Operating System Interview Questions

Index

What is Operating System & Types of OS

- An Operating System is a software that acts as an interface between computer hardware and user applications.
- It manages the resources and provides services for the efficient and secure execution of programs.
- The primary functions of an operating system include process management, memory management, file system management, device management and user interface.

Types of OS:

1. **Windows:**
    a. Developed by Microsoft (Founder: Bill Gates, Current CEO: Satya Nadella)
    b. Widely used OS personal computers.
2. **macOS:**
    a. Developed by Apple (Founder: Steve Jobs, Current CEO: Tim Cook)
    b. It is used on Apple Mac Computers.
3. **Linux:**
    a. An open-source OS that is highly customizable.
    b. Widley used in server environments and embedded systems.
    c. It is known for its stability, security and flexibility.
    d. Famous distribution: Ubuntu, Fedora, CentOS, Kali, etc.
4. **Unix:**
    a. A powerful multiuser operating system that serves as the foundation for many other operating systems including Linux and macOS.
    b. It is known for its stability and secure environment.
    c. Widely used in server applications.
5. Android:
    a. Developed by Google (Founder: Larry Page and Sergey Brin, Current CEO: Sundar Pichai)
    b. It is an open-source operating system maintained and governed by Google.
    c. Primarily designed for mobile devices such as smartphones and tablets, but now is used in Wear OS, Desktop, TV and Automative
6. iOS:
    a. Developed by Apple (Founder: Steve Jobs, Current CEO: Tim Cook)
    b. It is used in iPads and iPods.
7. Real Time Operating System (RTOS):

a. Widely used in embedded systems, control systems, and IoT devices.

Process vs Thread vs Program

Program

- A program is a set of instructions written in a programming language that performs a specific task or set of tasks.
- It is typically stored in a file on disk and represents an executable entity.
- Programs can be compiled or interpreted, and they serve as a blueprint for the execution of tasks on a computer system.

Process

- A process is an instance of a program in execution. When a program is loaded into memory and executed, it becomes a process.
- A process is an independent entity with its own PCB unit ie. memory space, resources, and execution context.
- PCB :
    - A Process Control Block (PCB) is a data structure used by the operating system to manage information about a process.
    - Process ID: A unique identifier for each process
    - Priority: The process's scheduling priority
    - State: The current state of the process (e.g., running, sleeping, waiting)
    - CPU Registers: The current values of CPU registers
    - Memory Pointers: Pointers to the process's memory segments
    - Open Files: A list of open files and their corresponding file descriptors
    - I/O Status: The current I/O status of the process
- Processes are managed by the operating system, and each process runs in its own protected memory space.
- Processes can be concurrent and communicate with each other through inter-process communication (IPC) mechanisms.

Thread

- A thread is a unit of execution within a process.
- It represents a sequence of instructions that can be scheduled and executed independently.
- Threads share the same memory space and resources within a process.
- Multiple threads within a process can run concurrently, allowing for parallel execution of tasks.

- Threads within the same process can communicate and share data more easily compared to inter-process communication (IPC).
- However, each thread has its own program counter and stack.

Multiprogramming vs Multiprocessing vs Multitasking vs Multithreading

Multiprogramming

- Multiprogramming is a technique where multiple programs are loaded into memory and executed concurrently by a single CPU.
- The CPU switches between programs, giving the illusion of simultaneous execution.
- Allows full utilization of CPU.
- **Example:** Consider a scenario where you have three programs: a text editor, a web browser, and a media player. In a multiprogramming environment, the CPU switches between these programs so that when one program is waiting for an I/O operation (like reading from a disk), the CPU can execute another program. This way, the CPU is kept busy almost all the time.

Multiprocessing

- Multiprocessing involves using multiple CPUs or CPU cores to execute multiple processes simultaneously.
- Each processor works on a different task or part of a task.
- **Example:** A modern computer with a quad-core processor running four different applications, each on its own core.

Multitasking

- Multitasking is the ability of an operating system to execute multiple tasks (processes) concurrently by rapidly switching between them.
- This creates the impression that multiple tasks are running simultaneously.
- Example: Using a smartphone to listen to music while browsing the web and receiving text messages.

Multithreading

- Multithreading is a programming concept where a single process is divided into multiple threads that can execute concurrently.
- Threads share the same memory space but can run independently.

Differences

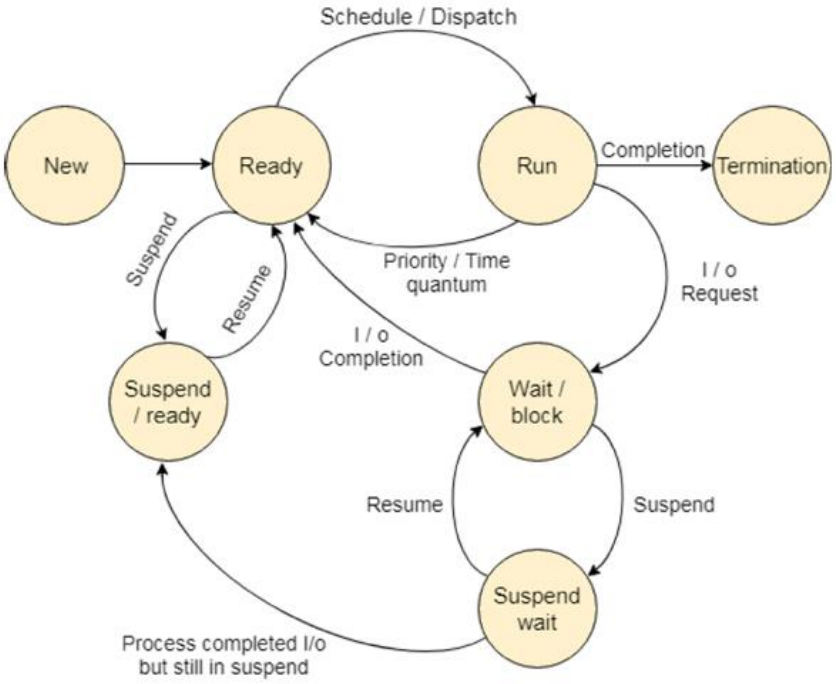
- Multiprogramming vs Multiprocessing:

    - Multiprogramming uses a single CPU to run multiple programs.
    - Multiprocessing uses multiple CPUs or cores to run multiple programs simultaneously.
- Multiprogramming vs Multitasking:
    - Multiprogramming focuses on running multiple programs concurrently.
    - Multitasking is a broader concept that includes both process-level and thread-level concurrency.
- Multiprocessing vs Multithreading:
    - Multiprocessing involves multiple CPUs working on separate tasks.
    - Multithreading involves multiple threads within a single process sharing the same memory space.
- Multiprocessing vs Multitasking:
    - Multiprocessing uses multiple CPUs to execute tasks simultaneously.
    - Multitasking rapidly switches between tasks to create the illusion of simultaneous execution, potentially on a single CPU.
- Multitasking vs Multithreading:
    - Multitasking deals with multiple processes with separate memory spaces.
    - Multithreading deals with multiple threads within a single process sharing the same memory space.

Summary of Differences

Concept	Definition	Example	Key Difference
Multiprogramming	Running multiple programs in memory and switching between them to maximize CPU utilization	Running a text editor, web browser, and media player simultaneously	Single CPU switches between multiple programs
Multiprocessing	Using multiple CPUs to execute processes simultaneously	Quad-core processor handling different processes	Multiple CPUs working on different processes at the same time
Multitasking	Executing multiple tasks simultaneously by switching rapidly between them	Writing a document while listening to music and downloading files	Can occur on a single CPU through time-sharing
Multithreading	Running multiple threads within a single process concurrently	Web browser rendering a page, downloading files, and running JavaScript simultaneously	Multiple threads within the same process share the same memory space, enabling parallel execution

Process State



The states of a process are as follows:

- **New State:** In this step, the process is about to be created but not yet created. It is the program that is present in secondary memory that will be picked up by the OS to create the process.
- **Ready State:** New -> Ready to run. After the creation of a process, the process enters the ready state i.e. the process is loaded into the main memory, that is, the process gets into the queue, which states that the process is ready to run and is waiting to get the CPU time for its execution.
- Run State: The process is chosen from the ready queue by the CPU for execution and the instructions within the process are executed by any one of the available CPU cores.
- Blocked or Wait State: Whenever the process requests access to I/O or needs input from the user or needs access to a critical region(the lock for which is already acquired) it enters the blocked or waits state. The process continues to wait in the main memory and does not require CPU. Once the I/O operation is completed the process goes to the ready state.
- Terminated or Completed State: Process is killed as well as PCB is deleted. The resources allocated to the process will be released or deallocated.
- Suspend Ready: Process that was initially in the ready state but was swapped out of main memory(refer to Virtual Memory topic) and placed onto external storage by the scheduler is said to be in suspend ready state. The process will transition back to a ready state whenever the process is again brought onto the main memory.
- Suspend Wait or Suspend Blocked: Similar to suspend ready but uses the process which was performing I/O operation and lack of main memory caused them to move to secondary memory. When work is finished it may go to suspend ready.

PCB

CPU Scheduling Algorithms

Critical Section Problem

Process Synchronization

Process Synchronization Mechanisms

Deadlock

Deadlock Handling Techniques

Memory Management

Partition and Memory Allocation

Paging

Virtual Memory

Page Replacement Algorithms

Thrashing

---

* Segmentation

---

* Disk Management

---

* Disk Scheduling Algorithms
