

OS Quick Revision

1. What is os(operating System)?

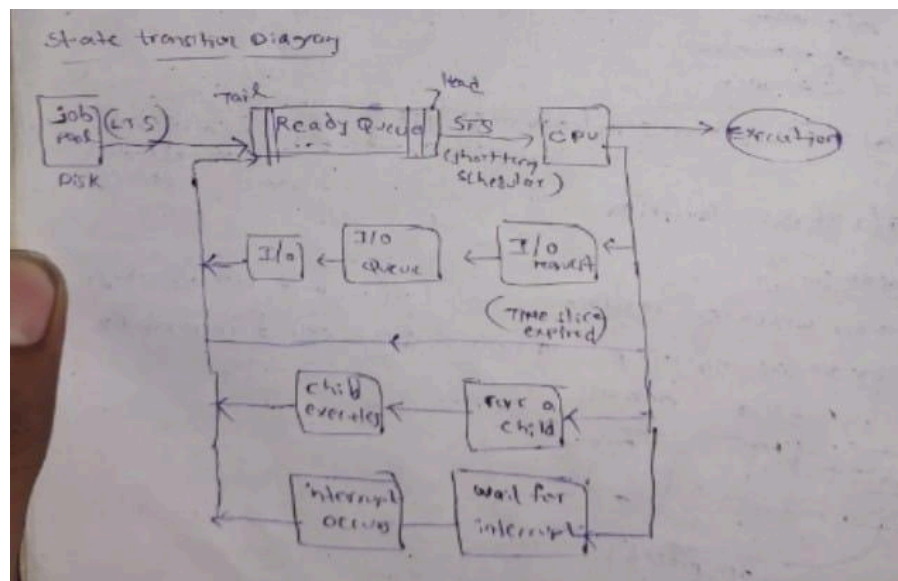
- An operating system is **system software** that manages computer hardware and software resources and provides a **platform for programs to run**. It acts as an **interface between the user and the computer hardware**.
- Key functions:
 - Process scheduling, memory management, file system management, device management, security and access control, user interface.

2. What is kernel?

- The core part of os that directly interacts with the hardware. It only includes low-level components like memory manager, scheduler. It doesn't interact with users.

3.Process scheduling:

- It is the activity of the os that decides which process runs at any given time. Crucial for multitasking, effective cpu utilization.
- Key concepts: cpu Scheduling (when multiple process are ready it determines which process need to be executed by cpu.
- Scheduling Criteria:
 - 1)Cpu utilization: keep the cpu as busy as possible.
 - 2)Throughput : Number os process completed in a given time.
 - 3)Turn around time: time taken from submission to competition
 - 4)Waiting time: time spend in ready queue.
 - 5)Response time: time from submission to first response.



- Preemptive Scheduling:
 - processes can be interrupted by the OS to give CPU time to others.
 - Used in most modern operating systems.
- Non-Preemptive (Cooperative):
 - A process must voluntarily yield control.
 - Simpler but can cause one process to dominate the CPU.

4. Cpu scheduling algorithms:

- FCFS(first come first serve) : simple but can cause long waiting times.
- SJF(shortest job first) : optimal for minimal average waiting time, but requires knowledge of future
- RR(round-robin) : Each process gets a fixed time slice (quantum), cycling through the ready queue. Fair for time sharing systems.
- Priority scheduling : Each process gets a priority assigned to it. The low priority process may be neglected for a longggg time.

5. Multiple processor scheduling: refers to how the OS handles scheduling of processes on a system with more than one cpu.

- Symmetric Multiprocessing (SMP):
 - All processors are treated equally.
 - Each processor runs its own scheduling algorithm.
 - Most modern OSes use SMP (e.g., Linux, Windows).
- Asymmetric Multiprocessing (AMP):
 - One processor is the master; others are slaves.
 - The master schedules and assigns tasks to slave processors.
 - Simpler but less scalable.

6. What is Inter-Process Communication(IPC)?

- Inter-Process Communication (IPC) is a mechanism that allows processes to communicate and share data with each other. This is essential when multiple processes need to coordinate their actions or exchange information.

7. Why IPC?

- Processes are isolated from each other. They often need to work together (e.g., a web server process and a database process).

8. What is race condition?

- Race condition occurs when two or more processes or threads access shared data at the same time, and the final result depends on the order in which they are executed.
- Simply If multiple threads are reading and writing to shared memory without proper synchronization (like locks or semaphores), they might interfere with each other, leading to unexpected or incorrect results.

9. What is a thread?

- A thread is a smallest unit of execution within a process.

10. What are Synchronization Primitives?

- Semaphores
 - Synchronization tools to control access to critical sections.
 - Two types:
 - Counting Semaphore: Allows multiple units (e.g., limited resources).
 - Binary Semaphore: Acts like a lock (0 or 1).
- Mutexes (Mutual Exclusion Locks)
 - A mutex is a locking mechanism used to protect shared resources.
 - Only one thread can hold the mutex at a time.
 - When a thread locks the mutex, other threads must wait until it is released.
- Monitors
 - High-level synchronization construct that combines mutual exclusion and condition variables.
 - Only one process can be active in the monitor at any time.
 - Simplifies handling multiple threads with safe access to shared data.

11. What is memory management?

- Allocating memory to process when they need it, removing from memory when it's no longer needed.
- Ensuring memory is used efficiently and securely.

12. What is swapping in memory management?

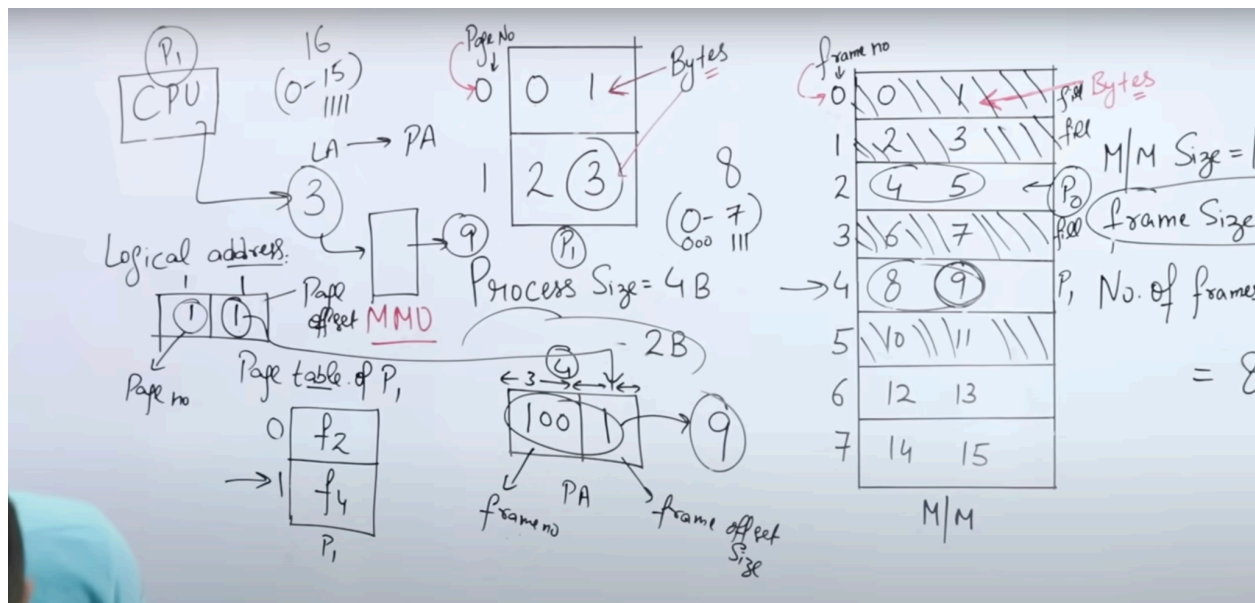
- Swapping is the process of temporarily transferring entire processes from main memory to disk (usually a swap file or partition) and back.
- Why it's used:
 - When RAM is full and another process needs to be loaded.
 - Supports multitasking.
- Draw backs:
 - Time-consuming due to slow disk access.
 - Can cause thrashing if done frequently
 - Thrashing means loading and unloading process from ram frequently.

13. Tell me about continuous memory allocation?

- Memory is allocated to a process in one single block of contiguous (adjacent) memory locations.
- Two methods:
- 1. Fixed Partitioning:
 - Memory is divided into fixed-size blocks (partitions).
 - Some memory may go unused (internal fragmentation).
- 2. Variable Partitioning:
 - Memory allocated dynamically based on process size.
 - Can lead to external fragmentation (free space scattered around).
- Example: If a 100 KB process is loaded into a 120 KB partition, 20 KB is wasted (internal fragmentation).
- Example(external fragmentation): Suppose the free memory is:
 - Block 1: 100 KB (free)
 - Block 2: 300 KB (used)
 - Block 3: 100 KB (free)
 - Now, a process needs 120 KB.
 - Total free memory = 100 KB + 100 KB = 200 KB
 - But both blocks are only 100 KB each, so:
 - Process can't be allocated, because no single block is 120 KB

14. What is paging?

- Paging is a non-contiguous memory allocation technique that eliminates external fragmentation by dividing memory into fixed-size blocks:
- Logical memory (process) is divided into blocks called pages.
- Physical memory (RAM) is divided into blocks called frames.
- Pages are mapped to available frames using a page table.



How Paging Works:


- Let's go step by step:
- 1. A process is divided into pages.
- 2. RAM is divided into frames of the same size as pages.
- 3. When the process is loaded:
 - The page table maps each page to a free frame.
 - Pages can be loaded anywhere in RAM, not necessarily contiguous.
- 4. The CPU uses:
 - Logical address = Page number + Offset
 - Translates it to a physical address using the page table.
- Example:
 - Page/frame size = 4 KB
 - Process size = 10 KB \rightarrow Requires 3 pages

- Free frames in memory: Frame 5, Frame 9, Frame 2
- The OS loads:
 - Page 0 → Frame 5
 - Page 1 → Frame 9
 - Page 2 → Frame 2
- Logical addresses are mapped to these physical frames via the page table.
- Advantages of Paging:
 - Eliminates external fragmentation
 - Efficient use of memory (process can be split across available frames)
 - Enables virtual memory (pages can be swapped in/out)

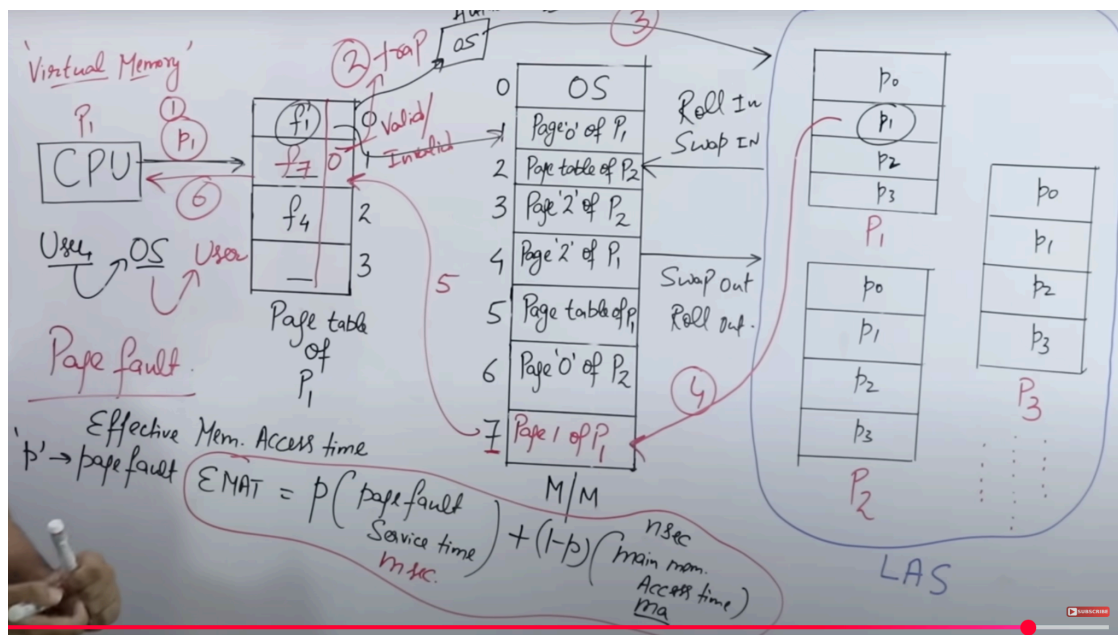
Paging don't has the external fragmentation.

15. What is segmentation?

- Segmentation is a memory management technique where a process is divided into variable-sized segments, based on the logical division of the program.
- Each segment represents a logical unit such as:
 - Code (instructions), Data (variables), Stack, Heap, Functions or arrays
- Disadvantages:
 - Can cause external fragmentation
 - Complex memory management compared to paging

Feature	Paging	Segmentation	
Memory Division	Divides memory into fixed-size pages.	Divides memory into variable-sized segments.	
Logical Unit	Pages are not related to the logical structure.	Segments correspond to logical divisions (e.g., code, data, stack).	
Size	Fixed size (e.g., 4KB).	Variable size.	
Address Translation	Uses a page number and offset.	Uses a segment number and offset.	
Fragmentation Type	May cause internal fragmentation .	May cause external fragmentation .	
Address Mapping	Page Table is used.	Segment Table is used.	
Transparency to User	Usually invisible to the programmer.	Exposed to the programmer (can define segments).	

16. What is virtual memory?



Virtual Memory is a memory management technique that allows a computer to run larger programs than the physical RAM by using a portion of the hard disk as if it were RAM.

It gives the illusion of a very large, continuous memory space to processes, even if the actual physical memory (RAM) is smaller.

How It Works:

The OS divides memory into pages.

Frequently used pages stay in RAM.

Less-used pages are moved to a swap space on the hard disk (called paging file or swap file).

When a process needs a page not in RAM, a page fault occurs, and the OS swaps it in from disk.

Key Components:

- **Logical Address Space:** The memory addresses a process uses (virtual addresses).

- Physical Address Space: Actual RAM.
- Page Table: Maps virtual addresses to physical memory (or disk).

Advantages:

- Run programs larger than RAM.
- Enables multitasking without worrying about exact memory limits.
- Isolates processes for better security and stability.

Disadvantages:

- Accessing disk is slower than RAM.
- Too much swapping can cause thrashing (system slows down severely).
- Requires careful management of page faults.

17. What is Page Replacement and its Algorithms?

When a page fault occurs (i.e., the page needed by a process is not in RAM), the OS must replace one of the pages currently in memory to bring in the required one. The strategy it uses is called a page replacement algorithm.

1. FIFO (First-In First-Out)

Oldest page in memory is replaced first.

Simple but may replace frequently used pages.

Example: Pages come in this order: 1, 2, 3, 4

Frame size = 3

After filling: [1, 2, 3]

Next page 4 comes → replace 1 (oldest)

Pros: Easy to implement

Cons: May remove useful pages (Belady's anomaly)

2. LRU (Least Recently Used)

Replaces the page that was used least recently.

Based on the idea that recently used pages will be used again soon.

Example: If pages 1, 2, and 3 are in memory, and page 2 was used long ago, it will be replaced first.

Pros: Good performance

Cons: Needs tracking of usage history (costly to implement)

3. Optimal Page Replacement (OPT)

Replaces the page that will not be used for the longest time in the future.

It gives the minimum number of page faults, but it's not practical (requires future knowledge).

Example: If pages needed in future are: 7, 0, 1, 2, 0, 3, 0, 4

If page 1 won't be needed for the longest time, it's replaced.

Pros: Best performance (theoretical)

Cons: Impossible in practice

18. What is Demand Paging?

Demand paging is a memory management technique where pages are loaded into RAM only when they are needed during program execution — not all at once.

It is a key concept in implementing virtual memory, helping systems run large programs even with limited physical RAM.

How It Works:

1. When a process starts, only a small portion (e.g., initial pages) is loaded into memory.
2. If the process tries to access a page that is not in memory:

A page fault occurs.

The operating system pauses the process.

The required page is fetched from disk (virtual memory) into RAM.

The process resumes execution.

19.What is Deadlock?

- A **deadlock** in an operating system occurs when a **two or more processes** are **stuck waiting** for each other to release resources, and **none of them can proceed**.
- It's a state where each process is **waiting indefinitely** for a resource that is held by another process, forming a **circular chain of dependency**.

Conditions for Deadlock

A deadlock can occur **only if** all of the following four conditions hold **simultaneously** (known as **Coffman's conditions**):

Condition	Description
Mutual Exclusion	At least one resource must be held in a non-shareable mode.
Hold and Wait	A process holding at least one resource is waiting to acquire more.
No Preemption	A resource cannot be forcibly taken from a process; it must be released voluntarily.
Circular Wait	A set of processes are waiting in a circular chain for resources.

Deadlock Handling Strategies

Strategy	Description
Deadlock Prevention	Prevent one of the four necessary conditions. E.g., disallow hold-and-wait.
Deadlock Avoidance	Use algorithms like Banker's Algorithm to avoid unsafe states.
Deadlock Detection	Allow deadlocks to occur and detect them using resource allocation graphs.
Deadlock Recovery	After detection, take actions like terminating processes or preempting resources.

20. explain Banker's algorithm?

Considering the following example of a system, check whether the system is safe or not, using Banker's algorithm. Determine the sequence if it is safe.

Process	Allocation	Max	Available	Need
P_0	0 1 0	7 5 3	3 3 2	7 4 3
P_1	2 0 0	3 2 2	5 3 2	1 2 2
P_2	3 0 2	9 0 2	4 4 3	6 0 0
P_3	2 1 1	2 2 2	7 4 5	0 1 1
P_4	0 0 2	4 3 3	10 5 7	4 3 1

Banker's algorithm:
 $Need_i \leq work \Rightarrow work = work + allocation_i$

P_0 $743 \leq 332$ X

P_1 $122 \leq 332$ ✓ $w = w + alloc = 332 + 200 = 532$

P_2 $600 \leq 532$ X

P_3 $011 \leq 532$ ✓ $w = w + alloc = 532 + 211 = 743$

P_4 $431 \leq 532$ ✓ $w = w + alloc = 743 + 002 = 745$

P_0 $743 \leq 745$ ✓ $w = w + alloc = 745 + 010 = 755$

P_2 $600 \leq 755$ ✓ $w = w + alloc = 755 + 302 = 1057$

$\langle P_1, P_3, P_4, P_0, P_2 \rangle$

21. What is Ostrich Algorithm?

- If there is deadlock occurs in the system, then the OS will just ignore the deadlock and reboot the system in order to function well.
- Use for deadlock recovery.

22. How to recover deadlock?

- If it is a Resource
 - Change non - primitive to Primitive
 - Roll back
- If it is a Process
 - Kill the process
 - Kill all the process