Week 4 Lab: Java Exercises & Problems on OOP (Encapsulation, Inheritance, and Polymorphism)

These exercises focus on **Encapsulation, Inheritance, and Polymorphism**, ensuring students understand **class relationships, method overriding, and access control**.

---

1. Encapsulation - Employee Salary Management

Problem:
Create an Employee class with private attributes name, salary, and position.

- Provide getter and setter methods with validation (salary should not be negative).
- Create an increaseSalary(double amount) method to increase the salary.
- Test the class by creating multiple employees and displaying their details.

---

2. Inheritance - Animal and Dog

Problem:
Create a superclass Animal with a method makeSound().

- Create a subclass Dog that overrides the makeSound() method.
- Create another subclass Cat with its own version of makeSound().
- Test polymorphism by calling makeSound() on different objects.

---

3. Constructor Overloading - Bank Account

Problem:
Create a BankAccount class with overloaded constructors:

- One constructor initializes an account with an opening balance.
- Another constructor initializes an account with no balance (default 0).
- Add deposit and withdrawal methods.

---

4. Polymorphism - Shape Area Calculation

Problem:
Create a superclass Shape with a method calculateArea().

- Create subclasses Circle and Rectangle, overriding calculateArea().
- Test by creating objects of Circle and Rectangle and displaying their area.

---

5. Method Overloading - Calculator

Problem:
Create a Calculator class with overloaded methods:

- add(int a, int b),
- add(double a, double b),
- add(int a, int b, int c).
  Test calling different versions of the method.

---

6. Super Keyword - Vehicle and Car

Problem:
Create a superclass Vehicle with attributes brand and speed.

- Create a subclass Car that extends Vehicle, adding fuelType.
- Use super() in the constructor to initialize the base class.
- Create a method displayDetails() in Car to print details.

---

7. Abstract Class - Employee Payroll

Problem:
Create an abstract class Employee with an abstract method calculateSalary().

- Create subclasses FullTimeEmployee and PartTimeEmployee, each implementing calculateSalary().
- Test by creating objects of both subclasses.

---

8. Interface - Payment System

Problem:
Create an interface Payment with a method processPayment().

- Implement it in CreditCardPayment and PayPalPayment classes.
- Call processPayment() on objects of both classes.

---

9. Final Keyword - Immutable User Data

Problem:
Create a class User with final fields username and email.

- Initialize these fields using a constructor.
- Provide only getter methods to make the class immutable.

---

10. Multiple Inheritance Using Interfaces - Flying Car

Problem:
Create an interface CarFeatures with a method drive().

- Create another interface AirplaneFeatures with fly().
- Create a class FlyingCar that implements both interfaces.
- Call both drive() and fly() methods on a FlyingCar object.

Expected Learning Outcomes

- **Encapsulation**: Private fields with getter and setter methods.
- **Inheritance**: Creating base and derived classes.
- **Polymorphism**: Method overriding and overloading.
- **Abstract Classes and Interfaces**: Implementing abstract methods.