- Process Synchronization is a concept in operating systems that refers to the coordination and control of multiple concurrent processes or threads to ensure proper and orderly execution.
- In a multi-process or multi-threaded environment, various processes often need to access shared resources or critical sections of code simultaneously.
- Process synchronization mechanisms are employed to prevent race conditions, data inconsistencies, and conflicts that can arise when multiple processes attempt to access shared resources simultaneously.
- The main goal of process synchronization is to enforce mutual exclusion, which ensures that only one process can access a shared resource or critical section at a time.
- This prevents data corruption and ensures that the results of concurrent operations are consistent and predictable.

Common mechanisms for process synchronization include:

1. Locks (Mutex): Mutexes (short for mutual exclusion) are used to protect shared resources. A process must acquire a mutex before accessing the shared resource and release it after completing the operation.
2. Semaphores: Semaphores are used to control access to shared resources. They can be used to allow multiple processes to access a resource up to a specified limit or to block access until the resource is available.
3. Conditional Variables: Conditional variables are used to signal between processes to coordinate their activities and to avoid busy-waiting.
4. Atomic Operations: Some hardware architectures provide atomic operations, which are low-level instructions that are guaranteed to be executed as a single, indivisible unit, ensuring that concurrent access to shared resources does not lead to data

inconsistency.