Assignment 1

Student Name: _____

Registration No: _____

Instructor Name: Laiba Akhund

Marks: 05

Instructions:

- Perform all tasks in separate .java files.
- Place all files in a folder named your Name_Reg.No. (e.g. LaibaAkhund_44146).
- Compress the folder in a single (.zip) file named same as your folder.
- Submit compressed file on LMS (Moodle App).
- **-100** policies for plagiarism.

Question 1:

Scenario:

You are building a library system with books and different types of members (e.g., regular members and premium members).

1. Create a Book class with properties like title, author, ISBN, and isAvailable.
2. Define a Member class with details like name, memberId, and borrowBook().
3. Extend Member into two subclasses: RegularMember and PremiumMember. Premium members can borrow multiple books at once, while regular members can borrow only one at a time.

Task:

Implement these classes and demonstrate how a PremiumMember and RegularMember would borrow books differently.

Question 2:

Scenario:

A video streaming service has different types of content: Movie and Series. Each has a method to play(), but the behavior varies between the two (e.g., Movie plays one file while Series might play an episode list).

Task:

1. Create a superclass Content with a play() method.
2. Create subclasses Movie and Series, each with their own overridden play() methods to reflect the specific playback functionality.
3. Write code to demonstrate polymorphism, where both Movie and Series are treated as Content but exhibit different behaviors.

Question 3:

Scenario:

You are designing a vehicle management system that keeps track of vehicles' details, such as speed, fuel level, and mileage.

1. Create a Vehicle class with private properties: speed, fuelLevel, and mileage.
2. Implement public getter and setter methods for each property, ensuring appropriate data validation (e.g., speed cannot be negative, fuel level cannot exceed the tank capacity).

Task:

Write a program to create a Vehicle object, set its speed, fuel level, and mileage using setters, and retrieve the values using getters. Demonstrate validation by trying to set invalid values.

Question 4:

Scenario:

In an employee management system, employees can be categorized into full-time and part-time.

1. Create an Employee class with attributes name, id, salary, and a method calculateSalary().
2. Extend the Employee class into FullTimeEmployee and PartTimeEmployee, each having different logic in the calculateSalary() method.

Task:

Implement these classes and demonstrate how the salary is calculated differently for FullTimeEmployee and PartTimeEmployee.

Question 5:

Scenario:

In a payment processing system, different payment methods like CreditCard and Paypal have unique processes to validate payment details.

Task:

1. Create an abstract class Payment with an abstract method validatePayment().
2. Implement subclasses CreditCard and Paypal, each providing its own implementation of validatePayment().
3. Use an interface Refundable with a method refund(). Implement this interface in CreditCard but not in Paypal.
4. Write code to demonstrate the use of both the abstract class and the interface.

Question 6:

Scenario:

A banking application needs to manage transactions, ensuring that no transaction results in a negative balance.

Task:

1. Design a BankAccount class with an attribute balance and methods deposit(double amount) and withdraw(double amount).
2. In the withdraw method, throw an exception if the amount is greater than the current balance.
3. Demonstrate exception handling by writing code to attempt a withdrawal that exceeds the balance, and catch the exception to display an appropriate error message.

Question 7:

Scenario:

A car manufacturing company wants to keep track of different Car models and their components, such as Engine and Transmission.

Task:

1. Design a Car class that has attributes for make and model, and contains objects of Engine and Transmission classes as its components.
2. Define basic attributes and methods in the Engine and Transmission classes, such as Engine having horsepower and Transmission having gear type.
3. Demonstrate composition by creating an instance of Car with specific Engine and Transmission objects.