23 Lecture - CS506

Important Subjective

Certainly, here are 10 short subjective questions related to Multithreading along with their answers:

Question 1: What is multithreading?

Answer: Multithreading is a programming technique where multiple threads execute independently within a single process, enabling concurrent execution of tasks.**

Question 2: What is the difference between a process and a thread?

Answer: A process is an independent program with its own memory space, while a thread is a lightweight execution unit within a process, sharing the same memory space.**

Question 3: What are the advantages of using multithreading in a program?

Answer: Advantages include improved CPU utilization, better responsiveness, efficient resource sharing, and simplified program design for concurrent tasks.**

Question 4: What is a race condition in multithreading?

Answer: A race condition occurs when multiple threads access and modify shared data concurrently, leading to unpredictable and undesirable outcomes due to improper synchronization.**

Question 5: Explain the concept of synchronization in multithreading.

Answer: Synchronization ensures that multiple threads access shared resources or critical sections in an orderly manner, preventing conflicts and ensuring data consistency.**

Question 6: What is a deadlock in multithreading?

Answer: Deadlock happens when two or more threads are blocked, each waiting for a resource that another thread holds, resulting in a standstill where none of the threads can proceed.**

Question 7: How does multithreading contribute to responsiveness in graphical user interfaces (GUIs)?

Answer: Multithreading allows tasks like handling user input and updating the GUI to run in separate threads, ensuring that the interface remains responsive even during heavy processing.

Question 8: What is the purpose of a mutex in multithreading?

Answer: A mutex (mutual exclusion) is a synchronization primitive used to control access to shared resources. It ensures that only one thread can access the resource at a time.

Question 9: How can you prevent race conditions in multithreaded programs?

Answer: Preventing race conditions involves proper synchronization using techniques like mutexes, semaphores, and thread-safe data structures.

Question 10: What are thread pools, and why are they useful in multithreading?

Answer: A thread pool is a collection of pre-initialized threads that can be used to execute tasks concurrently. They improve performance by avoiding the overhead of creating and destroying threads frequently.