**Problem 1 – Payment System (Method Overriding)**

Create a base class **Payment** with properties Amount and Currency.

* Add a **virtual method** ProcessPayment() that prints a generic message.
* Create two derived classes:
  1. **CreditCardPayment** → Add a property CardNumber and override the method to show payment details.
  2. **PayPalPayment** → Add a property Email and override the method to show payment details.
* In the Main() method, create objects of both classes but reference them using the base class, then call ProcessPayment().

**Problem 2 – Employee Bonus (Method Overriding)**

Create a base class **Employee** with properties Name and Salary.

* Add a **virtual method** CalculateBonus() that returns 5% of the salary.
* Create two derived classes:
  1. **Manager** → override the method to return 15% of salary.
  2. **Developer** → override the method to return 10% of salary.
* In Main(), create objects of Manager and Developer and display their names with calculated bonuses.

**Problem 3 – Calculator (Method Overloading)**

Create a class **Calculator** and overload a method Add() in the following ways:

1. **Add(int a, int b)** → returns sum of two integers.
2. **Add(double a, double b)** → returns sum of two doubles.
3. **Add(int a, int b, int c)** → returns sum of three integers.

* In Main(), test all overloaded methods and print the results.

**Problem 4 – Student (Constructor Overloading with Chaining)**

Create a class **Student** with properties Name and Age.

* Create two constructors:
  1. A constructor that accepts only Name and sets Age to a default value of 18.
  2. A constructor that accepts both Name and Age but uses **constructor chaining** to call the first constructor.
* In Main(), create two student objects: one with only a name, and one with both name and age. Display their details.

**Problem 5 – Vector (Operator Overloading)**

Create a class **Vector** with properties X and Y.

* Overload the + operator to add two vectors.
* Overload the - operator to subtract two vectors.
* Overload the == and != operators to compare two vectors.
* Override ToString() to return vector format (X, Y).
* In Main(), create two vectors and test addition, subtraction, and equality comparison.

**Problem 6 – Sealed Method (Prevent Further Overriding)**

Create a base class **BaseClass** with a **virtual method** ShowMessage().

* Create a derived class **MiddleClass** that overrides the method and mark it as **sealed**.
* Try to override the method again in another class **DerivedClass** (it should cause an error).
* In Main(), create an object of MiddleClass and call ShowMessage().