Лабораторна робота 2

1. Ввести n чисел Серед чисел знайти число-паліндром. Якщо таких чисел більше одного, знайти друге.

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int n;

System.out.print("Vedit kilkist chisel: ");

n = scanner.nextInt();

int firstPalindrome = -1;

int secondPalindrome = -1;

int countPalindromes = 0;

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

System.out.print("Vedit chislo: ");

int number = scanner.nextInt();

if (isPalindrome(number)) {

if (countPalindromes == 0) {

firstPalindrome = number;

countPalindromes++;

} else if (countPalindromes == 1) {

secondPalindrome = number;

countPalindromes++;

}

}

}

if (countPalindromes >= 2) {

System.out.println("pershii polinom: " + firstPalindrome);

System.out.println("drygii polinom: " + secondPalindrome);

} else if (countPalindromes == 1) {

System.out.println("lishe 1 polinom: " + firstPalindrome);

} else {

System.out.println("ne znaideno polinomiv.");

}

scanner.close();

}

public static boolean isPalindrome(int number) {

int originalNumber = number;

int reversedNumber = 0;

while (number > 0) {

int digit = number % 10;

reversedNumber = reversedNumber \* 10 + digit;

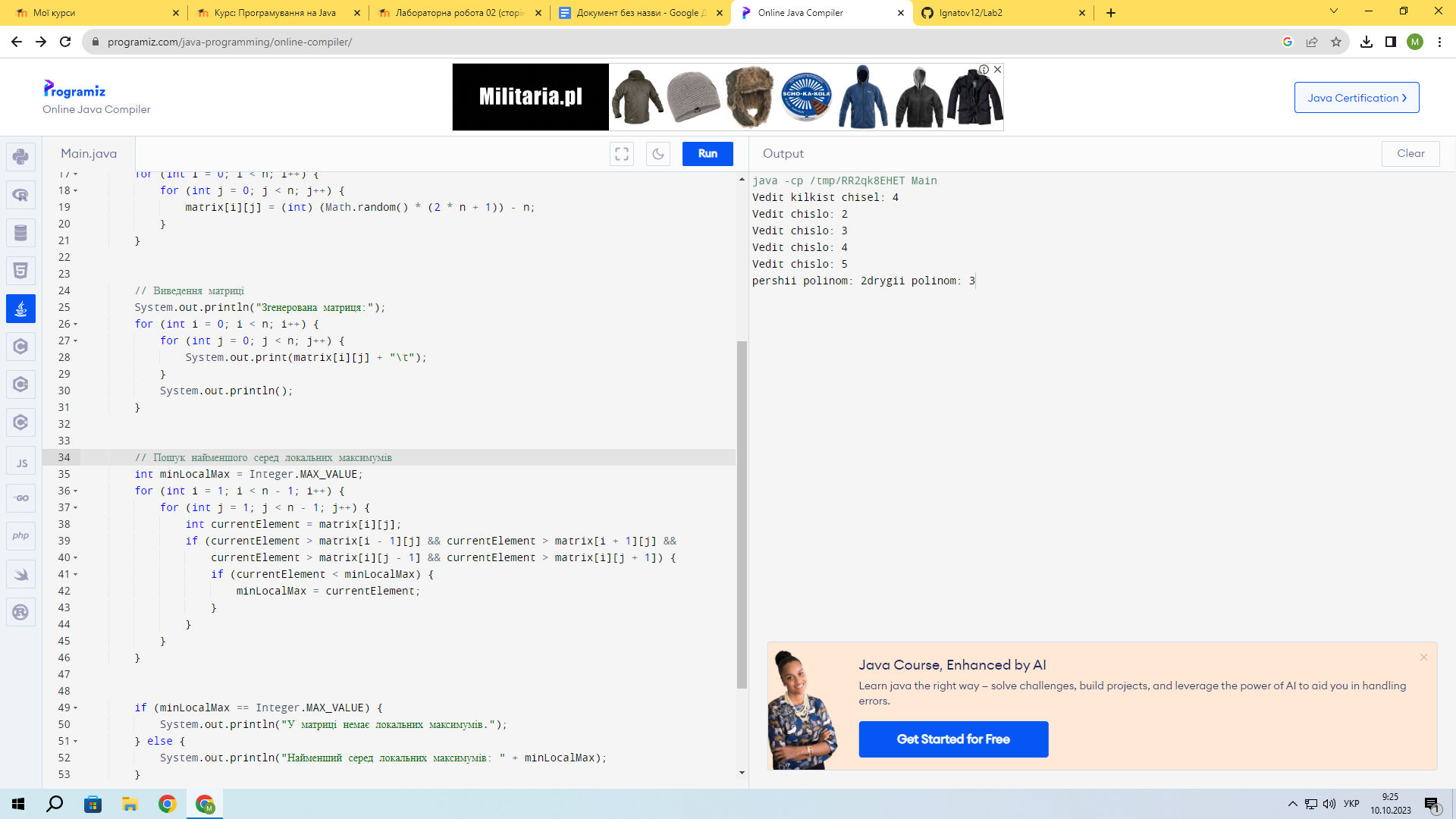
number /= 10;

}

return originalNumber == reversedNumber;

}

}



1. Вивести числа від 1 до k у вигляді матриці N x N зліва направо і зверху вниз

public class Main {

public static void main(String[] args) {

int k = 16; // Задайте значення k (кількість чисел)

int N = (int) Math.ceil(Math.sqrt(k)); // Розрахунок розміру матриці NxN

int[][] matrix = new int[N][N];

int num = 1;

// Заповнення матриці числами від 1 до k

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

for (int j = 0; j < N; j++) {

if (num <= k) {

matrix[i][j] = num;

num++;

}

}

}

// Виведення матриці

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

for (int j = 0; j < N; j++) {

System.out.print(matrix[i][j] + "\t");

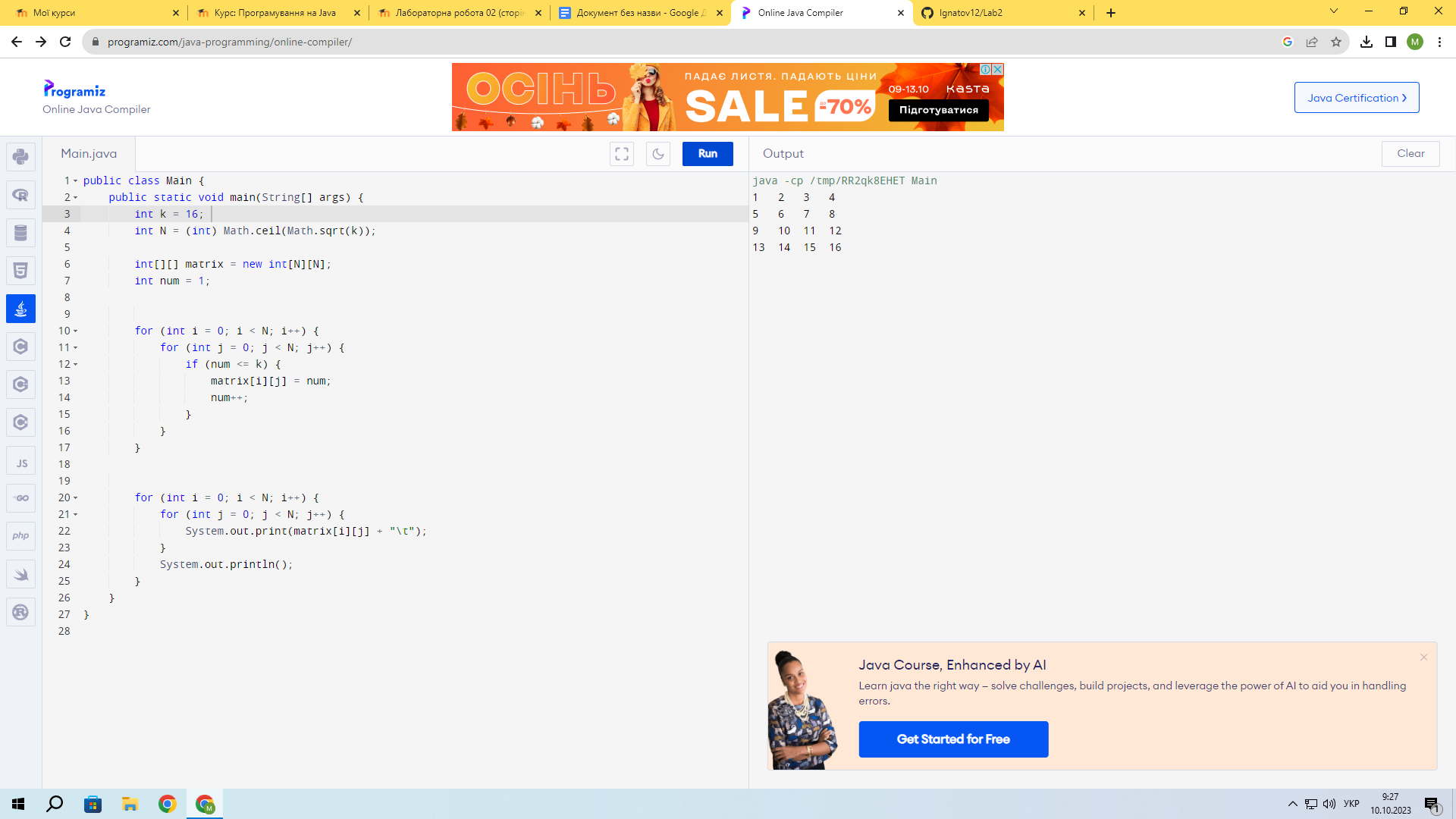
}

System.out.println();

}

}

}



1. Ввести з консолі n – розмірність матриці a[n] [n]. Задати значення елементів матриці в інтервалі значень від – n до n за допомогою датчика випадкових чисел. Знайти найменший серед локальних максимумів (елемент матриці називається локальним мінімумом, якщо він строго менше всіх своїх сусідів).

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Введіть розмірність матриці (n): ");

int n = scanner.nextInt();

int[][] matrix = new int[n][n];

// Заповнення матриці випадковими числами в інтервалі від -n до n

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

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

matrix[i][j] = (int) (Math.random() \* (2 \* n + 1)) - n;

}

}

// Виведення матриці

System.out.println("Згенерована матриця:");

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

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

System.out.print(matrix[i][j] + "\t");

}

System.out.println();

}

// Пошук найменшого серед локальних максимумів

int minLocalMax = Integer.MAX\_VALUE;

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

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

int currentElement = matrix[i][j];

if (currentElement > matrix[i - 1][j] && currentElement > matrix[i + 1][j] &&

currentElement > matrix[i][j - 1] && currentElement > matrix[i][j + 1]) {

if (currentElement < minLocalMax) {

minLocalMax = currentElement;

}

}

}

}

if (minLocalMax == Integer.MAX\_VALUE) {

System.out.println("У матриці немає локальних максимумів.");

} else {

System.out.println("Найменший серед локальних максимумів: " + minLocalMax);

}

scanner.close();

}

}

