Mason K Utt

Professor Kalysa Wilson

CS320

December 7, 2022

Project Two

**Summary:**

1. Describe your unit testing approach for each of the three features.

My unit testing approach for each of the three features was to encompass all of the requirements with at least one test. I made sure to test all the failing conditions in order to establish a passing condition. For example, all three features test the length of string input and will not allow too long or too short of strings. My overall quality of my Junit test is very high as I have a coverage percentage of 100% on all three of my features.

1. Describe your experience writing Junit tests.

My experience with Junit tests was overall a good time. I had worked with unit tests with python, c, and a few other languages before. I knew how to write tests prior to this class but now I know more about why. I ensured that my code was technically sound and that line 50 in TaskServiceTest demonstrates that. It tests the ability to add multiple Task class objects to a list. If the Task has a unique ID it allows it to add, but fails in the correct way if the user tries to add an ID that already exists. It tests that the values at the locations are the correct values also. Line 48 of TaskTest shows how my tests are efficient. It is able to see if the setTask function works as intended. It is able to test all three scenarios, a pass case, and the two failing cases in one function.

**Reflection:**

1. Testing Techniques

I employed static testing for each of the milestones. To do this I would break down each of the requirements and make a test for each of them. I used a Top-down technique for all of the milestones so far. In this technique you test components that call other components. I did this by testing to see if the constructor worked which calls several functions lower. Other techniques include Bing-bang integration and Bottom-up integration. Bing-bang integration tests everything all at once. This can be a poor choice as it introduces risk as you do not know where the problem is. Bottom-up integration is similar to Top-down but is the opposite. The lowest functions are tested first then the higher tests, this is the most thorough tests but are usually the most complex and time consuming. Each of these techniques have their own uses for different projects and situations. The benefit of Bing-bang integration is that all of the functions are written before testing starts. I would use this to go back and test code I had already written. The benefit of Top-down is that it is simple and non-data intensive and still captures all the tests. I would use this approach on code I have written before and am fairly confident in, especially if errors are not devastating. The benefit of Bottom-up testing is that it fully tests every possibility. I would use this in projects that need high levels of security and need to not have bad data passed.

1. Mindset

The mindset I took on as a tester was to break everything, I tried to think of how I like to play some videogames. For example, I love to go to the edge of the world and see if I cannot skip ahead of where I am supposed to be, find all the glitches and what not. I did not use caution I tried to do everything I could to break my code and then I would fix it. A good way to limit bias in your code is to pretend like it was someone else who wrote the code. This was a problem I had in English class, I would write essays and not notice the grammatical errors because I knew what it was supposed to say. To solve this my teacher recommended revising a couple days after writing the rough draft. In the case of programming writing tests days after the initial code was written may help. Discipline is extremely important when it comes to software from a business perspective and from a moral perspective. If testers do not test the quality of code, they are distributing low quality code. This is a bad business plan as if you become notorious for poor code quality no one will want to purchase your services. This is bad from a moral aspect as if the code quality is poor security aspects come into question, if your code accesses a database and allows xml or sql injections other people’s data may be at risk.

Doc:

[*Software Testing: An ISTQB-BCS Certified Tester Foundation Guide opens in new window*](https://ezproxy.snhu.edu/login?url=https://ebookcentral.proquest.com/lib/snhu-ebooks/detail.action?docID=5837074), Chapter 2

*Top-down vs bottom-up integration testing - javatpoint*. www.javatpoint.com. (n.d.). Retrieved November 26, 2022, from https://www.javatpoint.com/top-down-vs-bottom-up-integration-testing