1. Implement a ComplexNumber class that supports various operations likeaddition and multiplication. Use method overloading to provide differentways to perform these operations.

A.

```
class ComplexNumber {
    private double real;
    private double imaginary;
    public ComplexNumber(double real, double imaginary) {
        this.real = real;
        this.imaginary = imaginary;
    }
    public ComplexNumber add(ComplexNumber other) {
        double newReal = this.real + other.real;
        double newImaginary = this.imaginary + other.imaginary;
        return new ComplexNumber(newReal, newImaginary);
    }
    public ComplexNumber add(double real) {
        double newReal = this.real + real;
        double newImaginary = this.imaginary;
        return new ComplexNumber(newReal, newImaginary);
    }
    public ComplexNumber multiply(ComplexNumber other) {
        double newReal = this.real * other.real - this.imaginary * other.imaginary;
        double newImaginary = this.real * other.imaginary + this.imaginary * other.real;
        return new ComplexNumber(newReal, newImaginary);
    }
    public ComplexNumber multiply(double real) {
        double newReal = this.real * real;
        double newImaginary = this.imaginary * real;
        return new ComplexNumber(newReal, newImaginary);
    }
    public void print() {
```

```
        System.out.println(this.real + " + " + this.imaginary + "i");
    }

    public static void main(String[] args) {
        ComplexNumber c1 = new ComplexNumber(2, 3);
        ComplexNumber c2 = new ComplexNumber(-1, 2);
        System.out.print("Sum of c1 and c2: ");
        c1.add(c2).print();
        System.out.print("c1 + 5: ");
        c1.add(5).print();
        System.out.print("Product of c1 and c2: ");
        c1.multiply(c2).print();

        System.out.print("c2 * 2.5: ");
        c2.multiply(2.5).print();
    }
}
```

2. Create an abstract Shape class with an abstract method area(). Implementsubclasses Circle, Rectangle, and Triangle, each overriding the area()method to calculate the area specific to the shape. Additionally, provideoverloaded constructors for each subclass to handle different input types for the shape dimensions.

A.

```
abstract class Shape {
    public abstract double area();
}
class Circle extends Shape {
    private double radius;
    public Circle(double radius) {
        this.radius = radius;
    }
    public Circle(int diameter) {
        this.radius = diameter / 2.0;
```

```
    }
    public double area() {
        return Math.PI * radius * radius;
    }
}
class Rectangle extends Shape {
    private double length;
    private double width;
    public Rectangle(double length, double width) {
        this.length = length;
        this.width = width;
    }
    public Rectangle(double side) {
        this.length = side;
        this.width = side;
    }
    public double area() {
        return length * width;
    }
}
class Triangle extends Shape {
    private double base;
    private double height;
    public Triangle(double base, double height) {
        this.base = base;
        this.height = height;
    }
    public double area() {
        return 0.5 * base * height;
    }
}
```

```
public class ShapeTest {
    public static void main(String[] args) {
        Circle circle1 = new Circle(5.0);
        Circle circle2 = new Circle(10); // Using diameter constructor
        Rectangle rectangle1 = new Rectangle(4.0, 6.0);
        Rectangle square = new Rectangle(5.0); // Using square side constructor
        Triangle triangle = new Triangle(3.0, 4.0);
        System.out.println("Area of Circle (radius 5.0): " + circle1.area());
        System.out.println("Area of Circle (diameter 10): " + circle2.area());
        System.out.println("Area of Rectangle (4.0 x 6.0): " + rectangle1.area());
        System.out.println("Area of Square (side 5.0): " + square.area());
        System.out.println("Area of Triangle (base 3.0, height 4.0): " + triangle.area());
    }
}
```

3. Create a class hierarchy for an educational institution. The base classMember should have subclasses Student, Teacher, and Staff. Each subclassshould implement a method getDetails() to provide specific details and attributes relevant to the member type.

A.

```
class Member {
    protected String name;
    protected int age;
    protected String address;
    public Member(String name, int age, String address) {
        this.name = name;
        this.age = age;
        this.address = address;
    }
    public void getDetails() {
        System.out.println("Name: " + name);
        System.out.println("Age: " + age);
        System.out.println("Address: " + address);
```

```
    }
}
class Student extends Member {
    private int rollNumber;
    private String course;
    public Student(String name, int age, String address, int rollNumber, String course) {
        super(name, age, address);
        this.rollNumber = rollNumber;
        this.course = course;
    }
    public void getDetails() {
        super.getDetails();
        System.out.println("Roll Number: " + rollNumber);
        System.out.println("Course: " + course);
    }
}
class Teacher extends Member {
    private String subject;
    private double salary;
    public Teacher(String name, int age, String address, String subject, double salary) {
        super(name, age, address);
        this.subject = subject;
        this.salary = salary;
    }
    public void getDetails() {
        super.getDetails();
        System.out.println("Subject: " + subject);
        System.out.println("Salary: $" + salary);
    }
}
```

```
class Staff extends Member {
    private String department;
    private double salary;
    public Staff(String name, int age, String address, String department, double salary) {
        super(name, age, address);
        this.department = department;
        this.salary = salary;
    }
    public void getDetails() {
        super.getDetails();
        System.out.println("Department: " + department);
        System.out.println("Salary: $" + salary);
    }
}
public class EducationalInstitution {
    public static void main(String[] args) {
        Student student = new Student("John Doe", 20, "123 Main St, City", 101, "Computer Science");
        Teacher teacher = new Teacher("Jane Smith", 35, "456 Park Ave, Town", "Mathematics", 55000);
        Staff staff = new Staff("Michael Johnson", 28, "789 Broad Rd, Village", "Administration", 40000);
        System.out.println("Details of Student:");
        student.getDetails();
        System.out.println();
        System.out.println("Details of Teacher:");
        teacher.getDetails();
        System.out.println();
        System.out.println("Details of Staff:");
        staff.getDetails();
    }
}
```

4. Design a File class with subclasses TextFile, ImageFile, and VideoFile.Implement methods open(), close(), and getInfo(). Each subclass shouldoverride these methods to provide specific functionalities and attributes likeresolution for ImageFile and duration for VideoFile.

A.

```
abstract class File {
    protected String fileName;
    protected String fileType;
    public File(String fileName, String fileType) {
        this.fileName = fileName;
        this.fileType = fileType;
    }
    public abstract void open();
    public abstract void close();
    public abstract void getInfo();
}
class TextFile extends File {
    private int numberOfLines;
    public TextFile(String fileName, int numberOfLines) {
        super(fileName, "Text");
        this.numberOfLines = numberOfLines;
    }
    public void open() {
        System.out.println("Opening Text File: " + fileName);
    }
    public void close() {
        System.out.println("Closing Text File: " + fileName);
    }
    public void getInfo() {
        System.out.println("File Name: " + fileName);
        System.out.println("File Type: " + fileType);
        System.out.println("Number of Lines: " + numberOfLines);
    }
}
class ImageFile extends File {
```

```
    private int resolutionWidth;
    private int resolutionHeight;
    public ImageFile(String fileName, int resolutionWidth, int resolutionHeight) {
        super(fileName, "Image");
        this.resolutionWidth = resolutionWidth;
        this.resolutionHeight = resolutionHeight;
    }
    public void open() {
        System.out.println("Opening Image File: " + fileName);
    }
    public void close() {
        System.out.println("Closing Image File: " + fileName);
    }
    public void getInfo() {
        System.out.println("File Name: " + fileName);
        System.out.println("File Type: " + fileType);
        System.out.println("Resolution: " + resolutionWidth + "x" + resolutionHeight);
    }
}
class VideoFile extends File {
    private int duration;
    public VideoFile(String fileName, int duration) {
        super(fileName, "Video");
        this.duration = duration;
    }

    public void open() {
        System.out.println("Opening Video File: " + fileName);

    }
```

```
    public void close() {
        System.out.println("Closing Video File: " + fileName);
    }
    public void getInfo() {
        System.out.println("File Name: " + fileName);
        System.out.println("File Type: " + fileType);
        System.out.println("Duration: " + duration + " seconds");
    }
}
public class FileTest {
    public static void main(String[] args) {
        TextFile textFile = new TextFile("Document.txt", 100);
        ImageFile imageFile = new ImageFile("Photo.jpg", 1920, 1080);
        VideoFile videoFile = new VideoFile("Movie.mp4", 7200);
        textFile.open();
        textFile.getInfo();
        textFile.close();
        System.out.println();
        imageFile.open();
        imageFile.getInfo();
        imageFile.close();
        System.out.println();
        videoFile.open();
        videoFile.getInfo();
        videoFile.close();
    }
}
```