Міністерство освіти і науки України

Івано-Франківський національний технічний університет нафти і газу

*Кафедра програмного забезпечення автоматизованих систем*

**Лабораторна робота №2**

Тема:

**«Знаходження числових характеристик вибірки»**

***Виконав*:**

ст. групи ПіТ-15-3

Свирид О.Б.

***Перевірив:***

Незамай Б.С.

Івано-Франківськ

2015

**Завдання:** знайти числові характеристики вибірки, такі як : середнє арифметичне, медіана, мода.

**Хід роботи**

Обсяг вибірки N – підібрані тестові дані.

Код програми на мові Java:

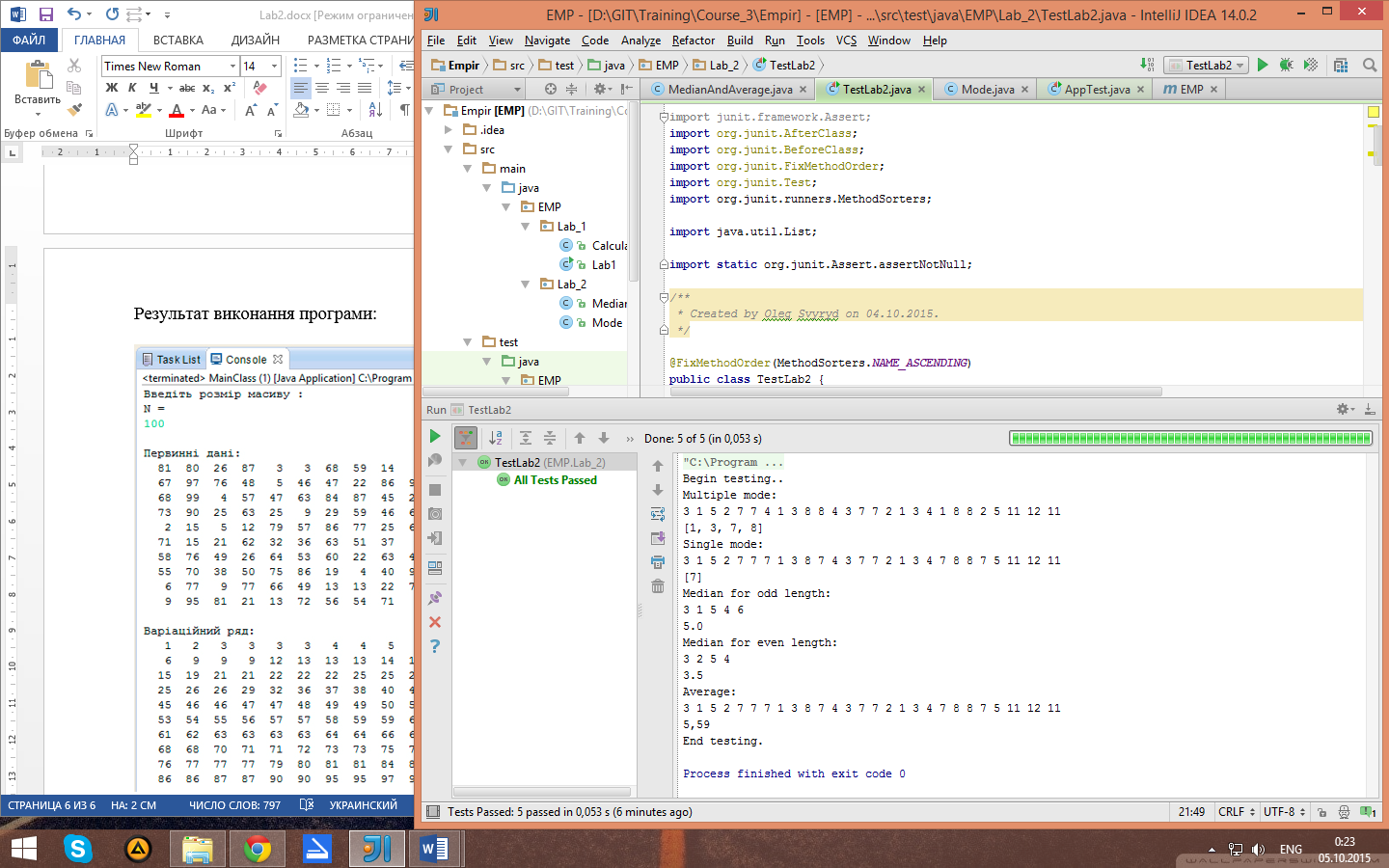
**public class** Mode {  
 */\*\*  
 \* Array frequencies of init array  
 \** ***@param array*** *Row of numbers(array)  
 \*/* **private static** Map<Integer, Integer> getFrequencies(**int**[] array) {  
 *//create map that contains modes* Map<Integer, Integer> modes = **new** HashMap<Integer, Integer>();  
  
 **for**(**int** i = 0; i < array.**length**; i++) {  
  
 *//get from map value from unsorted array* Integer value = modes.get(array[i]);  
  
 *//if value isn't exist* **if**(value == **null**) {  
  
 *//put instead value 0* modes.put(array[i], 0);  
 }  
  
 *//else if value is exist* **else** {  
  
 *//get amount of this values* **int** count = value.intValue();  
  
 *//increment amount* count++;  
  
 *//put new incremented value* modes.put(array[i], count);  
 }  
 }  
 **return** modes;  
 }  
  
 */\*\*  
 \* Get maximum frequencies or frequency  
 \** ***@param setValues*** *Set of frequencies  
 \** ***@return*** *Max frequencies or frequency  
 \*/* **private static int** getMaxFrequency(Collection<Integer> setValues) {  
  
 *//set max element* **int** max = 0;  
  
 *//flag if value is first in collection* **boolean** isFirstEnter = **false**;  
  
 *//iterating collection of frequencies* **for**(Iterator<Integer> iterator = setValues.iterator(); iterator.hasNext();) {  
  
 *//get concrete value* Integer value = iterator.next();  
  
 *//check if iteration is first* **if**(!isFirstEnter) {  
  
 *//set max value first value from collection* max = value;  
  
 *//flag that was first iteration* isFirstEnter = **true**;  
 }  
  
 *//check if max value not largest that previous value* **if**(max < value) {  
  
 *//set max value to actual value* max = value;  
 }  
 }  
  
 *//return max value or values* **return** max;  
 }  
  
 */\*\*  
 \* Return modes of array  
 \** ***@param array*** *Row of numbers(array)  
 \*/* **public** List<Integer> getMode(**int**[] array) {  
 *//map with frequencies* Map<Integer, Integer> modes = **new** HashMap<Integer, Integer>();  
  
 *//get frequencies* modes = *getFrequencies*(array);  
  
 *//get max frequency* **int** maxFreq = *getMaxFrequency*(modes.values());  
  
 *//create list of modes* List<Integer> listOfModes = **new** ArrayList<Integer>();  
  
 *//iterate frequencies* **for**(Map.Entry<Integer, Integer> entry : modes.entrySet()) {  
  
 *//set actual frequency* Integer actual = entry.getValue();  
  
 *//check if actual equals max frequency* **if**(actual == maxFreq) {  
  
 *//add mode into list* listOfModes.add(entry.getKey());  
 }  
 }  
  
 *//return list of modes* **return** listOfModes;  
 }  
}

**public class** MedianAndAverage {  
  
 */\*\*  
 \* Return median for even arrays  
 \** ***@param array*** *Init array of digits  
 \** ***@return*** *median  
 \*/* **private static float** getMedianForEvenArrays(**int**[] array) {  
 **return** ((**float**) (array[((array.**length** / 2) - 1)] + array[((array.**length** / 2))]) / 2);  
 }  
  
 */\*\*  
 \* Return median for odd arrays  
 \** ***@param array*** *Init array of digits  
 \** ***@return*** *median  
 \*/* **private static int** getMedianForOddArrays(**int**[] array) {  
 **return** (array[((array.**length** / 2))]);  
 }  
  
 */\*\*  
 \* Check length of array and return median  
 \** ***@param array*** *Init array of digits  
 \** ***@return*** *median  
 \*/* **public float** getMedian(**int**[] array) {  
 **return** ( (array.**length** % 2) == 0) ? *getMedianForEvenArrays*(array) : *getMedianForOddArrays*(array);  
 }  
  
 */\*\*  
 \* Return average of array  
 \** ***@param array*** *Init array of digits  
 \** ***@return*** *average  
 \*/* **public float** getAverage(**int**[] array) {  
 **float** sum = 0;  
 **for**(**int** i : array) {  
 sum += i;  
 }  
 **return** sum / array.**length**;  
 }  
}

Юніт тест:

@FixMethodOrder(MethodSorters.***NAME\_ASCENDING***)  
**public class** TestLab2 {  
  
 **private static** Mode *actualMode* = **new** Mode();  
 **private static** MedianAndAverage *actualMedianAndAverage* = **new** MedianAndAverage();  
  
 **private static final int**[] ***testMode1*** = **new int**[] { 3, 1, 5, 2, 7, 7, 4, 1, 3, 8, 8, 4, 3, 7, 7, 2, 1, 3, 4, 1, 8, 8, 2, 5, 11, 12, 11};  
 **private static final int**[] ***testMode2*** = **new int**[] { 3, 1, 5, 2, 7, 7, 7, 1, 3, 8, 7, 4, 3, 7, 7, 2, 1, 3, 4, 7, 8, 8, 7, 5, 11, 12, 11};  
 **private static final int**[] ***testOddMedian*** = **new int**[] { 3, 1, 5, 4, 6};  
 **private static final int**[] ***testEvenMedian*** = **new int**[] { 3, 2, 5, 4};  
  
 @BeforeClass  
 **public static void** runBeforeClass() {  
 System.***out***.println(**"Begin testing.."**);  
 }  
  
 @AfterClass  
 **public static void** runAfterClass() {  
 *actualMode* = **null**;  
 *actualMedianAndAverage* = **null**;  
 System.***out***.println(**"End testing."**);  
 }  
  
 @Test  
 **public void** test01MultipleModes() {  
 List<Integer> modes = *actualMode*.getMode(***testMode1***);  
  
 System.***out***.println(**"Multiple mode: "**);  
 *printTestArray*(***testMode1***);  
 System.***out***.println(modes);  
  
 *assertNotNull*(modes);  
 }  
  
 @Test  
 **public void** test02SingleMode() {  
 List<Integer> modes = *actualMode*.getMode(***testMode2***);  
  
 System.***out***.println(**"Single mode: "**);  
 *printTestArray*(***testMode2***);  
 System.***out***.println(modes);  
  
 *assertNotNull*(modes);  
 }  
  
 @Test  
 **public void** test03OddMedian() {  
 **float** median = *actualMedianAndAverage*.getMedian(***testOddMedian***);  
 System.***out***.println(**"Median for odd length: "**);  
 *printTestArray*(***testOddMedian***);  
 System.***out***.println(median);  
  
 *assertNotNull*(median);  
 }  
  
 @Test  
 **public void** test04EvenMedian() {  
 **float** median = *actualMedianAndAverage*.getMedian(***testEvenMedian***);  
 System.***out***.println(**"Median for even length: "**);  
 *printTestArray*(***testEvenMedian***);  
 System.***out***.println(median);  
  
 *assertNotNull*(median);  
 }  
  
 @Test  
 **public void** test05Average() {  
 **float** average = *actualMedianAndAverage*.getAverage(***testMode2***);  
  
 System.***out***.println(**"Average: "**);  
 *printTestArray*(***testMode2***);  
 System.***out***.format(**"%2.2f\n"**, average);  
  
 *assertNotNull*(average);  
 }  
  
 **private static void** printTestArray(**int**[] arr) {  
 **for**(**int** i : arr) {  
 System.***out***.print(i + **" "**);  
 }  
 System.***out***.println();  
 }  
}

Результат виконання програми:



**Висновок:** в ході лабораторної роботи я реалізував програму на мові Java яка вираховує деякі чисельні характеристики вибірки ( середнє арифметичне, моду і медіану ).