**Problem 1 – Student Data Processor**

Create a program to manage student information for a small training course.

**Requirements:**

1. Create a **Student** class with properties:
   * Name (string, automatic property)
   * Age (nullable int)
   * Scores (List of int)
2. Constructor should:
   * Accept Name and optionally Age (nullable).
   * Initialize Scores list.
3. Add a method AddScores(params int[] scores) to add multiple scores at once.
4. Add a method GetAverageScore() that:
   * Uses TryParse if scores are read as strings before adding.
   * Returns the average score (or null if there are no scores).
5. In Main():
   * Create a list of students using var.
   * Add scores for each student.
   * Print student details using an **anonymous type** { Name, Age, AverageScore }.
   * Use a **lambda expression** to get all students with an average score above 80.

**Problem 2 – Product Box Inventory**

Create a generic inventory system for products.

**Requirements:**

1. Create a **generic class** Box<T> with:
   * Property Item of type T.
   * Method ShowInfo() that prints details of the Item.  
     *(Hint: you can use dynamic to access unknown members)*
2. Create a Product class with:
   * Name (string)
   * Price (double)
3. In Main():
   * Create several Box<Product> objects.
   * Initialize Product objects using object initializers or anonymous types where appropriate.
   * Use **ref** to update the price of a product.
   * Use a **lambda expression** to filter products above a certain price.

**Problem 3 – Mixed Data Utility**

Write a small utility program that:

1. Accepts a list of **mixed-type data** using object[].
2. Uses **var**, **dynamic**, and **object** to:
   * Store values temporarily.
   * Safely process integers and strings separately.
3. Use TryParse to:
   * Convert all string numbers into integers and sum them.
4. Use a **lambda expression** to:
   * Extract only the numbers greater than 50 from the list.
5. Display all results.