Inheritance & Abstraction Tutorial

Question 1 - Online Shopping System

Imagine you are building a simplified online shopping system. You have different types of products to sell, such as **electronics, clothing, and books**. Each product has some common properties like a **name, price, and description**, but they also have some unique characteristics.

Your task is to create an abstract class called Product to represent these common properties and behaviors. Then, create concrete subclasses for three types of products: Electronics, Clothing, and Books. Each subclass should inherit from the Product class and provide their own specific implementations for any unique properties or methods.

1. Define an abstract class Product with the following attributes and methods:

   Attributes: name (String), price (double), and description (String).
   An abstract method calculateShippingCost that returns a double representing the shipping cost for the product.

2. Now, create three concrete subclasses (Electronics, Clothing, and Books) that inherit from the Product class. Each subclass should include specific properties or methods related to their product type
   - Electronics should have an additional attribute warrantyPeriod (int in months) and a method isWarrantyValid that returns a boolean.
   - Clothing should have an additional attribute size (String) and a method getSize that returns the size.
   - Books should have an additional attribute author (String) and a method getAuthor that returns the author's name.

3. Now create a non abstract method in the abstract class printProductDescription which prints the product description in the console. (Use System.out.print)

4. Additionally, implement the calculateShippingCost method in each subclass:
   - For Electronics, the shipping cost is 2% of the product price.
   - For Clothing, the shipping cost is a flat rate of $5.00.
   - For Books, the shipping cost is 5% of the product price with a minimum of $2.00.

5. Finally have a main method and create few objects to each child class and test out the functionality of your code

Question 2 - Payment Processing System

You are tasked with developing a payment processing system for an e-commerce platform. The platform accepts various payment methods, including credit cards and digital wallets. Each payment method has common functionality, such as processing payments and refunding transactions, but they may also have unique features.

Your task is to design an interface, PaymentMethod, that defines the common methods for processing payments and refunds. Then, create two classes that implement this interface: CreditCardPayment and DigitalWalletPayment, each with their own specific implementation.

1. Define an interface called PaymentMethod with the following methods:

   PaymentMethod Interface:
   - processPayment(double amount): A method that processes a payment for the given amount and returns a boolean indicating whether the payment was successful.
   - refundPayment(double amount): A method that refunds a payment for the given amount and returns a boolean indicating whether the refund was successful.

2. Now, create two classes, CreditCardPayment and DigitalWalletPayment, that implement the PaymentMethod interface:

   CreditCardPayment Class:

   - Implement the methods processPayment and refundPayment for credit card payments. (you should provide simple print statements within the methods to indicate that the payment or refund is being processed)
   - Include an attribute creditCardNumber (String) to represent the credit card number.
   - Implement a constructor that takes the credit card number as a parameter.

   DigitalWalletPayment Class:

   - Implement the methods processPayment and refundPayment for digital wallet payments. (you should provide simple print statements within the methods to indicate that the payment or refund is being processed)
   - Include an attribute walletAccountID (String) to represent the digital wallet account ID.
   - Implement a constructor that takes the wallet account ID as a parameter.

3. Finally have a main method and create few objects to each child class and test out the functionality of your code