The Future Track Computer Education

Operating Systems: Detailed Notes

1. Definition and Purpose

Operating System (OS): A software that acts as an interface between computer hardware and users.

Purpose:

- Manage hardware resources (CPU, memory, storage, etc.).
- Provide a user interface (GUI or CLI) to interact with the system.
- Enable the execution of applications.

2. Types of Operating Systems

Batch Operating Systems:

- Jobs are processed in batches without user interaction.
- Example: Early IBM systems.

Time-Sharing Operating Systems:

- Multiple users access the system simultaneously through time-sharing.
- Example: UNIX.

Distributed Operating Systems:

- Resources and tasks are distributed across multiple machines.
- Example: Google's cloud infrastructure.

Real-Time Operating Systems (RTOS):

- Tasks are processed within strict time constraints.

- Example: Systems in medical devices, robotics.

Embedded Operating Systems:

- Designed for specific devices with limited resources.

- Example: OS for IoT devices.

Network Operating Systems (NOS):

- Facilitates networking capabilities.

- Example: Windows Server, Linux.

Mobile Operating Systems:

- Optimized for mobile devices.

- Example: Android, iOS.

3. Core Components of an OS

Kernel:

- The core part of the OS, managing hardware resources and system calls.

- Types:

- Monolithic Kernel: Single large process (e.g., Linux).

- Microkernel: Minimal functions with other services running in user space (e.g., Minix).

Process Management:

- Handles creation, scheduling, and termination of processes.

- Concepts: Threads, Multitasking, Multithreading.

Memory Management:

- Manages allocation and deallocation of memory.
- Techniques: Paging, Segmentation, Virtual Memory.

File System Management:

- Organizes and stores data on storage devices.
- Examples: NTFS, FAT32, ext4.

Device Drivers:

- Facilitate communication between the OS and hardware devices.

User Interface:

- Graphical User Interface (GUI): e.g., Windows, macOS.
- Command Line Interface (CLI): e.g., Linux terminal.

4. Features of Operating Systems

- Multitasking: Running multiple processes simultaneously.
- Multithreading: Multiple threads within a single process.
- Multiprocessing: Support for multiple CPUs.
- Security and Access Control: Authentication, authorization, encryption.
- Device Independence: Uniform device access.

5. Popular Operating Systems

Windows:

- User-friendly GUI, widely used in personal and enterprise settings.
- Versions: Windows 11, 10, etc.

Linux:

- Open-source, highly customizable.- Distributions: Ubuntu, Fedora, CentOS.

macOS:

- Proprietary OS by Apple, known for its stability and aesthetics.

Android:

- Linux-based mobile OS by Google.

iOS:

- Apple's mobile OS, optimized for performance and security.

6. System Security

- User Authentication: Passwords, biometrics.

- Access Control Lists (ACLs): Define permissions for files and resources.

- Firewall and Antivirus: Protect against malware and unauthorized access.

7. Virtualization and Cloud OS

Virtual Machines (VMs):

- Simulate physical hardware to run multiple OS instances.

Hypervisors:

- Type 1 (Bare Metal): Runs directly on hardware (e.g., VMware ESXi).

- Type 2 (Hosted): Runs on top of a host OS (e.g., VirtualBox).

Cloud OS: Manages cloud infrastructure (e.g., Microsoft Azure, AWS).

8. OS Development and Programming

- Programming languages for OS: C, C++, Rust, Assembly.

- Common OS development practices:

- Low-level programming.

- Emphasis on performance and reliability.

9. Future Trends

- AI Integration: Intelligent task scheduling and resource management.

- Quantum OS: Designed for quantum computing platforms.

- Edge Computing OS: Optimized for edge devices.