Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра інформаційні систем та технологій

Лабораторна робота №3

Java

Варіант 3 (103)

Виконав:
Студент групи ІА-33
Ничик О. О.

Перевірив:
Лесик В. О.

Київ - 2025

Мета роботи: закріплення на практиці знань з принципів об’єктно орієнтованого програмування: абстракція, інкапсуляція, поліморфізм та наслідування; створення додатку з використанням архітектурного шаблону model-view-controller.

Хід роботи

Main.java

```java
import lab3.model.*;
import lab3.view.ShapeView;
import lab3.controller.ShapeController;

public class Main {
    Run | Debug
    public static void main(String[] args) {

        Shape[] shapes = {
                new Rectangle(color: "Red", width: 3, height: 4),
                new Circle(color: "Blue", radius: 2),
                new Triangle(color: "Green", base: 3, height: 5),
                new Rectangle(color: "Black", width: 5, height: 6),
                new Circle(color: "Yellow", radius: 4),
                new Triangle(color: "White", base: 6, height: 2),
                new Rectangle(color: "Red", width: 1, height: 2),
                new Circle(color: "Blue", radius: 3),
                new Triangle(color: "Green", base: 4, height: 4),
                new Rectangle(color: "Pink", width: 7, height: 3),
        };

        ShapeView view = new ShapeView();
        ShapeController controller = new ShapeController(shapes, view);

        controller.showAll();
        controller.totalArea();
        controller.totalAreaByType(type: Rectangle.class);
        controller.totalAreaByType(type: Circle.class);
        controller.totalAreaByType(type: Triangle.class);

        controller.sortByArea();
        controller.sortByColor();
    }
}
```

ShapeController.java

```java
public class ShapeController {
    private final Shape[] shapes;
    private final ShapeView view;

    public ShapeController(Shape[] shapes, ShapeView view) {
        this.shapes = shapes;
        this.view = view;
    }

    public void showAll() {
        view.displayMessage(msg: "Усі фігури:");
        view.displayShapes(shapes);
    }

    public void sortByArea() {
        Arrays.sort(shapes, Comparator.comparingDouble(Shape::calcArea));
        view.displayMessage(msg: "Сортування за площею:");
        view.displayShapes(shapes);
    }

    public void sortByColor() {
        Arrays.sort(shapes, Comparator.comparing(Shape::getShapeColor));
        view.displayMessage(msg: "Сортування за кольором:");
        view.displayShapes(shapes);
    }

    public void totalArea() {
        double sum = Arrays.stream(shapes)
                .mapToDouble(Shape::calcArea).sum();
        view.displayMessage("Сумарна площа всіх фігур: " + sum + "\n");
    }

    public void totalAreaByType(Class<?> type) {
        double sum = Arrays.stream(shapes)
                .filter(s -> s.getClass() == type)
                .mapToDouble(Shape::calcArea)
                .sum();

        view.displayMessage("Сумарна площа " + type.getSimpleName() + ": " + sum + "\n");
    }
}
```

Circle.java

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

    public Circle(String color, double radius) {
        super(color);
        this.radius = radius;
    }

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

    @Override
    public String toString() {
        return super.toString() +
                " (radius=" + radius + ")";
    }
}
```

Drawable.java

```java
public interface Drawable {
    void draw();
}
```

Rectangle.java

```java
public class Rectangle extends Shape {
    private double width;
    private double height;

    public Rectangle(String color, double width, double height) {
        super(color);
        this.width = width;
        this.height = height;
    }

    @Override
    public double calcArea() {
        return width * height;
    }

    @Override
    public String toString() {
        return super.toString() +
                " (width=" + width +
                ", height=" + height + ")";
    }
}
```

Shape.java

```java
public abstract class Shape implements Drawable {
    protected String shapeColor;

    public Shape(String shapeColor) {
        this.shapeColor = shapeColor;
    }

    public abstract double calcArea();

    @Override
    public void draw() {
        System.out.println(this);
    }

    @Override
    public String toString() {
        return getClass().getSimpleName() +
                " [color=" + shapeColor +
                ", area=" + calcArea() + "]";
    }

    public String getShapeColor() {
        return shapeColor;
    }

}
```

Triangle.java

```java
public class Triangle extends Shape {
    private double base;
    private double height;

    public Triangle(String color, double base, double height) {
        super(color);
        this.base = base;
        this.height = height;
    }

    @Override
    public double calcArea() {
        return (base * height) / 2;
    }

    @Override
    public String toString() {
        return super.toString() +
                " (base=" + base +
                ", height=" + height + ")";
    }
}
```

Shapeview.java

```java
import lab3.model.Shape;

public class ShapeView {
    public void displayShapes(Shape[] shapes) {
        for (Shape s : shapes) {
            System.out.println(s);
        }
        System.out.println();
    }

    public void displayMessage(String msg) {
        System.out.println(msg);
    }
}
```

https://github.com/iamthegoose/java-fifth-semester/tree/lab3

Висновок: В ході виконання даної лабораторної роботи ми закріпили на практиці знання з принципів об’єктно орієнтованого програмування: абстракцію, інкапсуляцію, поліморфізм та наслідування; створення додатку з використанням архітектурного шаблону model-view-controller.