Operating System
Assignment - 3

Ayush Gupta ①
2301010206 B.Tech CSE (Core)

Ans 1) Race Condition & Mutual Exclusive

→ Race Condition Example (Bank): Two people simultaneously withdrawing the last $100 from a $100 account. Both read $100 deduct, and write $0, resulting in an incorrect, overdrawn balance.

→ Mutual Exclusive : Ensures only one person / process can update the shared balance at a time. The first process locks the balance until the transaction is complete, preventing the second process from reading the initial $100 value.

Ans 2) Peterson's Solution vs Semaphores

Feature	Peterson's Solution	Semaphores
→ Implementation Complexity	low. Simple logic with standard variables (flag, turn).	Moderate to high. Requires kernel support for system calls (wait / signal) and process queue management.
→ Hardware Dependency	Moderate. Requires atomic read / write guarantees (eg:- memory barriers)	Low. Implemented as OS kernel primitives; less dependent on specific CPU architecture.

Ans 3) Monitors provide a higher level abstraction that automatically handles locking and signalling within their structure. This often allows for finer-grained or optimized locking in multi core systems, resulting in better performance and less contention compared to the typically coarse-grained locking associated with basic semaphore implementations.

Ans4) Starvation in Reader - Writer Problem

→ Starvation: Occur when a writer is continually blocked by a constant stream of newly arriving readers in a Reader - Priority Scheme. The writer is perpetually delayed.

→ Prevention: Implement a writer - priority scheme. when a writer arrives, no new readers are allowed to start. The writer is guaranteed access as soon as the currently active readers finish, preventing indefinite delay.

Ans5) Drawback of eliminating "Hold and wait".

→ Elimination Method: Force a process to request utilization. Resources are tied up for the entire duration of a process, even if only briefly. This restricts other processes and reduces overall system throughput.

Ans6) Banker's Algorithm Simulation

→ Total Instances : A=10, B=5, C= 7

→ Allocation : A = 7, B=2, C=5

→ Initial Available : (10-7, 5-2, 7-5) = (3, 3, 2)

a) Calculate the Need Matrix.

Process	Need (A, B, C)
P_0	7, 4, 3
P_1	1, 2, 2
P_2	6, 0, 0
P_3	0, 0, 0
P_4	5, 3, 1

b) Determine if the system is in a safe state.

Yes, the system is in a safe state.

Safe Sequence (one possibility):

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

$\rightarrow (3,3,2) \rightarrow P_3\ (0,0,0) \rightarrow (5,4,3) \rightarrow P_1\ (1,2,2) \rightarrow (7,4,3)$

$\rightarrow P_4\ (5,3,1) \rightarrow (7,4,5) \rightarrow P_2\ (6,0,0) \rightarrow (10,4,7) \rightarrow$

$P_0\ (7,4,3) \rightarrow (10,5,7)$.

c) If the P_1 request $(1,0,2)$, check whether it can be granted immediately.

1. Check Request $\leq$ Need : $(1,0,2) \leq (1,2,2)$ — True.
2. Check Request $\leq$ Available : $(1,0,2) \leq (3,3,2)$ — True.
3. Simulate Grant: New Available is $(3-1, 3-0, 2-2) = (2,3,0)$
4. Safety check with New State: A safe sequence can still be found (eg:- $\langle P3, P1_{new}, P_4, P_2, P_0 \rangle$)

∴ The request $(1,0,2)$ can be granted immediately because the resulting state remain safe.

Ans7) Dining Philosophers:

→ Deadlock Scenario: All 5 philosophers (P_0 to P_4) simultaneously acquire their left chopstick (C_0 to C_4) and then wait for their right chopstick (C_1 to C_0). This establishes a circular wait (P_0 waits for P_1, P_1 waits for P_2 ..., P_4 waits for P_0). Deadlock.

→ Deadlock Avoidance Modification (Break Circular wait): Impose an asymmetric ordering for one philosophers.

→ P_0 to P_3: Acquire left chopstick. then Right chopstick.

Ans 8) I/O system Analysis

a) Calculate CPU time spent handling interrupts per second.

i) Interrupts per second (N):

$$N = \frac{\text{Data Rate}}{\text{Block size}} = \frac{500\text{nB}}{100\text{Byte}} = \frac{5:12000\ \text{B/s}}{100\text{B/interrupt}} = 5120\ \text{interrupts/sec}$$

2) Total CPU time = (T_{CPU}):

$T_{CPU} : N \times T_{int} = 5120 \times 5\mu s = 25{,}600\ \mu s = 0.0256\ \text{sec}$

b) Suggest one improvement to reduce CPU overhead.

→ Improvement: Use direct Memory Access (DMA)

→ Reason: DMA allows data transfer directly between the device and memory, by passing the CPU. The CPU is interrupted only once per transfer buffer (not every 100 bytes) drastically reducing overhead.

Ans 9) Air traffic control System

a) Critical Sections and IPC Mechanisms

→ Critical Section:

→ Shared flight database updates: Require Mutual exclusion for data integrity.

→ Radio Transmittive / Receiver Access: Require exclusive access

b) Deadlock detection and Recovery Strategy.

→ Detection: Use a resource Allocation graph Algorithm to check for cycles involving RDA and FPC processes.

→ Recovery 1) Select Victim

2) Preempt

3) Roll back

Satindra 26/11/25