What I Learned:

I’m already quite familiar with JUnit tests, but something new I learned was creating one from scratch. I usually only have to import JUnit tests for projects. I now know what to do to have the Eclipse IDE populate test stubs for the methods of a class.

Issues I Encountered:

When adding scores to the gradebook, I added less than the gradebook could hold (5). My addScore test kept failing, and I didn’t know why. I was in the process of creating a “throwaway” class to find out what was happening, but before I could, I remembered that an array of type double will automatically populate default values of 0.0. So, instead of displaying “70.0 90.0 ”, it was displaying “70.0 90.0 0.0 0.0 0.0 ”. This was easy to fix by changing the condition in the for loop to i < scoresSize instead of scores.length. This taught me how important it is to create a wide range of test cases. Had I only added scores until the gradebook was full, I wouldn’t have realized the flaw in my code.

What I Would’ve Done Differently:

I would’ve been more thoughtful about the data I’m working with. If I had established from the beginning that the array is already filled with 0.0’s once instantiated, I would’ve anticipated that the toString method wouldn’t work if it iterated through the whole loop.

How I’ll Apply JUnit Tests in the Future:

I have no choice but to utilize them for projects, which is fine. They’re a useful way to figure out errors in code. As the textbook instructs, we should create core methods and test them before continuing the rest of the program. Isolating what needs to be tested helps pinpoint where errors are.

