1. Create a Java class named Calculator with two methods:  
i) multiply method that takes two integers and returns their product.

ii) multiply method overload that takes three doubles and returns their product.

Write a simple program to demonstrate the use of method overloading by calling both versions of the multiply method and printing the results

public class Calculator {

public int multiply(int num1, int num2) {

return num1 \* num2;

}

public double multiply(double num1, double num2, double num3) {

return num1 \* num2 \* num3;

}

public static void main(String[] args) {

Calculator calculator = new Calculator();

int productInt = calculator.multiply(5, 6);

System.out.println("Product of integers: " + productInt);

double productDouble = calculator.multiply(2.5, 3.4, 1.2);

System.out.println("Product of doubles: " + productDouble);

}

}

2.  Create a class hierarchy representing different types of employees in a company. Design a base class**Employee** with fields for the employee's name, employee ID, and a method named calculateSalary() that returns the basic salary. Implement two subclasses: **Manager and Developer.**

Manager class should have an additional field for the bonus percentage. Developer class should have an additional field for the programming language.

Override the calculateSalary() method in both the Manager and Developer classes to include the bonus for managers and an extra allowance for developers. The basic salary for all employees is $50,000.

Write a program to create instances of managers and developers, call the calculateSalary method on each, and print the details

class Employee {

protected String name;

protected int employeeId;

protected double basicSalary = 50000;

public Employee(String name, int employeeId) {

this.name = name;

this.employeeId = employeeId;

}

public double calculateSalary() {

return basicSalary;

}

}

class Manager extends Employee {

private double bonusPercentage;

public Manager(String name, int employeeId, double bonusPercentage) {

super(name, employeeId);

this.bonusPercentage = bonusPercentage;

}

@Override

public double calculateSalary() {

double bonus = basicSalary \* bonusPercentage / 100;

return basicSalary + bonus;

}

}

class Developer extends Employee {

private String programmingLanguage;

private double allowance;

public Developer(String name, int employeeId, String programmingLanguage, double allowance) {

super(name, employeeId);

this.programmingLanguage = programmingLanguage;

this.allowance = allowance;

}

@Override

public double calculateSalary() {

return basicSalary + allowance;

}

}

public class Main {

public static void main(String[] args) {

Manager manager1 = new Manager("John Smith", 1234, 10);

Developer developer1 = new Developer("Jane Doe", 5678, "Python", 5000);

System.out.println("Manager " + manager1.name + " (ID: " + manager1.employeeId + ") earns $" + manager1.calculateSalary());

System.out.println("Developer " + developer1.name + " (ID: " + developer1.employeeId + ") with " + developer1.programmingLanguage + " skills earns $" + developer1.calculateSalary());

}

}

3. Implement a class hierarchy with a base class**Vehicle**and two derived classes **Car and Motorcycle**.

The Vehicle class should have a method named calculateSpeed() that returns the speed of the vehicle. Override it in other two classes, where**the speed is calculated as the product of the vehicle's speed and the number of passengers or wheels.**

Note:

a) Car class should have an additional field for the number of passengers.

b) Motorcycle class should have an additional field for the number of wheels.

Write a program to create instances of car and motorcycle, call the calculateSpeed method on each, and determine the vehicle with the highest effective speed.

class Vehicle {

protected double speed;

public Vehicle(double speed) {

this.speed = speed;

}

public double calculateSpeed() {

return speed;

}

}

class Car extends Vehicle {

private int numPassengers;

public Car(double speed, int numPassengers) {

super(speed);

this.numPassengers = numPassengers;

}

@Override

public double calculateSpeed() {

return super.calculateSpeed() \* numPassengers;

}

}

class Motorcycle extends Vehicle {

private int numWheels;

public Motorcycle(double speed, int numWheels) {

super(speed);

this.numWheels = numWheels;

}

@Override

public double calculateSpeed() {

return super.calculateSpeed() \* numWheels;

}

}

public class Main {

public static void main(String[] args) {

Car car = new Car(100, 4); // Speed = 100, Passengers = 4

Motorcycle motorcycle = new Motorcycle(80, 2); // Speed = 80, Wheels = 2

double carSpeed = car.calculateSpeed();

double motorcycleSpeed = motorcycle.calculateSpeed();

System.out.println("Car's effective speed: " + carSpeed);

System.out.println("Motorcycle's effective speed: " + motorcycleSpeed);

String fastestVehicle = carSpeed > motorcycleSpeed ? "Car" : "Motorcycle";

System.out.println("Vehicle with the highest effective speed: " + fastestVehicle);

}

}