ASSIGMENT DAY-2

Rishikesh RR

A776647

1.Create a product class having proid, proname and price as data members and display as a member function. Display 5 products with its details.

```
class Product {
int proid;
String proname;
double price;
Product(int proid, String proname, double price) {
this.proid = proid;
this.proname = proname;
this.price = price;
}
void display() {
System.out.println("Product ID: " + proid);
System.out.println("Product Name: " + proname);
System.out.println("Price: " + price);
System.out.println();
}
public static void main(String[] args) {
Product p1 = new Product(1, "TV", 75000.00);
Product p2 = new Product(2, "AC", 25000.00);
Product p3 = new Product(3, "FRIDGE", 15000.00);
Product p4 = new Product(4, "ironbox", 5000.00);
Product p5 = new Product(5, "fan", 2000.00);
p1.display();
p2.display();
p3.display();
p4.display();
p5.display();
}
}
```

2.Create an account class with accno, name, balance as data members and deposit, withdraw and check account balances as their methods. Based on the customers request, it needs to perform deposit, withdraw and balance checking and displays the accno, name, balance as the output.

```
class Account {
    int accno;
    String name;
    double balance;
    Account(int accno, String name, double balance) {
        this.accno = accno;
        this.name = name;
        this.balance = balance;
    }
    void deposit(double amount) {
        if (amount > 0) {
            balance += amount;
            System.out.println("Deposited: " + amount);
        } else {
            System.out.println("Invalid deposit amount");
        }
    }
    void withdraw(double amount) {
        if (amount > 0 && amount <= balance) {
            balance -= amount;
            System.out.println("Withdrawn: " + amount);
         } else {
            System.out.println("Invalid withdraw amount or insufficient balance");
        }}
    void checkBalance() {
        System.out.println("Account Balance: " + balance);
    }
    void display() {
        System.out.println("Account Number: " + accno);
        System.out.println("Account Holder: " + name);
        System.out.println("Account Balance: " + balance);
        System.out.println();
    }
    public static void main(String[] args) {
        Account acc1 = new Account(101, "John Doe", 5000.00);
        acc1.display();
        acc1.deposit(2000.00);
        acc1.checkBalance();
        acc1.withdraw(1500.00);
        acc1.checkBalance();
        acc1.withdraw(6000.00);
        acc1.display();
    }
```

3. Create a person class with attributes name,gender and age. Consider name as private and other two are having public access specifier. In what way, you access private variable. Finally, display the person details.

```
class Person {
    private String name;
    public String gender;
    public int age;
    Person(String name, String gender, int age) {
        this.name = name;
        this.gender = gender;
        this.age = age;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    void display() {
        System.out.println("Name: " + getName());
        System.out.println("Gender: " + gender);
        System.out.println("Age: " + age);
    }

    public static void main(String[] args) {
        Person person1 = new Person("Alice", "Female", 30);
        person1.display();
        person1.setName("Alice Johnson");
        person1.display();
    }
}
```

4. You are building a system to manage vehicles in a parking lot. Create a base class called Vehicle with attributes registrationNumber and brand. Derive two classes, Car and Motorcycle, from the Vehicle class. Each derived class should have additional attributes specific to the type of vehicle, such as numDoors for cars and engineType for motorcycles. Implement methods in each class to calculate the parking fee for the vehicle based on its type. Create instances of these classes and calculate the parking fees.

```
class Vehicle {
    String registrationNumber;
    String brand;
    Vehicle(String registrationNumber, String brand) {
        this.registrationNumber = registrationNumber;
        this.brand = brand;
    }
    double calculateParkingFee() {
        return 0.0;
    }
}
class Car extends Vehicle {
    int numDoors;
    Car(String registrationNumber, String brand, int numDoors) {
        super(registrationNumber, brand);
        this.numDoors = numDoors;
    }
    double calculateParkingFee() {
        return 20.0;
    }
}
class Motorcycle extends Vehicle {
    String engineType;
    Motorcycle(String registrationNumber, String brand, String engineType) {
        super(registrationNumber, brand);
        this.engineType = engineType;
    }
    double calculateParkingFee() {
        return 10.0;
    }
}
public class ParkingLot {
    public static void main(String[] args) {
        Vehicle car = new Car("ABC123", "Toyota", 4);
        Vehicle motorcycle = new Motorcycle("XYZ789", "Yamaha", "V-Twin");
        System.out.println("Car Parking Fee: $" + car.calculateParkingFee());
        System.out.println("Motorcycle Parking Fee: $" + motorcycle.calculateParkingFee());
    }
```

```
}
```

5.You are tasked with creating a class called AreaCalculator that can calculate the areas of various geometric shapes. The class provides multiple overloaded calculateArea methods for different shapes: square, rectangle, and circle. a) Implement the AreaCalculator class with overloaded calculateArea methods for squares, rectangles, and circles. b) In the main method, demonstrate the use of these methods to calculate the areas of a square with a side length of 5.0, a rectangle with dimensions 4.0 by 6.0, and a circle with a radius of 3.0. Print the results.

```
class AreaCalculator {
 double calculateArea(double side) {
        return side * side;
    }
    double calculateArea(double length, double width) {
        return length * width;
    }
    double calculateArea(double radius) {
        return Math.PI * radius * radius;
    }
    public static void main(String[] args) {
        AreaCalculator calculator = new AreaCalculator();
        System.out.println("Area of square: " + calculator.calculateArea(5.0));
        System.out.println("Area of rectangle: " + calculator.calculateArea(4.0, 6.0));
        System.out.println("Area of circle: " + calculator.calculateArea(3.0));
    }
}
```

6.Implement Multilevel inheritance with an example

```
class Vehicle {
    void start() {
        System.out.println("Vehicle is starting.");
    }
}
class Car extends Vehicle {
    void drive() {
        System.out.println("Car is driving.");
    }
}
class ElectricCar extends Car {
    void charge() {
        System.out.println("Electric car is charging.");
    }
}
public class MultilevelInheritanceWithCar {
    public static void main(String[] args) {
        ElectricCar tesla = new ElectricCar();
        tesla.start();
        tesla.drive();
        tesla.charge();
    }
}
```

7.Implement Hierarchical inheritance with an example

```
class Vehicle {
    void start() {
        System.out.println("Vehicle is starting.");
    }
}
class Car extends Vehicle {
    void drive() {
        System.out.println("Car is driving.");
    }
}
class ElectricCar extends Vehicle {
    void charge() {
        System.out.println("Electric car is charging.");
    }
}
class SportsCar extends Vehicle {
    void accelerate() {
        System.out.println("Sports car is accelerating.");
    }
}
public class HierarchicalInheritanceWithCar {
    public static void main(String[] args) {
        Car regularCar = new Car();
        ElectricCar tesla = new ElectricCar();
        SportsCar ferrari = new SportsCar();
        regularCar.start();
        regularCar.drive();
        tesla.start();
        tesla.charge();
        ferrari.start();
        ferrari.accelerate();
    }
}
```