
🚀 Advanced-Level Exercises Across All Topics

🔷 1. Interface + Inheritance + Polymorphism – Payment Gateway System

Task:

- Create an interface `IPaymentProcessor` with `ProcessPayment()`.
- Create an abstract class `PaymentGateway` with `GatewayName` and a method `Validate()`.
- Implement `Razorpay`, `PayPal`, and `Stripe` classes.
- Process multiple payments polymorphically from a list of `IPaymentProcessor`.

🔷 2. Collection of Objects + LINQ – Student Rank List

Task:

- Create `Student` class with `Name`, `Marks`, `Grade`.
- Add 10 students to a `List<Student>`.
- Use LINQ to:
  - Sort by marks (descending)
  - Group by grade
  - Project the top 3 students using `.Take(3)`

🔷 3. Composition + File I/O – Bank Account Logger

Task:

- Create `BankAccount` class with `Deposit()` and `Withdraw()`.
- Store all transactions in a `List<Transaction>`.
- Save and load transaction history from a `.txt` file.
- Display summary on restart.

🔷 4. Thread with Dynamic Work Assignment

Task:

- Start a thread that waits for work via a shared `Queue<string>`.
- From `Main()`, enqueue 5 tasks while the thread runs.
- Use `lock` to synchronize access to the queue.
- Exit when all tasks are processed.

🔷 5. Async Task WhenAny – Fastest Server Simulation

Task:

- Create 3 async methods simulating responses from different servers (`Server1`, `Server2`, `Server3`) with random delay.
- Use `Task.WhenAny()` to get the fastest.
- Cancel the remaining using `CancellationTokenSource`.

🔹 6. Custom Collection Filter with LINQ – Inventory System

Task:

- Create an `Item` class with `Name`, `Type`, `Stock`, `Price`.
- Add 15 items to `List<Item>`.
- Use LINQ to:
  - Filter low stock items
  - Group by Type
  - Find highest priced item in each group

🔹 7. Abstract Class + Strategy Pattern (Manual)

Task:

- Create an abstract class `CompressionStrategy` with method `Compress(string inputPath, string outputPath)`.
- Implement `ZipCompression`, `RarCompression`.
- In a `Compressor` class, pass the strategy object.
- Use polymorphism to switch compression algorithm at runtime.

🔹 8. async/await + Parallel.ForEachAsync – Batch Image Processing

Task:

- Simulate processing 20 image files using `Parallel.ForEachAsync`.
- Use `await Task.Delay(200ms)` per image to simulate processing.
- Print progress percentage in the console.

🔹 9. Exception Aggregation – Parallel Task Errors

Task:

- Start 5 tasks using `Task.WhenAll()`.
- Two of them should throw exceptions.
- Catch `AggregateException` and log individual error messages.
- Still show summary after all tasks complete.

🔹 10. LINQ + Dictionaries – Word Frequency Analyzer

Task:

- Take a paragraph as string input.
- Break into words using `.Split()`.
- Use `Dictionary<string, int>` to count frequencies.
- Sort and display top 5 most frequent words using LINQ.
