НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

ІМ. ІГОРЯ СІКОРСЬКОГО

КАФЕДРА АВТОМАТИКИ ТА УПРАВЛІННЯ В ТЕХНІЧНИХ СИСТЕМАХ

Звіт

з дисципліни «Сучасні технології програмування - 1»

за результатами виконання лабораторної роботи № 2

Виконав:

студент групи ІТ-51

Сопов О. О.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Перевірив:

ас. Федорчук В. В.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Київ – 2017

**Завдання:**

1. Find the number of positive items;

Find the sum of the elements that are larger than 3;

Find the maximum element of the array;

Find the number of negative elements;

Find the sum of negative elements;

Find the minimum element that is multiple 5;

* 1. Given an integer square matrix of order 8. Find the least of the values of the column items that have the largest amount of modules elements of If there are several columns, then take the first one.;
  2. The given positive integer n is an integer square matrix of order n Obtain bi, ..., bn, where bi is: a) the least of the values of the elements that are at the beginning of the i-th matrix line to the element belonging to the main diagonal including; b) the value of the first in the order of the positive element of the i-th line (if there are no such elements, then take bi = l).

**Вихідний код:**

***Program.java***

**public class Program** {  
 **public static void** main(**String**[] args) {  
 *solveOneDimensionalArray*();  
 *solveMultiDimensionalArray*();  
 }  
  
 **private static void** solveOneDimensionalArray() {  
 **ArrayInitializer** arrayInitializer = **new** ArrayInitializer();  
  
 **int** arraySize = arrayInitializer.requestArraySize();  
 **int**[] array = arrayInitializer.initializeOneDimensionalArray(arraySize);  
  
 **ArrayPrinter**.*printArray*(array);  
 **OneDimensionalArraySolver** oneDimensionalArraySolver = **new** OneDimensionalArraySolver(array);  
  
 **System**.***out***.println("The number of positive items: " + oneDimensionalArraySolver.getCountOfPositive());  
 **System**.***out***.println("Sum of elements larger than 3: " + oneDimensionalArraySolver.getSumOfLargerThanThree());  
 **System**.***out***.println("Maximum element of array: " + oneDimensionalArraySolver.getMaximum());  
 **System**.***out***.println("The number of negative items: " + oneDimensionalArraySolver.getCountOfNegative());  
 **System**.***out***.println("The sum of negative items: " + oneDimensionalArraySolver.getSumOfNegative());  
 **System**.***out***.println("The minimum element that is multiple 5: " + oneDimensionalArraySolver.getMinElementMultipleFive());  
 }  
  
 **private static void** solveMultiDimensionalArray() {  
 **System**.***out***.println();  
 **ArrayInitializer** arrayInitializer = **new** ArrayInitializer();  
  
 *// 8 - by task* **int**[][] array1 = arrayInitializer.initializeTwoDimensionalArray(8);  
 **ArrayPrinter**.*printArray*(array1);  
 **TwoDimensionalArraySolver** twoDimensionalArraySolver = **new** TwoDimensionalArraySolver(array1);  
 **System**.***out***.println("The least element of max by module column: " + twoDimensionalArraySolver.getLeastElementOfMaxByModuleColumn());  
  
 **int** arraySize = arrayInitializer.requestArraySize();  
 **int**[][] array2 = arrayInitializer.initializeTwoDimensionalArray(arraySize);  
  
 **ArrayPrinter**.*printArray*(array2);  
 twoDimensionalArraySolver = **new** TwoDimensionalArraySolver(array2);  
  
 **System**.***out***.println("Minimums to diagonal: ");  
 **ArrayPrinter**.*printArray*(twoDimensionalArraySolver.getMinimumsToDiagonal());  
  
 **System**.***out***.println("\nFirst positives to diagonal: ");  
 **ArrayPrinter**.*printArray*(twoDimensionalArraySolver.getFirstPositivesToDiagonal());  
 }  
}

***OneDimensionalArraySolver.java***

**import java.util.Arrays**;  
  
  
**class** OneDimensionalArraySolver {  
 **private int**[] arrayData;  
 **private int**[] sortedData;  
  
 OneDimensionalArraySolver(**int**[] data) {  
 arrayData = data;  
  
 sortedData = **new int**[arrayData.length];  
 **System**.*arraycopy*(arrayData, 0, sortedData, 0, arrayData.length );  
 **Arrays**.*sort*(sortedData);  
 }  
  
 **int** getCountOfPositive() {  
 **int** count = 0;  
 **for** (**int** i = sortedData.length - 1; i >= 0; i--) {  
 **if** (sortedData[i] > 0) {  
 count++;  
 }  
 **else** {  
 **break**;  
 }  
 }  
  
 **return** count;  
 }  
 **int** getSumOfLargerThanThree() {  
 **int** sum = 0;  
 **for** (**int** value : arrayData) {  
 **if** (value > 3) {  
 sum += value;  
 }  
 }  
  
 **return** sum;  
 }  
 **int** getMaximum() {  
 **return** sortedData[sortedData.length - 1];  
 }  
 **int** getCountOfNegative() {  
 **int** count = 0;  
 **for** (**int** aSortedData : sortedData) {  
 **if** (aSortedData < 0) {  
 count++;  
 } **else** {  
 **break**;  
 }  
 }  
  
 **return** count;  
 }  
 **int** getSumOfNegative() {  
 **int** sum = 0;  
 **for** (**int** aSortedData : sortedData) {  
 **if** (aSortedData < 0) {  
 sum += aSortedData;  
 } **else** {  
 **break**;  
 }  
 }  
  
 **return** sum;  
 }  
 **int** getMinElementMultipleFive() {  
 **int** min = arrayData[0];  
 **for** (**int** value : arrayData) {  
 **if** (value % 5 == 0 && value < min) {  
 min = value;  
 }  
 }  
  
 **return** min;  
 }  
}

***TwoDimensionalArraySolver.java***

**class TwoDimensionalArraySolver** {  
 **private int**[][] arrayData;  
  
 TwoDimensionalArraySolver(**int**[][] data) {  
 arrayData = data;  
 }  
  
 **int** getLeastElementOfMaxByModuleColumn() {  
 **int** [] dataOfMaxByModuleColumn = **new int**[arrayData[0].length];  
 **int** maxByModuleColumn = getMaxByModuleColumn();  
  
 **for** (**int** i = 0; i < arrayData[0].length; i++) {  
 dataOfMaxByModuleColumn[i] = arrayData[i][maxByModuleColumn];  
 }  
  
 **return** getMin(dataOfMaxByModuleColumn);  
 }  
 **int**[] getMinimumsToDiagonal() {  
 **int**[] minimumsToDiagonal = **new int**[arrayData[0].length];  
  
 **for** (**int** i = 0; i < arrayData[0].length; i++) {  
 **int**[] currentRowToDiagonalData = **new int**[i + 1];  
 **System**.*arraycopy*(arrayData[i], 0, currentRowToDiagonalData, 0, i + 1);  
 minimumsToDiagonal[i] = getMin(currentRowToDiagonalData);  
 }  
  
 **return** minimumsToDiagonal;  
 }  
 **int**[] getFirstPositivesToDiagonal() {  
 **int**[] positivesToDiagonal = **new int**[arrayData[0].length];  
  
 **for** (**int** i = 0; i < arrayData[0].length; i++) {  
 **boolean** isPositiveToDiagonal = **false**;  
 **for** (**int** j = 0; j <= i; j++) {  
 **if** (arrayData[i][j] > 0) {  
 positivesToDiagonal[i] = arrayData[i][j];  
 isPositiveToDiagonal = **true**;  
 **break**;  
 }  
 }  
  
 **if** (!isPositiveToDiagonal) {  
 positivesToDiagonal[i] = 1;  
 }  
 }  
  
 **return** positivesToDiagonal;  
 }  
  
 **private int** getMaxByModuleColumn(){  
 **int**[] moduleColumnSum = **new int**[arrayData[1].length];  
  
 **for** (**int** i = 0; i < arrayData[0].length; i++) {  
 **for** (**int** j = 0; j < arrayData[1].length; j++) {  
 moduleColumnSum[j] += **Math**.*abs*(arrayData[i][j]);  
 }  
 }  
  
 **return** getIndexOfMax(moduleColumnSum);  
 }  
 **private int** getMin(**int**[] data) {  
 **int** min = data[0];  
  
 **for** (**int** value : data) {  
 **if** (value < min) {  
 min = value;  
 }  
 }  
  
 **return** min;  
 }  
 **private int** getIndexOfMin(**int**[] data) {  
 **int** min = getMin(data);  
  
 **for** (**int** i = 0; i < data.length; i++) {  
 **if** (data[i] == min) {  
 **return** i;  
 }  
 }  
  
 **return** -1;  
 }  
 **private int** getMax(**int**[] data) {  
 **int** max = data[0];  
  
 **for** (**int** value : data) {  
 **if** (value > max) {  
 max = value;  
 }  
 }  
  
 **return** max;  
 }  
 **private int** getIndexOfMax(**int**[] data) {  
 **int** max = getMax(data);  
  
 **for** (**int** i = 0; i < data.length; i++) {  
 **if** (data[i] == max) {  
 **return** i;  
 }  
 }  
  
 **return** -1;  
 }  
}

***ArrayInitializer.java***

**import java.io.BufferedReader**;  
**import java.io.IOException**;  
**import java.io.InputStreamReader**;  
**import java.util.Random**;  
  
  
**class ArrayInitializer** {  
 **private BufferedReader** bufferedReader;  
  
 ArrayInitializer() {  
 bufferedReader = **new** BufferedReader(  
 **new** InputStreamReader(**System**.***in***)  
 );  
 }  
  
 **int** requestArraySize() {  
 **System**.***out***.print("Enter an array size: ");  
 **int** arraySize = requestInteger();  
  
 **if** (arraySize < 1) {  
 **System**.***out***.println("Array size must be at least 1. Try again.");  
 **return** requestArraySize();  
 }  
  
 **return** arraySize;  
 }  
 **int**[] initializeOneDimensionalArray(**int** length) {  
 **if** (length < 1) {  
 **throw new** IllegalArgumentException("Parameter length must be at least 1.");  
 }  
  
 **System**.***out***.println("Enter values of array: ");  
  
 **int**[] result = **new int**[length];  
 **for** (**int** i = 0; i < result.length; i++) {  
 result[i] = requestInteger();  
 }  
  
 **return** result;  
 }  
 **int**[][] initializeTwoDimensionalArray(**int** length) {  
 **if** (length < 1) {  
 **throw new** IllegalArgumentException("Parameter length must be at least 1.");  
 }  
  
 **Random** random = **new** Random();  
 **int**[][] result = **new int**[length][length];  
  
 **for** (**int** i = 0; i < result[0].length; i++) {  
 **for** (**int** j = 0; j < result[1].length; j++) {  
 result[i][j] = random.nextInt() % 100;  
 }  
 }  
  
 **return** result;  
 }  
  
 **private int** requestInteger() {  
 **int** result = 0;  
  
 **try** {  
 result = **Integer**.*parseInt*(bufferedReader.readLine());  
 } **catch** (**IOException** ioe) {  
 **System**.***out***.println("IO exception has occurred during runtime.");  
 **System**.*exit*(1);  
 } **catch**(**NumberFormatException** nfe) {  
 **System**.***out***.println("Incorrect format of value. Try again.");  
 **return** requestInteger();  
 }  
  
 **return** result;  
 }  
}

***ArrayPrinter.java***

**class ArrayPrinter** {  
 **private** ArrayPrinter() {}  
  
 **static void** printArray(**int**[][] arrayData) {  
 **for** (**int** i = 0; i < arrayData[0].length; i++) {  
 **for** (**int** j = 0; j < arrayData[1].length; j++) {  
 **System**.***out***.print(arrayData[i][j] + "\t");  
 }  
 **System**.***out***.println();  
 }  
  
 **System**.***out***.println();  
 }  
 **static void** printArray(**int**[] arrayData) {  
 **for** (**int** value : arrayData) {  
 **System**.***out***.print(value + "\t");  
 }  
  
 **System**.***out***.println();  
 }  
}