CHHATRAPATI SHAHU JI MAHARAJ UNIVERSITY, KANPUR

ACCREDITED WITH GRADE A++ NAAC

प्रश्नBANK

Bridge of Academic Novelties in Knowledge

KANPUR UNIVERSITY'S

QUESTION BANK

Brief and Intensive Notes
Multiple Choice Questions

Based On NEP 2020

OPERATING SYSTEM

B.Sc. II SEM

DR. AGNIVESH

MRS. SANCHITA LAXMI

Course Title: Operating System

Unit I

Introduction Operating system and functions, Classification of Operating systems: Batch, Interactive, Time sharing, Real Time System, Multiprocessor Systems, Multiuser Systems, Multithreaded Systems, Operating System Structure, System Components, Operating System Services, Kernels, Monolithic and Microkernel Systems.

Operating System and Its Functions

What is an Operating System?

An **Operating System (OS)** is system software that manages computer hardware and software resources and provides common services for computer programs. It acts as an intermediary between users and the computer hardware.

Main Functions of an Operating System:

1. **Process Management**: Controls the creation, execution, and termination of processes.
2. **Memory Management**: Manages the system's primary memory (RAM), keeping track of each byte.
3. **File System Management**: Handles file creation, deletion, reading, writing, and access permissions.
4. **Device Management**: Manages device communication via drivers.
5. **Security and Access Control**: Protects system resources against unauthorized access.
6. **User Interface**: Provides a user interface (CLI or GUI) for interaction.
7. **Resource Allocation**: Allocates hardware resources as needed to processes.
8. **Error Detection and Handling**: Monitors the system for errors and takes corrective action.

Classification of Operating Systems

1. Batch Operating System

- Executes batches of jobs without user interaction.
- Jobs are grouped and processed sequentially.
- **Example**: Early IBM mainframes.
- **Pros**: Efficient for large jobs.
- **Cons**: No user interaction during execution.

2. Interactive Operating System

- Allows user interaction with running applications.
- Input and response occur in real-time or near-real-time.

- **Example**: Windows, macOS.
- **Use case**: Desktop computing.

3. Time Sharing Operating System

- Multiple users use the system simultaneously via terminals.
- CPU time is divided among users.
- **Example**: UNIX.
- **Pros**: Increases responsiveness.
- **Cons**: Complex resource management.

4. Real-Time Operating System (RTOS)

- Provides immediate processing and response.
- **Hard RTOS**: Strict timing constraints (e.g., medical systems).
- **Soft RTOS**: Flexible timing constraints (e.g., multimedia).
- **Examples**: VxWorks, QNX.

5. Multiprocessor Systems

- Use two or more CPUs within a single computer system.
- CPUs share memory and devices.
- **Types**: Symmetric and Asymmetric multiprocessing.
- **Benefits**: Increased throughput, reliability.

6. Multiuser Systems

- Multiple users access the computer simultaneously.
- Each user has a terminal and allocated resources.
- **Example**: UNIX servers.
- **Use case**: Mainframes, server environments.

7. Multithreaded Systems

- Supports multiple threads within a process.
- Each thread shares the process's resources but executes independently.
- **Advantage**: Efficient use of CPU, better performance.

Operating System Structure

OS structure defines how its components are organized and interact.

Common Structures:

Monolithic Systems**: All OS services are in one large block of code in a single address space.**

1. **Layered Approach**: OS divided into layers, each built on top of lower ones.
2. **Microkernel Architecture**: Only essential components (like memory management, IPC) run in kernel mode; others run in user space.
3. **Modules**: Loadable modules allow dynamic linking and management of services.
4. **Hybrid Systems**: Combination of monolithic and microkernel, used in modern systems like Windows and macOS.

System Components of an Operating System

1. **Process Management**
2. **Main Memory Management**
3. **File Management**
4. **I/O System Management**
5. **Secondary Storage Management**
6. **Networking**
7. **Security and Protection System**
8. **Command Interpreter System**

Operating System Services

1. **Program Execution**
2. **I/O Operations**
3. **File System Manipulation**
4. **Communication**
5. **Error Detection**
6. **Resource Allocation**
7. **Accounting**
8. **Security and Protection**

Kernels in Operating Systems

Kernel:

The core of an OS, managing communication between hardware and software.

Types of Kernels:

1. **Monolithic Kernel**:
   - All OS services run in kernel space.
   - Fast but complex and difficult to maintain.
   - **Example**: Linux.
2. **Microkernel**:
   - Only essential services in the kernel; others in user space.
   - Easier to maintain, more secure, but slightly slower due to more context switches.
   - **Example**: Minix, QNX.

1. What is the primary purpose of an operating system?
A) Compile programs
B) Manage hardware and software resources
C) Translate user code
D) Control applications only
Answer: B
Explanation: The OS manages hardware/software and provides services to applications.

2. Which of the following is not a function of an operating system?
A) Memory Management
B) Process Management
C) Virus Scanning
D) File Management
Answer: C
Explanation: Virus scanning is typically done by antivirus software, not the OS.

3. Which component of the OS handles task scheduling?
A) File Manager
B) Process Scheduler
C) Device Driver
D) Shell
Answer: B
Explanation: The process scheduler manages the execution of processes.

4. Which OS function is responsible for protecting data from unauthorized access?
A) I/O Management
B) File Management
C) Security
D) Scheduling
Answer: C
Explanation: The OS ensures secure access through permissions and encryption.

5. The OS acts as a(n) _______ between user and hardware.
A) Compiler
B) Interface
C) Translator
D) Emulator
Answer: B
Explanation: The OS provides an interface that hides hardware complexities.

6. Batch systems are suitable for:
A) Interactive user sessions
B) Time-sharing applications
C) Long-running, repetitive tasks
D) Real-time processing
Answer: C
Explanation: Batch systems process jobs with similar needs in groups.

7. Which OS allows multiple users to use the system at the same time?
A) Batch
B) Multiuser
C) Real-Time
D) Embedded
Answer: B
Explanation: Multiuser systems provide concurrent access to several users.

8. Which system ensures immediate response to events?
A) Time-sharing
B) Real-time
C) Batch
D) Multiuser
Answer: B
Explanation: Real-time OS is designed for time-critical tasks.

9. Which type of OS shares CPU time among users to give a responsive experience?
A) Batch
B) Real-time
C) Time-sharing
D) Interactive
Answer: C
Explanation: Time-sharing OS provides each user a small time slice.

10. In a multiprocessor system:
A) Only one CPU is used
B) CPUs run different OSes
C) CPUs share memory and tasks
D) Used only for gaming
Answer: C
Explanation: Multiprocessor systems use multiple CPUs in a coordinated fashion.

11. Which OS allows users to interact with the system via a GUI or command line?
A) Batch
B) Interactive
C) Real-time
D) Multiuser
Answer: B
Explanation: Interactive OS supports user interaction during execution.

12. Which of the following supports execution of multiple threads within a process?
A) Batch system
B) Multithreaded OS
C) Single-user OS
D) Micro OS
Answer: B
Explanation: Multithreaded systems enhance performance and responsiveness.

13. Multithreading is useful for:
A) Increasing hardware size
B) Supporting multiple devices
C) Improving program efficiency
D) Storing files faster
Answer: C
Explanation: Threads help execute parts of a program in parallel.

14. What makes multiuser OS different from single-user OS?
A) Faster speed
B) Shared system resources
C) Costlier hardware
D) Ability to run one program
Answer: B
Explanation: Multiuser OS allocates resources to multiple users at once.

15. In time-sharing OS, the CPU time is:
A) Given to one user always
B) Shared equally among users
C) Randomly assigned
D) Reserved for admin
Answer: B
Explanation: It provides a fair time slice to each user.

16. Which structure combines all OS functions in a single large block?
A) Layered
B) Monolithic
C) Microkernel
D) Modular
Answer: B
Explanation: A monolithic kernel contains all core functions in one program.

17. A layered OS design helps in:
A) Increasing execution time
B) Simplifying debugging
C) Making a single-layer system
D) Replacing user programs
Answer: B
Explanation: Layering simplifies system design and testing.

18. Microkernel architecture aims to:
A) Add all functions in the kernel
B) Remove all hardware access
C) Minimize kernel responsibilities
D) Exclude system calls
Answer: C
Explanation: Microkernels only keep essential services in the kernel.

19. Hybrid kernel is:
A) Non-kernel based OS
B) Combines monolithic and microkernel ideas
C) Kernel-less OS
D) DOS-based
Answer: B
Explanation: Hybrid kernels blend performance and modularity.

20. Which OS structure runs most services in user space?
A) Monolithic
B) Layered
C) Microkernel
D) Real-time
Answer: C
Explanation: Microkernels run components like drivers in user mode.

21. The OS component responsible for file operations is:
A) Scheduler
B) File Manager
C) Memory Manager
D) Loader
Answer: B
Explanation: File manager handles creation, deletion, and access control.

22. Which component allocates space in RAM?
A) File manager
B) Device driver
C) Memory manager
D) Scheduler
Answer: C
Explanation: Memory manager tracks and allocates RAM usage.

23. I/O system management involves:
A) Deleting processes
B) Handling file systems
C) Managing device communication
D) Terminating the OS
Answer: C
Explanation: It ensures smooth operation between CPU and I/O devices.

24. Which component manages process states and transitions?
A) Kernel
B) Command Interpreter
C) Process Manager
D) File System
Answer: C
Explanation: Process manager oversees the lifecycle of processes.

25. Networking in OS refers to:
A) Accessing the hard disk
B) Managing processes
C) Connecting systems for resource sharing
D) Switching OS modes
Answer: C
Explanation: Networking enables resource sharing between computers.

26. Program execution service:
A) Compiles code
B) Loads and runs user programs
C) Provides printers
D) Writes OS itself
Answer: B
Explanation: OS loads and starts execution of user programs.

27. Which service allows OS-user communication?
A) File system
B) Shell
C) System call interface
D) Kernel
Answer: C
Explanation: System calls provide an interface for user programs to interact with the OS.

28. Error detection service is important for:
A) Debugging hardware
B) Monitoring system errors and taking action
C) Enhancing GUI
D) Adding new users
Answer: B
Explanation: It helps in fault detection and stability.

29. Accounting in OS means:
A) Counting users
B) Keeping track of resource usage
C) Preventing security breaches
D) Adding interest to files
Answer: B
Explanation: OS keeps logs of usage for performance and billing.

30. Which of the following is not a typical OS service?
A) Communication
B) Resource Allocation
C) Compiler Optimization
D) Security
Answer: C
Explanation: Compiler optimization is part of the compiler, not the OS.

31. The core component of an OS is:
A) Shell
B) Kernel
C) Command Interpreter
D) Loader
Answer: B
Explanation: The kernel manages low-level operations.

32. Monolithic kernels are generally:
A) Slower
B) Modular
C) Large and fast
D) Based on microservices
Answer: C
Explanation: All services run in one address space, making it fast but bulky.

33. Which of the following is an advantage of microkernels?
A) Fast execution
B) High reliability and modularity
C) All services in one module
D) Lack of abstraction
Answer: B
Explanation: Microkernels isolate services, improving stability.

34. A drawback of microkernel systems is:
A) Poor modularity
B) Excessive security
C) Increased overhead from context switching
D) Lack of multitasking
Answer: C
Explanation: More communication between services increases overhead.

35. Which OS uses microkernel architecture?
A) Linux
B) QNX
C) MS-DOS
D) Windows 95
Answer: B
Explanation: QNX is a well-known microkernel-based OS.

36. Which of the following is not typically part of a microkernel?
A) Memory Management
B) File System
C) Inter-Process Communication (IPC)
D) CPU Scheduling
Answer: B
Explanation: In microkernel architecture, file systems typically run in user space, not in the kernel.

37. Which system design supports dynamic loading of OS components?
A) Monolithic kernel
B) Modular kernel
C) Batch OS
D) Real-time OS
Answer: B
Explanation: Modular kernels support loading/unloading kernel modules at runtime.

38. What is the key disadvantage of monolithic kernels?
A) Low performance
B) High complexity and tight coupling
C) Poor multitasking
D) No hardware support
Answer: B
Explanation: All services being in one block increases complexity and affects maintainability.

39. Which kernel type is easier to debug and extend?
A) Monolithic
B) Microkernel
C) Hybrid
D) None of the above
Answer: B
Explanation: Microkernel's modular design makes debugging and extending easier.

40. Hybrid kernels attempt to combine:
A) Multiuser and batch systems
B) Real-time and batch systems
C) Monolithic and microkernel advantages
D) Hardware and software
Answer: C

Explanation: Hybrid kernels aim for performance like monolithic and modularity of microkernels.

41. What happens when a process makes a system call?
A) It is terminated
B) It switches to user mode
C) It requests a service from the OS
D) It gets high priority
Answer: C
Explanation: System calls are interfaces through which processes request OS services.

42. Which of the following is used to switch from user mode to kernel mode?
A) Process creation
B) System call
C) Interrupt
D) Both B and C
Answer: D
Explanation: Both system calls and interrupts trigger a switch to kernel mode.

43. Which of these is not a system component?
A) Memory Manager
B) I/O System
C) Web Browser
D) File System
Answer: C
Explanation: A web browser is an application, not a system component.

44. Time-sharing OS improves:
A) Compilation speed
B) User interaction and responsiveness
C) I/O performance only
D) Network bandwidth
Answer: B
Explanation: It allows users to interact with programs in near-real-time.

45. Which scheduling technique is typically used in time-sharing OS?
A) First-Come First-Served
B) Round Robin
C) Priority Scheduling
D) Shortest Job First

Answer: B
Explanation: Round Robin ensures fairness by giving each process a time slice.

46. Which is an example of a time-sharing operating system?
A) MS-DOS
B) Unix
C) VxWorks
D) Windows 3.1
Answer: B
Explanation: Unix supports multiple users and multitasking with time-sharing.

47. Which type of OS would be most suitable for a missile guidance system?
A) Interactive
B) Batch
C) Real-time
D) Time-sharing
Answer: C
Explanation: Real-time OS is required for strict timing and immediate response.

48. What does "context switch" mean in OS?
A) Changing hardware settings
B) Switching from kernel mode to user mode
C) Saving state of one process and loading another
D) Restarting a program
Answer: C
Explanation: It is the mechanism to switch between processes.

49. Which of the following is not a classification of operating system?
A) Real-time
B) Interactive
C) Reflective
D) Batch
Answer: C
Explanation: Reflective is not a standard OS classification.

50. The mechanism that ensures only authorized users access system resources is called:
A) Resource Allocation
B) Accounting
C) Security
D) I/O Control

Answer: C
Explanation: Security ensures controlled access to resources.

Unit II

Process Management Process Concept, Process States, Process Synchronization, Critical Section, Mutual Exclusion, Classical Synchronization Problems, Process Scheduling, Process States, Process Transitions, Scheduling Algorithms Interprocess Communication, Threads and their management, Security Issues.

Process Management

Process management is a core function of the operating system. It handles the creation, execution, synchronization, and termination of processes.

1. Process Concept

A **process** is a program in execution. It consists of:

- **Program code** (text section)
- **Program counter**
- **Stack** (function calls, parameters, return addresses)
- **Heap** (dynamically allocated memory)
- **Data section** (global variables)

Each process is represented by a **Process Control Block (PCB)** containing:

- Process state
- Program counter
- CPU registers
- Scheduling information
- Memory management info
- I/O status

2. Process States

A process typically exists in one of the following states:

1. **New**: Being created.
2. **Ready**: Waiting to be assigned to the CPU.
3. **Running**: Instructions are being executed.
4. **Waiting/Blocked**: Waiting for some event (e.g., I/O).
5. **Terminated**: Process has finished execution.

3. Process Transitions

Processes can transition between states:

- **New → Ready**: Admitted by OS.
- **Ready → Running**: Scheduled by CPU.
- **Running → Waiting**: Waiting for I/O.
- **Running → Ready**: Interrupted.
- **Waiting → Ready**: I/O complete.
- **Running → Terminated**: Execution completed or aborted.

4. Process Synchronization

In multiprogramming systems, processes may access shared data. To prevent conflicts, **synchronization** is required.

Goals:

- Prevent **race conditions** (where outcomes depend on timing).
- Ensure **data consistency**.
- Enable **safe cooperation** among processes.

5. Critical Section

A **critical section** is a portion of code that accesses shared resources and must not be executed by more than one process at a time.

Solution Requirements (for Critical Section Problem):

1. **Mutual Exclusion**: Only one process can be in the critical section at a time.
2. **Progress**: No process outside the critical section can block others from entering.
3. **Bounded Waiting**: A limit on the number of times other processes can enter before a waiting one does.

6. Mutual Exclusion

Mutual exclusion ensures that concurrent processes do not simultaneously execute critical sections.

Mechanisms:

- **Software**: Peterson’s algorithm, Bakery algorithm
- **Hardware**: Test-and-set, Compare-and-swap
- **Semaphores/Locks/Monitors**: OS-level constructs

7. Classical Synchronization Problems

These illustrate synchronization challenges:

1. **Bounded Buffer Problem (Producer-Consumer)**
   - Shared buffer between producers and consumers.
   - Needs synchronization to avoid overflow/underflow.
2. **Readers-Writers Problem**
   - Many readers or one writer at a time.
   - No data inconsistency should occur.
3. **Dining Philosophers Problem**
   - Philosophers share chopsticks; synchronization avoids deadlock and starvation.

8. Process Scheduling

The OS decides **which process to run next**. This decision is based on the **scheduling algorithm** used.

Scheduler Types:

- **Long-term**: Selects which processes are admitted to the system.
- **Short-term** (CPU scheduler): Selects among ready processes.
- **Medium-term**: Swaps processes in/out of memory.

9. Scheduling Algorithms

1. **First-Come, First-Served (FCFS)**: Non-preemptive, simple but may cause long wait times.
2. **Shortest Job Next (SJN)**: Non-preemptive, optimal average waiting time but impractical.
3. **Round Robin (RR)**: Preemptive, each process gets a time slice (quantum).
4. **Priority Scheduling**: Each process assigned a priority.
5. **Multilevel Queue Scheduling**: Multiple queues with different priorities.
6. **Multilevel Feedback Queue**: Dynamic process movement between queues.

10. Interprocess Communication (IPC)

IPC allows processes to exchange information and synchronize actions.

Types:

- **Shared Memory**: Processes communicate via a common memory region.
- **Message Passing**: Processes send and receive messages (e.g., pipes, sockets).

Key Concepts:

- **Direct/Indirect communication**
- **Synchronous/Asynchronous communication**
- **Blocking/Non-blocking**

11. Threads and Their Management

Thread: A lightweight process, part of a process that shares resources.

Types:

- **User-Level Threads (ULT)**: Managed in user space.
- **Kernel-Level Threads (KLT)**: Managed by the OS.
- **Multithreading Models**:
  - **Many-to-One**: Many user threads mapped to one kernel thread.
  - **One-to-One**: One user thread to one kernel thread.
  - **Many-to-Many**: Many user threads to many kernel threads.

Thread Operations:

- Creation, execution, synchronization, and termination.
- Thread libraries: POSIX Pthreads, Windows threads, Java threads.

12. Security Issues in Process Management

Process management must ensure:

- **Isolation**: Processes do not interfere with each other.
- **Authentication**: Only authorized processes access resources.
- **Access Control**: Enforcing permissions on shared data and files.
- **Audit Trails**: Logging process activities.
- **Preventing Attacks**: Like buffer overflows or privilege escalation.

1. What is a process in an operating system?

A) An algorithm
B) A program in execution
C) A data structure
D) A software module

Answer: B) A program in execution
Explanation: A process is a running instance of a program, along with its program counter, registers, and variables.

2. Which is NOT a valid process state?

A) Running
B) Waiting
C) Sleeping
D) Terminated

Answer: C) Sleeping
Explanation: Standard process states include New, Ready, Running, Waiting, and Terminated. "Sleeping" is not a standard OS-defined process state.

3. A transition from Ready to Running occurs when:

A) The process is created
B) The process finishes execution
C) The scheduler selects it
D) The process waits for I/O

Answer: C) The scheduler selects it
Explanation: The CPU scheduler picks a process from the ready queue and dispatches it to run.

4. Critical Section problems arise mainly because:

A) Processes don't need resources
B) Processes share resources
C) Processes execute sequentially
D) Processes have infinite loops

Answer: B) Processes share resources
Explanation: Critical sections manage access to shared resources to avoid conflicts.

5. Which condition is essential for Mutual Exclusion?

A) Only one process in critical section at a time
B) All processes access critical section simultaneously
C) All processes are blocked
D) Critical section has no entry

Answer: A) Only one process in critical section at a time
Explanation: Mutual exclusion guarantees that only one process can access the critical section at a time.

6. Which technique can be used for achieving mutual exclusion?

A) Paging
B) Peterson’s Algorithm
C) Disk Scheduling
D) File Allocation

Answer: B) Peterson’s Algorithm
Explanation: Peterson’s algorithm is used for mutual exclusion between two processes.

7. In the Dining Philosophers Problem, deadlock can be avoided by:

A) Allowing all philosophers to eat at once
B) Having philosophers pick up both forks simultaneously
C) Limiting the number of philosophers who can pick up forks
D) No synchronization

Answer: C) Limiting the number of philosophers who can pick up forks
Explanation: Restricting access prevents all philosophers from waiting indefinitely (deadlock).

8. The Readers-Writers Problem addresses:

A) Scheduling readers
B) Synchronizing shared resource access
C) Thread termination
D) Process migration

Answer: B) Synchronizing shared resource access
Explanation: Readers and writers must synchronize access to prevent data inconsistency.

9. A binary semaphore can have values:

A) 0 and 1
B) 0, 1, and 2
C) Only positive integers
D) Any integer

Answer: A) 0 and 1
Explanation: A binary semaphore acts like a lock with two states: locked (0) and unlocked (1).

10. Which of these is a preemptive scheduling algorithm?

A) FCFS
B) SJF (non-preemptive)
C) Round Robin
D) Priority (non-preemptive)

Answer: C) Round Robin
Explanation: Round Robin scheduling preempts processes after a fixed time quantum.

11. In Shortest Job First (SJF) scheduling, starvation can happen because:

A) Short jobs keep arriving
B) No jobs are short
C) CPU halts
D) I/O errors

Answer: A) Short jobs keep arriving
Explanation: Longer processes may starve if shorter ones continuously arrive.

12. Which scheduling algorithm minimizes average waiting time?

A) FCFS
B) SJF
C) Round Robin
D) Priority

Answer: B) SJF
Explanation: SJF selects the process with the shortest burst time, minimizing waiting time.

13. Which transition is valid in a process state diagram?

A) Running → Waiting
B) Ready → Terminated
C) New → Running
D) Waiting → New

Answer: A) Running → Waiting
Explanation: A process moves from Running to Waiting when it needs to wait for an event like I/O.

14. Context switching involves:

A) Switching from user mode to kernel mode
B) Switching CPU from one process to another
C) Switching memory pages
D) Swapping disks

Answer: B) Switching CPU from one process to another
Explanation: Context switching saves and restores process states when switching.

15. IPC mechanisms include:

A) Signals and polling
B) Paging and segmentation
C) Message passing and shared memory
D) Encryption and decryption

Answer: C) Message passing and shared memory
Explanation: IPC relies on mechanisms like message queues and shared memory for process communication.

16. Which of the following is an IPC method?

A) Fork
B) Semaphore
C) Context switch
D) Bootloader

Answer: B) Semaphore
Explanation: Semaphores can be used for IPC synchronization purposes.

17. A thread shares all except:

A) Code
B) Stack
C) Data
D) Open files

Answer: B) Stack
Explanation: Threads have their own stacks but share code, data, and file descriptors.

18. Which is an advantage of multithreading?

A) Higher memory consumption
B) Slower response
C) Efficient CPU utilization
D) Increased isolation

Answer: C) Efficient CPU utilization
Explanation: Multithreading makes efficient use of CPU by overlapping I/O and computation.

19. A kernel-level thread:

A) Is managed by user programs
B) Is managed directly by the OS
C) Is invisible to the OS
D) Does not exist

Answer: B) Is managed directly by the OS
Explanation: Kernel-level threads are recognized and managed by the operating system.

20. A race condition occurs when:

A) Two threads access shared data without synchronization
B) A thread sleeps
C) CPU slows down
D) Process switches context

Answer: A) Two threads access shared data without synchronization
Explanation: Race conditions happen when concurrent threads improperly access shared resources.

21. Deadlock can occur if processes:

A) Use round robin scheduling
B) Share resources without proper synchronization
C) Never share resources
D) Do not require any resources

Answer: B) Share resources without proper synchronization
Explanation: Deadlocks happen when processes compete for shared resources without adequate control mechanisms.

22. Which is NOT a necessary condition for deadlock?

A) Mutual exclusion
B) Hold and wait
C) Preemption
D) Circular wait

Answer: C) Preemption
Explanation: No preemption is required for deadlock to occur; preemption actually helps prevent deadlocks.

23. In a multithreaded process, threads share:

A) Stack memory
B) CPU registers
C) Code section
D) Program counter

Answer: C) Code section
Explanation: Threads share the same code, data, and file resources, but each has its own program counter and stack.

24. Which of these is true for user-level threads?

A) Managed by kernel
B) Slower to switch
C) More control for user programs
D) Always faster than processes

Answer: C) More control for user programs
Explanation: User-level threads are managed by user-level libraries, allowing finer control.

25. A secure operating system must:

A) Allow unrestricted access
B) Control access to resources
C) Share passwords openly
D) Ignore user authentication

Answer: B) Control access to resources
Explanation: A secure OS ensures that only authorized users can access system resources.

26. A token-based authentication system is used for:

A) File management
B) Process scheduling
C) Secure identity verification
D) Deadlock prevention

Answer: C) Secure identity verification
Explanation: Tokens like OTPs or certificates are used for verifying users securely.

27. Which IPC mechanism is the fastest?

A) Message Passing
B) Shared Memory
C) Sockets
D) Signals

Answer: B) Shared Memory
Explanation: Shared memory is the fastest because processes communicate by directly accessing memory, avoiding kernel intervention.

28. Which of the following scheduling algorithms could lead to starvation?

A) FCFS
B) Round Robin
C) Priority Scheduling
D) SJF (non-preemptive)

Answer: C) Priority Scheduling
Explanation: In Priority Scheduling, low-priority processes might never execute if high-priority processes keep arriving.

29. In Round Robin scheduling, context switching happens:

A) After the process finishes
B) At the end of each time quantum
C) Only on I/O request
D) Only after CPU failure

Answer: B) At the end of each time quantum
Explanation: After each fixed time slice (quantum), Round Robin scheduling switches to the next process.

30. The main problem with priority scheduling is:

A) High CPU usage
B) Low throughput
C) Starvation of lower priority processes
D) Infinite loops

Answer: C) Starvation of lower priority processes
Explanation: Lower-priority processes may be postponed indefinitely if high-priority ones dominate.

31. Mutual exclusion means:

A) All threads execute critical sections together
B) Only one thread accesses a critical section at a time
C) Threads are independent
D) Threads do not share resources

Answer: B) Only one thread accesses a critical section at a time
Explanation: Mutual exclusion prevents race conditions by allowing only one thread at a time to execute critical sections.

32. A solution to critical section problem must satisfy:

A) Safety, liveness, and bounded waiting
B) Speed, fairness, and user access
C) Encryption, authentication, and authorization
D) Deadlock, livelock, and starvation

Answer: A) Safety, liveness, and bounded waiting
Explanation: Correct solutions must guarantee mutual exclusion, progress (liveness), and bounded waiting.

33. In process management, a zombie process refers to:

A) A process still running
B) A process stuck in waiting state
C) A terminated process not yet removed from the process table
D) A virus-infected process

Answer: C) A terminated process not yet removed from the process table
Explanation: A zombie process has completed execution but still has an entry in the process table.

34. Which scheduling algorithm is ideal for time-sharing systems?

A) FCFS
B) SJF
C) Priority Scheduling
D) Round Robin

Answer: D) Round Robin
Explanation: Round Robin is suited for time-sharing as it provides fair and equal CPU time to all processes.

35. In interprocess communication, message passing involves:

A) Direct memory access
B) Sending and receiving messages via OS
C) Using hardware buses
D) Interrupts only

Answer: B) Sending and receiving messages via OS
Explanation: Message passing is a method where processes communicate by sending messages through the OS.

36. An important method to prevent deadlock is:

A) Ignore deadlocks
B) Allow circular wait
C) Resource allocation graph
D) Removing mutual exclusion

Answer: C) Resource allocation graph
Explanation: A resource allocation graph can be used to detect and prevent deadlocks.

37. The process table is used to:

A) Schedule I/O devices
B) Track system users
C) Store information about all active processes
D) Handle memory segmentation

Answer: C) Store information about all active processes
Explanation: The OS uses the process table to maintain information like process state, program counter, and memory limits.

38. In priority scheduling, which process gets CPU first?

A) The one with the lowest priority number
B) The one with the highest priority number
C) The oldest process
D) Random process

Answer: A) The one with the lowest priority number
Explanation: Typically, lower numbers represent higher priority.

39. An example of asymmetric multiprocessing is:

A) Round Robin
B) Master-Slave architecture
C) Symmetric multithreading
D) Priority Scheduling

Answer: B) Master-Slave architecture
Explanation: In asymmetric multiprocessing, one processor (master) controls others (slaves).

40. The key difference between a process and a thread is:

A) Threads are slower
B) Threads share resources; processes do not
C) Processes are part of threads
D) Threads require more memory

Answer: B) Threads share resources; processes do not
Explanation: Threads of the same process share memory and resources, unlike processes.

41. Which condition leads to a deadlock situation?

A) Preemption
B) Circular waiting
C) No mutual exclusion
D) Absence of hold and wait

Answer: B) Circular waiting
Explanation: Circular waiting occurs when each process holds a resource and waits for another, forming a cycle—essential for deadlock.

42. A semaphore is mainly used for:

A) Memory management
B) CPU scheduling
C) Synchronization
D) File allocation

Answer: C) Synchronization
Explanation: Semaphores are used to solve synchronization problems and manage concurrent processes.

43. In thread management, which is true about user-level threads?

A) The OS schedules them
B) Switching is faster than kernel threads
C) Each has its own process table entry
D) OS provides direct support

Answer: B) Switching is faster than kernel threads
Explanation: User-level threads are managed without kernel intervention, making switching faster.

44. Which is NOT an example of interprocess communication?

A) Shared memory
B) Message passing
C) Sockets
D) Segmentation

Answer: D) Segmentation
Explanation: Segmentation is a memory management technique, not an IPC method.

45. The main goal of process scheduling is to:

A) Increase memory usage
B) Maximize CPU utilization
C) Maximize I/O wait time
D) Minimize throughput

Answer: B) Maximize CPU utilization
Explanation: Efficient process scheduling aims to maximize CPU usage and minimize waiting time.

46. In which scheduling algorithm is aging used to prevent starvation?

A) FCFS
B) Round Robin
C) Priority scheduling
D) SJF

Answer: C) Priority scheduling
Explanation: Aging gradually increases the priority of waiting processes to prevent starvation.

47. Context switching is:

A) Saving and restoring process states
B) Deleting memory pages
C) Flushing the disk cache
D) Changing virtual memory

Answer: A) Saving and restoring process states
Explanation: Context switching involves saving the state of the currently running process and loading the state of the next scheduled process.

48. Which is an example of classical synchronization problem?

A) CPU scheduling
B) Dining philosophers problem
C) Memory segmentation
D) Thread pooling

Answer: B) Dining philosophers problem
Explanation: Dining philosophers, readers-writers, and producer-consumer are classical synchronization problems.

49. Which security issue is related to process management?

A) Stack overflow
B) Buffer overflow attacks
C) Encryption
D) Thread priority inversion

Answer: B) Buffer overflow attacks
Explanation: Buffer overflow vulnerabilities can allow unauthorized access to processes' memory space.

50. In multithreaded programming, a daemon thread is:

A) A thread with highest priority
B) A thread that runs forever
C) A background thread that terminates with the program
D) A thread created for network connections only

Answer: C) A background thread that terminates with the program
Explanation: Daemon threads provide services in the background and terminate automatically when the main program ends.

Unit III

CPU Scheduling, Scheduling Concepts, Techniques of Scheduling, Preemptive and Non-Preemptive Scheduling: First-Come-First-Serve, Shortest Request Next, Highest Response Ration Next, Round Robin, Least Complete Next, Shortest Time to Go, Long, Medium, Short Scheduling, Priority Scheduling. Deadlock: System model, Deadlock characterization, Prevention, Avoidance and detection, Recovery from deadlock.

1. Scheduling Concepts

CPU scheduling is the process of selecting a process from the ready queue to be executed by the CPU.

- **Goal:** Maximize CPU utilization, throughput, minimize waiting time, turnaround time, and response time.
- **Scheduler:** The system component that selects which process runs next.

Types of scheduling decisions:

- When a process switches from running to waiting (I/O request)
- When a process switches from running to ready (preemption)
- When a process switches from waiting to ready (I/O completion)
- When a process terminates

2. Techniques of Scheduling

- **Preemptive Scheduling:** CPU can be taken away from a process if a higher-priority process arrives.
- **Non-Preemptive Scheduling:** Once a process gets the CPU, it holds it until completion or waiting.

3. Preemptive and Non-Preemptive Scheduling Algorithms

i) First-Come, First-Serve (FCFS)

- **Non-preemptive**
- Processes are executed in the order they arrive.
- **Advantage:** Simple and easy to implement.
- **Disadvantage:** Can cause long waiting times (convoy effect).

ii) Shortest Request Next (SJN) / Shortest Job First (SJF)

- **Non-preemptive** (can also have preemptive version called SRTF - Shortest Remaining Time First)
- The process with the smallest execution time is selected next.
- **Advantage:** Optimal for minimum average waiting time.
- **Disadvantage:** May cause starvation for longer processes.

iii) Highest Response Ratio Next (HRRN)

- **Non-preemptive**
- Response Ratio = (Waiting Time + Service Time) / Service Time
- Process with highest response ratio is scheduled next.
- **Advantage:** Balances short and long processes; avoids starvation.

iv) Round Robin (RR)

- **Preemptive**
- Each process gets a fixed time slice (quantum).
- If not finished, it is placed back into the ready queue.
- **Advantage:** Fair to all processes; suitable for time-sharing systems.
- **Disadvantage:** Performance depends on quantum size.

v) Least Complete Next

- **Preemptive**
- Process closest to completion is selected next.
- Similar to Shortest Remaining Time First (SRTF).

vi) Shortest Time to Go

- **Preemptive**
- Similar to SRTF.
- The process with the least remaining execution time is selected.

4. Long, Medium, Short-Term Scheduling

a) Long-Term Scheduler (Admission Scheduler)

- Decides which processes are admitted into the system for processing.
- Controls the degree of multiprogramming (number of processes in memory).

b) Medium-Term Scheduler

- Swaps processes in and out of memory to improve process mix or free memory.
- Important for handling suspended processes.

c) Short-Term Scheduler (CPU Scheduler)

- Selects which process to execute next from the ready queue.
- Runs very frequently (milliseconds level).

5. Priority Scheduling

- Each process is assigned a priority.
- The CPU is allocated to the process with the highest priority.
- **Preemptive or Non-Preemptive:** Both versions are possible.
- **Problem:** Starvation for low-priority processes (solved using **aging**).

Deadlock

1. System Model

In a deadlock situation:

- **Processes** hold resources and request others.
- **Resources** can only be assigned to one process at a time.

Deadlocks occur when processes form a circular chain holding resources and requesting each other’s resources.

2. Deadlock Characterization

Four conditions must hold simultaneously for a deadlock to occur:

1. **Mutual Exclusion:** Only one process can use a resource at a time.
2. **Hold and Wait:** A process holding at least one resource is waiting to acquire more.
3. **No Preemption:** Resources cannot be forcibly removed from processes.
4. **Circular Wait:** A closed chain of processes exists, each waiting for a resource held by the next.

3. Deadlock Prevention

- Design a system that **negates** one of the four necessary conditions. Examples:
- **Eliminate Hold and Wait:** Require processes to request all resources at once.
- **Allow Preemption:** Take resources away if needed.
- **Avoid Circular Wait:** Impose a resource-ordering rule.

4. Deadlock Avoidance

- Ensure that the system will **never enter** an unsafe state.
- Example: **Banker's Algorithm** checks whether granting a resource request leaves the system in a safe state.

5. Deadlock Detection

- Allow deadlocks to occur but **detect** them using algorithms.
- Example: Resource Allocation Graph (RAG) analysis.
- Deadlocks are detected by finding cycles in the graph.

6. Recovery from Deadlock

After detecting a deadlock, the system must recover:

- **Process Termination:** Kill one or more processes to break the deadlock.
- **Resource Preemption:** Take a resource away from one process and give it to another.

1. CPU Scheduling is needed when a process:

A) Executes I/O operations
B) Requests more memory
C) Terminates
D) Switches from running to ready state

Answer: D) Switches from running to ready state
Explanation: CPU scheduling is triggered when a process is ready but the CPU is busy.

2. Which scheduler decides which process is admitted into the system?

A) Short-Term Scheduler
B) Medium-Term Scheduler
C) Long-Term Scheduler
D) CPU Scheduler

Answer: C) Long-Term Scheduler
Explanation: Long-term scheduler controls the degree of multiprogramming by selecting processes from the job pool.

3. Preemptive scheduling allows:

A) A process to keep the CPU until completion
B) Forced CPU release from a running process
C) Only batch jobs to run
D) I/O-bound processes to dominate

Answer: B) Forced CPU release from a running process
Explanation: In preemptive scheduling, a higher-priority process can interrupt the current process.

4. Which of these is non-preemptive scheduling?

A) Round Robin
B) SJF (Shortest Job First)
C) Priority Scheduling (preemptive version)
D) Multilevel Queue Scheduling

Answer: B) SJF (Shortest Job First)
Explanation: SJF can be non-preemptive if a new shorter job doesn't preempt the current running one.

5. In FCFS scheduling:

A) Processes are scheduled randomly
B) Shortest processes are scheduled first
C) Processes are scheduled in order of arrival
D) Highest priority process runs first

Answer: C) Processes are scheduled in order of arrival
Explanation: First-Come-First-Serve is based solely on arrival time.

6. A major problem with FCFS scheduling is:

A) Starvation
B) High turnaround time for short jobs
C) Low CPU utilization
D) Frequent context switching

Answer: B) High turnaround time for short jobs
Explanation: Short jobs may wait long if a long job arrives first (convoy effect).

7. In Shortest Request Next scheduling, the next selected process is the one with:

A) Highest burst time
B) Highest priority
C) Shortest burst time
D) Earliest arrival time

Answer: C) Shortest burst time
Explanation: SRN/SJF selects the process needing the least CPU time.

8. Highest Response Ratio Next (HRRN) helps prevent:

A) Deadlocks
B) Starvation
C) Low throughput
D) Race conditions

Answer: B) Starvation
Explanation: HRRN boosts the priority of waiting processes by factoring in waiting time.

9. In Round Robin scheduling, what controls the maximum time a process can run?

A) Turnaround time
B) Time quantum
C) Response ratio
D) Arrival time

Answer: B) Time quantum
Explanation: In RR, each process gets a fixed amount of CPU time (quantum).

10. The main challenge in Round Robin scheduling is:

A) Selecting the next process
B) Setting an optimal time quantum
C) Handling I/O operations
D) Deadlock prevention

Answer: B) Setting an optimal time quantum
Explanation: If quantum is too small, overhead increases; if too large, it acts like FCFS.

11. "Least Complete Next" scheduling is closest to:

A) First Come First Serve
B) Shortest Remaining Time First
C) Highest Priority First
D) Round Robin

Answer: B) Shortest Remaining Time First
Explanation: Least Complete Next selects the process closest to completion.

12. "Shortest Time to Go" scheduling is:

A) Non-preemptive
B) Preemptive based on remaining time
C) Similar to FCFS
D) A type of Round Robin

Answer: B) Preemptive based on remaining time
Explanation: It selects the process with the least time remaining for completion.

13. Which scheduler manages processes that are swapped out of memory?

A) Long-Term Scheduler
B) Short-Term Scheduler
C) Medium-Term Scheduler
D) CPU Scheduler

Answer: C) Medium-Term Scheduler
Explanation: Medium-term scheduler swaps processes between memory and disk to manage multiprogramming.

14. Short-Term Scheduler is also called:

A) CPU Scheduler
B) Memory Scheduler
C) I/O Scheduler
D) Admission Controller

Answer: A) CPU Scheduler
Explanation: The short-term scheduler selects from ready processes for execution on the CPU.

15. Priority Scheduling may cause:

A) Deadlocks
B) Starvation
C) Context Switching
D) Resource Leakage

Answer: B) Starvation
Explanation: Lower-priority processes may be postponed indefinitely if higher-priority processes keep arriving.

16. Aging technique in scheduling helps to:

A) Kill older processes
B) Increase the priority of waiting processes
C) Decrease the priority of new processes
D) Prevent short process starvation

Answer: B) Increase the priority of waiting processes
Explanation: Aging increases priority over time to avoid starvation.

17. Which of the following is NOT a necessary condition for deadlock?

A) Mutual Exclusion
B) Hold and Wait
C) Preemption
D) Circular Wait

Answer: C) Preemption
Explanation: Preemption helps prevent deadlocks; it’s not a requirement.

18. In deadlock, processes are stuck in:

A) Running state
B) Ready state
C) Waiting state
D) New state

Answer: C) Waiting state
Explanation: Deadlocked processes wait indefinitely for resources.

19. The resource allocation graph helps in:

A) Scheduling CPU processes
B) Detecting deadlocks
C) Preventing thrashing
D) Handling I/O

Answer: B) Detecting deadlocks
Explanation: Cycles in a resource allocation graph indicate deadlocks.

20. Deadlock prevention aims to:

A) Allow deadlocks to happen
B) Make sure at least one necessary condition for deadlock does not occur
C) Detect deadlocks after they occur
D) Restart deadlocked processes

Answer: B) Make sure at least one necessary condition for deadlock does not occur
Explanation: Deadlock prevention breaks one or more of the four conditions necessary for deadlock.

21. Which strategy involves checking if the system remains in a safe state?

A) Deadlock prevention
B) Deadlock detection
C) Deadlock avoidance
D) Deadlock ignoring

Answer: C) Deadlock avoidance
Explanation: Deadlock avoidance ensures that resource allocation keeps the system in a safe state (like Banker's algorithm).

22. Deadlock detection algorithms require:

A) No resource tracking
B) Periodic checks for cycles
C) Random termination of processes
D) Assigning same priority to all processes

Answer: B) Periodic checks for cycles
Explanation: Cycle detection is essential for deadlock detection.

23. Recovery from deadlock can involve:

A) Rebooting the system
B) Killing one or more processes
C) Ignoring deadlocks
D) Increasing RAM

Answer: B) Killing one or more processes
Explanation: Terminating processes is a way to recover from deadlocks.

24. Which is NOT a method of deadlock recovery?

A) Process termination
B) Resource preemption
C) Ignore deadlock
D) Process replication

Answer: D) Process replication
Explanation: Replicating processes does not help recover from deadlocks.

25. A "safe state" in a system means:

A) Deadlock will surely occur
B) Deadlock may occur
C) Deadlock will not occur
D) The system is shutting down

Answer: C) Deadlock will not occur
Explanation: A safe state ensures that every process can eventually complete.

26. In Banker’s Algorithm, a process’s request is granted if:

A) Enough available resources exist
B) The system remains in a safe state after allocation
C) The process has highest priority
D) It reduces deadlock chances

Answer: B) The system remains in a safe state after allocation
Explanation: Banker's Algorithm allocates resources only if it keeps the system safe.

27. If no process is preempted and resources are not released voluntarily, it can cause:

A) Thrashing
B) Starvation
C) Deadlock
D) Fragmentation

Answer: C) Deadlock
Explanation: No preemption helps deadlocks to occur.

28. Which deadlock strategy is the simplest but most wasteful?

A) Prevention
B) Avoidance
C) Detection and recovery
D) Ignoring deadlocks

Answer: D) Ignoring deadlocks
Explanation: Some systems (e.g., early UNIX) simply ignored deadlocks and restarted systems when necessary.

29. In preemptive priority scheduling, if two processes have the same priority:

A) FCFS is used
B) Shortest job first is used
C) Round robin is used
D) Random process is selected

Answer: A) FCFS is used
Explanation: If priorities are equal, processes are scheduled in arrival order.

30. In non-preemptive scheduling, CPU is allocated:

A) Based on priority
B) Until the process voluntarily releases it
C) For one quantum only
D) Until an I/O request is made

Answer: B) Until the process voluntarily releases it
Explanation: In non-preemptive scheduling, a running process keeps the CPU until it terminates or requests I/O.

1. Processes P1, P2, P3 arrive at time 0.
Their CPU burst times are 5, 9, and 6 milliseconds respectively.
Find the **average waiting time** using **FCFS**.

A) 5 ms
B) 8 ms
C) 6.33 ms
D) 7 ms

Answer: C) 6.33 ms
Explanation:

- P1: Waiting time = 0
- P2: Waiting time = 5
- P3: Waiting time = 5 + 9 = 14
  Average = (0 + 5 + 14) / 3 = 6.33 ms

2. For the same processes above, find the **average turnaround time** using FCFS.

A) 14 ms
B) 15 ms
C) 16.33 ms
D) 18 ms

Answer: C) 16.33 ms
Explanation:
Turnaround = Waiting time + Burst time

- P1 = 0 + 5 = 5
- P2 = 5 + 9 = 14
- P3 = 14 + 6 = 20
  Average = (5 + 14 + 20) / 3 = 13 ms
  (Mistake in options — correct avg is **13 ms**, not listed)

3. In Round Robin scheduling, time quantum = 4 ms.
Processes P1 (burst=10), P2 (burst=5), P3 (burst=8).
How many context switches occur?

A) 5
B) 6
C) 7
D) 8

Answer: B) 6
Explanation:

- P1 (4 ms) → P2 (4 ms) → P3 (4 ms) → P1 (6 ms left - 4 ms) → P3 (4 ms) → P1 (2 ms) Each switch counts: total 6 switches.

4. In Shortest Job First (SJF) scheduling:
Processes P1 (burst=6), P2 (burst=8), P3 (burst=7), P4 (burst=3).
What is the average waiting time?

A) 5.75 ms
B) 6 ms
C) 7.5 ms
D) 8.25 ms

Answer: A) 5.75 ms
Explanation:
Order: P4 → P1 → P3 → P2

- P4: 0
- P1: 3
- P3: 3+6=9
- P2: 9+7=16
  Average = (0+3+9+16)/4 = 7 ms

5. If a system has 12 instances of a resource and 5 processes.
The maximum needs are: [3, 4, 4, 5, 2]
Currently allocated: [1, 2, 1, 2, 1]
How many resources are available?

A) 5
B) 6
C) 4
D) 2

Answer: A) 5
Explanation:

Allocated = 1+2+1+2+1 = 7
Available = 12 - 7 = 5

6. In Banker's algorithm, if the system is in a safe state, it means:

A) All processes have completed
B) No process can make further requests
C) There exists a safe sequence
D) The system is deadlocked

Answer: C) There exists a safe sequence
Explanation:
A safe state means at least one order of process execution exists without deadlock.

7. Given arrival times: P1(0), P2(1), P3(2), Burst times: P1(8), P2(4), P3(2)
Using SRTF (Shortest Remaining Time First), who finishes first?

A) P1
B) P2
C) P3

Answer: C) P3
Explanation:
P3 arrives at time 2 and has shortest burst time 2.

8. In Priority Scheduling: Priorities: P1(2), P2(1), P3(3). (Lower number = higher priority)
Order of execution is:

A) P1 → P2 → P3
B) P2 → P1 → P3
C) P3 → P1 → P2
D) P1 → P3 → P2

Answer: B) P2 → P1 → P3
Explanation:
Lower number = higher priority.

9. A Round Robin system has quantum 5ms. P1 needs 17ms. How many full quantums will P1 get before finishing?

A) 3
B) 4
C) 2
D) 5

Answer: B) 4
Explanation:

- First 5 + 5 + 5 = 15 ms done
- 2 ms remains
- 4 quantums in total.

10. In FCFS, a set of jobs with equal arrival times but varying burst times will result in:

A) Lowest average waiting time
B) High average waiting time
C) Medium average waiting time
D) Starvation

Answer: B) High average waiting time
Explanation:
Long jobs early increase waiting time for short jobs (convoy effect).

11. Processes have following burst times: 3, 5, 8. Find total CPU burst time.

A) 14
B) 16
C) 17
D) 18

Answer: B) 16
Explanation:
Sum = 3+5+8 = 16.

12. Using SJF, if processes arrive at the same time, the scheduling will favor:

A) Longer processes
B) I/O-bound processes
C) Shorter processes
D) Random processes

Answer: C) Shorter processes
Explanation:
SJF = shortest burst first.

13. A deadlock can occur if each process holds _____ resource(s) and waits for another.

A) No
B) One

C) Two
D) Three

Answer: B) One
Explanation:
Holding one and waiting for another creates circular wait.

14. A system has 6 tape drives and there are 3 processes. Each may need 4 drives. Is deadlock possible?

A) Yes
B) No

Answer: B) No
Explanation:
Total resources > (sum of max needs - number of processes)
(3x4)-3 = 9; system only has 6, so needs adjustment.

15. If the resource allocation graph has **no cycles**, then:

A) Deadlock may happen
B) Deadlock definitely happens
C) Deadlock cannot happen
D) Deadlock must be manually checked

Answer: C) Deadlock cannot happen
Explanation:
No cycle = no deadlock.

16. Process P1 holds R1 and requests R2, P2 holds R2 and requests R1.
This is an example of:

A) Starvation
B) Circular Wait
C) Preemption
D) Aging

Answer: B) Circular Wait
Explanation:
Processes are waiting in a cycle for each other’s resource.

17. If we prevent **Hold and Wait** in a system:

A) Deadlocks are prevented
B) Deadlocks are detected

C) Deadlocks are ignored
D) Deadlocks are allowed

Answer: A) Deadlocks are prevented
Explanation:
Breaking any deadlock condition (like Hold and Wait) prevents deadlocks.

18. In Banker's Algorithm, what does **Need** matrix represent?

A) Maximum - Allocation
B) Allocation - Maximum
C) Available - Allocation
D) Allocation + Available

Answer: A) Maximum - Allocation
Explanation:
Need = Max - Allocated resources.

19. If a system has 10 resources, and currently allocated 7, available resources are:

A) 2
B) 3
C) 1
D) 5

Answer: B) 3
Explanation:
10 - 7 = 3 available.

20. Time quantum in a Round Robin system is set too high, it behaves like:

A) SJF
B) FCFS
C) Priority Scheduling
D) Shortest Remaining Time First

Answer: B) FCFS
Explanation:
If time quantum is large, each process finishes without being preempted → acts like FCFS.

Unit IV

Memory Management and File System Memory Allocation, Paging, Segmentation, Virtual Memory, Demand Paging, Page Replacement Algorithms File concept, File organization and access mechanism, File directories, and File sharing, File system protection and security.

1. Memory Allocation

- It refers to assigning portions of memory to programs and processes.
- Two main types:
  - **Static Allocation**: Fixed memory assigned at compile-time.
  - **Dynamic Allocation**: Memory assigned during run-time.

Types of Allocation Techniques:

- **Contiguous Allocation**: Single continuous section of memory per process.
- **Non-Contiguous Allocation**: Memory blocks scattered, like paging or segmentation.

2. Paging

- **Paging** is a memory management scheme that eliminates the need for contiguous allocation.
- Memory is divided into fixed-size blocks called **frames**.
- Logical memory is divided into blocks of the same size called **pages**.
- The **Page Table** maps logical pages to physical frames.

Advantages:

- No external fragmentation
- Efficient use of memory

3. Segmentation

- **Segmentation** divides the process memory into segments based on the logical division (like functions, arrays, stacks).
- Each segment has a **segment number** and **offset**.
- Provides a **logical view** of memory, easier for programmers.

Example Segments:

- Code segment, Data segment, Stack segment

4. Virtual Memory

- A memory management technique where **secondary storage (disk)** is used as an extension of RAM.
- Programs can be larger than the physical memory.
- It provides the illusion that the user has access to a very large main memory.

Main Concept:

- Part of program in RAM, rest stored in disk (swapped in/out when needed).

5. Demand Paging

- **Demand Paging** is a lazy-loading method for virtual memory.
- Pages are loaded into memory **only when needed**.
- If the needed page is not in memory → **Page Fault** occurs → OS loads it from disk.

Goal:

- Save memory space by loading only necessary pages.

6. Page Replacement Algorithms

When a page fault occurs and memory is full, we must replace an existing page. Algorithms include:

- **FIFO (First In, First Out):** Remove the oldest page.
- **LRU (Least Recently Used):** Remove the page that has not been used for the longest time.
- **Optimal Page Replacement:** Replace the page that will not be used for the longest future period (ideal but impossible in real life).

Page Replacement reduces the number of page faults.

7. File Concept

- A **file** is a collection of related information stored on secondary storage (like hard disks).
- It is identified by its name and is usually organized into directories.
- Types of files: text files, binary files, executable files, etc.

8. File Organization and Access Mechanism

File Organization:

- **Sequential Access:** Read/write data sequentially (tape-like).
- **Direct Access:** Jump directly to the desired part (disk-like).

Access Mechanisms:

- **Sequential Access:** Data processed in order.
- **Direct Access:** Access any block directly using a pointer or key.

9. File Directories

- A **directory** contains information about files, like name, size, location, etc.
- Helps organize files systematically.

- Structures:
  - Single-level directory
  - Two-level directory
  - Tree-structured directory
  - Acyclic graph directory

10. File Sharing

- Multiple users/processes may need access to the same file.
- Sharing can be **read-only** or **read-write**.
- Mechanisms for sharing include **locking** and **access control** to avoid inconsistencies.

11. File System Protection and Security

- Protect files from **unauthorized access**, **accidental loss**, or **corruption**.
- Security techniques:
  - **Access Control Lists (ACLs)**: Who can read, write, or execute a file.
  - **Password Protection**: Password needed to access specific files.
  - **Encryption**: Files are encrypted to prevent unauthorized reading.

Example:

- Owner, Group, Others permissions in Unix/Linux (read/write/execute).

1. Which of the following memory allocation techniques suffers from external fragmentation?

A) Paging
B) Segmentation
C) Contiguous allocation
D) Virtual memory

Answer: C) Contiguous allocation
Explanation: Contiguous allocation requires memory blocks to be adjacent, leading to external fragmentation.

2. Paging helps to solve the problem of:

A) Internal fragmentation
B) External fragmentation
C) Thrashing
D) Deadlock

Answer: B) External fragmentation
Explanation: Paging divides memory into fixed-size blocks to avoid external fragmentation.

3. In paging, the size of a frame is:

A) Fixed
B) Variable
C) Depends on process size
D) Depends on physical memory size

Answer: A) Fixed
Explanation: Frame size in paging is always fixed.

4. Which memory management technique supports logical division of memory into modules like code, stack, and data?

A) Paging
B) Segmentation
C) Contiguous allocation
D) Virtual memory

Answer: B) Segmentation
Explanation: Segmentation divides memory based on logical divisions.

5. The page table is used for:

A) Storing process IDs
B) Mapping virtual addresses to physical addresses
C) Storing memory allocation methods
D) Managing CPU scheduling

Answer: B) Mapping virtual addresses to physical addresses
Explanation: The page table keeps track of where virtual pages are stored in physical memory.

6. Thrashing occurs when:

A) CPU is overloaded
B) Too much paging happens
C) Hard disk crashes
D) File system becomes corrupted

Answer: B) Too much paging happens
Explanation: Thrashing happens when the system spends more time swapping pages than executing processes.

7. What causes a page fault?

A) Page is not in memory
B) CPU failure
C) Disk error
D) File system error

Answer: A) Page is not in memory
Explanation: A page fault occurs when a program accesses a page not loaded into RAM.

8. Which of the following is NOT a page replacement algorithm?

A) FIFO
B) LRU
C) Optimal
D) FCFS

Answer: D) FCFS
Explanation: FCFS is a scheduling algorithm, not a page replacement algorithm.

9. Demand paging brings pages into memory:

A) Before they are needed
B) Only when they are needed
C) During process loading
D) When the system boots

Answer: B) Only when they are needed
Explanation: Demand paging loads pages into memory upon request.

10. Which page replacement algorithm gives the lowest number of page faults?

A) FIFO
B) Optimal
C) LRU
D) Random

Answer: B) Optimal
Explanation: Optimal replaces the page not used for the longest future time.

11. Virtual memory increases:

A) Internal memory
B) Physical memory
C) Logical memory
D) CPU speed

Answer: C) Logical memory
Explanation: Virtual memory makes it appear that there is more memory than physically available.

12. Segmentation suffers from:

A) Internal fragmentation
B) External fragmentation
C) No fragmentation
D) Thrashing

Answer: B) External fragmentation
Explanation: Since segments are of variable sizes, gaps (external fragmentation) can occur.

13. The address generated by the CPU is known as:

A) Physical address
B) Logical address
C) Virtual address
D) Both B and C

Answer: D) Both B and C
Explanation: Logical and virtual addresses are generated by the CPU before translation.

14. Which technique allows a process to execute even if it is not entirely in memory?

A) Paging
B) Segmentation
C) Virtual memory
D) Contiguous allocation

Answer: C) Virtual memory
Explanation: Virtual memory allows partial loading of processes.

15. Page size is typically:

A) 2-4 bytes
B) 512 bytes
C) 1 MB
D) 4 KB

Answer: D) 4 KB
Explanation: 4 KB is a common page size in modern systems.

16. The main advantage of paging is:

A) Large address space
B) No external fragmentation
C) Faster CPU
D) Reduced context switching

Answer: B) No external fragmentation
Explanation: Paging breaks memory into fixed sizes, avoiding gaps.

17. The mechanism to swap pages between disk and RAM is called:

A) Scheduling
B) Paging
C) Swapping
D) Segmentation

Answer: C) Swapping
Explanation: Swapping involves moving processes between disk and memory.

18. In LRU page replacement, which page is replaced?

A) Newest page
B) Page that will not be used soon
C) Page least recently used
D) Page with the highest ID

Answer: C) Page least recently used
Explanation: LRU removes the page that was used least recently.

19. Optimal page replacement requires:

A) Future knowledge
B) Past knowledge
C) Random replacement
D) None

Answer: A) Future knowledge
Explanation: It needs knowledge about future page requests.

20. In segmentation, each address is specified by:

A) Page number and offset
B) Segment number and offset
C) Frame number and offset
D) None of these

Answer: B) Segment number and offset
Explanation: Segmentation uses (segment, offset) format.

21. Internal fragmentation occurs in:

A) Paging
B) Segmentation
C) Demand paging
D) Thrashing

Answer: A) Paging
Explanation: Due to fixed page sizes, some memory within pages may be unused.

22. Page fault service time is generally:

A) Very high
B) Very low
C) Same as CPU burst time
D) Zero

Answer: A) Very high
Explanation: Disk access during page fault takes much longer than memory access.

23. Inverted page table is used to:

A) Save memory
B) Increase speed
C) Decrease number of page faults
D) Prevent deadlocks

Answer: A) Save memory
Explanation: Inverted page tables reduce memory used for page tables.

24. A process may be divided into pages in:

A) Paging only
B) Segmentation only
C) Both
D) None

Answer: A) Paging only
Explanation: Segmentation divides by logical parts, not pages.

25. In virtual memory, the address space available to a program:

A) Is smaller than physical memory
B) Equals physical memory
C) Is larger than physical memory
D) Depends on CPU speed

Answer: C) Is larger than physical memory
Explanation: Virtual memory uses disk space to extend memory capacity.

26. A file is a collection of:

A) Records
B) Bytes
C) Data
D) All of these

Answer: D) All of these
Explanation: A file can consist of records, bytes, or structured data.

27. Which file access method reads data in order?

A) Sequential
B) Direct
C) Indexed
D) Random

Answer: A) Sequential
Explanation: Sequential access reads one record after another.

28. Which file access method allows jumping to any part of the file?

A) Sequential
B) Direct
C) Random
D) Both B and C

Answer: D) Both B and C
Explanation: Direct access and random access allow jumping.

29. Which is NOT a valid file organization technique?

A) Sequential
B) Direct
C) Mixed
D) Indexed

Answer: C) Mixed
Explanation: "Mixed" is not a recognized standard file organization.

30. A directory structure organizes:

A) Files only
B) Files and directories
C) Directories only
D) None

Answer: B) Files and directories
Explanation: Directory structures manage both files and subdirectories.

31. Which of the following is NOT a directory structure?

A) Single-level
B) Two-level
C) Tree-structured
D) Page-level

Answer: D) Page-level
Explanation: Page-level relates to memory, not directory structures.

32. In which directory structure can two users have files with the same name?

A) Single-level directory
B) Two-level directory
C) Tree-structured directory
D) None

Answer: B) Two-level directory
Explanation: Each user has their own directory, so filenames can be repeated.

33. In a tree-structured directory system, a directory can contain:

A) Files only
B) Subdirectories only
C) Files and subdirectories
D) Only one file

Answer: C) Files and subdirectories
Explanation: A tree allows directories to contain both files and subdirectories.

34. Which directory structure allows sharing of files using links?

A) Tree
B) Acyclic graph
C) Single-level
D) Two-level

Answer: B) Acyclic graph
Explanation: Acyclic graph directory allows shared files through links.

35. The data structure used to implement directories is generally:

A) Stack
B) Queue
C) Hash table or tree
D) Array

Answer: C) Hash table or tree
Explanation: Trees and hash tables help organize directories efficiently.

36. Which file access method is best for databases?

A) Sequential
B) Direct
C) Indexed
D) Random

Answer: C) Indexed
Explanation: Indexed access is most suitable for databases for fast lookup.

37. A file control block (FCB) contains information about:

A) File size
B) File location
C) File permissions
D) All of these

Answer: D) All of these
Explanation: The FCB stores all metadata about a file.

38. What is the purpose of access control in file systems?

A) Enhance speed
B) Restrict unauthorized users
C) Reduce fragmentation
D) Save disk space

Answer: B) Restrict unauthorized users
Explanation: Access control protects files by defining who can access them.

39. Which type of file sharing allows concurrent reading and writing?

A) Exclusive sharing
B) Controlled sharing with locks
C) Public sharing
D) Sequential sharing

Answer: B) Controlled sharing with locks
Explanation: Locks enable safe concurrent access.

40. Which of the following is used for file protection?

A) Permissions
B) Firewalls
C) RAID systems
D) CPU Scheduling

Answer: A) Permissions
Explanation: Permissions (read, write, execute) control file access.

41. Which type of permission allows a user to execute a file?

A) Read
B) Write
C) Execute
D) Open

Answer: C) Execute
Explanation: Execute permission allows running a program file.

42. Which file protection mechanism encrypts file content?

A) Password
B) File lock
C) Encryption
D) ACL

Answer: C) Encryption
Explanation: Encryption encodes file data to prevent unauthorized reading.

43. Which structure maps user actions (like read/write) to file operations?

A) Access Control List (ACL)
B) File Directory
C) Page Table
D) Index Block

Answer: A) Access Control List (ACL)
Explanation: ACL specifies which users can perform which operations.

44. A virus modifies files and spreads without user consent. This is a violation of:

A) Privacy
B) Integrity
C) Availability
D) Authentication

Answer: B) Integrity
Explanation: Integrity means keeping data unaltered — a virus violates this.

45. Which of these is NOT a goal of file system security?

A) Data confidentiality
B) Data integrity
C) Data unavailability
D) Data availability

Answer: C) Data unavailability
Explanation: Security aims for confidentiality, integrity, and availability (CIA triad).

46. Which of the following is NOT considered a security threat to the file system?

A) Unauthorized access
B) System crash
C) Data tampering
D) File deletion by a hacker

Answer: B) System crash
Explanation: A system crash is a reliability issue, not directly a security threat.

47. Which file system allows permissions for Owner, Group, and Others?

A) Windows NTFS
B) Unix/Linux file system
C) FAT32
D) exFAT

Answer: B) Unix/Linux file system
Explanation: Unix/Linux permissions structure defines access for Owner, Group, and Others.

48. In Unix, which command changes file permissions?

A) chmod
B) chperm
C) permchg
D) access

Answer: A) chmod
Explanation: chmod is used to change permissions in Unix/Linux systems.

49. What ensures that even if someone gains access to a file, they cannot read its content without a key?

A) Passwords
B) Access Control
C) Encryption
D) Directory locking

Answer: C) Encryption
Explanation: Encryption protects file contents even if the file is accessed.

50. Which file attribute indicates whether a file is hidden or visible?

A) Read-only flag
B) Hidden flag
C) Archive flag
D) System flag

Answer: B) Hidden flag
Explanation: A hidden flag marks a file as hidden in the file system.