



JAVA LAB

Prepared by:

Name of Student : JEEVAN NAIDU

Roll No:150096723015

Batch: 2023-27

Dept. of CSE

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING



**INSTITUTE OF TECHNOLOGY AND MANAGEMENT
SKILLS UNIVERSITY,
KHARGHAR, NAVI MUMBAI**

CERTIFICATE

This is to certify that Mr. JEEVAN NAIDU

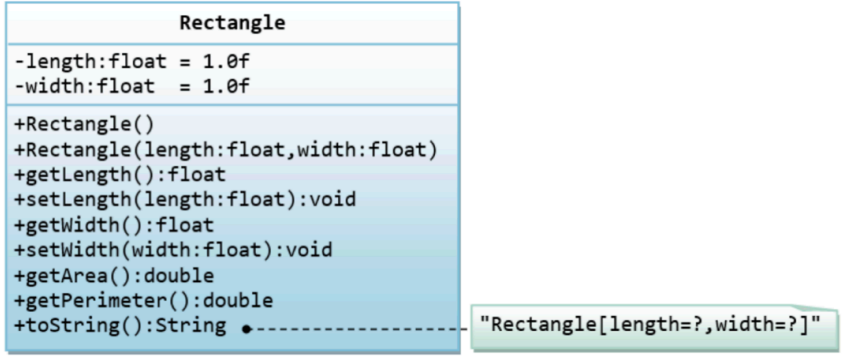
Roll No. 150096723015 Semester III of B.Tech Computer Science & Engineering,
ITM Skills University, Kharghar, Navi Mumbai , has completed the term work
satisfactorily in subject JAVA for the academic year 2024 - 2025 as prescribed in the
curriculum.

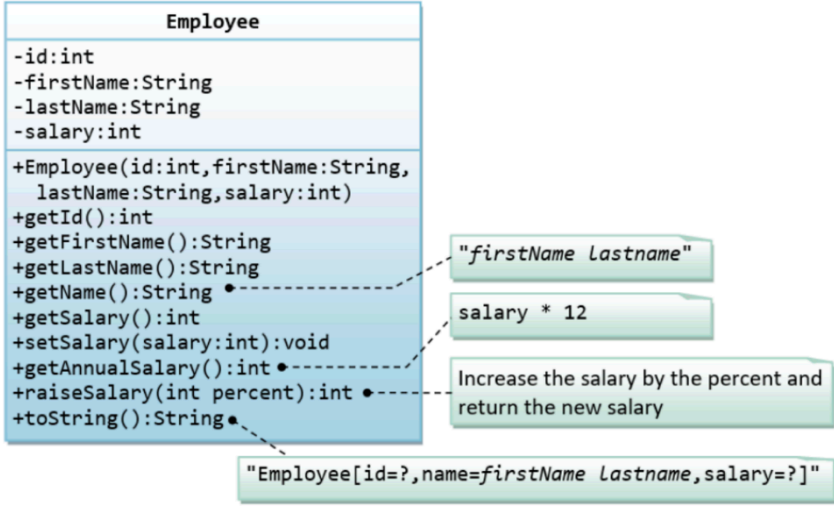
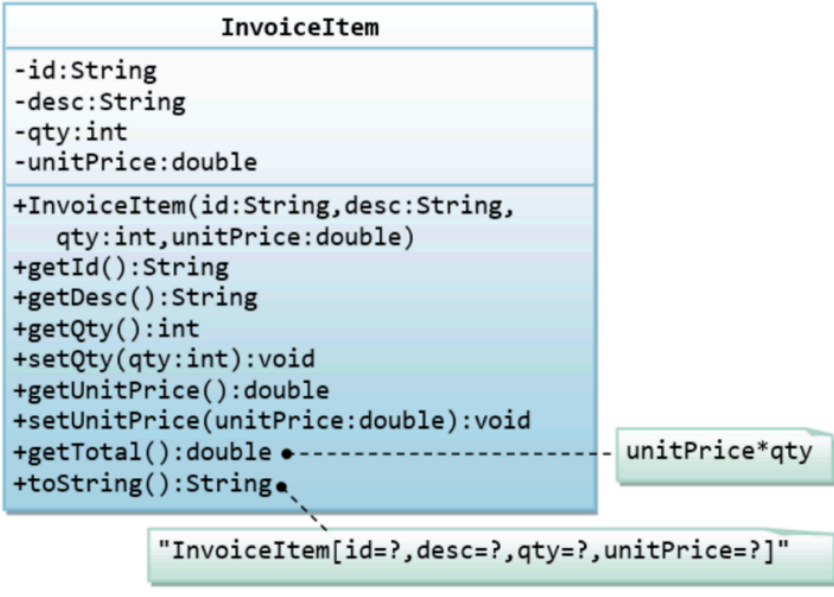
Place: NAVI MUMBAI

Date: 25 AUGUST 2024

Subject I/C

HOD

Exp. No	List of Experiment	Date of Submission	Sign
1	<p>WAP to create a class called Circle. It contains:</p> <ul style="list-style-type: none"> • private instance variable: radius (of the type double) with default value of 1.0. • Two overloaded constructors - a default constructor with no argument, and a constructor which takes a double argument for radius. • Two public methods: getRadius(), calculateArea(), calculateCircumference() which return the radius, calculate and return area, and circumference respectively. <p>Hint: Use Math.PI for calculating area and circumference</p>	25 August 2024	
2	<p>A class called Rectangle, which models a rectangle with a length and a width (in float), is designed as shown in the following class diagram. Write the Rectangle class as per UML diagram.</p>  <pre> classDiagram class Rectangle { -length:float = 1.0f -width:float = 1.0f +Rectangle() +Rectangle(length:float,width:float) +getLength():float +setLength(length:float):void +getWidth():float +setWidth(width:float):void +getArea():double +getPerimeter():double +toString():String } Rectangle "1" -- "*" Rectangle : Rectangle[length=?,width=?] </pre> <p>The UML diagram shows a class named Rectangle. It has two private attributes: -length:float = 1.0f and -width:float = 1.0f. The methods include a default constructor +Rectangle(), a parameterized constructor +Rectangle(length:float,width:float), and several accessor and mutator methods: +getLength():float, +setLength(length:float):void, +getWidth():float, +setWidth(width:float):void, +getArea():double, +getPerimeter():double, and +toString():String. A multiplicity diagram at the bottom right shows a dashed line with a filled circle at one end and an open circle at the other, labeled "Rectangle[length=?,width=?]".</p>	25 August 2024	

3	<p>A class called Employee, which models an employee with an ID, name and salary, is designed as shown in the following class diagram. The method raiseSalary(percent) increases the salary by the given percentage. Write the Employee class and its driver class.</p> 	25 August 2024	
4	<p>A class called InvoiceItem, which models an item of an invoice, with ID, description, quantity and unit price, is designed as shown in the following class diagram. It has a method getTotal which calculates total value (total=quantity*unit price). Write the InvoiceItem class and it's driver class.</p> 	25 August 2024	

5	<p>A class called Author (as shown in the class diagram) is designed to model a book's author. It contains:</p> <ul style="list-style-type: none"> • Three private instance variables: name (String), email (String), and gender (char of either 'm' or 'f'); • One constructor to initialize the name, email and gender with the given values; public Author (String name, String email, char gender) {.....} (There is no default constructor for Author, as there are no defaults for name, email and gender.) • public getters/setters: getName(), getEmail(), setEmail(), and getGender(); (There are no setters for name and gender, as these attributes cannot be changed.) • A toString() method that returns "Author[name=?,email=?,gender=?]", e.g., "Author[name=Abc ,email=Abc@gmail.com, gender=m]". 	25 August 2024	
---	---	----------------	--

6	<p>WAP to create a class time having default constructor, parameterized constructor whose specifications are as follows:</p> <ul style="list-style-type: none"> • Instance variable: hr, min, sec • Constructors: <ul style="list-style-type: none"> ▪ default (with no parameters passed; should initialize the represented time to 12:0:0) ▪ a constructor with three parameters: hours, minutes, and seconds. ▪ a constructor with one parameter: the value of time in seconds since midnight (it should be converted into the time value in hours, minutes, and seconds) • Instance methods: <ul style="list-style-type: none"> ▪ setClock() with one parameter seconds since midnight (to be converted into the time value in hours, minutes, and seconds as above). ▪ tick() with no parameters that increments the time stored in a Clock object by one second. ▪ tickDown() which decrements the time stored in a Clock object by one second. ▪ displaytime() displays the time in the format hr: min:sec e.g: 05:45:23 	25 August 2024	
---	---	----------------	--

7	<p>Write a Java class Complex for dealing with complex number. Your class must have the following features:</p> <ul style="list-style-type: none"> • Instance variables : <ul style="list-style-type: none"> o real for the real part of type double o imag for imaginary part of type double. • Constructor: <ul style="list-style-type: none"> o public Complex (): A default constructor, it should initialize the number to 0, 0) o public Complex (double real, double imag: A constructor with parameters, it creates the complex object by setting the two fields to the passed values. • Instance methods:public Complex add (Complex n): This method will find the sum of the current complex number and the passed complex number. The methods returns a new Complex number which is the sum of the two. <ul style="list-style-type: none"> o public Complex subtract (Complex n): This method will find the difference of the current complex number and the passed complex number. The methods returns a new Complex number which is the difference of the two. 	25 August 2024	
---	--	----------------	--

8	Create a Geometry class and overload calculateArea method for square, circle, and rectangle.	25 August 2024	
9	<p>WAP to implement Box class (Hint: Refer Java PPT). Inherit Box class in BoxWt whose</p> <p>a. instance variable is weight and b. method is print_BoxWt() c. constructors: default, parameterized and BoxWt(BoxWt ob)</p> <p>Use super() to invoke superclass constructors.</p>	25 August 2024	
10	WAP the demonstrate that super class variable can point to object of subclass.	25 August 2024	
11	WAP to demonstrate multilevel inheritance by creating a BoxColor class and inheriting BoxWt class. BoxColor class has an instance variable color of String type.	25 August 2024	
12	WAP to demonstrate the sequence of execution of constructors in multilevel inheritance.	25 August 2024	

13	WAP to demonstrate the concept of method overriding.	25 August 2024	
14	WAP to demonstrate the use of super keyword to call overridden methods and instance variables.	25 August 2024	
15	<p>Create a class Animal with</p> <ul style="list-style-type: none"> • instance variables: • boolean vegetarian • String food • int numOfLegs • Create a no-argument constructor and parameterized constructor. (Use "this") • Create getters and setters • Create a toString() method for animal class • Create a subclass Cat with instance variable: • String color • Create a no-argument constructor and parameterized constructor which has all four • parameters (Use this and super) • • Create a toString() method for Cat class • Create a subclass Cow with instance variable: • String breed • • Create a no-argument constructor and parameterized constructor which has all four • parameters. (Use this and super) • • Create a toString() method for Cow class 	25 August 2024	
16	<p>Create a figure class with instance variable dim1, dim2 and method as area(). Create two derived class rectangle and triangle with constructors defined for initializing instance variable. Override area in both derived classes. WAP to demonstrate dynamic method dispatch (Run time polymorphism).</p>	25 August 2024	

17	Write a program to implement the concept of multiple inheritance through interface. Create a Figure and Draw interface. In Figure interface, define members PI, area(), perimeter(). In Draw interface, define members draw_shape(). Implement it in Circle class and Rectangle class.	25 August 2024	
----	--	----------------	--

18	<p>Write a Java program that creates a class hierarchy for employees of a company. The base class should be Employee, with subclasses Manager and Developer.</p> <ul style="list-style-type: none"> • Employee class should have private attributes such as name, address, salary, and job title. <p>Implement public methods:</p> <ul style="list-style-type: none"> • public double calculateBonus(): return 0.0 • public String generatePerformanceReport(): return "No performance report available." • Create getters • Create no arg and parameterized constructors • • Manager class has an attribute numberOfSubordinates. It has a method: mangeProject(), • and overridden methods: calculateBonus() and generatePerformanceReport(). • • public double calculateBonus(): provides a custom implementation for bonus • calculation for managers. In this case, it calculates the bonus as 15% of the • manager's salary. (Hint: Use getter for salary) • • public String generatePerformanceReport(): It returns a specific performance • report message for managers, including the manager's name and an "Excellent" • rating. (Hint: Use getter for name) • • public void manageProject(): This is a custom method specific to the "Manager" • class. It simulates the action of a manager managing a project by printing a message • to the console. (Hint: Use getter for name) • • Create no arg and parameterized constructors and getters • • Developer class has a private attribute programmingLanguage • public double calculateBonus(): provides a custom implementation for bonus • calculation for developers. In this case, it calculates the bonus as 10% of the • developer's salary. (Hint: Use getter for salary) • • public String generatePerformanceReport(): It returns a specific performance report • message for developers, including the developer's name and a "Good" rating. (Hint: • Use getter for name) • • public void writeCode(): This is a custom method specific to the "Developer" class. • It simulates the action of a developer writing code in their specialized programming 	25 August 2024	
----	---	----------------	--

19	Write a Java program to create an abstract class <code>Animal</code> with an abstract method called <code>sound()</code> .	25 August 2024	
20	Create subclasses <code>Lion</code> and <code>Cat</code> that extend the <code>Animal</code> class and implement the <code>sound()</code> method to make a specific sound for each animal.	25 August 2024	
21	<p>Create an Abstract Class “Shape” and 2 subclasses, <code>SemiCircle</code> and <code>Circle</code> that extend the <code>Shape</code> class and implement the respective methods to calculate the area and perimeter of each shape.</p> <ul style="list-style-type: none"> • <code>Shape</code> also defines 2 abstract methods <code>calculateArea()</code> and <code>calculateCircumference()</code>. Both return doubles representing the area and circumference of the shapes respectively. • For each shape you will have to create the necessary values need to calculate area and circumference and getters and setters for each of the values. You will also need to create constructors for each class, and the instructions for those will be included with each class. 	25 August 2024	
22	Write a program to search an element in an array using for each loop.	25 August 2024	
23	WAP to calculate sum and average of elements stored in one D array.	25 August 2024	

24	WAP to input elements in one D array using scanner class.	25 August 2024	
25	Write a Java program to find the maximum and minimum value of an array.	25 August 2024	
26	Write a Java program to sort an array of integers.	25 August 2024	

Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 1

Title: WAP to create a class called Circle. It contains:

- **private instance variable: radius (of the type double) with default value of 1.0.**
- **Two overloaded constructors - a default constructor with no argument, and a constructor**

which takes a double argument for radius.

- **Two public methods: getRadius(), calculateArea(), calculateCircumference() which return**

the radius, calculate and return area, and circumference respectively.

Hint: Use Math.PI for calculating area and circumference

Code:

```
package Assignment2;

class Circle {
    private double radius;

    public Circle() {
        this.radius = 1.0;
    }

    public Circle(double radius) {
        this.radius = radius;
    }

    public double getRadius() {
        return this.radius;
    }

    public double calculateArea() {
        return Math.PI * Math.pow(radius, 2);
    }

    public double calculateCircumference() {
        return 2 * Math.PI * radius;
    }
}
```

```

public class Question1 {

    public static void main(String[] args) {
        Circle circle1 = new Circle();
        System.out.println("Circle 1:");
        System.out.println("Radius: " + circle1.getRadius());
        System.out.println("Area: " + circle1.calculateArea());
        System.out.println("Circumference: " +
circle1.calculateCircumference());

        Circle circle2 = new Circle(2.5);
        System.out.println("\nCircle 2:");
        System.out.println("Radius: " + circle2.getRadius());
        System.out.println("Area: " + circle2.calculateArea());
        System.out.println("Circumference: " +
circle2.calculateCircumference());
    }
}

```

Output:

```

apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question1
Circle 1:
Radius: 1.0
Area: 3.141592653589793
Circumference: 6.283185307179586

Circle 2:
Radius: 2.5
Area: 19.634954084936208
Circumference: 15.707963267948966
apple@Jeevans-MacBook-Air JAVA %

```

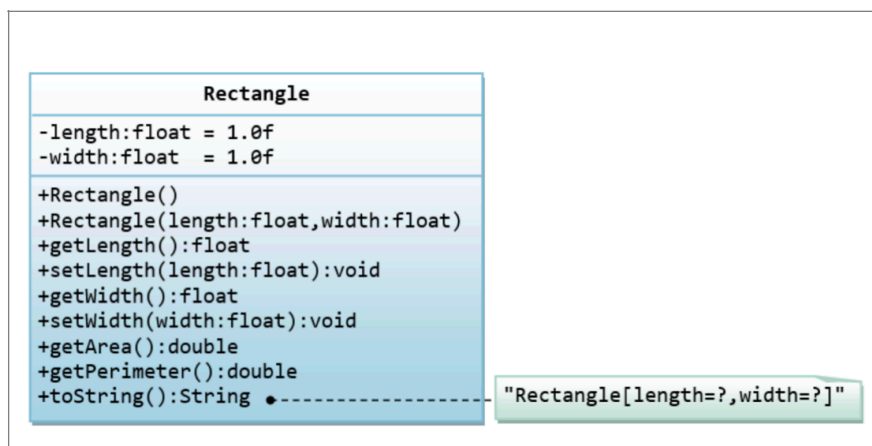
Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 2

Title: A class called Rectangle, which models a rectangle with a length and a width (in float), is

designed as shown in the following class diagram. Write the Rectangle class as per UML diagram.



Code:

```
package Assignment2;

class Rectangle {
    private float length = 1.0f;
    private float width = 1.0f;

    public Rectangle() {}

    public Rectangle(float length, float width) {
        this.length = length;
        this.width = width;
    }

    public float getLength() {
        return length;
    }

    public void setLength(float length) {
```



```

        this.length = length;
    }

    public float getWidth() {
        return width;
    }

    public void setWidth(float width) {
        this.width = width;
    }

    public double getArea() {
        return length * width;
    }

    public double getPerimeter() {
        return 2 * (length + width);
    }

    @Override
    public String toString() {
        return "Rectangle[length=" + length + ",width=" + width +
    "];
    }
}

public class Question2 {
    public static void main(String[] args) {
        Rectangle rect = new Rectangle(5.0f, 3.0f);
        System.out.println(rect.toString());
        System.out.println("Area: " + rect.getArea());
        System.out.println("Perimeter: " + rect.getPerimeter());
    }
}

```

Output:

```

apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question2
Rectangle[length=5.0,width=3.0]
Area: 15.0
Perimeter: 16.0
apple@Jeevans-MacBook-Air JAVA %

```

Name of Student: JEEVAN NAIDU

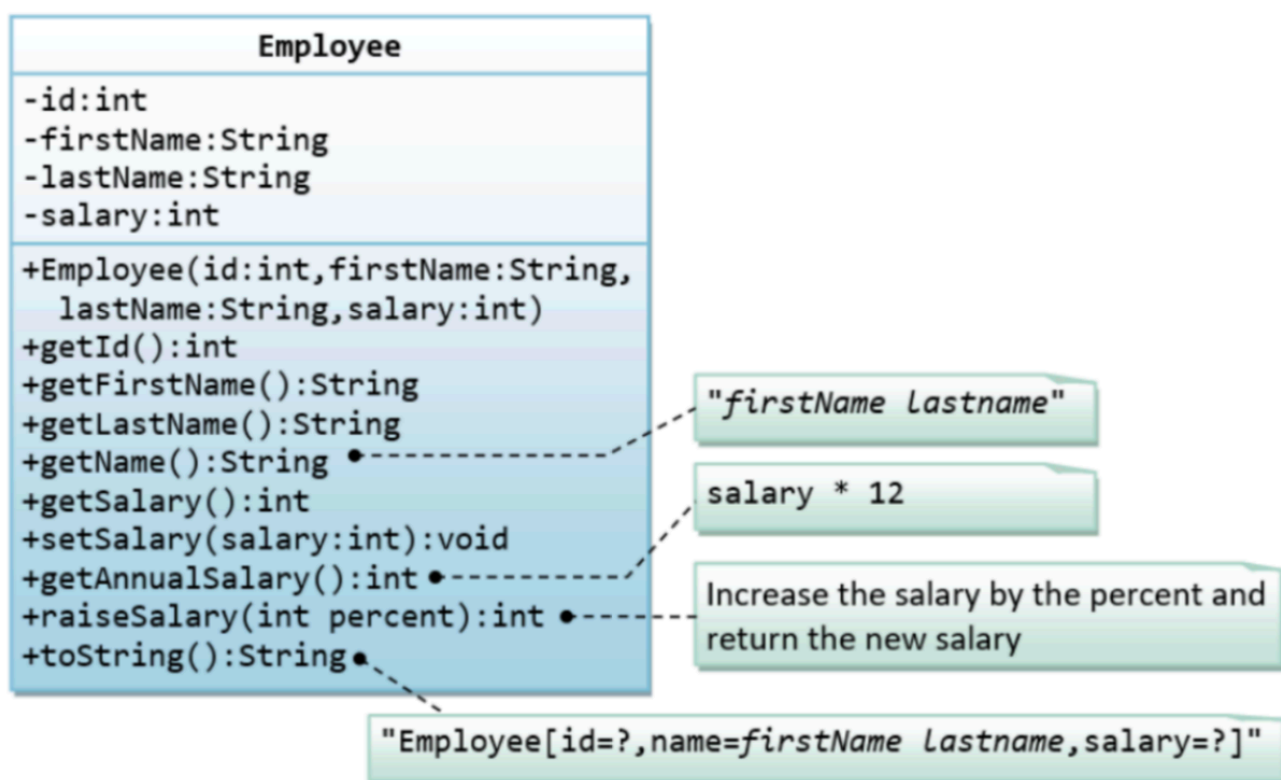
Roll Number: 15

Experiment No: 3

Title: A class called Employee, which models an employee with an ID, name and salary, is designed

as shown in the following class diagram. The method raiseSalary(percent) increases the salary by

the given percentage. Write the Employee class and its driver class.



Code:

```
package Assignment2;

class Employee {
    private int id;
```

```
private String firstName;  
private String lastName;  
private int salary;
```

```
public Employee(int id, String firstName, String lastName, int  
salary) {  
    this.id = id;  
    this.firstName = firstName;  
    this.lastName = lastName;  
    this.salary = salary;  
}
```

```
public int getId() {  
    return id;  
}
```

```
public String getFirstName() {  
    return firstName;  
}
```

```
public String getLastName() {  
    return lastName;  
}
```

```
public String getName() {  
    return firstName + " " + lastName;  
}
```

```
public int getSalary() {  
    return salary;  
}
```

```
public void setSalary(int salary) {  
    this.salary = salary;  
}
```

```
public int getAnnualSalary() {  
    return salary * 12;  
}
```

```
public int raiseSalary(int percent) {  
    this.salary += this.salary * percent / 100;  
    return this.salary;  
}
```

```
@Override  
public String toString() {  
    return "Employee[id=" + id + ",name=" + getName() +  
",salary=" + salary + "];"
```

```

    }
}

public class Question3 {
    public static void main(String[] args) {
        Employee emp = new Employee(1, "John", "Doe", 100000);
        System.out.println(emp.toString());
        System.out.println("Annual Salary: " +
emp.getAnnualSalary());
        System.out.println("New Salary after 10% raise: " +
emp.raiseSalary(35));
    }
}

```

Output:

```

apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question3
Employee[id=1,name=John Doe,salary=100000]
Annual Salary: 1200000
New Salary after 10% raise: 135000
apple@Jeevans-MacBook-Air JAVA %

```

Name of Student: JEEVAN NAIDU

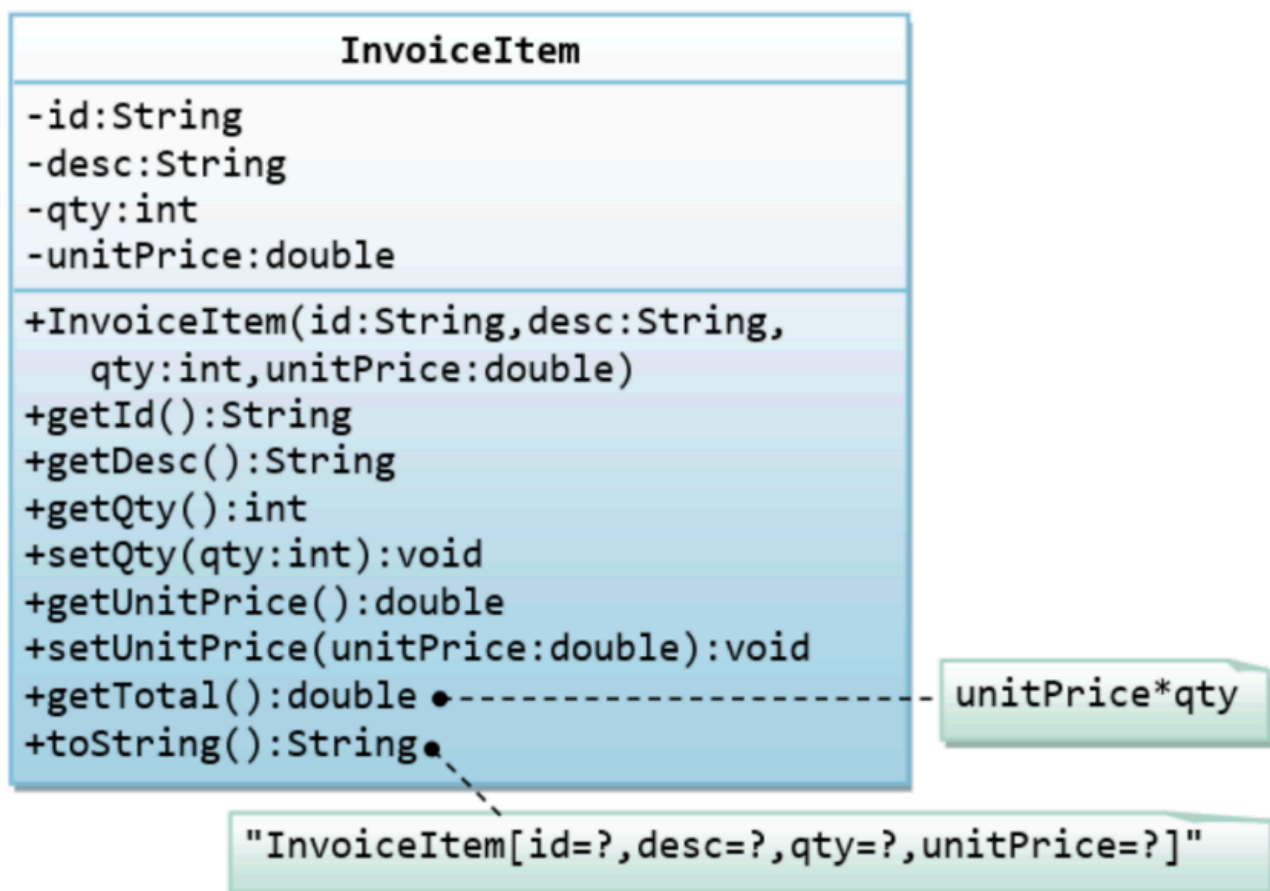
Roll Number: 15

Experiment No: 4

Title:A class called InvoiceItem, which models an item of an invoice, with ID, description, quantity

and unit price, is designed as shown in the following class diagram. It has a method getTotal which

calculates total value (total=quantity*unit price). Write the InvoiceItem class and it's driver class.



Code:

```

package Assignment2;

class InvoiceItem {
    private String id;
    private String desc;
    private int qty;
    private double unitPrice;

    public InvoiceItem(String id, String desc, int qty, double
unitPrice) {
        this.id = id;
        this.desc = desc;
        this.qty = qty;
        this.unitPrice = unitPrice;
    }

    public String getId() {
        return id;
    }

    public String getDesc() {
        return desc;
    }

    public int getQty() {
        return qty;
    }

    public void setQty(int qty) {
        this.qty = qty;
    }

    public double getUnitPrice() {
        return unitPrice;
    }

    public void setUnitPrice(double unitPrice) {
        this.unitPrice = unitPrice;
    }

    public double getTotal() {
        return qty * unitPrice;
    }

    @Override
    public String toString() {
        return "InvoiceItem[id=" + id + ",desc=" + desc + ",qty=" +
qty + ",unitPrice=" + unitPrice + "];"
    }
}

```

```

}
public class Question4 {
    public static void main(String[] args) {
        InvoiceItem item = new InvoiceItem("1", "Laptop", 2,
850.50);
        System.out.println(item.toString());
        System.out.println("Total: " + item.getTotal());
    }
}

```

Output:

```

apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question4
InvoiceItem[id=1,desc=Laptop,qty=2,unitPrice=850.5]
Total: 1701.0
apple@Jeevans-MacBook-Air JAVA %

```

'Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 5

Title:A class called Author (as shown in the class diagram) is designed to model a book's author. It

contains:

- **Three private instance variables:** name (String), email (String), and gender (char of

either 'm' or 'f');

- **One constructor to initialize the name, email and gender with the given values;**
public Author

(String name, String email, char gender) {.....}

(There is no default constructor for Author, as there are no defaults for name, email and

gender.)

- **public getters/setters:** getName(), getEmail(), setEmail(), and getGender();

(There are no setters for name and gender, as these attributes cannot be changed.)

- **A toString() method that returns "Author[name=?,email=?,gender=?]", e.g.,
"Author[name=Abc ,email=Abc@gmail.com, gender=m]".**

Code:

```
package Assignment2;
```

```
public class Question5 {  
    private String name;  
    private String email;  
    private char gender;
```

```
    public Question5(String name, String email, char gender) {  
        this.name = name;
```



```

        this.email = email;
        this.gender = gender;
    }

```

```

    public String getName() {
        return name;
    }

```

```

    public String getEmail() {
        return email;
    }

```

```

    public void setEmail(String email) {
        this.email = email;
    }

```

```

    public char getGender() {
        return gender;
    }

```

```

    @Override
    public String toString() {
        return "Author[name=" + name + ",email=" + email +
",gender=" + gender + "];"
    }

```

```

    public static void main(String[] args) {
        Question5 author = new Question5("john doe",
"joedoe@gmail.com", 'm');
        System.out.println(author.toString());
        author.setEmail("HelloWorld@example.com");
        System.out.println("Updated Email: " + author.getEmail());
    }
}

```

Output:

```

apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question5
Author[name=john doe,email=joedoe@gmail.com,gender=m]
Updated Email: HelloWorld@example.com
apple@Jeevans-MacBook-Air JAVA %

```

Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 6

Title: WAP to create a class time having default constructor, parameterized constructor whose

specifications are as follows:

- **Instance variable: hr, min, sec**
- **Constructors:**
 - **default (with no parameters passed; should initialize the represented time to 12:0:0)**
 - **a constructor with three parameters: hours, minutes, and seconds.**
 - **a constructor with one parameter: the value of time in seconds since midnight (it should be converted into the time value in hours, minutes, and seconds)**
- **Instance methods:**
 - **setClock() with one parameter seconds since midnight (to be converted into the time value in hours, minutes, and seconds as above).**
 - **tick() with no parameters that increments the time stored in a Clock object by one second.**
 - **tickDown() which decrements the time stored in a Clock object by one second.**
 - **displaytime() displays the time in the format hr: min:sec e.g: 05:45:23**

Code:

```
package Assignment2;  
  
public class Question6 {  
    private int hr, min, sec;
```

```
public Question6() {  
    this.hr = 12;  
    this.min = 0;  
    this.sec = 0;  
}
```

```
public Question6(int hr, int min, int sec) {  
    this.hr = hr;  
    this.min = min;  
    this.sec = sec;  
}
```

```
public Question6(int secondsSinceMidnight) {  
    setClock(secondsSinceMidnight);  
}
```

```
public void setClock(int secondsSinceMidnight) {  
    this.hr = (secondsSinceMidnight / 3600) % 24;  
    this.min = (secondsSinceMidnight / 60) % 60;  
    this.sec = secondsSinceMidnight % 60;  
}
```

```
public void tick() {  
    setClock((hr * 3600 + min * 60 + sec + 1) % 86400);  
}
```

```
public void tickDown() {  
    setClock((hr * 3600 + min * 60 + sec - 1 + 86400) % 86400);  
}
```

```
public void displayTime() {  
    System.out.printf("%02d:%02d:%02d\n", hr, min, sec);  
}
```

```
public static void main(String[] args) {  
    Question6 time1 = new Question6();  
    time1.displayTime();
```

```
    Question6 time2 = new Question6(5, 45, 23);  
    time2.displayTime();
```

```
    Question6 time3 = new Question6(55555);  
    time3.displayTime();
```

```
    time3.tick();  
    time3.displayTime();
```

```
    time3.tickDown();  
    time3.displayTime();
```

```
}  
}
```

Output:

```
apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C  
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question6  
12:00:00  
05:45:23  
15:25:55  
15:25:56  
15:25:55  
apple@Jeevans-MacBook-Air JAVA %
```

Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 7

Title: Q7. Write a Java class Complex for dealing with complex number. Your class must have the

following features:

- **Instance variables :**

- o **real for the real part of type double**

- o **imag for imaginary part of type double.**

- **Constructor:**

- o **public Complex (): A default constructor, it should initialize the number to 0, 0)**

- o **public Complex (double real, double imag: A constructor with parameters, it creates the complex object by setting the two fields to the passed values.**

- **Instance methods:**

- o **public Complex add (Complex n): This method will find the sum of the current complex number and the passed complex number. The methods returns a new Complex number which is the sum of the two.**

- o **public Complex subtract (Complex n): This method will find the difference of the current complex number and the passed complex number. The methods returns a new Complex number which is the difference of the two.**

- o **public void setReal(double real):** Used to set the real part of this complex number.
- o **public void setImag(double image):** Used to set the imaginary part of this complex number.
- o **public double getReal():** This method returns the real part of the complex number
- o **public double getImag():** This method returns the imaginary part of the complex number
- o **public String toString():** This method allows the complex number to be easily printed out to the screen

Write a separate class ComplexDemo with a main() method and test the Complex class methods.

Code:

```
package Assignment2.Question7;
```

```
public class Complex {
    private double real;
    private double imag;
```

```
    public Complex() {
        this.real = 0;
        this.imag = 0;
    }
```

```
    public Complex(double real, double imag) {
        this.real = real;
        this.imag = imag;
    }
```

```
    public Complex add(Complex n) {
        return new Complex(this.real + n.real, this.imag + n.imag);
    }
```

```
    public Complex subtract(Complex n) {
        return new Complex(this.real - n.real, this.imag - n.imag);
    }
```

```
public void setReal(double real) {
    this.real = real;
}
```

```
public void setImag(double imag) {
    this.imag = imag;
}
```

```
public double getReal() {
    return this.real;
}
```

```
public double getImag() {
    return this.imag;
}
```

```
@Override
public String toString() {
    return this.real + " + " + this.imag + "i";
}
}
```

```
package Assignment2.Question7;
```

```
public class ComplexDemo {
    public static void main(String[] args) {
        Complex num1 = new Complex(3.5, 2.5);
        Complex num2 = new Complex(1.5, 4.5);

        System.out.println("Number 1: " + num1);
        System.out.println("Number 2: " + num2);

        Complex sum = num1.add(num2);
        System.out.println("Sum: " + sum);

        Complex difference = num1.subtract(num2);
        System.out.println("Difference: " + difference);
    }
}
```

Output:

```
apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question7.ComplexDemo
Number 1: 3.5 + 2.5i
Number 2: 1.5 + 4.5i
Sum: 5.0 + 7.0i
Difference: 2.0 + -2.0i
apple@Jeevans-MacBook-Air JAVA % █
```

Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 8

Title: Create a Geometry class and overload calculateArea method for square, circle, and rectangle.

Code:

```
package Assignment2;

public class Question8 {
    public double calculateArea(double side) {
        return side * side;
    }
    public double calculateArea(double radius, boolean isCircle) {
        if (isCircle) {
            return Math.PI * radius * radius;
        } else {
            throw new IllegalArgumentException("For a circle, use the radius and set isCircle to true.");
        }
    }
    public double calculateArea(double length, double width) {
        return length * width;
    }

    public static void main(String[] args) {
        Question8 geometry = new Question8();

        double squareArea = geometry.calculateArea(2);
        System.out.println("Area of the square: " + squareArea);

        double circleArea = geometry.calculateArea(5, true);
        System.out.println("Area of the circle: " + circleArea);

        double rectangleArea = geometry.calculateArea(2, 4);
        System.out.println("Area of the rectangle: " + rectangleArea);
    }
}
```


Output:

```
apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question8
Area of the square: 4.0
Area of the circle: 78.53981633974483
Area of the rectangle: 8.0
apple@Jeevans-MacBook-Air JAVA %
```

Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 9

Title:

WAP to implement Box class (Hint: Refer Java PPT). Inherit Box class in BoxWt whose

a. instance variable is weight and

b. method is print_BoxWt()

c. constructors: default, parameterized and BoxWt(BoxWt ob)

Use super() to invoke superclass constructors.

Code:

```
package Assignment2.Question9;
```

```
public class Box {  
    double length;  
    double width;  
    double height;
```

```
    Box() {  
        this.length = 1.0;  
        this.width = 1.0;  
        this.height = 1.0;  
    }
```

```
    Box(double length, double width, double height) {  
        this.length = length;  
        this.width = width;  
        this.height = height;  
    }
```

```
    Box(Box ob) {  
        this.length = ob.length;
```

```

        this.width = ob.width;
        this.height = ob.height;
    }

    double volume() {
        return length * width * height;
    }
}

package Assignment2.Question9;

public class BoxWt extends Box {
    double weight;

    BoxWt() {
        super();
        this.weight = 1.0;
    }

    BoxWt(double length, double width, double height, double
weight) {
        super(length, width, height);
        this.weight = weight;
    }

    BoxWt(BoxWt ob) {
        super(ob);
        this.weight = ob.weight;
    }

    void print_BoxWt() {
        System.out.println("Box dimensions: " + length + "x" +
width + "x" + height);
        System.out.println("Weight: " + weight);
    }

    public static void main(String[] args) {
        BoxWt box1 = new BoxWt();
        box1.print_BoxWt();

        BoxWt box2 = new BoxWt(2.5, 3.0, 4.5, 5.0);
        box2.print_BoxWt();

        BoxWt box3 = new BoxWt(box2);
        box3.print_BoxWt();
    }
}

```

Output:

```
apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question9.BoxWt
Box dimensions: 1.0x1.0x1.0
Weight: 1.0
Box dimensions: 2.5x3.0x4.5
Weight: 5.0
Box dimensions: 2.5x3.0x4.5
Weight: 5.0
apple@Jeevans-MacBook-Air JAVA %
```

Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 10

Title:

WAP the demonstrate that super class variable can point to object of subclass.

Code:

```
package Assignment2;

class Animal {
    void sound() {
        System.out.println("Animal makes a sound");
    }
}

class Dog extends Animal {
    @Override
    void sound() {
        System.out.println("Dog barks");
    }
}

public class Question10 {
    public static void main(String[] args) {
        Animal myAnimal = new Animal();
        myAnimal.sound();

        Animal myDog = new Dog();
        myDog.sound();
    }
}
```

Output:

```
apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question10
Animal makes a sound
Dog barks
apple@Jeevans-MacBook-Air JAVA % █
```

Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 11

Title:

WAP to demonstrate multilevel inheritance by creating a BoxColor class and inheriting BoxWt class. BoxColor class has an instance variable color of String type.

Code:

```
package Assignment2;

class Box {
    double width;
    double height;
    double depth;

    Box() {
        width = 0;
        height = 0;
        depth = 0;
    }

    Box(double w, double h, double d) {
        width = w;
        height = h;
        depth = d;
    }

    double volume() {
        return width * height * depth;
    }
}

class BoxWt extends Box {
    double weight;

    BoxWt() {
        super();
        weight = 0;
    }
}
```

```

    BoxWt(double w, double h, double d, double wg) {
        super(w, h, d);
        weight = wg;
    }
}

class BoxColor extends BoxWt {
    String color;

    BoxColor() {
        super();
        color = "unknown";
    }

    BoxColor(double w, double h, double d, double wg, String c) {
        super(w, h, d, wg);
        color = c;
    }
}

public class Question11 {
    public static void main(String[] args) {
        BoxColor box = new BoxColor(5, 4, 3, 2.5, "Red");
        System.out.println("Volume: " + box.volume());
        System.out.println("Weight: " + box.weight);
        System.out.println("Color: " + box.color);
    }
}

```

Output:

```

apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question11
Volume: 60.0
Weight: 2.5
Color: Red
apple@Jeevans-MacBook-Air JAVA %

```

Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 12

Title:

WAP to demonstrate the sequence of execution of constructors in multilevel inheritance.

Code:

```
package Assignment2;

class A {
    A() {
        System.out.println("Constructor of class A");
    }
}

class B extends A {
    B() {
        System.out.println("Constructor of class B");
    }
}

class C extends B {
    C() {
        System.out.println("Constructor of class C");
    }
}

public class Question12 {
    public static void main(String[] args) {
        C obj = new C();
    }
}
```

Output:

```
apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question12
Constructor of class A
Constructor of class B
Constructor of class C
apple@Jeevans-MacBook-Air JAVA %
```


Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 13

Title: WAP to demonstrate the concept of method overriding.

Code:

```
package Assignment2;

class Parent {
    void display() {
        System.out.println("Display from Parent");
    }
}

class Child extends Parent {
    @Override
    void display() {
        System.out.println("Display from Child");
    }
}

public class Question13 {
    public static void main(String[] args) {
        Parent obj = new Child();
        obj.display();
    }
}
```

Output:

```
apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question13
Display from Child
apple@Jeevans-MacBook-Air JAVA %
```

Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 14

Title: WAP to demonstrate the use of super keyword to call overridden methods and instance variables.

Code:

```
package Assignment2.Question14;

class Child extends Parent {
    String name = "Child";

    void display() {
        System.out.println("Display from Child");
    }

    void show() {
        System.out.println("Name in Child: " + name);
        System.out.println("Name in Parent: " + super.name);
        super.display();
        display();
    }
}

package Assignment2.Question14;

class Parent {
    String name = "Parent";

    void display() {
        System.out.println("Display from Parent");
    }
}

package Assignment2.Question14;

public class Question14 {
    public static void main(String[] args) {
        Child obj = new Child();
        obj.show();
    }
}
```

}

Output:

```
apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question14.Question14
Name in Child: Child
Name in Parent: Parent
Display from Parent
Display from Child
apple@Jeevans-MacBook-Air JAVA %
```

Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 15

Title:

Create a class Animal with

- **instance variables:**
 - **boolean vegetarian**
 - **String food**
 - **int numOfLegs**
 - **Create a no-argument constructor and parameterized constructor. (Use “this”)**
 - **Create getters and setters**
 - **Create a toString() method for animal class**
 - **Create a subclass Cat with instance variable:**
 - **String color**

 - **Create a no-argument constructor and parameterized constructor which has all four**
- parameters (Use this and super)**
- **Create a toString() method for Cat class**
 - **Create a subclass Cow with instance variable:**
 - **String breed**
 - **Create a no-argument constructor and parameterized constructor which has all four**
- parameters. (Use this and super)**
- **Create a toString() method for Cow class**

Code:

```
package Assignment2.Question15;

public class Animal {
    private boolean vegetarian;
    private String food;
    private int numOfLegs;

    public Animal() {
        this.vegetarian = false;
        this.food = "Unknown";
        this.numOfLegs = 0;
    }

    public Animal(boolean vegetarian, String food, int numOfLegs) {
        this.vegetarian = vegetarian;
        this.food = food;
        this.numOfLegs = numOfLegs;
    }

    public boolean isVegetarian() {
        return vegetarian;
    }

    public void setVegetarian(boolean vegetarian) {
        this.vegetarian = vegetarian;
    }

    public String getFood() {
        return food;
    }

    public void setFood(String food) {
        this.food = food;
    }

    public int getNumOfLegs() {
        return numOfLegs;
    }

    public void setNumOfLegs(int numOfLegs) {
        this.numOfLegs = numOfLegs;
    }

    @Override
    public String toString() {
        return "Animal [vegetarian=" + vegetarian + ", food=" +
        food + ", numOfLegs=" + numOfLegs + "]\n";
    }
}
```

```

}
package Assignment2.Question15;

public class Cat extends Animal {
    private String color;

    public Cat() {
        super();
        this.color = "Unknown";
    }

    public Cat(boolean vegetarian, String food, int numOfLegs,
String color) {
        super(vegetarian, food, numOfLegs);
        this.color = color;
    }

    public String getColor() {
        return color;
    }

    public void setColor(String color) {
        this.color = color;
    }

    @Override
    public String toString() {
        return super.toString() + ", Cat [color=" + color + "];
    }
}

```

```

package Assignment2.Question15;

```

```

public class Cow extends Animal {
    private String breed;

```

```

    public Cow() {
        super();
        this.breed = "Unknown";
    }

```

```

    public Cow(boolean vegetarian, String food, int numOfLegs,
String breed) {
        super(vegetarian, food, numOfLegs);
        this.breed = breed;
    }

```

```

    public String getBreed() {
        return breed;
    }

```

```

    public void setBreed(String breed) {
        this.breed = breed;
    }

    @Override
    public String toString() {
        return super.toString() + ", Cow [breed=" + breed + "]";
    }
}

package Assignment2.Question15;

public class Question15 {
    public static void main(String[] args) {
        Cat cat = new Cat(true, "Fish", 4, "Black");
        Cow cow = new Cow(false, "Grass", 4, "Holstein");

        System.out.println(cat);
        System.out.println(cow);
    }
}

```

Output:

```

apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question15.Question15
Animal [vegetarian=true, food=Fish, numOfLegs=4], Cat [color=Black]
Animal [vegetarian=false, food=Grass, numOfLegs=4], Cow [breed=Holstein]
apple@Jeevans-MacBook-Air JAVA %

```

Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 16

Title:

**Create a figure class with instance variable dim1, dim2 and method as area().
Create two**

**derived class rectangle and triangle with constructors defined for initializing
instance variable.**

**Override area in both derived classes. WAP to demonstrate dynamic method
dispatch (Run time
polymorphism).**

Code:

```
package Assignment2;

public class Question16 {

    static class Figure {
        protected double dim1;
        protected double dim2;

        public Figure(double dim1, double dim2) {
            this.dim1 = dim1;
            this.dim2 = dim2;
        }

        public double area() {
            return 0;
        }
    }

    static class Rectangle extends Figure {

        public Rectangle(double length, double width) {
            super(length, width);
        }
    }
}
```



```

        @Override
        public double area() {
            return dim1 * dim2;
        }
    }
}

```

```

static class Triangle extends Figure {

```

```

    // Constructor
    public Triangle(double base, double height) {
        super(base, height);
    }

```

```

        @Override
        public double area() {
            return 0.5 * dim1 * dim2;
        }
    }
}

```

```

public static void main(String[] args) {

```

```

    Figure fig;

```

```

    // Rectangle object
    fig = new Rectangle(5, 3);
    System.out.println("Area of Rectangle: " + fig.area());

```

```

    // Triangle object
    fig = new Triangle(4, 6);
    System.out.println("Area of Triangle: " + fig.area());

```

```

}

```

Output:

```

apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question16
Area of Rectangle: 15.0
Area of Triangle: 12.0
apple@Jeevans-MacBook-Air JAVA % █

```

Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 17

Title:

Write a program to implement the concept of multiple inheritance through interface. Create a

Figure and Draw interface. In Figure interface, define members PI, area(), perimeter(). In Draw

interface, define members draw_shape(). Implement it in Circle class and Rectangle class.

Code:

```
package Assignment2.Question17;

public interface Figure {
    double PI = 3.14159;

    double area();
    double perimeter();
}

package Assignment2.Question17;

public interface Draw {
    void draw_shape();
}

package Assignment2.Question17;

public class Circle implements Figure, Draw {
    private double radius;

    public Circle(double radius) {
        this.radius = radius;
    }

    @Override
    public double area() {
        return PI * radius * radius;
    }
}
```

```
@Override
public double perimeter() {
    return 2 * PI * radius;
}
```

```
@Override
public void draw_shape() {
    System.out.println("Drawing a Circle");
}
}
```

```
package Assignment2.Question17;
```

```
public class Rectangle implements Figure, Draw {
    private double length;
    private double width;
```

```
    public Rectangle(double length, double width) {
        this.length = length;
        this.width = width;
    }
```

```
@Override
public double area() {
    return length * width;
}
```

```
@Override
public double perimeter() {
    return 2 * (length + width);
}
```

```
@Override
public void draw_shape() {
    System.out.println("Drawing a Rectangle");
}
}
```

```
package Assignment2.Question17;
```

```
public class Question17 {
    public static void main(String[] args) {
        Figure circleFigure = new Circle(5);
        Draw circleDraw = (Draw) circleFigure;

        System.out.println("Circle:");
        System.out.println("Area: " + circleFigure.area());
        System.out.println("Perimeter: " +
circleFigure.perimeter());
    }
}
```

```

        circleDraw.draw_shape();

        Figure rectangleFigure = new Rectangle(4, 6);
        Draw rectangleDraw = (Draw) rectangleFigure;

        System.out.println("\nRectangle:");
        System.out.println("Area: " + rectangleFigure.area());
        System.out.println("Perimeter: " +
rectangleFigure.perimeter());
        rectangleDraw.draw_shape();
    }
}

```

Output:

```

apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question17.Question17
Circle:
Area: 78.53975
Perimeter: 31.4159
Drawing a Circle

Rectangle:
Area: 24.0
Perimeter: 20.0
Drawing a Rectangle
apple@Jeevans-MacBook-Air JAVA %

```

Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 18

Title:

Q18. Write a Java program that creates a class hierarchy for employees of a company. The base

class should be Employee, with subclasses Manager and Developer.

- **Employee class should have private attributes such as name, address, salary, and job title.**

Implement public methods:

- **public double calculateBonus(): return 0.0**

- **public String generatePerformanceReport(): return "No performance report available."**

- **Create getters**

- **Create no arg and parameterized constructors**

- **Manager class has an attribute numberOfSubordinates. It has a method: manageProject(),**

and overridden methods: calculateBonus() and generatePerformanceReport().

- **public double calculateBonus(): provides a custom implementation for bonus calculation for managers. In this case, it calculates the bonus as 15% of the manager's salary. (Hint: Use getter for salary)**

- **public String generatePerformanceReport(): It returns a specific performance report message for managers, including the manager's name and an "Excellent" rating. (Hint: Use getter for name)**

- **public void manageProject(): This is a custom method specific to the "Manager"**

class. It simulates the action of a manager managing a project by printing a message

to the console. (Hint: Use getter for name)

- Create no arg and parameterized constructors and getters
- Developer class has a private attribute `programmingLanguage`
- `public double calculateBonus():` provides a custom implementation for bonus calculation for developers. In this case, it calculates the bonus as 10% of the developer's salary. (Hint: Use getter for salary)
- `public String generatePerformanceReport():` It returns a specific performance report

message for developers, including the developer's name and a "Good" rating. (Hint:

Use getter for name)

- `public void writeCode():` This is a custom method specific to the "Developer" class.

It simulates the action of a developer writing code in their specialized programming

language by printing a message to the console. (Hint: Use getter for name)

- Create no arg and parameterized constructors and getters

Code:

```
package Assignment2.Question18;
```

```
public class Employee {  
    private String name;  
    private String address;  
    private double salary;  
    private String jobTitle;
```

```
    public Employee() {  
        // No-arg constructor  
    }
```

```

    public Employee(String name, String address, double salary,
String jobTitle) {
        this.name = name;
        this.address = address;
        this.salary = salary;
        this.jobTitle = jobTitle;
    }

    public double calculateBonus() {
        return 0.0;
    }

    public String generatePerformanceReport() {
        return "No performance report available.";
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getAddress() {
        return address;
    }

    public void setAddress(String address) {
        this.address = address;
    }

    public double getSalary() {
        return salary;
    }

    public void setSalary(double salary) {
        this.salary = salary;
    }

    public String getJobTitle() {
        return jobTitle;
    }

    public void setJobTitle(String jobTitle) {
        this.jobTitle = jobTitle;
    }
}

package Assignment2.Question18;

```

```

public class Manager extends Employee {
    private int numberOfSubordinates;

    public Manager() {

    }

    public Manager(String name, String address, double salary,
String jobTitle, int numberOfSubordinates) {
        super(name, address, salary, jobTitle);
        this.numberOfSubordinates = numberOfSubordinates;
    }

    @Override
    public double calculateBonus() {
        return 0.15 * getSalary();
    }

    @Override
    public String generatePerformanceReport() {
        return "Manager " + getName() + " has an Excellent
performance rating.";
    }

    public void manageProject() {
        System.out.println("Manager " + getName() + " is managing a
project.");
    }

    public int getNumberOfSubordinates() {
        return numberOfSubordinates;
    }

    public void setNumberOfSubordinates(int numberOfSubordinates) {
        this.numberOfSubordinates = numberOfSubordinates;
    }
}

package Assignment2.Question18;

public class Developer extends Employee {
    private String programmingLanguage;

    public Developer() {

    }
}

```



```

    public Developer(String name, String address, double salary,
String jobTitle, String programmingLanguage) {
        super(name, address, salary, jobTitle);
        this.programmingLanguage = programmingLanguage;
    }

```

```

@Override
public double calculateBonus() {
    return 0.10 * getSalary();
}

```

```

@Override
public String generatePerformanceReport() {
    return "Developer " + getName() + " has a Good performance
rating.";
}

```

```

public void writeCode() {
    System.out.println("Developer " + getName() + " is writing
code in " + programmingLanguage + ".");
}

```

```

public String getProgrammingLanguage() {
    return programmingLanguage;
}

```

```

public void setProgrammingLanguage(String programmingLanguage)
{
    this.programmingLanguage = programmingLanguage;
}
}

```

```

package Assignment2.Question18;

```

```

public class Question18 {
    public static void main(String[] args) {
        Manager manager = new Manager("Alice", "123 Manager St",
80000, "Project Manager", 5);
        Developer developer = new Developer("Bob", "456 Developer
Rd", 70000, "Software Developer", "Java");

```

```

        System.out.println("Manager Details:");
        System.out.println("Name: " + manager.getName());
        System.out.println("Address: " + manager.getAddress());
        System.out.println("Salary: " + manager.getSalary());
        System.out.println("Job Title: " + manager.getJobTitle());
        System.out.println("Number of Subordinates: " +
manager.getNumberOfSubordinates());
        System.out.println("Bonus: " + manager.calculateBonus());

```

```

        System.out.println("Performance Report: " +
manager.generatePerformanceReport());
manager.manageProject();

        System.out.println("\nDeveloper Details:");
        System.out.println("Name: " + developer.getName());
        System.out.println("Address: " + developer.getAddress());
        System.out.println("Salary: " + developer.getSalary());
        System.out.println("Job Title: " +
developer.getJobTitle());
        System.out.println("Programming Language: " +
developer.getProgrammingLanguage());
        System.out.println("Bonus: " + developer.calculateBonus());
        System.out.println("Performance Report: " +
developer.generatePerformanceReport());
        developer.writeCode();
    }
}

```

Output:

```

apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question18.Question18
Manager Details:
Name: Alice
Address: 123 Manager St
Salary: 80000.0
Job Title: Project Manager
Number of Subordinates: 5
Bonus: 12000.0
Performance Report: Manager Alice has an Excellent performance rating.
Manager Alice is managing a project.

Developer Details:
Name: Bob
Address: 456 Developer Rd
Salary: 70000.0
Job Title: Software Developer
Programming Language: Java
Bonus: 7000.0
Performance Report: Developer Bob has a Good performance rating.
Developer Bob is writing code in Java.
apple@Jeevans-MacBook-Air JAVA %

```

Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 19

Title:

Write a Java program to create an abstract class Animal with an abstract method called sound().

Code:

```
package Assignment2.Question19;

abstract class Animal {
    abstract void sound();

    void sleep() {
        System.out.println("This animal is sleeping.");
    }
}

package Assignment2.Question19;

class Cat extends Animal {
    @Override
    void sound() {
        System.out.println("The cat meows.");
    }
}

package Assignment2.Question19;

class Dog extends Animal {
    @Override
    void sound() {
        System.out.println("The dog barks.");
    }
}

package Assignment2.Question19;

public class Question19 {
    public static void main(String[] args) {
```

```
Animal myDog = new Dog();
Animal myCat = new Cat();
```

```
myDog.sound();
myCat.sound();
```

```
myDog.sleep();
myCat.sleep();
}
```

```
}
```

Output:

```
apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question19.Question19
The dog barks.
The cat meows.
This animal is sleeping.
This animal is sleeping.
apple@Jeevans-MacBook-Air JAVA %
```

Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 20

Title:

Create subclasses Lion and Cat that extend the Animal class and implement the sound()

method to make a specific sound for each animal.

Code:

```
package Assignment2.Question20;

abstract class Animal {
    abstract void sound();

    void sleep() {
        System.out.println("This animal is sleeping.");
    }
}

package Assignment2.Question20;

class Cat extends Animal {
    @Override
    void sound() {
        System.out.println("The cat meows.");
    }
}

package Assignment2.Question20;

class Lion extends Animal {
    @Override
    void sound() {
        System.out.println("The lion roars.");
    }
}

package Assignment2.Question20;

public class Question20 {
    public static void main(String[] args) {
        Animal myLion = new Lion();
    }
}
```

```
Animal myCat = new Cat();
```

```
myLion.sound();  
myCat.sound();
```

```
myLion.sleep();  
myCat.sleep();
```

```
}
```

```
}
```

Output:

```
apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question20.Question20
The lion roars.
The cat meows.
This animal is sleeping.
This animal is sleeping.
apple@Jeevans-MacBook-Air JAVA %
```

Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 21

Title:

Create an Abstract Class “Shape” and 2 subclasses, SemiCircle and Circle that extend the

Shape class and implement the respective methods to calculate the area and perimeter of each

shape.

- **Shape also defines 2 abstract methods calculateArea() and calculateCircumference(). Both**

return doubles representing the area and circumference of the shapes respectively.

- **For each shape you will have to create the necessary values need to calculate area and**

circumference and getters and setters for each of the values. You will also need to create

constructors for each class, and the instructions for those will be included with each class.

Code:

```
package Assignment2.Question21;

abstract class Shape {
    abstract double calculateArea();
    abstract double calculateCircumference();
}

package Assignment2.Question21;

class SemiCircle extends Shape {
    private double radius;
```

```
public SemiCircle() {  
    this.radius = 1.0;  
}
```

```
public SemiCircle(double radius) {  
    this.radius = radius;  
}
```

```
public double getRadius() {  
    return radius;  
}
```

```
public void setRadius(double radius) {  
    this.radius = radius;  
}
```

```
@Override  
double calculateArea() {  
    return 0.5 * Math.PI * radius * radius;  
}
```

```
@Override  
double calculateCircumference() {  
    return Math.PI * radius + 2 * radius;  
}  
}
```

```
package Assignment2.Question21;
```

```
class Circle extends Shape {  
    private double radius;
```

```
public Circle() {  
    this.radius = 1.0;  
}
```

```
public Circle(double radius) {  
    this.radius = radius;  
}
```

```
public double getRadius() {  
    return radius;  
}
```

```
public void setRadius(double radius) {  
    this.radius = radius;  
}
```



```

@Override
double calculateArea() {
    return Math.PI * radius * radius;
}

@Override
double calculateCircumference() {
    return 2 * Math.PI * radius;
}
}

package Assignment2.Question21;

public class Question21 {
    public static void main(String[] args) {
        Circle circle = new Circle(5);
        SemiCircle semiCircle = new SemiCircle(5);

        System.out.println("Circle:");
        System.out.println("Radius: " + circle.getRadius());
        System.out.println("Area: " + circle.calculateArea());
        System.out.println("Circumference: " +
circle.calculateCircumference());

        System.out.println("\nSemiCircle:");
        System.out.println("Radius: " + semiCircle.getRadius());
        System.out.println("Area: " + semiCircle.calculateArea());
        System.out.println("Circumference: " +
semiCircle.calculateCircumference());
    }
}

```

Output:

```

apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question21.Question21
Circle:
Radius: 5.0
Area: 78.53981633974483
Circumference: 31.41592653589793

SemiCircle:
Radius: 5.0
Area: 39.269908169872416
Circumference: 25.707963267948966
apple@Jeevans-MacBook-Air JAVA %

```

Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 22

Title:

Write a program to search an element in an array using for each loop.

Code:

```
package Assignment2;

public class Question22 {
    public static void main(String[] args) {
        int[] numbers = {10, 20, 30, 40, 50};
        int searchElement = 30;

        boolean found = searchInArray(numbers, searchElement);

        if (found) {
            System.out.println("Element " + searchElement + " is
present in the array.");
        } else {
            System.out.println("Element " + searchElement + " is
not present in the array.");
        }
    }

    public static boolean searchInArray(int[] array, int element) {
        for (int value : array) {
            if (value == element) {
                return true;
            }
        }
        return false;
    }
}
```

Output:

```
apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question22
Element 30 is present in the array.
apple@Jeevans-MacBook-Air JAVA %
```

Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 23

Title:

WAP to calculate sum and average of elements stored in one D array.

Code:

```
package Assignment2;

public class Question23 {
    public static void main(String[] args) {
        int[] numbers = {10, 20, 30, 40, 50};

        int sum = 0;
        for (int num : numbers) {
            sum += num;
        }

        double average = (double) sum / numbers.length;

        System.out.println("Sum: " + sum);
        System.out.println("Average: " + average);
    }
}
```

Output:

```
apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question23
Sum: 150
Average: 30.0
apple@Jeevans-MacBook-Air JAVA %
```

Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 24

Title:

WAP to input elements in one D array using scanner class.

Code:

```
package Assignment2;

import java.util.Scanner;

public class Question24 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the number of elements: ");
        int size = scanner.nextInt();

        int[] numbers = new int[size];

        System.out.println("Enter " + size + " elements:");

        for (int i = 0; i < size; i++) {
            System.out.print("Element " + (i + 1) + ": ");
            numbers[i] = scanner.nextInt();
        }

        System.out.println("You entered:");
        for (int num : numbers) {
            System.out.println(num);
        }

        scanner.close();
    }
}
```

Output:

```
apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question24
Enter the number of elements: 5
Enter 5 elements:
Element 1: 1
Element 2: 2
Element 3: 3
Element 4: 5
Element 5: 4
You entered:
1
2
3
5
4
apple@Jeevans-MacBook-Air JAVA % █
```

Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 25

Title:

Write a Java program to find the maximum and minimum value of an array.

Code:

```
package Assignment2;

public class Question25 {
    public static void main(String[] args) {
        int[] numbers = {45, 78, 23, 56, 89, 12, 67};

        if (numbers.length == 0) {
            System.out.println("Array is empty.");
            return;
        }

        int max = numbers[0];
        int min = numbers[0];

        for (int num : numbers) {
            if (num > max) {
                max = num;
            }
            if (num < min) {
                min = num;
            }
        }

        System.out.println("Maximum value: " + max);
        System.out.println("Minimum value: " + min);
    }
}
```

Output:

```
apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question25
Maximum value: 89
Minimum value: 12
apple@Jeevans-MacBook-Air JAVA % █
```

Name of Student: JEEVAN NAIDU

Roll Number: 15

Experiment No: 26

Title:

Write a Java program to sort an array of integers.

Code:

```
package Assignment2;

import java.util.Arrays;

public class Question26 {
    public static void main(String[] args) {
        int[] numbers = {34, 7, 23, 32, 5, 62, 32};

        System.out.println("Original array:");
        for (int num : numbers) {
            System.out.print(num + " ");
        }
        System.out.println();

        Arrays.sort(numbers);

        System.out.println("Sorted array:");
        for (int num : numbers) {
            System.out.print(num + " ");
        }
    }
}
```

Output:

```
apple@Jeevans-MacBook-Air JAVA % /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-11.jdk/Contents/Home/bin/java -cp /Users/apple/Library/Application\ Support/C
ode/User/workspaceStorage/065d3d50a5046553aff44f758dd56b2d/redhat.java/jdt_ws/JAVA_4bc477a0/bin Assignment2.Question26
Original array:
34 7 23 32 5 62 32
Sorted array:
5 7 23 32 32 34 62
apple@Jeevans-MacBook-Air JAVA %
```

