BESAGAS, JOHN PAUL
TN36

- **Explain the importance of thread synchronization in Java. Provide an example where lack of synchronization could lead to incorrect results in a multi-threaded application.**

  Thread synchronization in Java helps make sure that when multiple threads are trying to access the same resources, they don't interfere into each other and cause issues. Without it, simultaneous updates by two threads can lead to inconsistent results.

  For example, when developers update a shared database at the same time without coordination, one may overwrite the other's changes, causing inaccuracies. In Java, thread synchronization prevents this by allowing only one thread to access shared resources at a time, ensuring data remains accurate.

- **Compare and contrast the newFixedThreadPool and newCachedThreadPool methods in the Java Executor Framework. In what scenarios would you prefer one over the other?**

  In Java's Executor Framework, when you use newFixedThreadPool, you set up a thread pool with a set number of threads that just keep getting reused to tackle tasks. On the other hand, newCachedThreadPool is more flexible since it can grow or shrink its size as needed, creating new threads when there's a lot to do and reusing.

  I prefer using newFixedThreadPool for tasks that are steady and long-lasting because it helps manage resources more predictably. For quick bursts of short tasks, I prefer newCachedThreadPool since it adjusts easily to what's happening.