**1. Introduction to Operating System Process Scheduling**

* **Definition and Purpose:**
  + What is process scheduling?
  + Why is process scheduling important in operating systems?
  + Overview of scheduling objectives (e.g., CPU utilization, throughput, turnaround time, waiting time, and response time).

**2. Non-preemptive Scheduling Techniques**

* **Definition:**
  + Non-pre-emptive scheduling overview: A process cannot be interrupted and is allowed to complete its execution before another process is scheduled.
* **Common Non-preemptive Scheduling Techniques:**
  + **First-Come, First-Served (FCFS):**
    - Description: Processes are scheduled in the order they arrive.
    - Example: A queue where processes enter and are executed in sequence.
    - Pros and Cons: Simple to implement, but can lead to the "convoy effect" (where short processes wait for long ones to complete).
  + **Shortest Job Next (SJN) or Shortest Job First (SJF):**
    - Description: The process with the smallest execution time is scheduled next.
    - Example: If process A needs 5 units of time and process B needs 2 units, process B will execute first.
    - Pros and Cons: Minimizes average waiting time but requires knowing the execution time in advance and can cause starvation for longer processes.
  + **Priority Scheduling:**
    - Description: Each process is assigned a priority, and the process with the highest priority is scheduled next.
    - Example: If processes have priorities assigned, the one with the highest priority is executed first.
    - Pros and Cons: Can be preemptive or non-preemptive. Non-preemptive priority scheduling can lead to starvation for low-priority processes.
* **Illustrations and Examples:**
  + Use simple diagrams to demonstrate each technique.
  + Walk through a few example problems to show how each scheduling technique works in practice.

**3. Understanding Race Conditions (20 minutes)**

* **Definition:**
  + What is a race condition?
  + Explanation: A situation where the outcome of a process depends on the sequence or timing of uncontrollable events.
* **Examples of Race Conditions:**
  + Simple example: Two processes trying to update the same bank account balance.
  + Classic Example: The "Producer-Consumer" problem.
* **Impact of Race Conditions:**
  + Data inconsistency.
  + Unexpected behavior in processes.
* **Mitigation Strategies:**
  + **Mutual Exclusion:**
    - Explanation: Only one process can access a critical section at a time.
    - Example: Using locks or semaphores to ensure mutual exclusion.
  + **Synchronization Mechanisms:**
    - **Locks:** Ensuring only one process can access the critical section at a time.
    - **Semaphores:** Counting or binary semaphores to manage access to resources.
    - **Monitors:** High-level synchronization constructs that combine locking and condition variables.
* **Illustrations and Examples:**
  + Use diagrams or pseudocode to show how race conditions can occur and how synchronization mechanisms can resolve them.
  + Simple code examples demonstrating race conditions and their solutions.

**4. Q&A and Recap (10 minutes)**

* **Summary of Key Points:**
  + Review non-preemptive scheduling techniques and their characteristics.
  + Recap the concept of race conditions and strategies to handle them.