Name → AJAY NATH JHA Roll. No → 23010170
Course → B.Tech CSE Core Page No. 01
Section → C Semester → V Date 03/12/25

Subject :→ Operating System

Assignment - 3 :→

Part - A : Short Answer Type →

(1.) Race conditions and Mutual Exclusion →

A race condition occurs when two or more entities attempt to change a shared resource at the same time, leading to inconsistent outcomes. A real-world example is two people trying to withdraw money from same bank account simultaneously; if both withdrawls occur without coordination, the final balance becomes incorrect.

Mutual Exclusion resolves this by ensuring that only one entity accesses the shared resource at a time. When one person's withdrawal is being processed, the other must wait, guaranteeing correct and predictable results.

(2.) Peterson's Solution vs Semaphores →

Peterson's solution is purely software-based and requires strict assumptions such as atomic read/write operations, making it harder to implement correctly on modern hardware. It is mainly useful for two-process situations.

Semaphores are more flexible, easier to implement, and widely supported. They often rely on hardware primitives such as test-and-set, making them more reliable and scalable. However, semaphores introduce more complexity in terms of correct usage, as misuse can lead to deadlocks.

(3.) Advantage of monitors in multi-core Systems :→
In Multi-core systems, monitors provide automatic handling of mutual exclusion by associating locks with the monitored object. This reduces the programmer's responsibility of explicitly acquiring and releasing locks. The monitor's internal synchronization ensures that only one thread executes its critical section at a time, improving both safety & clarity in concurrent execution.

(4.) Starvation in the Reader-Writer problem and its prevention.
Starvation can occur when writers are repeatedly delayed because readers continuously arrive and obtain priority, preventing writers from ever gaining access. One method to prevent Starvation is to use a fair Scheduling policy, such as giving writers priority once they appear. This ensures that once a writer requests access, new readers are temporarily blocked until the writer completes, guaranteeing fairness.

(5.) Drawback of eliminating 'Hold and Wait' in deadlock prevention.
Eliminating the 'Hold & Wait' condition requires processes to request all needed resources at once before execution begins. This leads to poor resource utilization because processes hold resources much earlier than necessary, leaving them unutilized for long periods. As a result, System throughput decreases

Name → AJAY NATH JHA
Roll. No → 2301010170

Page No. 03
Date 03 12 25

Part-B: Application / Numerical Based →

⑥ Banker's Algorithm:

(a) Need Matrix = Max - Allocation.

P0: (7-0, 5-1, 3-0) = 7, 4, 3
P1: (3-2, 2-0, 2-0) = 1, 2, 2
P2: (9-3, 0-0, 2-2) = 6, 0, 0
P3: (4-2, 2-1, 2-1) = 2, 1, 1
P4: (5-0, 3-0, 3-2) = 5, 3, 1

(b) Safe State Check:

Available = (Total - Allocation)
Allocated Sum = A:7, B:2, C:5 → Available = (3, 3, 2)
Safe sequence exists: P1 → P3 → P4 → P0 → P2
So, the system is in a Safe state.

(c) P1 requests (1, 0, 2)

Request ≤ Need and Request ≤ Available
(1, 0, 2) ≤ (1, 2, 2) and (1, 0, 2) ≤ (3, 3, 2)
After granting, System remains Safe.
Thus, the request can be granted immediately.

⑦ Dining Philosophers with Semaphores :→

Deadlock scenario (all philosophers pick left fork):
Each philosopher executes:
wait (left_fork) wait (right_fork)
All five pick their left fork. No one releases it, and all wait indefinitely for the right fork → deadlock.
Modified solution to avoid deadlock:
Enforce ordering, such as allowing one philosopher (Say philosopher 4) to pick the right fork first:
For philosophers 0-3:
wait (left); wait (right)
For philosopher 4:

wait(right); wait(left)
This breaks circular wait & ensures deadlock does not occur.

(8.) I/O System Analysis →
Data transfer rate = 500 KB/Sec
Block Size = 100 bytes
Interrupts per second = (500 × 1024) ÷ 100
= 5120 interrupts/sec
Interrupt handling time = 5 μs

(a) CPU time per second,
5120 × 5 μs = 25600 μs = 25.6 ms per second

(b) Improvement.
Use DMA (Direct Memory Access) to transfer data in larger blocks without involving CPU for each interrupt, thus reducing CPU overhead.

(9.) Air Traffic Control System Case Study :→

(a) Critical sections and suitable IPC →
Critical sections include :
1. Accessing radar data buffers. 2. Updating shared flight path tables. 3. Managing communication queues for pilot instructions.
A suitable IPC mechanism is message passing with priority scheduling, as it supports real-time responsiveness and avoid unnecessary blocking.

(b) Deadlock detection & recovery strategy →
The O.S can periodically run a deadlock detection algorithm to check for circular wait between radar acquisition & flight path processes. If detected, the system can recover by preempting or rolling back less critical process, freeing its resources and allowing critical task to continue. This minimizes disruption while maintaining system safety.