13.04.22 Vid\_2

Задание: программа для подсчета количества элементов !=0

Согласно методологии TDD:

* Создаём прототип класса, описывающего бизнес- логику:

package itStep.yandr.javaLessons.lesson29.task01a.model.logic;  
  
public class ArrayLogic {  
 public static int countNonZeroElements(int[]array){  
 int count = 0;   
 return count;  
 }   
}

* Создаем тесты:

package itStep.yandr.javaLessons.lesson29.task01a.model.logic;  
  
import org.junit.Test;  
  
  
import static org.junit.Assert.\*;  
  
public class ArrayLogicTest {

Позитивные сценарии:

Базовый  
 @Test   
 public void testCountNonZeroElementsBasic() {  
 int[] array = {2, 3, -4, 0, 5, 0, 6, 0, -22};  
 int expected = 6;  
 int actual = ArrayLogic.*countNonZeroElements*(array);  
 *assertEquals*(expected, actual);  
 }

Без элементов!= 0  
 @Test  
 public void testCountNonZeroElementsWithoutZeroElement() {  
 int[] array = {2, 3, -4, 5, 6, -22};  
 int expected = array.length;  
 int actual = ArrayLogic.*countNonZeroElements*(array);  
 *assertEquals*(expected, actual);  
 }

Все элементы = 0  
 @Test  
 public void testCountNonZeroElementsWithAllZeroElements(){  
 int[]array = {0,0,0,0,0,0,0};  
 int expected=0;  
 int actual = ArrayLogic.*countNonZeroElements*(array);  
 *assertEquals*(expected,actual);  
 }

Негативные сценарии:

Null  
 @Test  
 public void testCountNonZeroElementsWithNull(){  
 int expected = -1;  
 int actual = ArrayLogic.*countNonZeroElements*(null);  
 *assertEquals*(expected,actual);  
 }

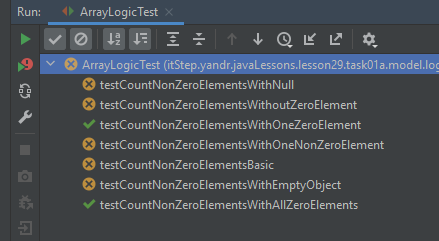
Создание массива как анонимного объекта

Или int[]array={};  
 Пустой массив int[]array=new int[0];  
 @Test  
 public void testCountNonZeroElementsWithEmptyObject(){  
 int expected=-1;  
 int actual = ArrayLogic.*countNonZeroElements*(new int[0]);  
 *assertEquals*(expected,actual);  
 }

Граничные значения (граница между позитивным и негативным сценариями, с массивами, чаще всего 1 элемент или 2, если это минимальное количество элементов необходимое для реализации бизнес-логики):  
 1 элемент==0  
 @Test  
 public void testCountNonZeroElementsWithOneZeroElement(){  
 int[]array = {0};  
 int expected= 0;  
 int actual = ArrayLogic.*countNonZeroElements*(array);  
 *assertEquals*(expected,actual);  
 }

1 элемент!=0  
 @Test  
 public void testCountNonZeroElementsWithOneNonZeroElement(){  
 int[]array = {2};  
 int expected= 1;  
 int actual = ArrayLogic.*countNonZeroElements*(array);  
 *assertEquals*(expected,actual);  
 }  
}

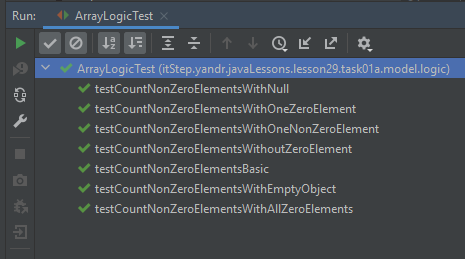
Тесты заведомо провалятся, кроме тех, ожидаемое значение которых ==0 (значение возвращаемое прототипом).



* реализовываем бизнес- логику:

package itStep.yandr.javaLessons.lesson29.task01a.model.logic;  
  
public class ArrayLogic {  
 public static int countNonZeroElements(int[] array) {  
 if (array == null || array.length == 0) {  
 return -1;  
 }  
 int count = 0;  
 for (int element : array) {  
 if (element != 0) {  
 count++;  
 }  
 }  
 return count;  
 }  
}

* повторно запускаем тесты:



* Создаем сервисный класс по наполнению массива значениями :пользовательский ввод и рандомный: C:\Users\Юлия\ITStepJavaClassroomPractice

package itStep.yandr.javaLessons.lesson29.task01a.util;  
  
import java.util.Random;  
import java.util.Scanner;  
  
public class ArrayInitializer {  
 private static final Scanner *SCN* ;  
 private static final Random *RND* ;  
 static {  
 *SCN* = new Scanner(System.*in*);  
 }  
 static {  
 *RND* = new Random();  
 }  
  
 public static void randomInit(int[] array, int min, int max) {  
 for (int i = 0; i < array.length; i++) {  
 array[i] = *RND*.nextInt(max - min + 1) + min;  
 }  
 }  
  
 public static void userInit(int[] array, String msg) {  
 for (int i = 0; i < array.length; i++) {  
 System.*out*.printf("Input value %d: ",i);  
 array[i] = *SCN*.nextInt();  
 }  
 }  
  
 public static int inputIntValue(String msg) {  
 System.*out*.println(msg);  
 return *SCN*.nextInt();  
 }  
  
}

* Создаем контроллер
* package itStep.yandr.javaLessons.lesson29.task01a.controller;  
    
  import itStep.yandr.javaLessons.lesson29.task01a.model.logic.ArrayLogic;  
  import itStep.yandr.javaLessons.lesson29.task01a.util.ArrayInitializer;  
  import itStep.yandr.javaLessons.view.Printer;  
    
  import java.util.Arrays;  
    
  public class Main {  
   public static void main(String[] args) {  
   int size = ArrayInitializer.*inputIntValue*("Input size of array: ");  
   int []array = new int[size];  
   int min = ArrayInitializer.*inputIntValue*("Input min: ");  
   int max = ArrayInitializer.*inputIntValue*("Input max: ");  
    
   ArrayInitializer.*randomInit*(array,min,max);  
    
   int count = ArrayLogic.*countNonZeroElements*(array);  
   String msg = String.*format*("\nCount of non-zero array's elements = %d",count);  
   Printer.*print*("Array"+ Arrays.*toString*(array));  
   Printer.*print*(msg);  
    
   }  
  }

Vid\_2

Разбор одного из заданий Stage 14 + работа JUnit

Level F

2. Отсортировать элементы массива по убыванию до первого/последнего элемента , равного

-разрабатываем 2 метода которые будут искать индекс первого/последнего элементов , равных 0

package itStep.yandr.javaLessons.lesson29.task02f.model.logic;  
  
public class ArraySorter {  
 private static int findFirstZeroElementIndex(int[] array) {  
 if (array == null || array.length == 0) {  
 return -2;  
 }  
 for (int i = 0; i < array.length; i++) {  
 if (array[i] == 0) {  
 return i;  
 }  
 }  
 return -1;  
 }  
  
 private static int findLastZeroElementIndex(int[] array) {  
 if (array == null || array.length == 0) {  
 return -2;  
 }  
 for (int i = array.length - 1; i >= 0; i--) {  
 if (array[i] == 0) {  
 return i;  
 }  
 }  
 return -1;  
 }  
}

-разрабатываем 2 метода которые будут сортировать массив до соответствующего индекса

(Bubble sorted)

I < index – 1

index

public static void sortBeforeFirstZeroElement(int[] array) {  
 int index = *findFirstZeroElementIndex*(array);  
 if (index > 1) {  
 for (int i = 0; i < index -1; i++) {  
 for (int j = 0; j < index -1 - i; j++) {  
 if (array[j] < array[j + 1]) {  
 int temp = array[j];  
 array[j] = array[j + 1];  
 array[j + 1] = temp;  
 }  
  
 }  
 }  
 }  
}  
  
public static void sortBeforeLastZeroElement(int[] array) {  
 int index = *findLastZeroElementIndex*(array);  
 if (index > 1) {  
 for (int i = 0; i < index -1 ; i++) {  
 for (int j = 0; j < index -1 - i; j++) {  
 if (array[j] > array[j + 1]) {  
 int temp = array[j];  
 array[j] = array[j + 1];  
 array[j + 1] = temp;  
 }  
  
 }  
 }  
 }  
}

Но тк логика сортировки повторяется, выносим ее в отдельный метод …

package itStep.yandr.javaLessons.lesson29.task01f.model.logic;  
  
public class ArraySorter {  
 private static int findFirstZeroElementIndex(int[] array) {  
 if (array == null || array.length == 0) {  
 return -2;  
 }  
 for (int i = 0; i < array.length; i++) {  
 if (array[i] == 0) {  
 return i;  
 }  
 }  
 return -1;  
 }  
  
 private static int findLastZeroElementIndex(int[] array) {  
 if (array == null || array.length == 0) {  
 return -2;  
 }  
 for (int i = array.length - 1; i >= 0; i--) {  
 if (array[i] == 0) {  
 return i;  
 }  
 }  
 return -1;  
 }  
  
 private static void sortBeforeIndex(int[] array, int index) {  
 if (index > 1) {  
 for (int i = 0; i < index - 1; i++) {  
 for (int j = 0; j < index – 1 - i; j++) {  
 if (array[j] > array[j + 1]) {  
 int temp = array[j];  
 array[j] = array[j + 1];  
 array[j + 1] = temp;  
 }  
 }  
 }  
 }  
 }  
  
 public static void sortBeforeLastZeroElement(int[] array) {  
 int index = *findLastZeroElementIndex*(array);  
 *sortBeforeIndex*(array, index);  
 }  
  
 public static void sortBeforeFirstZeroElement(int[] array) {  
 int index = *findFirstZeroElementIndex*(array);  
 *sortBeforeIndex*(array, index);  
 }  
}

* Создаем тесты

package itStep.yandr.javaLessons.lesson29.task02f.model.logic;

import static org.junit.Assert.\*;  
import org.junit.Test;  
  
  
public class ArraySorterTest {  
 private final int[] *ARRAY* = {5, 9, 2, 4, 0, 7, 5, 1, 0, 8};  
  
 @Test  
 public void testSortBeforeFirstZeroElement() {  
 int[] expected = {2, 4, 5, 9, 0, 7, 5, 1, 0, 8};  
  
 ArraySorter.*sortBeforeFirstZeroElement*(*ARRAY*);  
 for (int i = 0; i < *ARRAY*.length; i++) {  
 if (*ARRAY*[i] != expected[i]) {  
 *fail*();  
 }  
 }  
 }  
  
 @Test  
 public void testSortBeforeLastZeroElement() {  
 int[] expected = {0, 1, 2, 4, 5, 5, 7, 9, 0, 8};  
  
 ArraySorter.*sortBeforeLastZeroElement*(*ARRAY*);  
 for (int i = 0; i < *ARRAY*.length; i++) {  
 if (*ARRAY*[i] != expected[i]) {  
 *fail*();  
 }  
 }  
 }  
  
 @Test  
 public void testSortBeforeLastZeroElementWithoutZeroElements() {  
 int[] array = {5, 9, 2, 4, 7, 5, 1, 8};  
 int[] expected = {5, 9, 2, 4, 7, 5, 1, 8};  
 ArraySorter.*sortBeforeLastZeroElement*(array);  
 *assertArrayEquals*(expected, array);  
 }  
  
 @Test  
 public void testSortBeforeFirstZeroElementWithoutZeroElements() {  
 int[] array = {5 , 9, 2, 4, 7, 5, 1, 8};  
 int[] expected = {5, 9, 2, 4, 7, 5, 1, 8};  
 ArraySorter.*sortBeforeFirstZeroElement*(array);  
 *assertArrayEquals*(expected, array);  
 }  
}