**🧵 Threading & Asynchronous Programming – Practice Exercises**

**🧠 Threads (Manual Threading)**

**📝 Exercise 1: Start, Pause, Resume, and Stop a Thread**

* Create a thread that prints numbers from 1 to 100.
* Use bool pause and bool stop flags to control the behavior.
* From Main(), pause for a few seconds, resume, then stop the thread.
* Use ThreadState to log status.

**📝 Exercise 2: Create Multiple Threads and Use Join**

* Create 3 threads that simulate independent tasks (like printing, computing, or waiting).
* Start all threads and use Join() to wait for all to complete before printing “All tasks done”.

**🧵 Thread Pool**

**📝 Exercise 3: Use ThreadPool to Run Lightweight Tasks**

* Create a method ProcessItem(object item) that prints a message with a delay.
* Use ThreadPool.QueueUserWorkItem to run this for 5 different items.

**⚡ Async / Await & Tasks**

**📝 Exercise 4: Simulate Parallel Downloads**

* Create DownloadFileAsync(string fileName, int delay)
* Simulate downloading 3 files in parallel.
* Use Task.WhenAll() to await them and print total download time.

**📝 Exercise 5: Async Method Returning a Value**

* Create GetTemperatureAsync(string city) that returns a random temp after delay.
* In Main(), call it for 3 cities and print the result.

**📝 Exercise 6: Use Task.WhenAny()**

* Create 3 tasks with different delays.
* Use Task.WhenAny() to find which one finishes first and print its name.

**📝 Exercise 7: Sequential vs Parallel Execution Time**

* Create SimulateWorkAsync(string name, int delay)
* First run 3 of them sequentially using await one after another.
* Then run all using Task.WhenAll().
* Compare total time taken for both runs using Stopwatch.

**📝 Exercise 8: Exception Handling in Async Code**

* Create RiskyTaskAsync(string name) where one of them throws an exception.
* Use try/catch around Task.WhenAll() and print which task failed.

**📝 Exercise 9: Cancellation Using CancellationToken**

* Create DoWorkAsync(CancellationToken token)
* Cancel the task from Main() after 2 seconds.
* Handle cancellation and print a message.