**Instance Class**

1. Create a class called Employee with properties for name, age, and salary. Implement a method to display employee details.

2. Create a class called BankAccount with properties for account number, account holder name, and balance. Implement methods for deposit, withdrawal, and displaying the account details.

**Static Class:**

3. Create a static utility class named MathHelper with a static method CalculateAverage that takes an array of integers as input and returns their average.

4. Implement a static logger class called Logger that has a method LogMessage for writing messages on console. Demonstrate its usage in a simple console application.

**Partial Class:**

5. Define a partial class Person with one part containing properties like FirstName and LastName, and another part with methods like PrintFullName to display the full name. Implement these parts in separate files.

6. Create a partial class Employee with properties representing employee details. In another part, implement methods for calculating salary based on different factors.

**Abstract Class:**

7. Define an abstract base class Shape with an abstract method CalculateArea. Derive classes like Circle and Rectangle from Shape and implement the area calculation methods for each.

8. Design an abstract class Animal with properties like Name and Age. Derive classes like Dog and Cat from Animal with their unique methods.

**Sealed Class:**

9. Create a base class Vehicle with methods like StartEngine and StopEngine. Derive a class Car from Vehicle and seal it. Try to create a class that inherits from Car and observe the behavior.

10. Design a class BankAccount with properties like AccountNumber and Balance. Implement a sealed class SavingsAccount that extends BankAccount with methods for interest calculation.