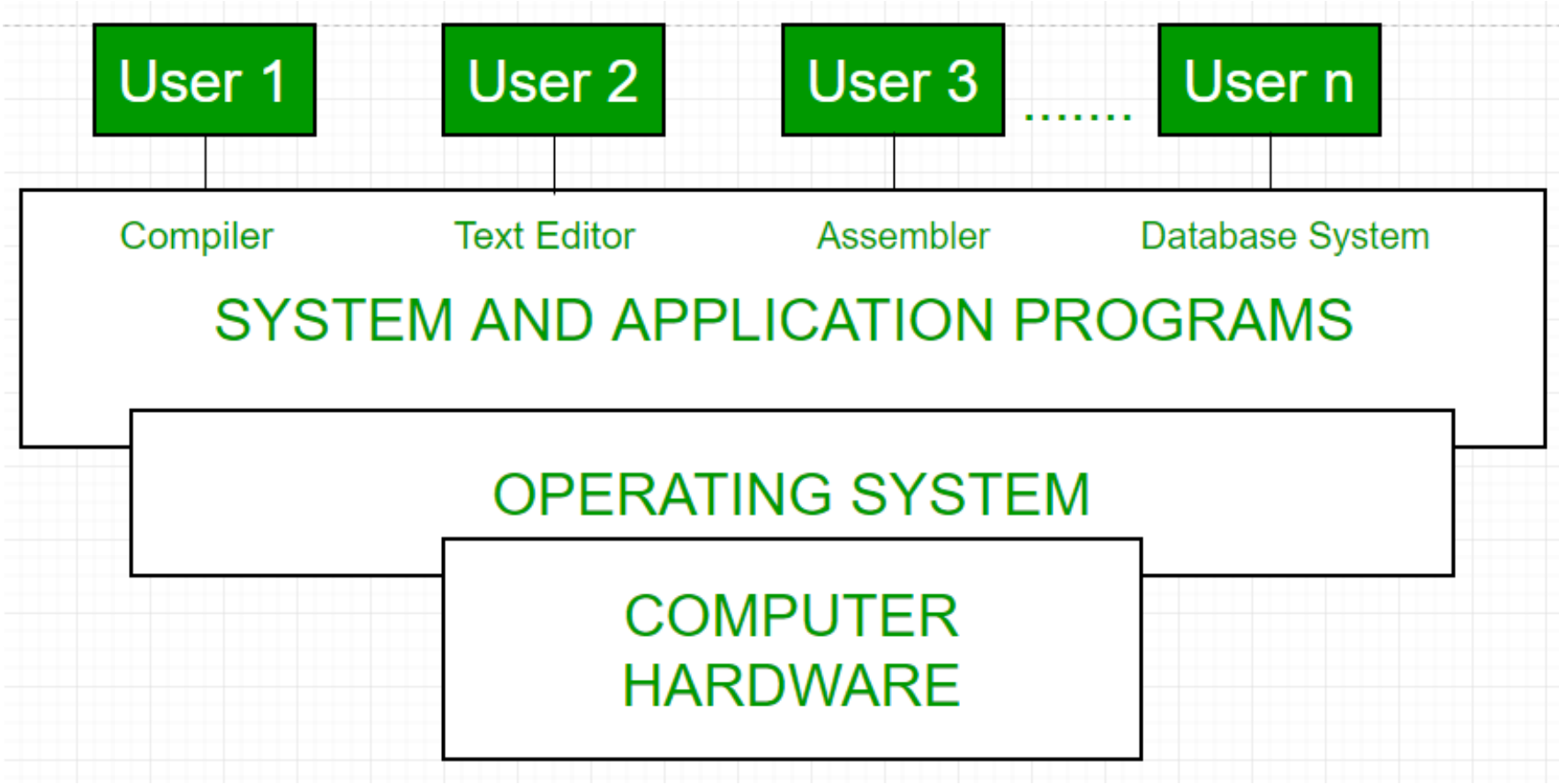


Figure: Basic Organization of the Operating System

Objectives of Operating System

- ✓ **Resource Management:** Efficiently allocate and manage hardware resources such as CPU, memory, and I/O devices.
- ✓ **User Convenience:** Provide a user-friendly interface to facilitate interactions between users and the computer.
- ✓ **Program Execution:** Enable smooth execution of user applications.
- ✓ **Error Detection and Handling:** Identify and respond to system errors to ensure stability.
- ✓ **Security and Protection:** Protect data and system resources from unauthorized access.
- ✓ **System Performance:** Optimize the overall performance of the computer system.

Types of Operating Systems (Based on Processing)

1. Multiprogramming OS:

- ✓ Technique in which multiple user programs are executed simultaneously by single processor or CPU.
- ✓ When two or more progams are provided to CPU for processing at same instant.
- ✓ Main purpose of multiprogramming is to increase the utilization of computer resources.
- ✓ Multiprocessing technique is possible due to the extremely high speed of CPU.
- ✓ The high processing speed of CPU helps it to allocate its time to every program in a **time sharing** mode.

Types of Operating Systems (Based on Processing)

2. Multiprocessing OS:

- ✓ Multiprocessing system has more than one processor, linked together in a coordinated way.
- ✓ Here, it supports running a program in more than one Central Processing Unit (CPU).
- ✓ Server systems are specially designed to support multiple processors.
- ✓ It is also known as **Parallel Processing,** wherein many processors work in parallel to solve one big task or many tasks simultaneously.
- ✓ Costly and complex implementation than multiprogramming.

Types of Operating Systems (Based on Processing)

3. Distributed OS:

- ✓ System that uses two or more independent computer systems via a commication link to serve multiple applications and users.
- ✓ It has its own OS and local memory.
- ✓ The processor communicates with each other through various communication lines like buses or network connection.

4. Network OS:

- ✓ System run on server and provide the capability to manage data, users, groups, security, applications and other networking functions.
- ✓ Allows shared access of files, printers, security, applications, and other networking functions over a small private network.

Types of Operating Systems (Based on Processing)

5. Batch Processing OS:

- ✓ Suitable in programs that require large computation time and with no need for user interaction or involvement.
- ✓ Here, user is not required to wait while the job is being processed.
- ✓ So, the user can submit their programs to operators and collect them later.
- ✓ Jobs are entered, stored on a disk in a batch or queue, and are executed one after another under the control of OS.

Types of Operating Systems (Based on User Interface)

1. Graphical User Interface (GUI)

- ✓ Allows the user to enter commands by pointing and clicking at objects that appear on screen.
- ✓ Uses the computer graphics capabilities to make program easier to use.
- ✓ Advantage of GUI is that they are easy to use for beginners.

Features:

- ✓ User friendly and easy to operate.
- ✓ User does not have to remember the commands.
- ✓ GUI environment is full of multimedia and animations.
- ✓ Requires larger memory size and faster processor to operate.

Types of Operating Systems (Based on User Interface)

2. Character User Interface (CUI)

- ✓ A Character User Interface (CUI) is a text-based interface where users interact with the system by **typing commands** or selecting options presented in text form.
- ✓ It is often referred to as a Command Line Interface (CLI).

Features:

- ✓ Less user friendly.
- ✓ Commands are used to instruct the computer.
- ✓ Unable to display graphics, icons, pictures, and multimedia.
- ✓ Faster than GUI system.
- ✓ It can be used in low memory and low processing speed computers.

Functions of Operating Systems

1. Process Management:

- ✓ Manages the execution of processes.
- ✓ Handles process scheduling, creation, and termination.
- ✓ Provides mechanisms for inter-process communication.

2. Memory Management:

- ✓ Allocates and deallocates memory space as required.
- ✓ Keeps track of memory usage.
- ✓ Manages virtual memory systems.

Functions of Operating Systems

3. File Management:

✓ Manages file storage, organization, retrieval, and updates.

✓ Provides directory structures and file access permissions.

4. Device Management:

✓ Controls and monitors hardware devices.

✓ Manages I/O devices and their drivers.

✓ Ensures proper communication between devices and the system.

5. Protection and Security:

✓ Ensures secure access to system resources.

✓ Protects data from unauthorized access.

✓ Implements authentication and encryption mechanisms.

Functions of Operating Systems

6. User Interface:

- ✓ Provides interfaces such as Command-Line Interface (CLI) and Graphical User Interface (GUI).
- ✓ Facilitates user interaction with the system.

Examples of Operating Systems

- ✓ **Desktop and Server OS**: Windows, Linux, macOS
- ✓ **Mobile OS:** Android, iOS

Classwork:

Q. What are the functions of Operating System?

New Trends in Software

1. **Increasing reliance on Artificial Intelligence (AI)**

- ✓ The increasing reliance on AI is largely driven by its ability to handle complex tasks efficiently, analyze vast amounts of data, and automate operations.
- ✓ As AI becomes more sophisticated, its role will likely expand even further, reshaping industries, creating new opportunities, and enhancing the quality of life.
- ✓ However, it also raises challenges, especially around ethics, transparency, and job displacement, which will need to be carefully managed.

New Trends in Software

2. Big Data Security

- ✓ Big data security is a crucial concern as organizations continue to collect and store massive amounts of data from various sources, such as customer interactions, social media, IoT devices, and more.
- ✓ The larger and more diverse the data set, the more complex the security challenges become.
- ✓ Ensuring that this data is protected from threats such as unauthorized access, breaches, and tampering is paramount.

New Trends in Software

3. Progressive web apps

- ✓ Progressive web apps are web applications that leverage modern web technologies to provide a native app-like experience while still being accessible through a web browser.
- ✓ It aims to combine the best features of websites and mobile applications, giving users a seamless, fast, and engaging experience.

New Trends in Software

4. Growing demand for outsourcing

- ✓ Outsourcing continues to grow due to its ability to deliver cost savings, efficiency, and access to specialized skills.
- ✓ As businesses face increasing competition, the need to innovate quickly, outsourcing has become an attractive option for organizations aiming to stay competitive.
- ✓ However, it requires careful planning, vendor selection, and management to avoid the potential downsides and ensure successful outcomes.

Classwork

Section A: Short Answer Questions (Answer in 1–3 sentences each)

1. What is software?
2. Define a computer program.
3. Write any two differences between system software and application software.
4. What is an operating system?
5. Write any two functions of an operating system.
6. Name any two types of operating systems.
7. What is utility software? Give one example.
8. What is a computer virus?
9. Write any two harmful effects of a computer virus.
10. What is antivirus software?
11. Name any two antivirus programs.
12. What is a programming language?
13. Name any two high-level programming languages.
14. What is a language translator? Name the three types of language translators.

Classwork

Section B: Long Answer Questions *(Answer in detail)*

1. Explain software and program. How are they related to each other?
2. Describe the types of software. Explain system software and application software with examples.
3. What is an operating system? Explain its major functions.
4. Explain different types of operating systems with suitable examples.
5. Define utility software. Explain its importance and uses in a computer system.
6. What is a computer virus? Explain its types and effects.
7. What is antivirus software? Explain how antivirus software protects a computer.
8. Define programming language. Explain different levels of programming languages.
9. What is a language translator? Explain compiler, interpreter, and assembler.
10. Differentiate between compiler and interpreter with examples.