1. What is an Operating System?

It is system software that manages computer hardware, software resources, and provides services for programs.

---

2. What are the types of Operating Systems?

- Batch OS
- Time-sharing OS
- Distributed OS
- Real-time OS
- Embedded OS

---

3. What is the primary function of an OS?

To act as an intermediary between users and computer hardware, managing resources like CPU, memory, storage, and I/O devices.

---

4. What is a Kernel?

The core part of the OS responsible for managing system resources and communication between hardware and software.

---

5. What is Multitasking?

The ability of an OS to run multiple tasks (processes) simultaneously by sharing resources like the CPU.

---

6. What is Multiprocessing?

The use of two or more CPUs within a single computer system to execute multiple processes.

---

7. What is the difference between Multitasking and Multithreading?

- Multitasking: Running multiple processes.
- Multithreading: Running multiple threads within a single process.

---

8. What are system calls?

Interfaces between user-level applications and the OS that allow programs to request services like file handling or process management.

---

9. What is Virtual Memory?

A memory management technique that uses both physical RAM and disk space to give an application the illusion of having more memory.

---

10. What is Paging?

A memory management scheme that divides memory into fixed-sized blocks (pages) for efficient allocation.

---

11. What is Segmentation?

A memory management technique that divides the memory into variable-sized segments based on the logical division of a program.

---

12. What is a Process?

A program in execution, including its code, data, and associated resources.

---

13. What are the states of a Process?

- New
- Ready
- Running
- Waiting
- Terminated

---

14. What is Context Switching?

The process of saving the state of a process and loading the state of another during multitasking.

---

15. What is a Thread?

A lightweight process that is a subset of a process, sharing resources like memory.

---

16. What is Deadlock?

A situation where two or more processes cannot proceed because each is waiting for resources held by the other.

---

17. What are the conditions for Deadlock?

- Mutual Exclusion
- Hold and Wait
- No Preemption
- Circular Wait

---

18. What is a Semaphore?

A synchronization tool used to control access to shared resources by multiple processes.

---

19. What is Mutual Exclusion?

Ensures that only one process accesses a critical section at a time.

---

20. What is Starvation?

When a process waits indefinitely for resources because higher-priority processes are always given preference.

---

21. What is a Scheduler?

An OS component that decides which process will run next on the CPU.

---

22. Types of Scheduling Algorithms?

- FCFS (First-Come-First-Serve)
- SJF (Shortest Job First)
- Round Robin
- Priority Scheduling

---

23. What is Process Synchronization?

A mechanism to coordinate processes to avoid data inconsistency in shared resources.

---

24. What is an Interrupt?

A signal to the CPU indicating an event that needs immediate attention, like hardware failure or I/O completion.

---

25. What are I/O Bound and CPU Bound Processes?

- I/O Bound: Processes spending more time on I/O.
- CPU Bound: Processes spending more time on computation.

---

26. What is a File System?

A method for storing and organizing files on a storage device.

---

27. What is Thrashing?

A condition where excessive paging reduces system performance as the CPU spends more time swapping pages.

---

28. What is Swapping?

Moving processes between main memory and secondary storage to manage memory usage.

---

29. What is Spooling?

A technique of placing data in a temporary buffer for efficient device management, commonly used in printing.

---

30. What is Cache Memory?

A small, high-speed memory between the CPU and RAM to store frequently accessed data.

---

31. What is Bootstrapping?

The process of loading the operating system into memory when the computer starts.

---

32. What is the difference between Monolithic and Microkernel?

- Monolithic: Single large process with all OS services.
- Microkernel: Minimal OS functionality in the kernel, other services run in user space.

---

33. What is RAID?

Redundant Array of Independent Disks, a data storage virtualization technology for redundancy and performance.

---

34. What is Disk Scheduling?

Techniques to optimize the order in which disk I/O requests are serviced.

---

35. What is FCFS Scheduling?

Processes are executed in the order they arrive.

---

36. What is Priority Scheduling?

Processes are executed based on their priority.

---

37. What is Preemptive Scheduling?

A process can be interrupted and moved to the ready queue to allow a higher-priority process to execute.

---

38. What is Non-Preemptive Scheduling?

Once a process starts execution, it cannot be preempted until it finishes.

---

39. What is Round Robin Scheduling?

Processes are executed in a cyclic order for a fixed time slice.

---

40. What is Belady's Anomaly?

Increased page faults when increasing the number of page frames in certain page replacement algorithms.

---

41. What are Page Replacement Algorithms?

- FIFO
- LRU (Least Recently Used)
- Optimal

---

42. What is Memory Fragmentation?

Unused memory space caused by inefficient memory allocation:

- Internal: Within allocated space.

- External: Outside allocated space.

---

43. What is Demand Paging?

Loading only required pages into memory, reducing memory usage.

---

44. What is a Critical Section?

A section of code where shared resources are accessed, requiring synchronization.

---

45. What is Time Sharing?

Allowing multiple users to share system resources interactively.

---

46. What is a System Call?

A request from a user program to the OS for a service, like file operations or process management.

---

47. What is the difference between User Mode and Kernel Mode?

- User Mode: Limited privileges.
- Kernel Mode: Full access to hardware.

---

48. What is Load Balancing?

Distributing tasks across multiple CPUs or servers to optimize performance.

---

49. What is a Daemon Process?

A background process running without user interaction.

---

50. What is Fork()?

A system call in UNIX to create a new process by duplicating an existing process.

---

51. What is Exec()?

Replaces the memory of a process with a new program.

---

52. What is a Zombie Process?

A process that has completed execution but still has an entry in the process table.

---

53. What is a Shell?

A command-line interpreter that provides an interface between the user and the OS.

---

54. What is DMA (Direct Memory Access)?

A technique where devices transfer data to/from memory without CPU intervention.

---

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

A mechanism to allow processes to communicate and synchronize with each other.

---

56. What are Pipes in OS?

An IPC mechanism for unidirectional data flow between processes.

---

57. What is the difference between Hard and Soft Real-Time Systems?

- Hard: Missed deadlines are unacceptable.
- Soft: Missed deadlines are tolerable.

---

58. What is a Clustered System?

A group of interconnected computers working together as a single system.

---

59. What is the purpose of an OSI Model?

To standardize communication between heterogeneous systems across seven layers.

---

60. What is a Virtual Machine?

A software emulation of a physical computer, allowing multiple OS instances on a single hardware platform.

Important questions with detailed answers

1. What's the main purpose of an OS? What are the different types of OS?

The main purpose of an OS is to execute user programs and make it easier for users to understand and interact with computers as well as run applications. It is specially designed to ensure that the computer system performs better by managing all computational activities. It also manages computer memory, processes, and operation of all hardware and software.

Types of OS:

- Batched OS (Example: Payroll System, Transactions Process, etc.)
- Multi-Programmed OS (Example: Windows O/S, UNIX O/S, etc.)
- Timesharing OS (Example: Multics, etc.)
- Distributed OS (LOCUS, etc.)
- Real-Time OS (PSOS, VRTX, etc.)

2. What are the benefits of a multiprocessor system?

A Multiprocessor system is a type of system that includes two or more CPUs. It involves the processing of different computer programs at the same time mostly by a computer system with two or more CPUs that are sharing single memory.

Benefits:

- Such systems are used widely nowadays to improve performance in systems that are running multiple programs concurrently.
- By increasing the number of processors, a greater number of tasks can be completed in unit time.
- One also gets a considerable increase in throughput and is cost-effective also as all processors share the same resources.
- It simply improves the reliability of the computer system.

3. What is a bootstrap program in OS?

It is generally a program that initializes OS during startup i.e., first code that is executed whenever computer system startups. OS is loaded through a bootstrapping process or program commonly known as booting. Overall OS only depends on the bootstrap program to perform and work correctly. It is fully stored in boot blocks at a fixed location on the disk. It also locates the kernel and loads it into the main memory after which the program starts its execution.

4. What is different between main memory and secondary memory.

Main memory: Main memory in a computer is RAM (Random Access Memory). It is also known as primary memory or read-write memory or internal memory. The programs and data that the CPU requires during the execution of a program are stored in this memory.
Secondary memory: Secondary memory in a computer are storage devices that can store data and programs. It is also known as external memory or additional memory or backup memory or auxiliary memory. Such storage devices are capable of storing high-volume data. Storage devices can be hard drives, USB flash drives, CDs, etc.

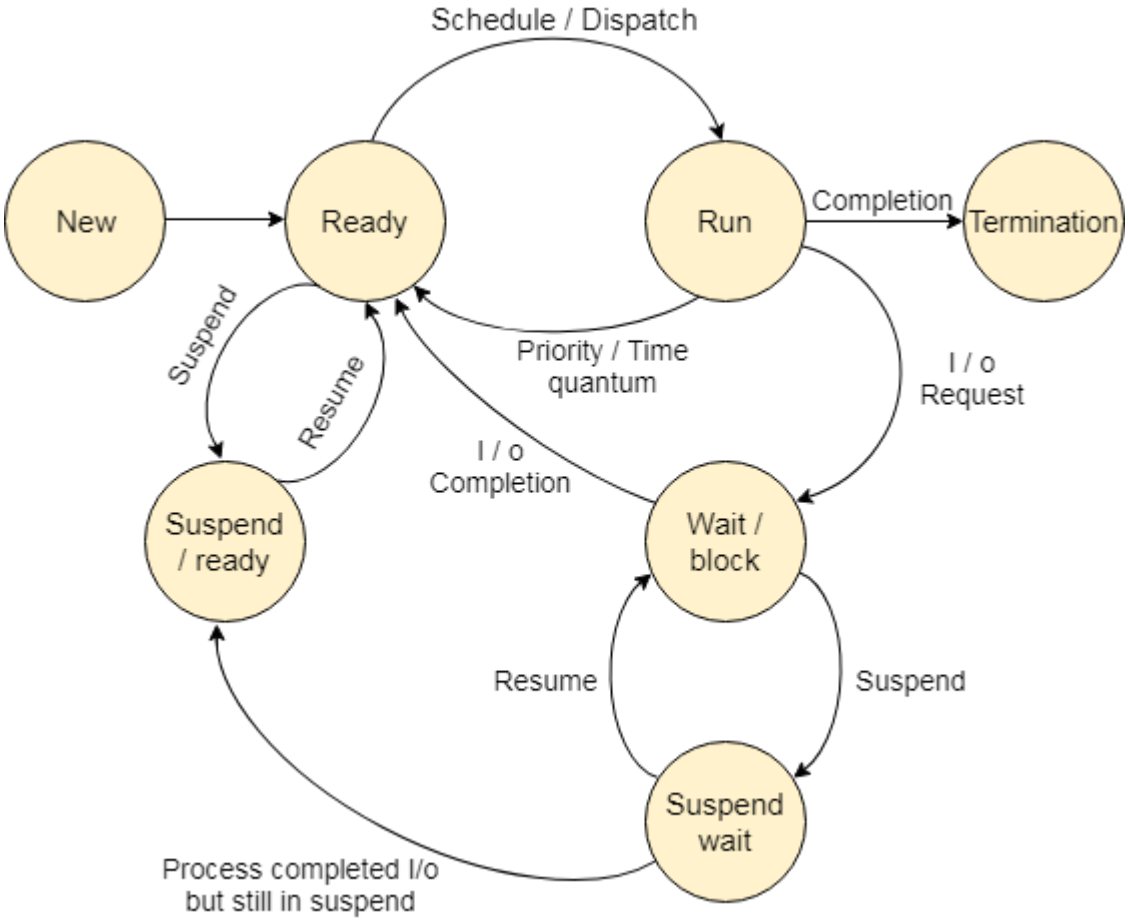
5. **What is the main objective of multiprogramming?**

It refers to the ability to execute or perform more than one program on a single processor machine. This technique was introduced to overcome the problem of underutilization of CPU and main memory. In simple words, it is the coordination of execution of various programs simultaneously on a single processor (CPU). The main objective of multiprogramming is to have at least some processes running at all times. It simply improves the utilization of the CPU as it organizes many jobs where the CPU always has one to execute.

6. **What is the difference between multitasking and multiprocessing OS?**

Multitasking	**Multiprocessing**
It performs more than one task at a time using a single processor.	It performs more than one task at a time using multiple processors.
In this, the number of CPUs is only one.	In this, the number of CPUs is more than one.
It is more economical.	It is less economical.
It is less efficient than multiprocessing.	It is more efficient than multitasking.
It allows fast switching among various tasks.	It allows smooth processing of multiple tasks at once.
It requires more time to execute tasks as compared to multiprocessing.	It requires less time for job processing as compared to multitasking.

7. Explain Process States



The process, from its creation to completion, passes through various states. The minimum number of states is five.

1. New

A program which is going to be picked up by the OS into the main memory is called a new process.

2. Ready

Whenever a process is created, it directly enters in the ready state, in which, it waits for the CPU to be assigned. The OS picks the new processes from the secondary memory and put all of them in the main memory.

The processes which are ready for the execution and reside in the main memory are called ready state processes. There can be many processes present in the ready state.

3. Running

One of the processes from the ready state will be chosen by the OS depending upon the scheduling algorithm. Hence, if we have only one CPU in our system, the number of running processes for a particular time will always be one. If we have n processors in the system then we can have n processes running simultaneously.

4. Block or wait

From the Running state, a process can make the transition to the block or wait state depending upon the scheduling algorithm or the intrinsic behavior of the process.

When a process waits for a certain resource to be assigned or for the input from the user then the OS move this process to the block or wait state and assigns the CPU to the other processes.

5. Completion or termination

When a process finishes its execution, it comes in the termination state. All the context of the process (Process Control Block) will also be deleted the process will be terminated by the Operating system.

6. Suspend ready

A process in the ready state, which is moved to secondary memory from the main memory due to lack of the resources (mainly primary memory) is called in the suspend ready state.

If the main memory is full and a higher priority process comes for the execution then the OS have to make the room for the process in the main memory by throwing the lower priority process out into the secondary memory. The suspend ready processes remain in the secondary memory until the main memory gets available.

7. Suspend wait

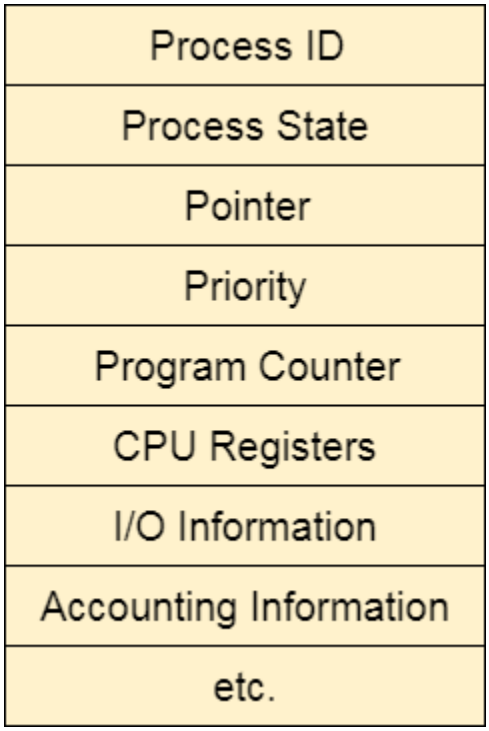
Instead of removing the process from the ready queue, it's better to remove the blocked process which is waiting for some resources in the main memory. Since it is already waiting for some resource to get available hence it is better if it waits in the secondary memory and make room for the higher priority process. These processes complete their execution once the main memory gets available and their wait is finished.

8. What is CPU Scheduling?

In Multiprogramming systems, the Operating system schedules the processes on the CPU to have the maximum utilization of it and this procedure is called CPU scheduling. The Operating System uses various scheduling algorithm to schedule the processes.

Process Control Block-

The Operating system maintains a process control block during the lifetime of the process. The Process control block is deleted when the process is terminated or killed. There is the following information which is saved in the process control block and is changing with the state of the process.



9. Explain Scheduling Algorithm?

1. First Come First Serve

It is the simplest algorithm to implement. The process with the minimal arrival time will get the CPU first. The lesser the arrival time, the sooner will the process gets the CPU. It is the non-preemptive type of scheduling.

2. Round Robin

In the Round Robin scheduling algorithm, the OS defines a time quantum (slice). All the processes will get executed in the cyclic way. Each of the process will get the CPU for a small amount of time (called time quantum) and then get back to the ready queue to wait for its next turn. It is a preemptive type of scheduling.

3. Shortest Job First

The job with the shortest burst time will get the CPU first. The lesser the burst time, the sooner will the process get the CPU. It is the non-primitive type of scheduling.

4. Shortest remaining time first

It is the pre-emptive form of SJF. In this algorithm, the OS schedules the Job according to the remaining time of the execution.

5. Priority based scheduling

In this algorithm, the priority will be assigned to each of the processes. The higher the priority, the sooner will the process get the CPU. If the priority of the two processes is same then they will be scheduled according to their arrival time.

6. Highest Response Ratio Next

In this scheduling Algorithm, the process with highest response ratio will be scheduled next. This reduces the starvation in the system.

10.What is process synchronization, and why is it needed?

Process synchronization ensures that multiple processes or threads execute in a way that maintains consistency and avoids data corruption when accessing shared resources. It is needed to handle critical sections, avoid race conditions, and ensure mutual exclusion.

11.What are the conditions for a Race Condition, and how can it be avoided?

Conditions: A race condition occurs when:

Two or more processes access shared resources simultaneously.

The execution order affects the program's outcome.

Avoidance: Use synchronization mechanisms like locks, semaphores, or monitors to control resource access.

12.What is a Semaphore, and how does it work?

A semaphore is a synchronization tool used to manage access to shared resources.

Types: Binary (0 or 1) and Counting (any value).

Working:

Wait(P): Decreases the semaphore value; blocks if the value is zero.

Signal(V): Increases the semaphore value, allowing blocked processes to proceed.

13.What is the difference between Mutex and Semaphore?

Aspect	Mutex	Semaphore
Ownership	Owned by a single thread.	No ownership concept.
Synchronization Type	For process threads only.	Can handle multiple threads.
Value Range	Binary (0 or 1).	Binary or counting.

14.What is a Deadlock, and how is it related to synchronization? How can it be prevented?

Deadlock: A situation where two or more processes are waiting indefinitely for resources held by each other.

Relation to Synchronization: Improper synchronization can lead to deadlock when processes fail to release or acquire resources in a controlled manner.

Prevention:

Avoid circular wait by resource ordering.

Use timeouts or preemption.

Apply a deadlock prevention algorithm like the Banker’s Algorithm.

15. What is a Deadlock, and what are its necessary conditions?

A deadlock is a state in a system where two or more processes are unable to proceed because each is waiting for resources held by the others.

Necessary Conditions (Coffman Conditions):

Mutual Exclusion: Only one process can access a resource at a time.

Hold and Wait: A process holding resources can request additional resources.

No Preemption: Resources cannot be forcibly taken from a process.

Circular Wait: A circular chain of processes exists, each waiting for a resource held by the next.

16. How can deadlocks be prevented, avoided, or detected?

Prevention: Break one or more of the necessary conditions:

Eliminate circular wait (impose resource ordering).

Allow preemption of resources.

Avoidance: Use algorithms like Banker's Algorithm to ensure the system remains in a safe state.

Detection and Recovery: Periodically check for deadlock (using resource allocation graphs) and take recovery actions, such as killing a process or forcibly preempting resources.

17.What is the difference between Deadlock and Starvation?

Deadlock: All involved processes are blocked, and none can proceed due to resource dependencies.

Starvation: A process waits indefinitely because high-priority processes are always favored.

Key Difference: Deadlock involves resource dependencies among processes, while starvation is due to unfair scheduling policies.

18.What is Memory Management in an Operating System?

Memory management is a core function of the operating system that handles the allocation and deallocation of memory to processes, ensuring efficient and safe use of the available memory.

19. What are the differences between Physical and Virtual Memory?

Physical Memory: Refers to the actual hardware RAM installed in the system.

Virtual Memory: A memory management technique that uses a combination of physical RAM and disk space to simulate more memory than physically available.

20. What is Paging, and why is it used?

Paging is a memory management scheme that divides physical memory into fixed-size blocks called pages and logical memory into blocks of the same size. It eliminates external fragmentation and allows processes to use non-contiguous memory.

21. What is the difference between Paging and Segmentation?

Paging: Divides memory into fixed-size pages.

Segmentation: Divides memory into variable-sized segments based on program structures like functions or data.

Key Difference: Paging focuses on fixed-size blocks, while segmentation focuses on logical divisions.

22.What is Fragmentation, and what are its types?

Fragmentation occurs when memory is inefficiently allocated, leaving unusable spaces.

Types:

Internal Fragmentation: Unused space within allocated memory blocks.

External Fragmentation: Unused space outside allocated memory blocks.

23.What is a Page Fault? How is it handled?

A page fault occurs when a process tries to access a page not currently in physical memory.

Handling:

OS pauses the process.

Brings the required page from disk to memory.

Restarts the process.

24. What are Page Replacement Algorithms?

Algorithms to decide which page to remove from memory when a page fault occurs.
Common algorithms include:

FIFO (First-In-First-Out)

LRU (Least Recently Used)

Optimal (minimizes future page faults, often theoretical).

25.What is Thrashing, and how can it be prevented?

Thrashing: Excessive paging activity occurs when a system spends more time swapping pages than executing processes.

Prevention:

Adjust the degree of multiprogramming.

Increase physical memory.

Use working set models to allocate sufficient memory to processes.

26. Paging in Memory Management full details

Paging is a memory management scheme that eliminates the need for contiguous allocation of physical memory, making memory utilization more efficient. It divides the process's logical memory and physical memory into fixed-size blocks called **pages** and **frames**, respectively.

Key Concepts in Paging

1. **Page**: A fixed-size block of logical memory.
2. **Frame**: A fixed-size block of physical memory.
3. **Page Table**: A data structure maintained by the OS that maps logical page numbers to physical frame numbers.
4. **Page Size**: The size of a page, equal to the frame size, typically in powers of 2 (e.g., 4 KB).

How Paging Works

1. **Logical Address**: Generated by the CPU, split into two parts:
    - **Page Number (P)**: Identifies the page in the logical address space.
    - **Page Offset (D)**: Identifies the specific byte within the page.

Logical Address = Page Number (P)+Page Offset (D)\text{Page Number (P)} + \text{Page Offset (D)}Page Number (P)+Page Offset (D)

2. **Physical Address**: Calculated by the OS using the page table:
    - Maps the page number to a frame number.
    - Combines the frame number and offset to generate the physical address.

Physical Address = Frame Number (F)+Page Offset (D)\text{Frame Number (F)} + \text{Page Offset (D)}Frame Number (F)+Page Offset (D)

Steps in Paging

1. The CPU generates a logical address.
2. The page number is extracted and used to index the page table.
3. The page table provides the corresponding frame number in physical memory.
4. The frame number is combined with the offset to get the physical address.
5. The physical address is used to access the desired location in RAM.

Example of Paging

Given:

- Logical Memory = 16 pages (each page is 1 KB).
- Physical Memory = 8 frames (each frame is 1 KB).

Logical to Physical Mapping:

Page Number	Frame Number
0	5
1	3
2	1
3	6
...	...

Logical Address:

- CPU generates a logical address: Page Number = 2, Offset = 200 bytes\text{Page Number = 2, Offset = 200 bytes}Page Number = 2, Offset = 200 bytes.

Translation:

1. Page table shows Page 2 maps to Frame 1.
2. Physical Address = Frame 1+Offset 200\text{Frame 1} + \text{Offset 200}Frame 1+Offset 200.
3. Physical Address = 1×1024+200=12241 \times 1024 + 200 = 12241×1024+200=1224.

Advantages of Paging

1. **Eliminates External Fragmentation**: Any free frame can be allocated.
2. **Efficient Memory Use**: No need for contiguous allocation.
3. **Ease of Swapping**: Pages can be swapped in and out of memory independently.

Disadvantages of Paging

1. **Internal Fragmentation**: Some memory within a frame may remain unused.

2. **Overhead**: Maintaining and accessing the page table requires additional time and memory.
3. **Page Faults**: If the required page is not in memory, fetching it from disk can slow down execution.
