**Lab #9**

Josue Ponce

Montgomery College

4/11/2018

Author’s Note

This lab report was prepared for CMSC 203 CRN #30672, taught by professor Ahmed Tarek

**Table of Contents**

Task #1 Code……………………………………………………………………………………2

Task #2 Code……………………………………………………………………………………3

Task #2 Screenshot of Successful Test Run and Sample Output……………………………….5

Copy of Gradebook.Java Code………………………………………………………………….6

**Task #1 Code**

// Added toString() method.

**public** String **toString**()

{

String result = "";

**for**(**int** i = **0**; i < scoresSize; i++)

{

result += scores[i] + " ";

}

System.out.println(result);

**return** result;

}

//Added getScoreSize

**public** **int** getScoreSize()

{

**return** scoresSize;

}

**Task #2 Code**

**import** **org.junit.After**;

**import** **org.junit.Before**;

**import** **org.junit.Test**;

**import** **static** org.junit.Assert.\*;

/\*\*

\* Junit test cases for lab #9. Added g3 gradebook object for fun and practice.

\* @author Josue Ponce

\* @version 1.0

\*

\*/

**public** **class** **GradebookTester**

{

**private** GradeBook g1;

**private** GradeBook g2;

//for fun

**private** GradeBook g3;

**@Before**

**public** **void** **setUp**()

{

// create two objects of GradeBook of size 5

g1 = **new** GradeBook(**5**);

g2 = **new** GradeBook(**5**);

g3=**new** GradeBook(**5**);

// Call the addScore method for each of the GradeBook objects

g1.addScore(**95.0**);

g1.addScore(**88.0**);

g1.addScore(**92.0**);

g2.addScore(**30.0**);

g2.addScore(**10.0**);

g2.addScore(**40.0**);

g2.addScore(**20.0**);

//for fun

g3.addScore(**50.0**);

g3.addScore(**75.0**);

}

**@After**

**public** **void** **tearDown**()

{

// set the two objects of GradeBook to null

g1 = **null**;

g2 = **null**;

//for fun

g3 = **null**;

}

**@Test**

**public** **void** **addScoreTest**()

{

// Use the toString method to compare the contents of what is in the scores array vs. what is expected to be in the scores array

assertTrue(g1.toString().equals("95.0 88.0 92.0 "));

assertTrue((g2.toString()).equals("30.0 10.0 40.0 20.0 "));

assertTrue((g3.toString()).equals("50.0 75.0 "));

//assertTrue((g3.toString()).equals("50.0 75.0 "));

// Compare the scoreSize to the expected number of scores entered.

assertEquals(**3**, g1.getScoreSize(), **0.001**);

assertEquals(**4**, g2.getScoreSize(), **0.001**);

assertEquals(**2**, g3.getScoreSize(), **0.001**);

}

**@Test**

**public** **void** **sumTest**()

{

// Compare what is returned by sum() to the expected sum of the scores entered.

assertEquals(**275**, g1.sum(), **0.0001**);

assertEquals(**100.0**, g2.sum(), **0.0001**);

assertEquals(**125.0**, g3.sum(), **0.0001**);

}

**@Test**

**public** **void** **minimumTest**()

{

// Compare what is returned by minimum() to the expected minimum of the scores entered.

assertEquals(**88.0**, g1.minimum(), **0.001**);

assertEquals(**10.0**, g2.minimum(), **0.001**);

assertEquals(**50.0**, g3.minimum(), **0.001**);

}

**@Test**

**public** **void** **finalScoreTest**()

{

// Compare what is returned by finalScore() to the expected finalScore of the scores entered.

assertEquals(**187.0**, g1.finalScore(), **0.001**);

assertEquals(**90.0**, g2.finalScore(), **0.001**);

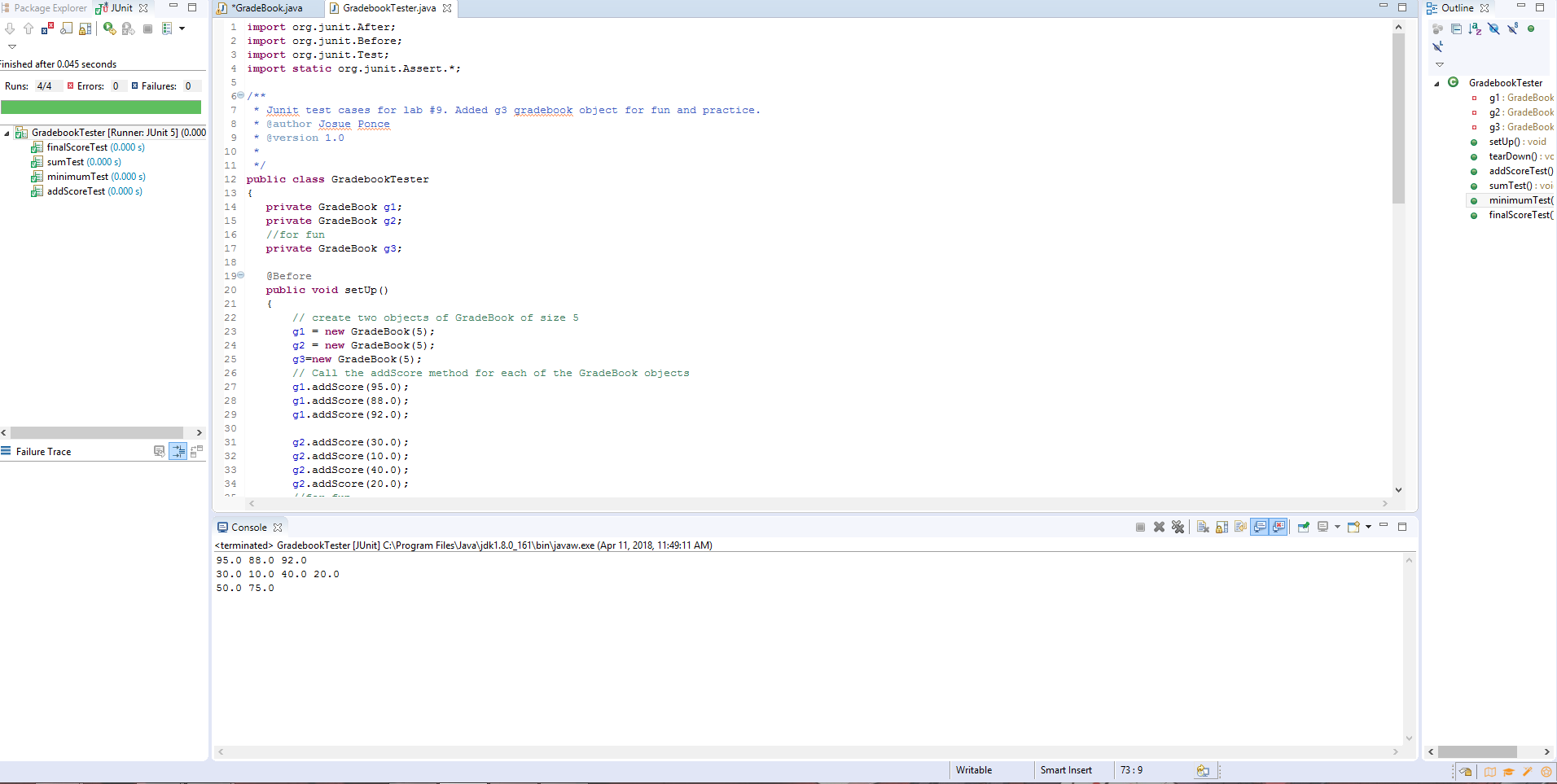
assertEquals(**75.0**, g3.finalScore(), **0.001**);

}

}

**Task #2 Screenshot of Successful Test Run and Sample Output**

After creating the test cases in Junit Test, the program was tested and the results came back green which ensures that the implementation of the new methods in gradebook.java worked as intended. It also shows that the test cases were implemented correctly which is an essential in test driven development. The test cases returned positive results which means there are no errors in the test cases or the program itself. Everything works as intended. The sample output as well as the screenshot of the results can be found down below.



**Copy of Gradebook.Java Code**

**import** **java.util.ArrayList**;

/\*\*

\* Lab #9 program that will be tested using Junit test.

\* @author Josue Ponce

\* @version 1.0

\*/

**public** **class** **GradeBook**

{

**private** **double**[] scores;

**private** **int** scoresSize;

/\*\*

Constructs a gradebook with no scores and a given capacity.

@capacity the maximum number of scores in this gradebook

\*/

**public** **GradeBook**(**int** capacity)

{

scores = **new** **double**[capacity];

scoresSize = **0**;

}

/\*\*

Adds a score to this gradebook.

@param score the score to add

@return true if the score was added, false if the gradebook is full

\*/

**public** **boolean** **addScore**(**double** score)

{

**if** (scoresSize < scores.length)

{

scores[scoresSize] = score;

scoresSize++;

**return** **true**;

}

**else**

**return** **false**;

}

//Added getScoreSize

**public** **int** **getScoreSize**()

{

**return** scoresSize;

}

/\*\*

Computes the sum of the scores in this gradebook.

@return the sum of the scores

\*/

**public** **double** **sum**()

{

**double** total = **0**;

**for** (**int** i = **0**; i < scoresSize; i++)

{

total = total + scores[i];

}

**return** total;

}

/\*\*

Gets the minimum score in this gradebook.

@return the minimum score, or 0 if there are no scores.

\*/

**public** **double** **minimum**()

{

**if** (scoresSize == **0**) **return** **0**;

**double** smallest = scores[**0**];

**for** (**int** i = **1**; i < scoresSize; i++)

{

**if** (scores[i] < smallest)

{

smallest = scores[i];

}

}

**return** smallest;

}

/\*\*

Gets the final score for this gradebook.

@return the sum of the scores, with the lowest score dropped if

there are at least two scores, or 0 if there are no scores.

\*/

**public** **double** **finalScore**()

{

**if** (scoresSize == **0**)

**return** **0**;

**else** **if** (scoresSize == **1**)

**return** scores[**0**];

**else**

**return** **sum**() - minimum();

}

// Added toString() method.

**public** String **toString**()

{

String result = "";

**for**(**int** i = **0**; i < scoresSize; i++)

{

result += scores[i] + " ";

}

System.out.println(result);

**return** result;

}

}