CS320 Summary and Reflections Report

Maya C. Neely

Southern New Hampshire University

Table of Contents

[Table of Contents 2](#_Toc117338013)

[1. Summary 3](#_Toc117338014)

[1a. Describe your unit testing approach for each of the three features. 3](#_Toc117338015)

[1b. Describe your experience writing the JUnit tests. 4](#_Toc117338016)

[2. Reflection 4](#_Toc117338017)

[2a. Testing Techniques 4](#_Toc117338018)

[2b. Mindset 5](#_Toc117338019)

[References 6](#_Toc117338020)

CS320 Summary and Reflections Report

# 1. Summary

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

### Contact Service.

#### To what extent was your approach aligned with the software requirements?

In Contact.java, lines 5-31 of my code is where all “Contact Class Requirements” were met. Five strings were created that could not be null or have more than ten digits (besides String phoneNum that had to have exactly ten digits). If a string did not meet these requirements, an “IllegalArgumentException” was thrown and an error message would appear saying “Invalid ‘string’.” In ContactService.java, I put “contacts” into an ArrayList where I was able to add, remove and update contacts based on a unique ID. For example, I used “public boolean remove(String id)” to test if a contact was present. If so, it would return true and the contact could be removed, if not, it would return false.

#### Defend the quality of your JUnit tests.

My Contact JUnit test performed 12 tests in total while my ContactService JUnit only performed 6. I believe this may have played a role in test coverage since my ContactTest performed more tests and had 85.5% coverage while ContactServiceTest performed half as many tests but tested at 100%. In my research, 80% coverage "is a good goal to aim for (Pittet)” which gave me reason to believe, along with all the tests passing, that my tests were effective.

### Task Service.

#### To what extent was your approach aligned with the software requirements?

Task Service was quite similar to Contact Service with a change in Strings. For example, in Task.java a description field was required that could not be null or exceed 50 characters. If so, an “IllegalArgumentException” was thrown and an error message would appear saying “Invalid description.”

#### Defend the quality of your JUnit tests.

The TaskTest.java conducted a total of 7 tests with an 81.8% coverage while the TaskServiceTest.java ran 6 tests with 100% coverage. Likewise with Contact Service, I expected to produce coverage of at least 80% to ensure my tests were effective.

### Appointment Service.

#### To what extent was your approach aligned with the software requirements?

Creating Appointment Service was a little tricky due to having to create a date field that could not be null or could not be in the past. In order to follow these requirements, I used “Date currentDate = new Date();” to initialize a new date and used “.before” to ensure the appointment was before the current date. If not, an “IllegalArgumentException” was thrown and an error message would appear saying “Invalid date” since an appointment can not be made for a date that has already past.

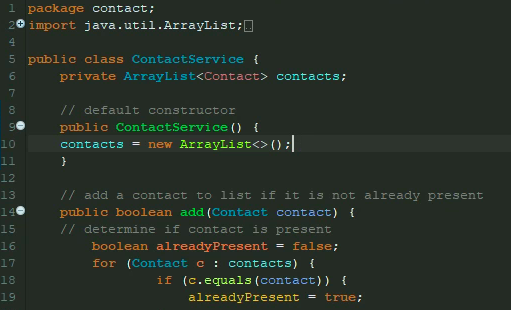
#### Defend the quality of your JUnit tests.

My AppointmentService.java and AppointmentSerciveTest.java both reached 100% coverage while my Appointment.javas only reached 77%. I believe the percentages of AppointmentService are higher since Appointment.java returns and sets strings only necessary for AppointmentService but this code is not used in the AppointmentTest therefore it is skipped during these tests, reducing the coverage percentage. This does not make my code and tests any more or less wrong also being as though all tests conducted passed.

## 1b. Describe your experience writing the JUnit tests.

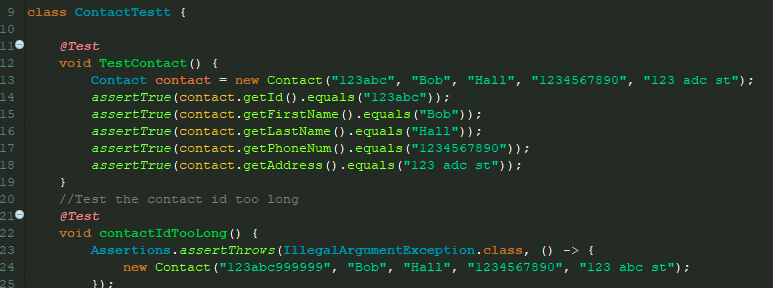
### How did you ensure that your code was technically sound?

I ensured my code was technically sound by including comments, importing and using data structures such as array and creating several tests to test all aspects of the requirements.



### How did you ensure that your test code was efficient?

I ensured my code was efficient with the use of my test cases.



In ContactTest.java, I first created a new contact object with an Id, first and last name, phone number and address. Then, I created five assertTrues which ensures the provided information is equal to the value I assigned. If this initial test failed, I would know that somehow my code is not accurately meeting the requirements and would need to be revised. This test was the first test I created and once it passed, I continued creating tests to test all aspects of the requirements like if a contact ID was too long.

# 2. Reflection

## 2a. Testing Techniques

### What were the software testing techniques that you employed in this project?

The main software testing techniques that I employed throughout the entire project were dynamic testing techniques. I performed JUnit tests which are a form of dynamic testing in which code needed to be written and executed in order to accurately perform my tests. Coverage tests were also performed and are a form of structured based testing that establishes how much code is covered versus how much has been written. Specification based techniques such as equivalence partitioning were used as well. For example, when creating objects such as a phone number I knew as long as the number had exactly ten digits the number would be valid so there was no need to test each ten-digit number.

### What are the other software testing techniques that you did not use for this project?

A software technique I was unable to use were experience based techniques. Since I had no experience creating JUnit tests, preforming tests such as error guessing, and exploratory testing were impracticable and have little to no knowledge of testing strengths and weaknesses.

Techniques opposite of what I have been using that I did not use throughout my project was black box testing. Black box testing is testing software from the users' point of view with the software system being concealed to me as the tester. I would be unable to perform this kind of test since I wrote the code and did not yet develop a UI. Also, without a UI, functional and usability testing cannot be performed. Functional testing tests that the system includes all intended requirements and usability testing tests the UI/UX design, ensuring that it is easy and enjoyable for users to navigate.

**For each technique you discussed, explain their practical uses and implications for different software projects and situations.**

The tests I performed occurred during the testing phase of the SDLC and these tests gave me an idea of how my software would perform once deployed. Static testing techniques such as reviews and static analysis occur before code development and are used to ensure code will be be functional and of upmost quality. Techniques such as black box testing would best be suited for use after the completion of a software to ensure the entire system functions as intended. Functional and usability testing would also be best suited for use after majority of the software is complete, since these tests are performed later in the SDLC. Although testing has its own phase in the SDLC, it occurs in all phases as it is the most important aspect of the SDLC, and is responsible for creating and delivering quality software.

## 2b. Mindset

### Assess the mindset you adopted working on this project.

My mindset, just like in many of my other classes, was to not lose track of the requirements expected of me. Including all necessary requirements of the project was my main priority alongside ensuring I produce functional code of good quality. To do so, I had to employ a substantial amount of caution. This involved a great deal of trial and error, ensuring my tests met the requirements, were functional and over 80% of my code was covered for each test. I had to employ the most caution when creating and testing the appointment class. Specifically, when creating a test “Date” I had to create a test date that would remain valid well after I ran the tests. I employed caution by using a date of 11/11/2024 which would be valid well after the submission of my project. Being able to initialize this test date and call it in other tests made me appreciate the interrelationships within my code.

### Assess the ways you tried to limit bias in your review of the code.

Bias can easily find itself present in a software since code is both developed and tested by its author. This makes it easy for the developer to write tests from their point of view when the test should be designed from the point of view of someone who does not have knowledge of the inner workings of the software. To limit the bias within my code, I used the rubric to assess me with the development of my tests instead of my code, to ensure each requirement was met. If a test were to fail, I would evaluate both the test and my code. Being biased in this situation would lead me to believe the test itself was causing the failure and that my code could not possibly be the reason for the errors. This would prolong both the development and testing phase while also wasting resources.

### Finally, evaluate the importance of being disciplined in your commitment to quality as a software engineering professional.

Self-discipline is a powerful skill that can not only help me throughout my software engineering journey, but in every aspect of my everyday life. Self-discipline is one's ability to manage one's thoughts and emotions to achieve their goals. This keeps me on task of progressing gradually and carefully through the development and testing phases, ensuring my software is of the utmost quality. Cutting corners in hopes of reducing development time can negatively impact the quality of code. Although this method may be easier to implement, if code is not thoroughly tested and reviewed, it may not meet all requirements and functionality. Therefore, it is important to have self-discipline, to remain patient and have the willpower to achieve any goal I have and overcome any obstacle presented to me.

References

Last Name, F. M. (Year). Article Title. *Journal Title*, Pages From - To.

Last Name, F. M. (Year). *Book Title.* City Name: Publisher Name.