Date
Page

Assignment - 4

Operating System

Name - Rishav Kumar
Roll No - 2301010424

Part - A

Q1 Ans> A race condition occurs when two people try to use the same shared resource at the same time causing an incorrect or unexpected result, for example - if two people try to take ₹500 from the same wallet simultaneously, each thinks the money is still there and both takes it, even though wallet had total ₹500. Mutual exclusion prevents this by ensuring that only One person can access the wallet at a time forcing the second person to wait, which guarantee a safe & correct outcome.

Q2 Ans
→ Peterson's Solution vs Semaphore
• Implementation complexity:
Peterson Solution
- More complexity is to implement correctly because it requires careful use of shared variables and is limited to two process.
Semaphore
- Simples and cleaner to use in programs, support many processes and offer standard operations (wait, signal)

→

• Hardware Dependency

Peterson's Solution
- Works only under strict assumptions and fails on modern CPU due to instruction reordering, not hardware - independent practice.

Semaphore
- Require minimal hardware support such as atomic instruction (Eg. test and set), make them reliable on modern systems.

Q3. Ans
→ Monitors automatically handle mutual exclusion, so in a multi-core system they reduce the chance of programming errors by ensuring that only one thread can execute the critical section at a time without manually manging semaphores.

Q4. Ans
→ In the Reader-writer problem, starvation happens when one type of process (usually writers) keeps getting delayed because the other type (readers) continuously arrives and is always given priority. As a result, a writer may wait indefinitely while readers keep accessing the shared resource.
One method to prevent it -
Use a fair (FIFO) scheduling policy where readers & writers are queued in the order

they arrives. This ensures that each process eventually gets its turn and no reader or writer waits forever.

Q5. Ans

→ Eliminating hold & wait forces a process to request all needed resource at once before it starts. Drawback: This leads to poor resource utilisation, because a process may hold resources it doesn't need yet, keeping them unavailable for others & reducing overall system efficiency.

Part - B

Part - B

Q6. Ans - Given:

S_1 : $P_1 \rightarrow P_2$, $P_3 \rightarrow P_4$

S_2 : $P_2 \rightarrow P_5$, $P_5 \rightarrow P_6$

S_3 : $P_6 \rightarrow P_1$

(a) Global wait for Graph:

$P_1 \rightarrow P_2 \rightarrow P_5 \rightarrow P_6 \rightarrow P_1$

$P_3 \rightarrow P_4$

(b) Deadlock Detection:

cycle exist : $P_1 \rightarrow P_2 \rightarrow P_5 \rightarrow P_6 \rightarrow P_1$

Deadlock Processes : P_1 , P_2 , P_5 , P_6

(c) Distributed Algorithm:

Chandy - Misra - Haas Algorithm for distributed deadlock detection.

Q.7. Ans - Distributed file System Performance

Given: local access : 5 ms

remote access : 25 ms

remote probability : 0.3

(a) Expected file access time E[T]:

$E[T] = (0.7 \times 5) + (0.3 \times 25$

$= 3.5 + 7.5$

$= 11$ ms

DATE: __/__/__
PAGE: _____

(b) caching strategy
→ Client - side caching - Store frequently accessed remote file locally.

→ Justification - reduces repeated remote access latency and network load.

Q8 Ans - Given :
full : 200 ms
Incremental : 50 ms
RPO : 1 s

(a) Optimal Mix : perform 1 full checkpoint every 1s, followed by incremental checkpoints every 250 ms

(b) Reasoning : Incremental checkpoints are faster, reducing overhead. full checkpoints ensure complete recovery. Combination meets RPO without blocking the system.

Q9. Ans - (a) Distributed Scheduling challenges: flash sales create sudden load spikes, uneven across region.

Suitable Algorithm : weighted Round Robin or Dynamic load balancing using least-loaded server.

Fault Tolerance Strategy :-

Cross - Redundant Deployment - replicate services across multiple Data centres.

RTO / RPO :- Use Synchronous replication for critical services (low RPO) and asynchronous (acceptable RTO)

Result : High availability even if a region fails.