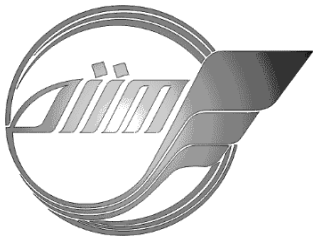
jav ajava МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ



Дніпровський національний університет
залізничного транспорту імені академіка В. Лазаряна

Кафедра «Комп'ютерні інформаційні технології»

Лабораторна робота № 13

з дисципліни «Об'єктно-орієнтоване програмування»

на тему: «Створення об’єктно-орієнтованих програм і консольних додатків з використанням Java.»

Виконав:
студент гр.ПЗ1911
Сафонов Д. Є.
Прийняла:
Демидович І.М.

Дніпро, 2021

Тема. Створення об’єктно-орієнтованих програм і консольних додатків з використанням Java.

Завдання. Написати об'єктно-орієнтовану програму на мові Java. Організувати консольний інтерфейс взаємодії з користувачем, структура діалогу — меню. В програмі повинно бути не менше трьох класів.

Індивідуальне завдання. 1. Дана дійсна матриця. Переставляючи її рядки і стовпці, домогтися того, щоб найбільший елемент (один з них) виявився в верхньому лівому кутку.

Діаграма класів.

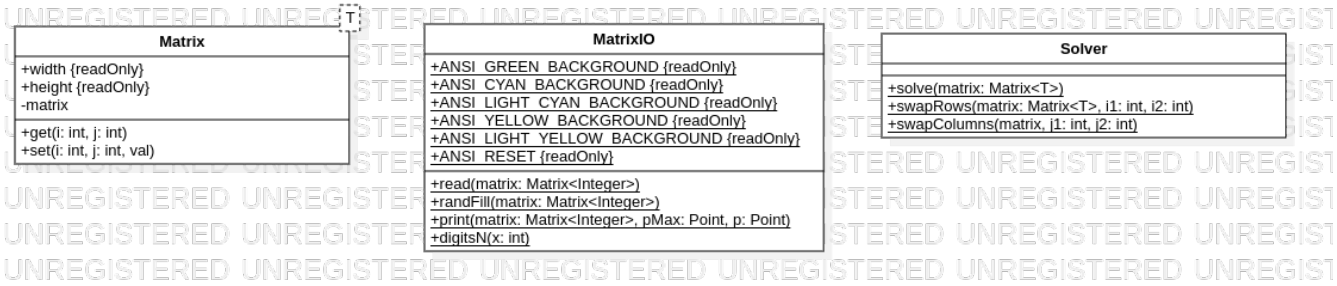


Рисунок 1

Алгоритм розв'язання задачі.

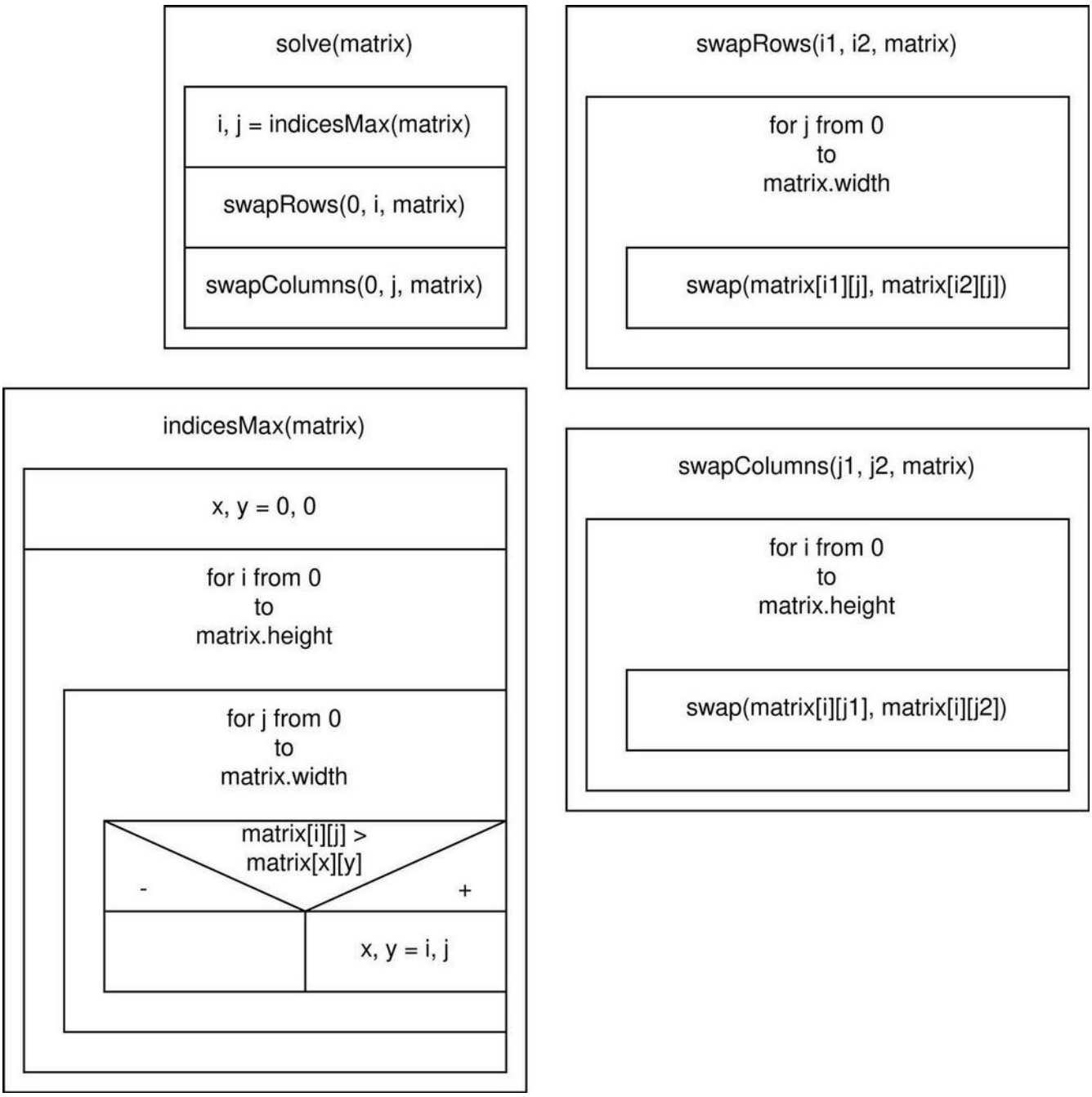


Рисунок 2

Текст програми.

“Main.java”

```
package oop13;

import java.awt.Point;
import java.util.Scanner;

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

		System.out.print("Input matrix' height: ");
		var h = in.nextInt();

		System.out.print("Input matrix' width: ");
		var w = in.nextInt();

		var in2 = new Scanner(System.in);
		System.out.print("Would you like to fill matrix yourself(Y - yourself, any other input -
random): ");
		var choice = in2.next();

		var matrix = new Matrix<Integer>(h, w, 0);

		if (choice.compareTo("Y") == 0) {
			MatrixIO.read(matrix);
		} else {
			MatrixIO.randFill(matrix);
		}

		var zero = new Point(0, 0);
		var indices = Solver.solve(matrix);

		var x = indices.x;
		var y = indices.y;

		MatrixIO.print(matrix, indices, zero);
		System.out.println();

		Solver.swapRows(matrix, 0, x);
		Solver.swapColumns(matrix, 0, y);
		MatrixIO.print(matrix, zero, indices);
	}
}
```

“Matrix.java”

```
package oop13;

import java.util.ArrayList;
import java.util.List;

public class Matrix <T> {
	public final int width;
	public final int height;
	private List<List<T>> matrix;

	public Matrix(int height, int width, T filler) {
		this.height = height;
		this.width  = width;

		this.matrix = new ArrayList<>(height);
		for (int i = 0; i < height; i++) {
			this.matrix.add(i, new ArrayList<>(width));
			for (int j = 0; j < width; j++) {
				this.matrix.get(i).add(filler);
			}
		}
	}

	public void set(int i, int j, T val) {
		this.matrix.get(i).set(j, val);
	}

	public T get(int i, int j) {
		return this.matrix.get(i).get(j);
	}
}
```

“MatrixIO.java”

```
package oop13;

import java.awt.Point;
import java.util.Random;
import java.util.Scanner;

public class MatrixIO {
	public static final String ANSI_GREEN_BACKGROUND = "\u001B[42m";
	public static final String ANSI_CYAN_BACKGROUND = "\u001B[46m";
	public static final String ANSI_LIGHT_CYAN_BACKGROUND = "\u001B[106m";
	public static final String ANSI_YELLOW_BACKGROUND = "\u001B[43m";
	public static final String ANSI_LIGHT_YELLOW_BACKGROUND = "\u001B[103m";
	public static final String ANSI_RESET = "\u001B[0m";

	public static void read(Matrix<Integer> matrix) {
		var in = new Scanner(System.in);
		var maxLenI = digitsN(matrix.height - 1);
		var maxLenJ = digitsN(matrix.width - 1);
		for (int i = 0; i < matrix.height; i++) {
			for (int j = 0; j < matrix.width; j++) {
				System.out.printf("matrix[%" + maxLenI +  "d][" + "%" + maxLenJ + "d] = ", i, j);
				matrix.set(i, j, in.nextInt());
			}
		}
	}

	public static void randFill(Matrix<Integer> matrix) {
		var r = new Random();
		for (int i = 0; i < matrix.height; i++) {
			for (int j = 0; j < matrix.width; j++) {
				matrix.set(i, j, r.nextInt(100));
			}
		}
	}

	public static void print(Matrix<Integer> matrix, Point pMax, Point p) {
		var x = pMax.x;
		var y = pMax.y;

		var maxLen = digitsN(matrix.get(x, y));

		for (int i = 0; i < matrix.height; i++) {
			for (int j = 0; j < matrix.width; j++) {
				var color = ((i == x || j == y) && (i == p.x || j == p.y)) ? ANSI_GREEN_BACKGROUND
						:  (i == x && j == y)                              ? ANSI_CYAN_BACKGROUND
						:  (i == x || j == y)                              ? ANSI_LIGHT_CYAN_BACKGROUND
						:  (i == p.x && j == p.y)                          ? ANSI_YELLOW_BACKGROUND
						:  (i == p.x || j == p.y)                          ? ANSI_LIGHT_YELLOW_BACKGROUND
                                                                           : ANSI_RESET;
				System.out.printf(color + "%" + maxLen + "d ", matrix.get(i, j));
			}
			System.out.println(ANSI_RESET);
```

```
            }
        }

        public static int digitsN(int x) {
            return String.valueOf(x).length();
        }
}
```

“Solver.java”

```
package oop13;

import java.awt.Point;

public class Solver {
	public static <T extends Comparable<T>> Point solve(Matrix<T> matrix) {
		int imax = 0;
		int jmax = 0;
		for (int i = 0; i < matrix.height; i++) {
			for (int j = 0; j < matrix.width; j++) {
				if (matrix.get(i, j).compareTo(matrix.get(imax, jmax)) > 0) {
					imax = i;
					jmax = j;
				}
			}
		}
		return new Point(imax, jmax);
	}

	public static <T> void swapRows(Matrix<T> matrix, int i1, int i2) {
		for (int j = 0; j < matrix.width; j++) {
			var tmp = matrix.get(i1, j);
			matrix.set(i1, j, matrix.get(i2, j));
			matrix.set(i2, j, tmp);
		}
	}

	public static <T> void swapColumns(Matrix<T> matrix, int j1, int j2) {
		for (int i = 0; i < matrix.height; i++) {
			var tmp = matrix.get(i, j1);
			matrix.set(i, j1, matrix.get(i, j2));
			matrix.set(i, j2, tmp);
		}
	}
}
```

Приклад роботи програми.

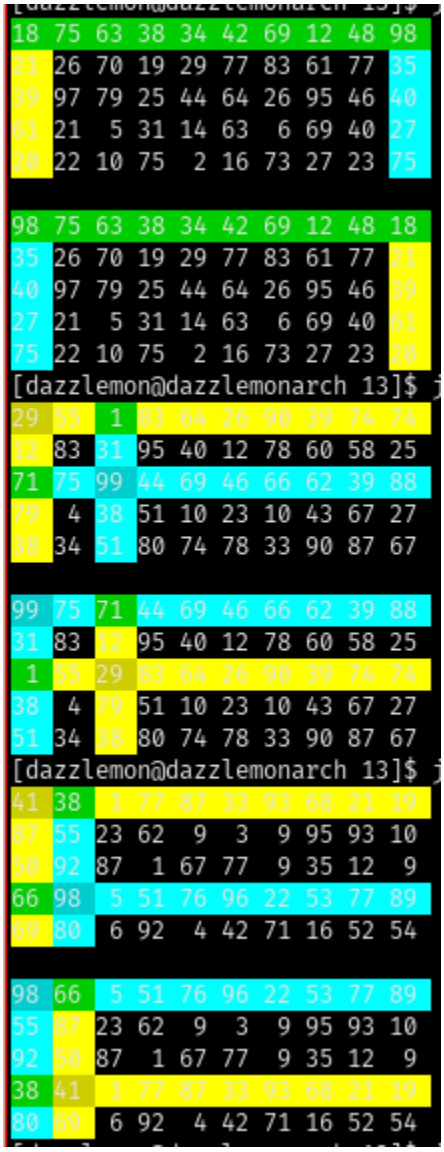


Рисунок 3(Алгоритм на випадково заповнених матрицях)

Синім кольором помічені рядок та стовпець першого примірники найбільшого елемента. Жовтим - нульовий рядок та стовпець, зеленим - місця перетину синіх та жовтих рядків/стовпців.

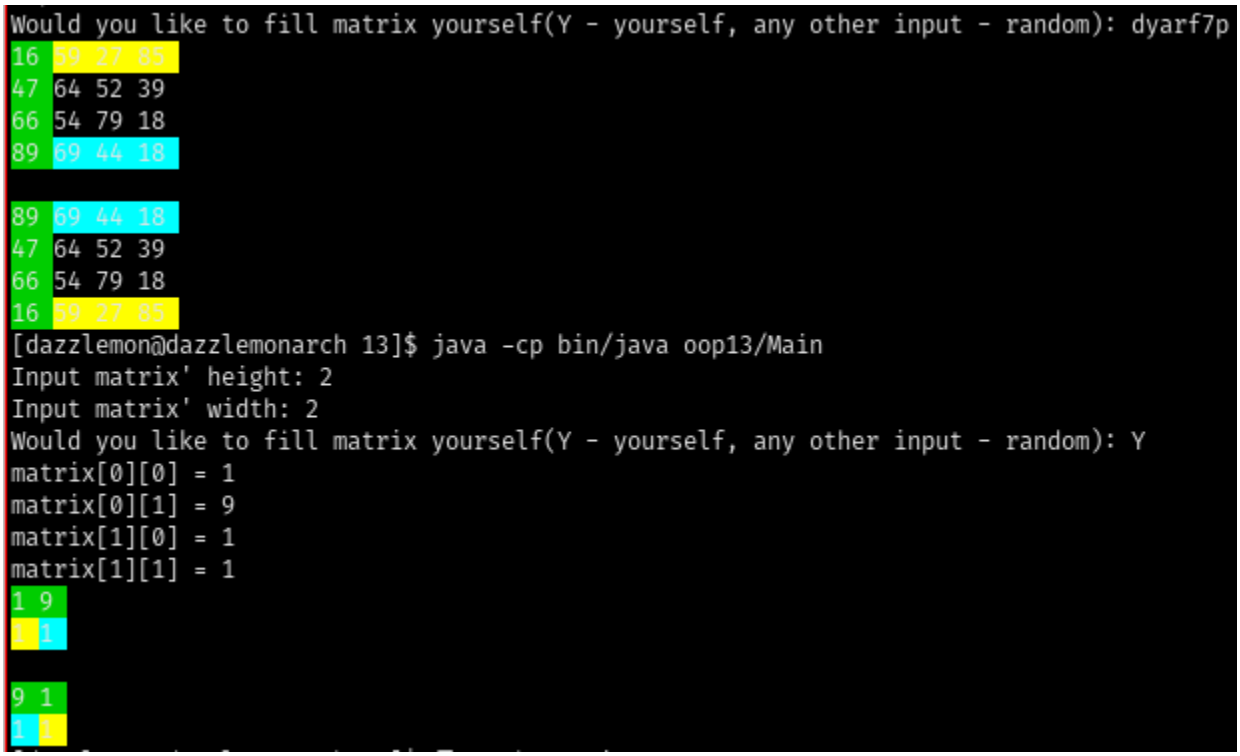


Рисунок 4(Взаємодія з користувачем)

Висновки.

Мова програмування Java потребує дуже конкретного опису програми, і дещо обмежує розробника. Наприклад клас може мати поле загального типу даних(generic), але якщо це поле - масив, то його не можна ініціалізувати у методах класу. І таких обмежень дуже багато. І хоч розробники цієї мови дають на них пояснення, вони не завжди є переконливими. наприклад у java немає загального класу, який би поєднував два типи(є у javafx), на це розробники рекомендують створювати невеликі класи. тож не дивлячись на те, що ця мова має вищий рівень абстракції відносно C++, вона не реалізує дуже багато стандартних речей, тож програміст має писати код, який міг бути включений у стандартну бібліотеку.