Ch 5 Concurrency: Mutual Exclusion and Synchronization

1. **List four design issues for which the concept of concurrency is relevant.**

   1. The OS must be able to keep track of the various processes. This is done with the use of process control blocks

   2. The OS must allocate and de-allocate various resources for each active process.
   At times, multiple processes want access to the same resource

   3. The OS must protect the data and physical resources of each process against unintended interference by other processes.
   This involves techniques that relate to memory, files, and I/O devices.

   4. ensure that the processes and outputs are independent of the processing speed.

2. **List the requirements for mutual exclusion.**

   1. mutual exclusion must be enforced
   2. A process that halts must do so without interfering with other processes
   3. No deadlock or starvation.
   4. A process must not be denied access to a critical section when there is no other process using it.
   5. No assumptions are made about relative process speeds or number of processes
   6. A process remains inside its critical section for a finite time only.

3. **List the three control problems associated with competing processes and briefly define each.**

   Mutual Exclusion:
   when one process is in a critical section that accesses shared resources, no other process may be in a critical section that accesses any of those shared resources.

   Deadlock:
   two or more processes are unable to proceed because each is waiting for one of the others to do something.

   Starvation:
   A runnable process is overlooked indefinitely by the scheduler; although it is able to proceed, it is never chosen.

4. **List three degrees of awareness between processes and briefly define each.**

   Unaware of each other:
   Competition.
   Problems: mutual exclusion, deadlock, starvation.

   Aware indirectly:
   Cooperation by sharing
   Problems: mutual exclusion, starvation, data coherence

   Aware directly:
   Cooperation by communication.
   Problems: deadlock, starvation

5. **What are three contexts in which concurrency arises?**

   • Multiple applications: invented to allow processing time to be shared among a number of active applications.

   • Structured applications:
   As an extension of the principles of modular design and structured programming

   • Operating system structure:
   The operating systems themselves as a set of processes or threads.

6.	**What conditions are generally associated with the readers/writers problem?**	1. Any number of readers may simultaneously read the file. 2. Only one writer at a time may write to the file. 3. If a writer is writing to the file, no reader may read it
7.	**What is a monitor?**	The monitor is a programming-language construct that provides equivalent functionality to that of semaphores and that is easier to control.
8.	**What is the basic requirement for the execution of concurrent processes?**	Mutual exclusive the ability to exclude all other processes from a course of action while one process is granted that ability.
9.	**What is the difference between binary and general semaphores?**	the semaphore value in binary just 0 and 1. general semaphores can set the value to other integers.
10.	**What is the difference between strong and weak semaphores?**	Strong semaphores put the blocked process into a FIFO queue thus first blocked process will be the first one be pick. Weak semaphores: the order in which processes are removed from the queue is not specified.
11.	**What is the distinction between blocking and nonblocking with respect to messages?**	• Blocking send, blocking receive: Both the sender and receiver are blocked until the message is delivered; this is sometimes referred to as a rendezvous. This combination allows for tight synchronization between processes. • Nonblocking send, blocking receive: Although the sender may continue on, the receiver is blocked until the requested message arrives.This is probably the **most useful combination**. It allows a process to send one or more messages to a variety of destinations as quickly as possible. A process that must receive a message before it can do useful work needs to be blocked until such a message arrives. An example is a server process that exists to provide a service or resource to other processes. • Nonblocking send, nonblocking receive: Neither party is required to wait
12.	**What is the distinction between competing processes and cooperating processes?**	Competing processes: compete for resources, like access to the same file or I/O device, processor time. Cooperating processes: share resources or communication. may or may not be ware of each other.
13.	**What operations can be performed on a semaphore?**	1. initialized to a nonnegative integer value. 2. semWait operation decrements the value 3. semSignal operation increments the value