**Operating System Important Questions**

**Unit 3**

1. Explain race condition with suitable example
2. Explain critical section problem. Describe Three requirement of the solution for critical section problem.
3. Describe two approaches used to handle critical section in operating system (preemptive kernels and non preemptive kernels)
4. Illustrate Peterson’s solution for critical section problem. which requirement of the solution for critical section problem it fulfils?
5. Illustrate Hardware solution for critical section problem using test and set instruction.
6. Illustrate Hardware solution for critical section problem using compare and swap.
7. Explain mutex lock.
8. Explain semaphore & types of semaphore.
9. Explain deadlock and starvation in semaphore.
10. Enlist classical problem of synchronization. Explain bounded buffer/reader-Writer/Dining philosopher problem in detail.
11. Explain Dining philosopher problem using semaphore. Describe possible remedies to avoid dead lock.
12. Explain Monitor.
13. Explain deadlock. Describe the sequence in which a process may utilize a resource (Expected answer: Request, use, release)
14. Describe necessary condition in which deadlock situation can arise.
15. Describe resource allocation graph with suitable diagram. (Assingment edge, request edge, resource instances, process states, cycles exist in the system/graph, deadlock or no deadlock)
16. Enlist Method of handling deadlock. Explain in detail deadlock prevention/deadlock avoidance/deadlock detection and recovery.
17. Explain safe and unsafe state with diagram.
18. Explain resource allocation-graph-algorithm in deadlock avoidance (claim edge)
19. Explain Bankes algorithm in deadlock. (problem: page no 332, 333, 340)
20. Explain safety & resource-request algorithm.
21. Explain deadlock detection (7.6.1, 7.6.2, 7.6.3)
22. How recovery from deadlock can be achieved. (7.7.1, 7.7.2)

\*\*\*