Question Bank for practice Only

Unit-1

5 marks questions

Q.1 What is the primary function of an Operating System?

Q.2 Define virtualization and containerization in computing environments.

Q.3 Explain the role of system calls in an Operating System.

Q.4 What are open source operating systems? Provide two examples.

Q.5 Differentiate between system boot and system calls.

10 marks questions

Q.1 Describe the history of operating systems and highlight key milestones in their evolution.

Q.2 Explain the structure of an operating system and how it interacts with hardware components.

Q.3 What are the different types of operating systems? Discuss with examples.

Q.4 Compare and contrast system calls in Windows and Unix operating systems with relevant examples.

Q.5 Discuss the functions and services of an operating system in a modern computing environment.

15 marks questions

Q.1 Explain in detail the concept of computing environments and the role of an operating system in managing these environments.

Q.2 How does virtualization enhance the efficiency of resource management in modern operating systems?

Q.4 What are the different types of system calls? Provide examples from both Unix and Windows operating systems.

Q.5 Discuss the process of system boot from powering on the computer to loading the operating system.

Q.4 Evaluate the importance of open-source operating systems in today’s technological landscape. Provide case studies where open-source operating systems are widely used.

Unit-2

5 Marks Questions:

1. Differentiate between a program and a process.
2. What is a Process Control Block (PCB), and why is it important?
3. Explain the concept of process context switching.
4. What is the difference between user-level threads and kernel-level threads?
5. List and briefly explain the types of process scheduling queues.

10 Marks Questions:

1. Describe the different process states and explain how a process transitions between them.
2. Explain the First Come First Serve (FCFS) scheduling algorithm with an example.
3. Compare and contrast preemptive and non-preemptive process scheduling with examples.
4. Discuss the benefits and challenges of multithreading in modern operating systems.
5. Explain priority scheduling (both preemptive and non-preemptive) and discuss the potential issues associated with priority inversion.

15 marks questions

Q.1 Round Robin (RR) Scheduling Problem: Consider the following set of processes with their burst times and arrival times. The time quantum for the Round Robin scheduling algorithm is 4 units.

Process	Arrival Time	Burst Time
P1	0	10
P2	1	4
P3	2	6
P4	3	8
P5	4	12

a) Draw the Gantt chart for the Round Robin scheduling algorithm.

b) Calculate the waiting time and turnaround time for each process.

c) Compute the average waiting time and average turnaround time.

Q.2 Shortest Job First (SJF) Scheduling Problem: Given the following processes with their burst times and arrival times, schedule them using the Shortest Job First (non-preemptive) scheduling algorithm.

Process	Arrival Time	Burst Time
P1	0	7
P2	2	4
P3	4	1
P4	5	4
P5	6	3

a) Draw the Gantt chart using the SJF scheduling algorithm.

b) Calculate the waiting time and turnaround time for each process.

c) Compute the average waiting time and average turnaround time.

Unit-3

5 Marks Questions:

1. What is Inter Process Communication (IPC), and why is it necessary in operating systems?
2. Differentiate between message passing and shared memory as IPC mechanisms.
3. What is a critical section, and why is it important in process synchronization?
4. Explain the concept of a race condition with an example.
5. What is the producer-consumer problem? Provide a brief description.

10 Marks Questions:

1. Explain how shared memory and message passing differ in terms of performance and complexity.

2. Describe pipes and named pipes in Linux. How do they enable communication between processes?
3. What are semaphores? Explain the difference between a binary semaphore and a counting semaphore.
4. What is the Critical Section Problem? Discuss the three requirements for a valid solution (mutual exclusion, progress, and bounded waiting).
5. Explain Peterson's solution for the critical section problem. How does it ensure mutual exclusion and avoid race conditions?

15 Marks Questions:

1. Producer-Consumer Problem with Semaphores: The producer-consumer problem involves two processes: one producing data and another consuming it. Implement a solution using semaphores for synchronization between the producer and the consumer.

a) Explain the producer-consumer problem and why synchronization is needed.

b) Write pseudo-code for solving this problem using semaphores (binary or counting).

c) How does the use of semaphores solve the issue of race conditions in this problem?

2. Classic Synchronization Problem – Dining Philosophers: The dining philosophers problem is a classic synchronization issue. Describe how semaphores or monitors can be used to prevent deadlock and ensure proper synchronization among the philosophers.

a) Explain the dining philosophers problem and its relevance to process synchronization.

b) Provide a solution using semaphores or monitors to solve this problem.

c) Discuss potential issues like deadlock and starvation in this context, and how the solution addresses them.

3. Critical Section Problem Solutions: Discuss different solutions to the critical section problem, including both hardware and software-based methods.

a) Compare hardware solutions like test-and-set and compare-and-swap with software solutions like Peterson's algorithm and the bakery algorithm.

b) Explain Peterson's solution and the bakery algorithm in detail with examples.

c) Discuss the advantages and disadvantages of these solutions in real-world systems.