1. **Vehicle Hierarchy**

   Scenario:

   A **vehicle manufacturing company** needs a system to manage different types of vehicles like **Cars, Bikes, and Trucks**. All vehicles have common properties like **brand, speed, and fuel type**, but specific behaviors differ.

   Tasks:

   - Create a base class Vehicle with attributes like brand, speed, and fuelType.
   - Create subclasses Car, Bike, and Truck with additional attributes and methods.
   - Implement method overriding for displayDetails().
   - Implement **multilevel inheritance**, such as ElectricCar inheriting from Car.
   - Demonstrate the use of constructors and the super keyword.

2. **E-Commerce Platform**

   Scenario:

   An online shopping website manages **Customers, Products, and Orders**. Customers can purchase multiple products, and an order contains multiple products.

   Tasks:

   - Create a base class User with name and email.
   - Create a Customer subclass with shoppingCart as an attribute.
   - Create a Product class with productID, name, and price.
   - Create an Order class that holds multiple Product objects.
   - Implement **associations** (Customer has multiple Orders).

3. **Car Rental System**

   Scenario:

   A car rental company offers **Economy Cars, SUVs, and Luxury Cars**. Each car has a **registration number, model, and rental price**, but pricing rules vary.

   Tasks:

   - Create a Car class with registrationNumber, model, and rentalPrice.
   - Create subclasses EconomyCar, SUV, and LuxuryCar.

- Implement method overriding for calculateRentalCost().
- Create a RentalSystem class to handle bookings.

4. **Ride-Sharing System**

   Scenario:

   A ride-sharing company offers different types of rides like **Economy, Luxury, and Shared Rides**. Each ride has a **driver, passenger, and fare**, but pricing varies.

   Tasks:

   - Create a Ride class with driver, passenger, and fare.
   - Create subclasses EconomyRide, LuxuryRide, and SharedRide.
   - Implement method overriding for calculateFare().
   - Create a RideBookingSystem that assigns drivers to passengers.

5. **Smart Home Automation**

   Scenario:

   A smart home system manages devices such as **Lights, Fans, and Air Conditioners**. All devices have an **ID, status (ON/OFF), and power consumption**, but their functionalities differ.

   Tasks:

   - Create a base class SmartDevice with deviceID, status, and powerConsumption.
   - Create subclasses Light, Fan, and AirConditioner.
   - Implement **method overriding** for turnOn() and turnOff().
   - Implement an interface RemoteControl with adjustSettings().