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.

public class product {

private int proid;

private String proname;

private double price;

public product(int proid, String proname, double price) {

this.proid = proid;

this.proname = proname;

this.price = price;

}

public void display() {

System.out.println("Product ID: " + proid + ", Product Name: " + proname + ", Price: " + price);

}

public static void main(String[] args) {

product product1 = new product(1, "Laptop", 1200);

product product2 = new product(2, "Smartphone", 800);

product product3 = new product(3, "Tablet", 400);

product product4 = new product(4, "Smartwatch", 200);

product product5 = new product(5, "Headphones", 100);

product1.display();

product2.display();

product3.display();

product4.display();

product5.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.

public class account {

private int accno;

private String name;

private int balance;

public account(int accno, String name, int balance) {

this.accno = accno;

this.name = name;

this.balance = balance;

}

public void deposit(int amount){

if(amount>0){

balance+=amount;

System.out.println("Deposit Successful"+ "new balance="+balance);

}

else{

System.out.println("Depsoit amount should be greater than zero");

}

}

public void withdraw(int amount){

if(amount>0 && balance>=amount){

balance-=amount;

System.out.println("withdraw successfull"+"new balance"+ balance);

}

else{

System.out.println("invalid amount is entered");;

}

}

public int checkbalance(){

System.out.println("accno:"+accno);

System.out.println("name:"+name);

System.out.println("balance:"+balance);;

return 0;

}

public static void main(String[] args) {

account acc=new account(123,"vishnupriyan",200000);

// Perform deposit

acc.deposit(500);

// Perform withdrawal

acc.withdraw(200);

// Check balance

acc.checkbalance();

}

}

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.

public class person {

private String name;

public int age;

public String gender;

public person(String name, int age, String gender){

this.name=name;

this.age=age;

this.gender=gender;

}

public String getName(){

return name;

}

public void display(){

System.out.println("Name: " + getName() + ", Gender: " + gender + ", Age: " + age);

}

public static void main(String[] args) {

// Create a person instance

person person = new person("vishnu",22,"Male");

// Display the person details

person.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.

// Base class Vehicle

class Vehicle {

private String registrationNumber;

private String brand;

public Vehicle(String registrationNumber, String brand) {

this.registrationNumber = registrationNumber;

this.brand = brand;

}

public String getRegistrationNumber() {

return registrationNumber;

}

public String getBrand() {

return brand;

}

public double calculateParkingFee() {

// Default implementation, should be overridden in derived classes

return 0.0;

}

}

// Derived class Car

class Car extends Vehicle {

private int numDoors;

public Car(String registrationNumber, String brand, int numDoors) {

super(registrationNumber, brand);

this.numDoors = numDoors;

}

public int getNumDoors() {

return numDoors;

}

@Override

public double calculateParkingFee() {

// Assuming a flat rate for cars

return 20.0;

}

}

// Derived class Motorcycle

class Motorcycle extends Vehicle {

private String engineType;

public Motorcycle(String registrationNumber, String brand, String engineType) {

super(registrationNumber, brand);

this.engineType = engineType;

}

public String getEngineType() {

return engineType;

}

@Override

public double calculateParkingFee() {

// Assuming a flat rate for motorcycles

return 10.0;

}

}

// Main class to test the implementation

public class ParkingLot {

public static void main(String[] args) {

Car car = new Car("ABC123", "Toyota", 4);

Motorcycle motorcycle = new Motorcycle("XYZ789", "Yamaha", "V-twin");

System.out.println("Parking fee for the car: rs:" + car.calculateParkingFee());

System.out.println("Parking fee for the motorcycle: rs:" + 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 {

// Method to calculate the area of a square

public double calculateArea(double side) {

return side \* side;

}

// Method to calculate the area of a rectangle

public double calculateArea(double length, double width) {

return length \* width;

}

public static void main(String[] args) {

AreaCalculator calculator = new AreaCalculator();

double squareSide = 5.0;

double rectangleLength = 4.0;

double rectangleWidth = 6.0;

double circleRadius = 3.0;

double squareArea = calculator.calculateArea(squareSide);

double rectangleArea = calculator.calculateArea(rectangleLength, rectangleWidth);

double circleArea = calculator.calculateArea(circleRadius);

System.out.println("Area of the square: " + squareArea);

System.out.println("Area of the rectangle: " + rectangleArea);

System.out.println("Area of the circle: " + circleArea);

}

}

6. Implement Multilevel inheritance with an example

class Animal {

public void eat() {

System.out.println("This animal eats food.");

}

}

class Mammal extends Animal {

public void walk() {

System.out.println("This mammal walks on land.");

}

}

class Dog extends Mammal {

public void bark() {

System.out.println("The dog barks.");

}

}

public class MultilevelInheritanceExample {

public static void main(String[] args) {

Dog dog = new Dog();

dog.eat();

dog.walk();

dog.bark();

}

}