Unit 4
Computer Software

By Kajol Ramtel

Introduction

- Software are the set of instructions or programs that tells the computer what to do and how to do it.
- Computer hardwares are basically nothing without the computer softwares.
- A computer can perform different jobs only if there is a set of instructions that can tell the computer what to solve a problem only if it is stated in the form of instruction which tell the computer what to do.
- Thus, softwares are the interface between the user of the computer and the computer hardware.

Types of Softwares

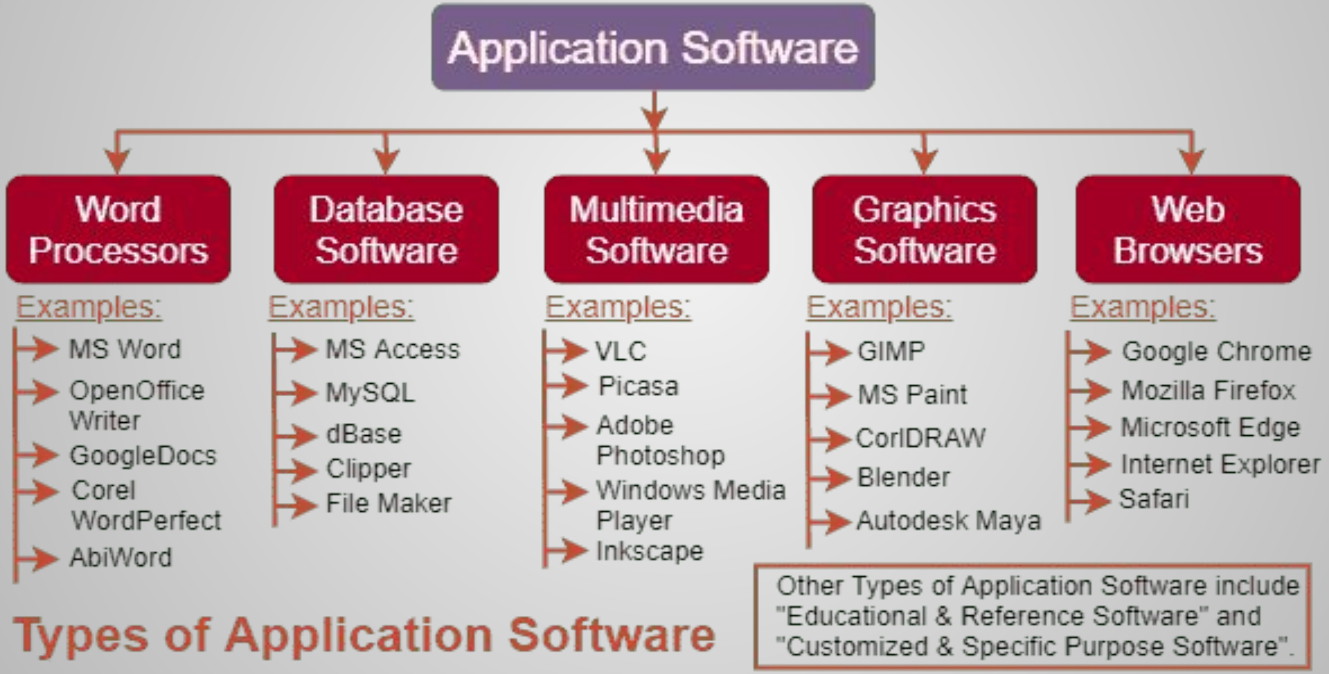
There are different types of softwares available for use in the computer.
They are:-

- Application Software
  - General Purpose / Packaged
  - Specific Purpose / Tailored
- System Software
  - Operating System
  - Language Processor
    - Assembler
    - Compiler
    - Interpreter
  - Utility Software/ program
  - Device Driver

Application Software

Application software is a type of computer software that is designed to help the user perform specific tasks. Examples of application software include word processors, spreadsheet programs, and media players. Unlike system software, which is responsible for the basic operation of the computer, application software is designed to perform more specialized tasks and is used by the user to complete specific work or personal projects.

Application Software



General Purpose Software / Packaged

General-purpose software is software that can be used for a wide range of tasks, rather than being designed specifically for a particular task or application.
These types of software are designed to be flexible and can be used to perform a variety of different tasks, depending on how they are programmed and configured.
In contrast, specialized software is designed for a specific task or application, and is not as flexible as general-purpose software.
Example: Microsoft Office package, Adobe Photoshop, VLC media player etc.

Special Purpose / Tailored Software

Tailored software, also known as custom software or bespoke software, is software that is developed specifically for a particular organization or user. It is designed to meet the specific needs and requirements of the organization or user, and is not generally available to the public.

Tailored software is often used to solve a specific problem or to automate a specific process within an organization, and is typically more expensive to develop than off-the-shelf software because it is developed specifically for a single client.

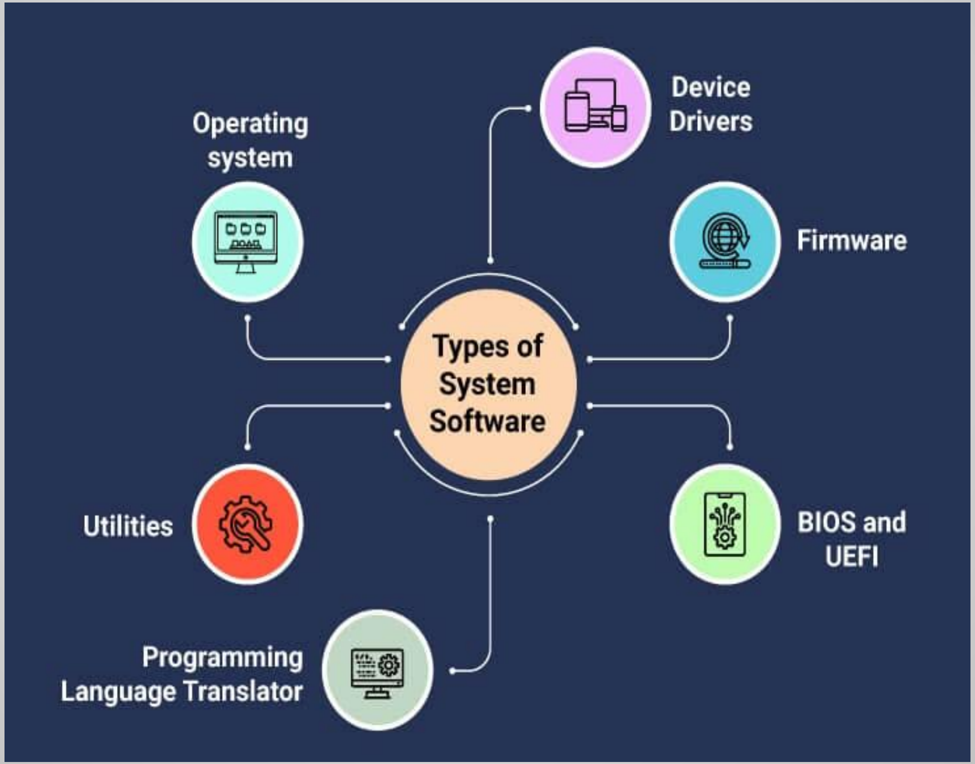
However, it can provide a number of benefits, such as increased efficiency, improved accuracy, and better integration with existing systems, that can outweigh the higher initial cost.

Example: Ticket Booking system, Employee Management system etc.

System Software

System software is a type of computer software that is designed to control the hardware and perform tasks that are essential for the operation of the computer. It is responsible for managing the computer's resources, such as memory, processors, and input/output devices, and providing a platform for other software to run on. Some examples of system software include :-
operating systems, device drivers, and utility programs.

System Software



Operating System

An operating system (OS) is a type of system software that manages the hardware and software resources of a computer. It is the most important type of system software in a computer, and it is responsible for controlling the hardware and providing a platform for other software to run on.
Some examples of operating systems include Windows, macOS, Linux, and Android.

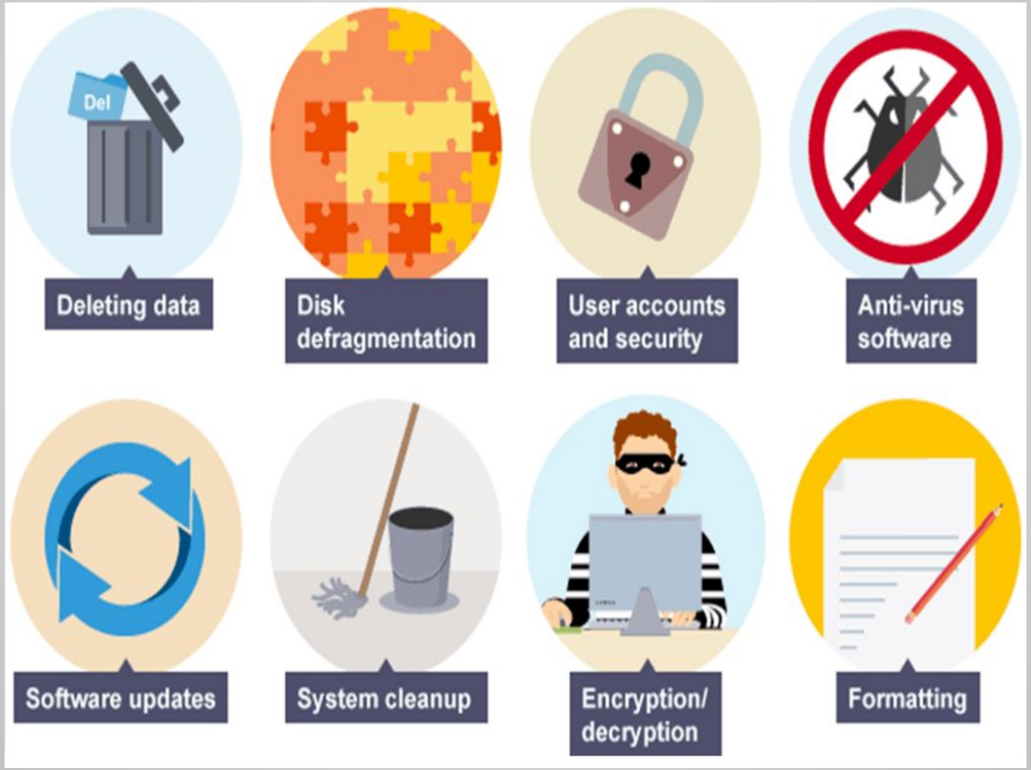
Utility Software

Utility software is a type of system software that performs a specific task or set of tasks to help the user maintain the computer or perform specific functions.

Utility programs are tools that help the user perform tasks such as disk defragmentation, virus scanning, and file compression. They are usually small programs that run in the background and are not typically used directly by the user.

Utility software is usually installed on the computer along with the operating system, but it can also be downloaded and installed separately.

Utility Software



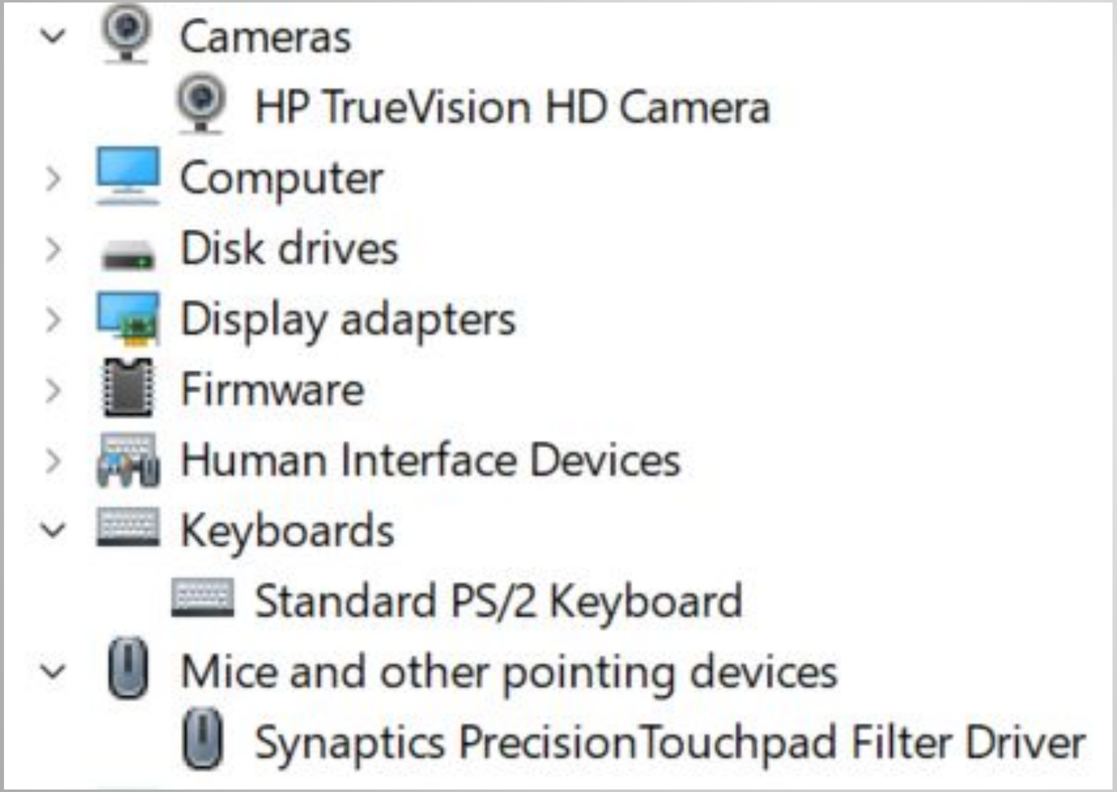
Device Drivers

A device driver is a piece of software that allows the operating system to communicate with a specific hardware device. It acts as an intermediary between the device and the operating system, translating the commands and requests of the operating system into actions that the device can understand and vice versa.

Device drivers are essential for the proper functioning of hardware devices and are usually provided by the manufacturer of the device.

They are usually specific to a particular device and operating system, and must be installed on the computer before the device can be used.

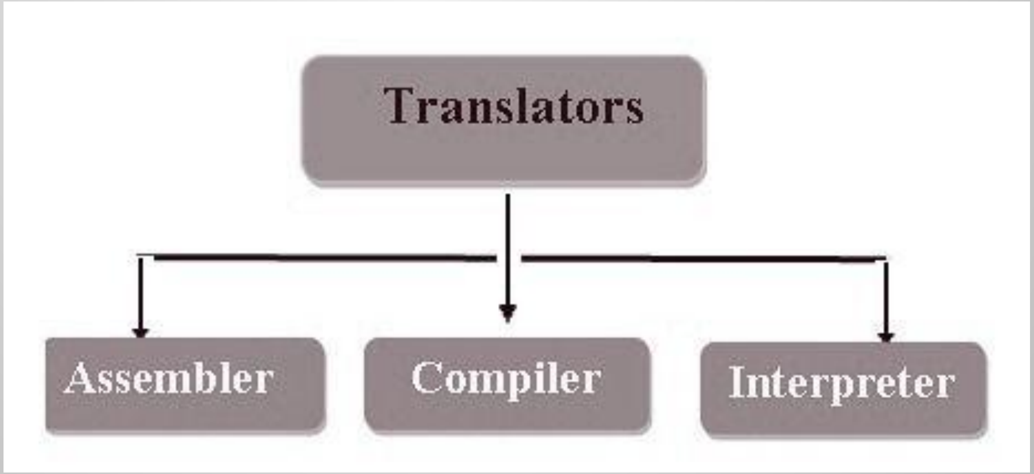
Device Drivers



Cameras
HP TrueVision HD Camera
Computer
Disk drives
Display adapters
Firmware
Human Interface Devices
Keyboards
Standard PS/2 Keyboard
Mice and other pointing devices
Synaptics PrecisionTouchpad Filter Driver

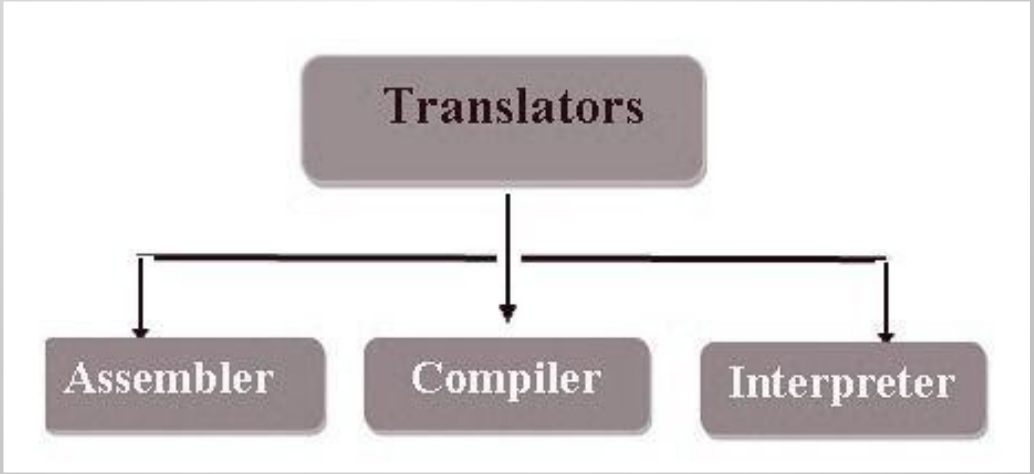
Language Translator

Language translators allow computer programmers to write sets of instructions in specific programming languages. These instructions are converted by the language translator into machine code. The computer system then reads these machine code instructions and executes them.



Language Translator

Language translators allow computer programmers to write sets of instructions in specific programming languages. These instructions are converted by the language translator into machine code. The computer system then reads these machine code instructions and executes them.



Assignment

What is language translator? What are the types of language translator . Explain each of them.

Software Acquisition

Software acquisition" refers to the process of obtaining software for use in an organization or by an individual. This process involves various stages, including planning, procurement, implementation, and maintenance. Here are the key steps typically involved in software acquisition:

1. Needs Assessment:

Identify the specific requirements and needs of the organization or individual.

Consider the purpose of the software, user requirements, system compatibility, and any other relevant factors.

2. Budgeting and Planning:

Determine the budget available for software acquisition.

Create a plan outlining the goals, timeline, and resources required for the acquisition process.

Software Acquisition

3. Market Research:
Explore the available software options in the market that meet the identified needs. Compare features, prices, licensing models, and support options of different software solutions.

4. Request for Proposal (RFP) or Quotation (RFQ):
If applicable, create and send out a Request for Proposal (RFP) or Request for Quotation (RFQ) to potential vendors.
Gather and evaluate proposals from different vendors.

Software Acquisition

5. Vendor Selection:
Evaluate vendors based on factors such as reputation, product quality, support services, and cost-effectiveness.
Select the vendor that best aligns with the organization's needs and budget.

6. Contract Negotiation:
Negotiate the terms and conditions of the contract with the chosen vendor.
Clarify licensing agreements, support and maintenance terms, and any other relevant details.

Software Acquisition

7. Pilot Testing (Optional):
Conduct a pilot test of the software in a controlled environment to assess its functionality, performance, and compatibility.

8. Deployment:
Install and configure the software in the production environment.
Train users on how to use the software effectively.

9. Monitoring and Evaluation:
Continuously monitor the performance of the software after deployment.
Gather feedback from users and address any issues that may arise.

Software Acquisition

10. Updates and Maintenance:
Stay informed about updates and patches released by the software vendor. Implement regular maintenance to ensure the software remains secure and functional.

11. End-of-Life Planning:
Plan for the eventual replacement of the software when it reaches the end of its life cycle.

It's essential to follow a systematic approach to software acquisition to ensure that the selected software meets the organization's needs, aligns with its goals, and provides value for the investment.

Programming Language and Development tools:

- A programming language is a formal language that consists of a set of instructions that can be used to create a computer program. These instructions specify how to perform a computation, and they can be executed by a computer or other type of machine.
- Some examples of popular programming languages include C++, Java, Python, and JavaScript. Each language has its own set of rules and syntax, and they are used to create different types of software, such as desktop applications, mobile apps, and web applications.
- A computer program that is used by the software developers for creating, editing, maintaining, supporting and debugging other applications, frameworks and programs – is termed as a Software Development Tool or a Software Programming Tool.
- Development tools can be of many forms like linkers, compilers, code editors, GUI designer, assemblers, debugger, performance analysis tools etc. There are certain factors to be considered while selecting the corresponding development tool, based on the type of the project.

Types of programming Language

- **Machine language:** This is the lowest-level programming language, and consists of binary code that can be directly executed by a computer's central processing unit (CPU). Machine language is difficult for humans to read and write, so it is not often used directly.
- **Assembly language:** This is a low-level programming language that is easier for humans to read and write than machine language. It is still a very basic language, and consists of simple instructions that are mapped to machine language code. E.g. ARM, MIPS etc.
- **High-level languages:** These languages are more abstract and easier for humans to read and write than low-level languages. They are closer to human languages and include programming languages such as C, C++, and Python. High-level languages are typically compiled or interpreted into machine code that can be run on a computer.

Most commonly used Programming Language

1. Python:

Python is one of the most widely used user-friendly programming languages. It is an open-source and easy to learn programming language developed in the 1990s. It is mostly used in Machine learning, Artificial intelligence, Big Data, GUI based desktop applications, and Robotics.

Advantages

- Python is easy to read, easy to understand, and easy to write.
- It integrates with other programming languages like C, C++, and Java.
- Python executes code line-by-line, so it is easy for the programmer to find the error that occurred in the code.
- Python is platform-independent means you can write code once and run it anywhere.

Most commonly used Programming Language

3. C

C is a popular, simple, and flexible general-purpose computer programming language. Dennis M Ritchie develops it in 1972 at AT&T. It is a combination of both low-level programming language as well as a high-level programming language. It is used to design applications like Text Editors, Compilers, Network devices, and many more.

Advantages

- C language is easy to learn.
- It is fast, efficient, portable, easy to extend, powerful, and flexible programming language.
- It is used to perform complex calculations and operations such as MATLAB.
- It provides dynamic memory allocation to allocate memory at the run time.

Most commonly used Programming Language

4. JavaScript

JavaScript is a type of scripting language that is used on both client-side as well as a server-side. It is developed in the 1990s for the Netscape Navigator web browser. It allows programmers to implement complex features to make web pages alive. It helps programmers to create dynamic websites, servers, mobile applications, animated graphics, games, and more.

Advantage

- JavaScript helps us to add behavior and interactivity on the web page.
- It can be used to decrease the loading time from the server.
- It has the ability to create attractive, dynamic websites, and rich interfaces.
- JavaScript is a simple, versatile, and lightweight programming language.

Devlopment Tools:

- PHPStorm
- VSCode
- Intellij IDEA
- Web storm
- Golang
- Net Beans
- SQLyog

Good programming practices:

Think of coding standards as a set of rules, techniques, and best practices to create cleaner, more readable, more efficient code with minimal errors. They offer a uniform format by which software engineers can use to build sophisticated and highly functional code.

Advantages of implementing Coding Standards

- Offers uniformity to the code created by different engineers.
- Enables the creation of reusable code.
- Makes it easier to detect errors.
- Make code simpler, more readable, and easier to maintain.
- Boost programmer efficiency and generates faster results.

Coding Standards & Best Practices To Follow

Readable code is easy to follow, optimizes space and time. Here are a few ways to achieve that:

- Write as few lines as possible.
- Use appropriate naming conventions.
- Use indentation to marks the beginning and end of control structures. Clearly specify the code between them.
- Don’t use lengthy functions. Ideally, a single function should carry out a single task.
- Use the DRY (Don’t Repeat Yourself) principle. Automate repetitive tasks whenever necessary. The same piece of code should not be repeated in the script.
- Avoid Deep Nesting. Too many nesting levels make code harder to read and follow.
- Capitalize SQL special words and function names to distinguish them from table and column names.
- Avoid long lines. It is easier for humans to read blocks of lines that are horizontally short and vertically long.

Operating System

- An operating system (OS) is a software program that manages the hardware and software resources of a computer.
- It acts as a intermediary between the computer's hardware and the programs that run on it, providing a layer of abstraction between the two.
- Some examples of operating systems include Windows, MacOS, and Linux.
- There are several advantages of an operating system, including:
- Resource management, security, compatibility, multitasking, memory management etc.

Objectives/Functions of Operating System

An operating system (OS) performs several functions to manage the resources of a computer and provide an interface for interacting with the computer. These functions include:

- **Memory management:** The OS manages the computer's memory, allocating memory to running programs and ensuring that the computer does not run out of memory.
- **Process management:** The OS manages the execution of processes, including creating, scheduling, and terminating processes. It includes allocating the processor to a process and deallocating when it no longer needs it. A process is the program in the state of execution. Process can be created, executed and stopped in a CPU.
    - Process Management Includes:-
        - Scheduling of process, process synchronizations, manage deadlock situation
        - Deadlock is situation where no single process do further operation.
- **Input/Output management:** The OS manages the communication between the computer and its input/output devices, such as the keyboard, mouse, and monitor.
- **File management:** The OS manages the organization and access of files on the computer's storage devices, including creating, deleting, and modifying files.

Objectives/Functions of Operating System

- **Security:** The OS provides security features, such as user authentication and file permissions, to prevent unauthorized access to the computer and its files.
- **Networking:** The OS manages the communication between the computer and other devices on a network.
- **Error detection and recovery:** The OS detects and recovers from errors, such as system crashes, to keep the computer running smoothly.
- **Device management:** The OS is responsible for managing the hardware devices, such as disk drives, printers, and other peripherals, and ensuring their proper functioning.
- **Resource allocation:** The OS allocates resources, such as CPU time and memory, to running programs to ensure that each program has the resources it needs to function properly.

Objectives/Functions of Operating System

- **User Interface: Provide** a user interface that allows users to interact with the computer system. May include command-line interfaces, graphical user interfaces (GUIs), or a combination of both.

Types of Operating System

- **Single-user, single-tasking:** These operating systems are designed to be used by one person at a time and can only run one program at a time. Examples include DOS and Windows 3.x.
- **Single-user, multi-tasking:** These operating systems are designed to be used by one person at a time, but can run multiple programs simultaneously. Examples include Windows and MacOS.
- **Multi-user:** These operating systems allow multiple users to access the computer simultaneously. Examples include UNIX and Linux.
- **Real-time:** These operating systems are designed to handle time-sensitive tasks, such as controlling industrial systems or managing scientific experiments. For example, the engine management system within a car uses a real-time operating system in order to react to the feedback from sensors placed throughout the engine. Examples include VxWorks, QNX, RTLINUX.

Types of Operating System

- **Batch Processing:** A batch processing operating system (OS) is a type of operating system that is specifically designed to manage and execute batch jobs. These systems are optimized for efficient processing of large volumes of data, and are often used in situations where a large number of similar tasks need to be executed.
- **Embedded:** These operating systems are designed to be used in small, specialized devices, such as projector, routers, and appliances like microwave oven, washing machines, traffic control machines etc. It is embedded in a device in the ROM. They are specific to a device and less resource intensive.
- **Mobile:** These operating systems are designed for mobile devices such as smartphones and tablets. Examples include Android and iOS.

Types of Operating System

- **Multiprogramming** is a method of executing multiple tasks or processes at the same time, using a single processor. The goal of multiprogramming is to increase the overall utilization of the processor by interleaving the execution of multiple tasks.
- **Multitasking** is a method of executing multiple tasks or processes at the same time, using a single processor or multiple processors. The goal of multitasking is to increase the overall performance and efficiency of a computer by allowing multiple tasks to be executed simultaneously.
- **Multiprocessing** is a method of executing multiple tasks or processes at the same time, using multiple processors or cores. The goal of multiprocessing is to increase the overall performance and speed of a computer by breaking down a large task into smaller, more manageable chunks and executing them simultaneously. Eg :- Linux, Unix, Windows 7/8/10/11.

Assignments

- Discuss about new trends in software.