

**RegNo:** 23MCA1030

**Name:** Vinayak Kumar Singh

**Java Programming Lab (PMCA502P)**

1. Write a Java program that demonstrates inheritance with constructors. Define a superclass Person with private attributes name and age. Implement a constructor to initialize these attributes. Create a subclass Student inheriting from Person with an additional attribute studentID. Implement a constructor in the Student class that initializes all attributes, including those inherited from Person.

Expected Input/Output:

Input:

Name: John

Age: 20

Student ID: 12345

Output:

Name: John

Age: 20

Student ID: 12345

Instructions:

Define the Person class with private attributes name and age, along with a constructor to initialize them.

Define the Student class inheriting from Person with an additional private attribute studentID.

Implement a constructor in the Student class that initializes all attributes, including those inherited from Person.

In the main method, create an instance of the Student class, passing appropriate values to the constructor, and print the details of the student, including name, age, and student ID.

**Code:**

```
// Superclass Person with private attributes name and age
class Person {
    private String name;
    private int age;
    // Constructor initialize name and age
    Person(String name, int age) {
        this.name = name;
        this.age = age;
    }
    // Methods for name and age
    public String getName() {
        return name;
    }
    public int getAge() {
        return age;
    }
}

// Subclass Student inheriting from Person with additional attribute
studentID
```

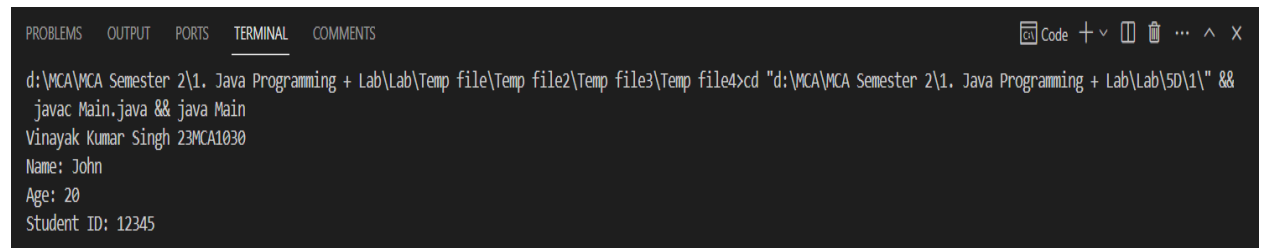
```
class Student extends Person {  
    private int studentID;  
  
    // Constructor initialize all attributes with those inherited from Person  
    Student(String name, int age, int studentID) {  
        super(name, age);  
        this.studentID = studentID;  
    }  
  
    // Method for studentID  
    public int getStudentID() {  
        return studentID;  
    }  
}  
  
// Main class  
public class Main {  
    public static void main(String[] args) {  
        // Create an instance of the Student class  
        Student student = new Student("John", 20, 12345);  
  
        // Print the details of the student  
        System.out.println("Vinayak Kumar Singh 23MCA1030");  
    }  
}
```

```

        System.out.println("Name: " + student.getName());
        System.out.println("Age: " + student.getAge());
        System.out.println("Student ID: " + student.getStudentID());
    }
}

```

## Output:



```

d:\MCA\MCA Semester 2\1. Java Programming + Lab\Lab\Temp file\Temp file2\Temp file3\Temp file4>cd "d:\MCA\MCA Semester 2\1. Java Programming + Lab\Lab\50\1\" &&
javac Main.java && java Main
Vinayak Kumar Singh 23MCA1030
Name: John
Age: 20
Student ID: 12345

```

## 2. Overriding Constructors:

Define a superclass Animal with private attributes name and type. Implement a constructor to initialize these attributes. Create a subclass Dog inheriting from Animal with an additional attribute breed. Override the constructor in the Dog class to initialize all attributes, including those inherited from Animal.

### Code:

```

// Superclass Animal with private attributes name and type
class Animal {
    private String name;
    private String type;
    // Constructor to initialize name and type
    Animal(String name, String type) {

```

```
    this.name = name;

    this.type = type;
}

// Methods for name and type
public String getName() {
    return name;
}

public String getType() {
    return type;
}
}

// Subclass Dog inheriting from Animal with an additional attribute
breed

class Dog extends Animal {
    private String breed;

    // Override the constructor to initialize all attributes, including those
    inherited from Animal

    Dog(String name, String type, String breed) {
        super(name, type);
        this.breed = breed;
    }

    // Method for breed
```

```

    public String getBreed() {
        return breed;
    }
}
// Main class
public class Main {
    public static void main(String[] args) {
        // Created instance of the Dog class
        Dog dog = new Dog("Tommy", "German ", " Shepherd");
        // Printing details of the dog
        System.out.println("Vinayak Kumar Singh 23MCA1030");
        System.out.println("Name: " + dog.getName());
        System.out.println("Type: " + dog.getType());
        System.out.println("Breed: " + dog.getBreed());
    }
}

```

## Output:

```

PROBLEMS OUTPUT PORTS TERMINAL COMMENTS
d:\MCA\MCA Semester 2\1. Java Programming + Lab\Lab\5D\2>cd "d:\MCA\MCA Semester 2\1. Java Programming + Lab\Lab\5D\2\" && javac Main.java && java Main
Vinayak Kumar Singh 23MCA1030
Name: Tommy
Type: German
Breed: Shepherd

```

### 3.Access Modifiers and Constructors:

Define a superclass Vehicle with private attributes make and model. Implement a constructor to initialize these attributes. Create a subclass Car inheriting from Vehicle with an additional attribute color. Implement a constructor in the Car class that initializes all attributes, including those inherited from Vehicle.

```
// Superclass Vehicle with private attributes make and model
class Vehicle {
    private String make;
    private String model;

    // Constructor to initialize make and model
    Vehicle(String make, String model) {
        this.make = make;
        this.model = model;
    }

    // Methods for make and model
    public String getMake() {
        return make;
    }

    public String getModel() {
        return model;
    }
}
```

```
// Subclass Car inheriting from Vehicle with an additional attribute
color

class Car extends Vehicle {
    private String color;

    // Constructor to initialize all attributes, including those inherited
    from Vehicle

    Car(String make, String model, String color) {
        super(make, model);
        this.color = color;
    }

    // Method for color

    public String getColor() {
        return color;
    }
}

// Main class

public class Main {
    public static void main(String[] args) {
        // Create an instance of the Car class

        Car car = new Car("Tesla", "Model X ", "Red");

        // Print the details of the car

        System.out.println("Vinayak Kumar Singh 23MCA1030");
    }
}
```



```

        System.out.println("Make: " + car.getMake());

        System.out.println("Model: " + car.getModel());

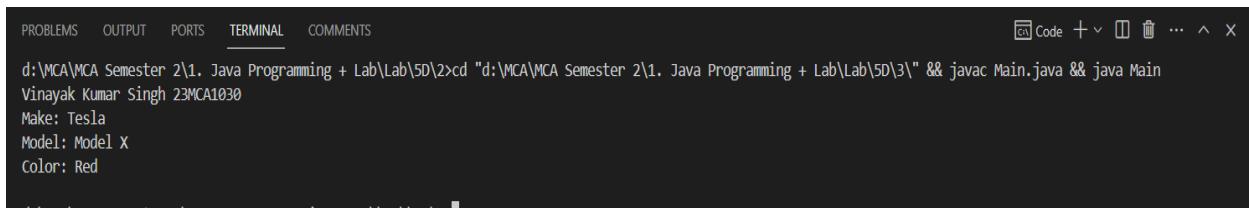
        System.out.println("Color: " + car.getColor());

    }

}

```

Output:



```

PROBLEMS OUTPUT PORTS TERMINAL COMMENTS
d:\MCA\MCA Semester 2\1. Java Programming + Lab\Lab\5D\2>cd "d:\MCA\MCA Semester 2\1. Java Programming + Lab\Lab\5D\3\" && javac Main.java && java Main
Vinayak Kumar Singh 23MCA1030
Make: Tesla
Model: Model X
Color: Red

```

#### 4. Constructor Chaining:

Define a superclass Shape with private attributes color and name. Implement a constructor to initialize these attributes. Create a subclass Circle inheriting from Shape with an additional attribute radius. Implement a constructor in the Circle class that initializes all attributes, including those inherited from Shape, using constructor chaining.

Code:

```

// Superclass Shape with private attributes color and name
class Shape {
    private String color;
    private String name;

    // Constructor to initialize color and name
    Shape(String color, String name) {

```

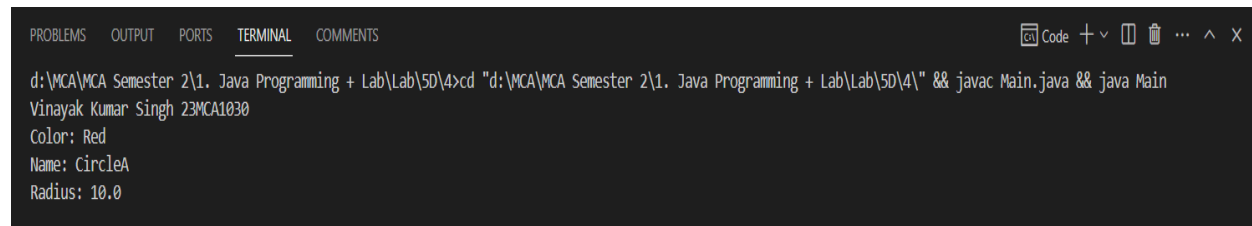
```
    this.color = color;
    this.name = name;
}
// Methods for color and name
public String getColor() {
    return color;
}
public String getName() {
    return name;
}
}
// Subclass Circle inheriting from Shape with an additional attribute
radius
class Circle extends Shape {
    private double radius;
    // Constructor to initialize all attributes, including those inherited from
    Shape, using constructor chaining
    Circle(String color, String name, double radius) {
        super(color, name);
        this.radius = radius;
    }
    // Method for radius
    public double getRadius() {
        return radius;
    }
}
```

```

}
// Main class
public class Main {
    public static void main(String[] args) {
        // Created instance of the Circle class
        Circle circle = new Circle("Red", "CircleA", 10.0);
        // Printing details of the circle
        System.out.println("Vinayak Kumar Singh 23MCA1030");
        System.out.println("Color: " + circle.getColor());
        System.out.println("Name: " + circle.getName());
        System.out.println("Radius: " + circle.getRadius());
    }
}

```

## Output:



```

PROBLEMS OUTPUT PORTS TERMINAL COMMENTS
d:\MCA\MCA Semester 2\1. Java Programming + Lab\Lab\5D\4>cd "d:\MCA\MCA Semester 2\1. Java Programming + Lab\Lab\5D\4" && javac Main.java && java Main
Vinayak Kumar Singh 23MCA1030
Color: Red
Name: CircleA
Radius: 10.0

```

5. Write a Java program that demonstrates the use of the super keyword and constructor chaining. Define a superclass Vehicle with private attributes make and model. Implement a constructor to initialize these attributes. Create a subclass Car inheriting from Vehicle with an additional attribute color. Implement a constructor in the Car class that initializes all attributes, including those inherited from Vehicle, using constructor chaining with the super keyword.

Expected Input/Output:

Input:

Make: Toyota

Model: Camry

Color: Red

Output:

Make: Toyota

Model: Camry

Color: Red

Instructions:

Define the Vehicle class with private attributes make and model, along with a constructor to initialize them.

Define the Car class inheriting from Vehicle with an additional private attribute color.

Implement a constructor in the Car class that initializes all attributes, including those inherited from Vehicle, using constructor chaining with the super keyword.

In the main method, create an instance of the Car class, passing appropriate values to the constructor, and print the details of the car, including make, model, and color.

Code:

```
// Superclass Vehicle with private attributes make and model
class Vehicle {
    private String make;
    private String model;
```

```
// Constructor initialize make and model
Vehicle(String make, String model) {
    this.make = make;
    this.model = model;
}

// Methods for make and model
public String getMake() {
    return make;
}

public String getModel() {
    return model;
}
}

// Subclass Car inheriting from Vehicle with an additional private
attribute color
class Car extends Vehicle {
    private String color;

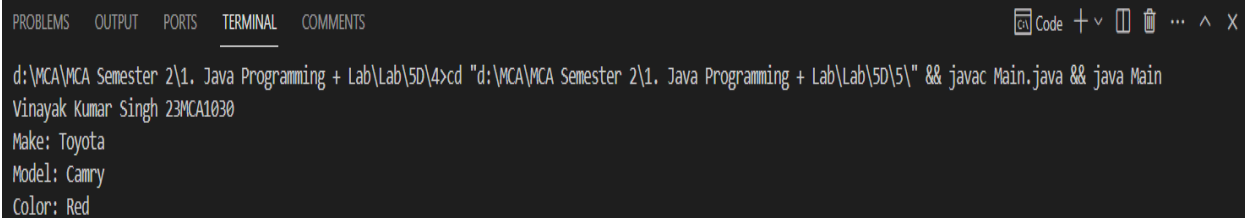
    // Constructor to initialize all attributes, including those inherited from
Vehicle, using constructor chaining with the super keyword
    Car(String make, String model, String color) {
        super(make, model);
        this.color = color;
    }

    // Method for color
    public String getColor() {
```

```
        return color;
    }
}

// Main class
public class Main {
    public static void main(String[] args) {
        // Created instance of the Car class
        Car car = new Car("Toyota", "Camry", "Red");
        // Printing details of the car
        System.out.println("Vinayak Kumar Singh 23MCA1030");
        System.out.println("Make: " + car.getMake());
        System.out.println("Model: " + car.getModel());
        System.out.println("Color: " + car.getColor());
    }
}
```

## Output:



The screenshot shows a terminal window with tabs for PROBLEMS, OUTPUT, PORTS, TERMINAL, and COMMENTS. The terminal displays the command to compile and run a Java program, followed by its output.

```
d:\MCA\MCA Semester 2\1. Java Programming + Lab\Lab\5D\4>cd "d:\MCA\MCA Semester 2\1. Java Programming + Lab\Lab\5D\5\" && javac Main.java && java Main
Vinayak Kumar Singh 23MCA1030
Make: Toyota
Model: Camry
Color: Red
```