Чернівецький національний університет імені Юрія Федьковича

Інститут фізико-технічних та комп’ютерних наук

Кафедра Математичних проблем управління і кібернетики

Лабораторна робота № 2

“Тема: Створення класів в Java”

з дисципліни “ Java”

Варіант № 18

Виконав:

ст. гр. 241 Фрасинюк О. Б.

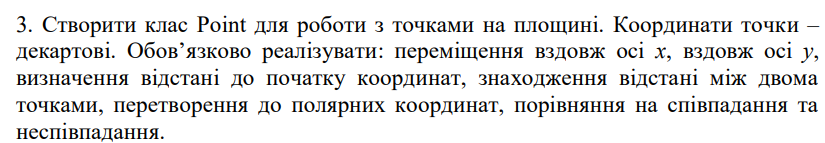
Прийняв:

Викладач Лазорик В.В.

Чернівці – 2024

<https://github.com/VLazorykOOP/jlab1-Oleh-Frasyniuk>

**Завдання 1**



**Код:**

import java.util.Objects;

public class Point {

    private double x;

    private double y;

    public Point(double x, double y) {

        this.x = x;

        this.y = y;

    }

    public double getX() {

        return x;

    }

    public double getY() {

        return y;

    }

    public void moveX(double deltaX) {

        this.x += deltaX;

    }

    // Переміщення вздовж осі y

    public void moveY(double deltaY) {

        this.y += deltaY;

    }

    //відстані до початку координат

    public double distanceToOrigin() {

        return Math.sqrt(x \* x + y \* y);

    }

    //відстані між двома точками

    public double distanceTo(Point other) {

        double dx = this.x - other.x;

        double dy = this.y - other.y;

        return Math.sqrt(dx \* dx + dy \* dy);

    }

    // Перетворення до полярних координат

    public double[] toPolarCoordinates() {

        double r = Math.sqrt(x \* x + y \* y); // Радіус

        double theta = Math.atan2(y, x);    // Кут (в радіанах)

        return new double[]{r, theta};

    }

    // Порівняння на співпадання

    @Override

    public boolean equals(Object o) {

        if (this == o) return true;

        if (o == null || getClass() != o.getClass()) return false;

        Point point = (Point) o;

        return Double.compare(point.x, x) == 0 && Double.compare(point.y, y) == 0;

    }

    // Порівняння на неспівпадання

    @Override

    public int hashCode() {

        return Objects.hash(x, y);

    }

    // Представлення точки у вигляді рядка

    @Override

    public String toString() {

        return "Point{" +

                "x=" + x +

                ", y=" + y +

                '}';

    }

    public static void main(String[] args) {

        Point p1 = new Point(3, 4);

        Point p2 = new Point(6, 8);

        System.out.println("P1: " + p1);

        System.out.println("P2: " + p2);

        // Переміщення точки

        p1.moveX(2);

        p1.moveY(-1);

        System.out.println("p1 after move: " + p1);

        // Відстань до початку координат

        System.out.println("Distance P1 to start coordinate: " + p1.distanceToOrigin());

        // Відстань між двома точками

        System.out.println("Distance p1 between p2: " + p1.distanceTo(p2));

        // Полярні координати

        double[] polar = p1.toPolarCoordinates();

        System.out.println("Polar coordinates p1: r = " + polar[0] + ", θ = " + polar[1]);

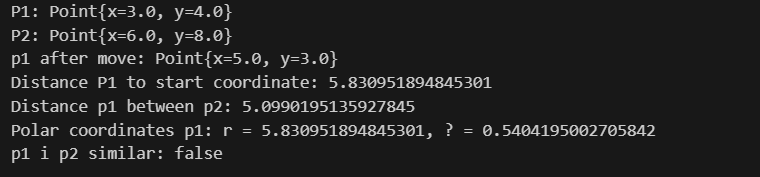
        // Порівняння точок

        System.out.println("p1 i p2 similar: " + p1.equals(p2));

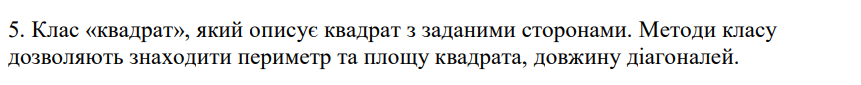
    }

}

**Результат:**



**Завдання 2**



**Код:**

public class Square {

    private double side;

    public Square(double side) {

        if (side <= 0) {

            throw new IllegalArgumentException("Side shouldn't equal to 0.");

        }

        this.side = side;

    }

    public double getSide() {

        return side;

    }

    // Сеттер для сторони

    public void setSide(double side) {

        if (side <= 0) {

            throw new IllegalArgumentException("Side shouldn't equal to 0.");

        }

        this.side = side;

    }

    public double getPerimeter() {

        return 4 \* side;

    }

    public double getArea() {

        return side \* side;

    }

    // Обчислення довжини діагоналі

    public double getDiagonal() {

        return Math.sqrt(2) \* side;

    }

    // Представлення квадрата у вигляді рядка

    @Override

    public String toString() {

        return "Square{" +

                "side=" + side +

                '}';

    }

    public static void main(String[] args) {

        Square square = new Square(5);

        System.out.println("Side: " + square.getSide());

        System.out.println("Peimetr: " + square.getPerimeter());

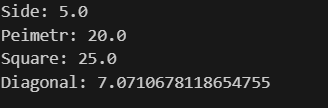
        System.out.println("Square: " + square.getArea());

        System.out.println("Diagonal: " + square.getDiagonal());

    }

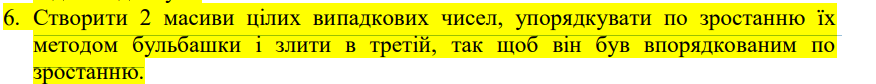
}

**Результат:**



Завдання 3:

**Код:**import java.util.Random;



import java.util.Objects;

public class MergeSortedArrays {

    public static void main(String[] args) {

        int[] array1 = generateRandomArray(10);

        int[] array2 = generateRandomArray(10);

        // Сортування методом бульбашки

        bubbleSort(array1);

        bubbleSort(array2);

        // Злиття двох масивів в один упорядкований

        int[] mergedArray = mergeArrays(array1, array2);

        System.out.println("Masive 1:");

        printArray(array1);

        System.out.println("Masive 2:");

        printArray(array2);

        System.out.println("Split Masive:");

        printArray(mergedArray);

    }

    public static int[] generateRandomArray(int size) {

        Random random = new Random();

        int[] array = new int[size];

        for (int i = 0; i < size; i++) {

            array[i] = random.nextInt(100); // Числа від 0 до 99

        }

        return array;

    }

    // Метод сортування бульбашкою

    public static void bubbleSort(int[] array) {

        int n = array.length;

        for (int i = 0; i < n - 1; i++) {

            for (int j = 0; j < n - i - 1; j++) {

                if (array[j] > array[j + 1]) {

                    int temp = array[j];

                    array[j] = array[j + 1];

                    array[j + 1] = temp;

                }

            }

        }

    }

    // Метод для злиття двох відсортованих масивів

    public static int[] mergeArrays(int[] array1, int[] array2) {

        int[] mergedArray = new int[array1.length + array2.length];

        int i = 0, j = 0, k = 0;

        // Злиття двох масивів

        while (i < array1.length && j < array2.length) {

            if (array1[i] <= array2[j]) {

                mergedArray[k++] = array1[i++];

            } else {

                mergedArray[k++] = array2[j++];

            }

        }

        // Додавання залишків з першого масиву

        while (i < array1.length) {

            mergedArray[k++] = array1[i++];

        }

        // Додавання залишків з другого масиву

        while (j < array2.length) {

            mergedArray[k++] = array2[j++];

        }

        return mergedArray;

    }

    // Метод для виведення масиву на екран

    public static void printArray(int[] array) {

        for (int num : array) {

            System.out.print(num + " ");

        }

        System.out.println();

    }

}

**Результат:**

