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

1. To what extent was your approach **aligned to the software requirements**? Support your claims with specific evidence.

My approach throughout the mobile application creation was aligned with all software requirements that were given to me by the customer. We needed to create a mobile app that would be used to store contact information through JUnit testing of small chunks of code. For the code regarding contact service, I approached it with a basic knowledge of how. We created specific unit testing by creating the object classes that stored the various strings details regarding the contact, such as first and last name. The junit testing helped to ensure that my strings were working properly, and I was able to store information as well as add/delete it. When it came to the task service portion of the project, we needed to ensure that we had character limits for storing tasks that the mobile app would develop. We used the unit testing similarly to what I did for storing contact services, where I tested the various task objects that required my strings to have certain characters numbers. The last part of my junit testing was for the portion of appointment services, where I tested the code for adding/deleting appointments and made sure the testing also applied to the id string being a limited number of characters.

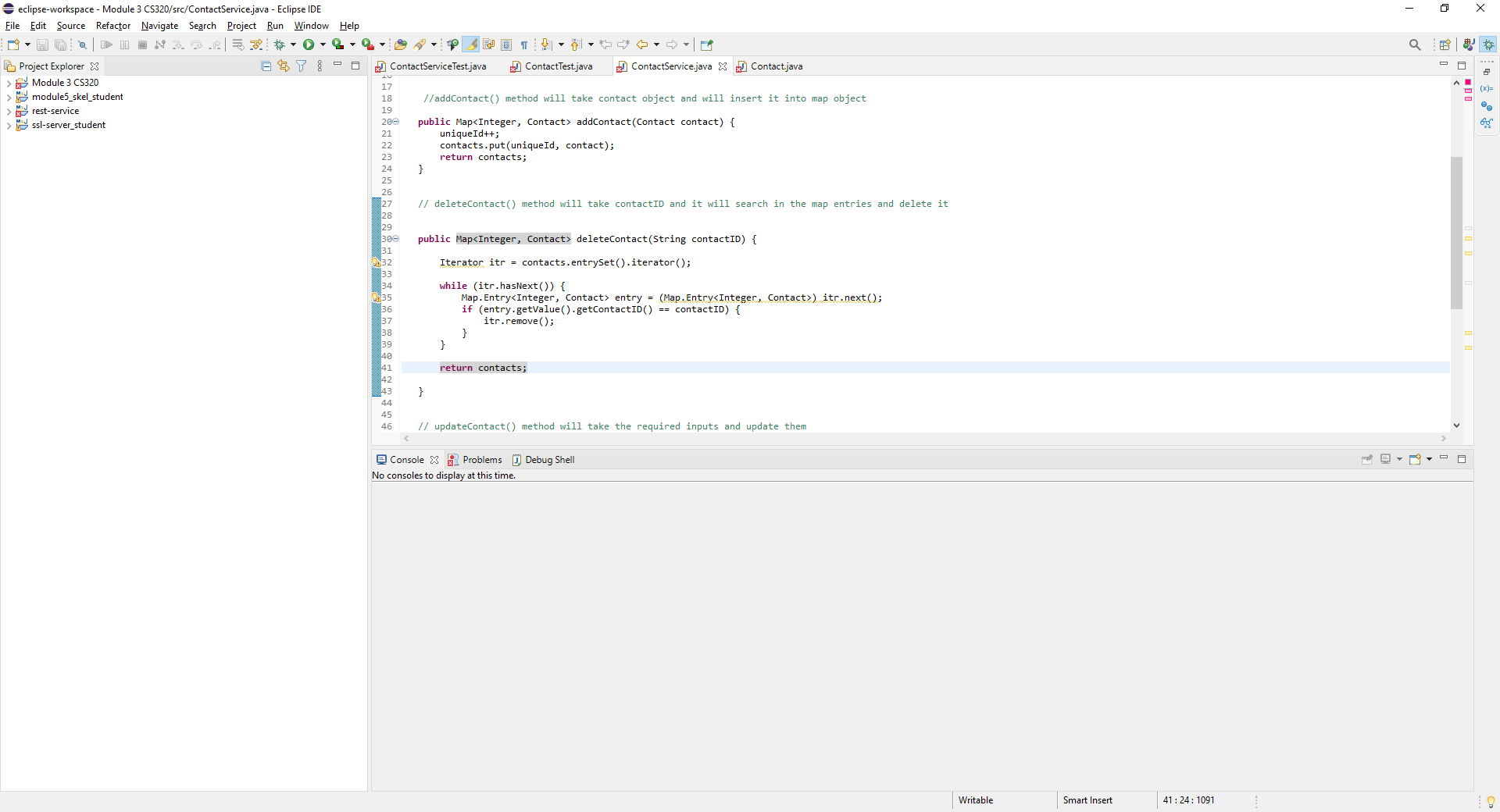
1. Defend the overall quality of your JUnit tests. In other words, how do you know your JUnit tests were **effective** based on the coverage percentage?

The overall quality of my junit testing was good regarding having the execution of my code come out correctly from the successful tested runs. I used a separate application known as cobetura, which helped me to get line coverage as well as branch coverage for my code, showing me that I had successfully had sufficient coverage within my code.

Describe your experience writing the JUnit tests.

1. How did you ensure that your code was **technically sound**? Cite specific lines of code from your tests to illustrate.

I ensured this by applying coding practices and techniques that I have gained through my timing coding and from previous courses in my bachelor's degree program. Technically sound code could be bias and have quite different meanings depending on who writes the code as well as who interprets it. Some of the widespread practice for sound code is the use of in-text citations explain the process, as well as proper indentation of code to make sure it's clean and efficient. Here is a decent example:



We can see from the code above a couple of examples of both efficient and technically sound code. We showed our delete contact string and if/else statement that helped to create the process on the application that deletes contacts within the mobile device.

Testing Techniques

1. What were the **software testing techniques** that you employed in this project? Describe their characteristics using specific details.

There were various software testing techniques that I used throughout the project, each playing a vital role in how the mobile application works. Some of the more common testing applications have to do with unit testing, acceptance testing, as well as integration and system testing. As for specific examples for the unit testing, we could go to the appointment service class for our specific code. We used unit testing to show that we have strings to add/delete appointments with functional code and testing (See code above). We used acceptance testing on the code throughout each of the milestones within the project, running the code to make sure that its functional so it could be delivered to mobile apps for the provider. On the last milestone for appointment testing, we used the black-box method of system testing, running the whole system to make sure we met the requirements laid out by the owner of grand strand systems. I used cobetura that also runs a similar testing apart from me as part of the development team.

1. What are the **other software testing techniques** that you did not use for this project? Describe their characteristics using specific details.

There are two other software testing techniques that I could not use within this certain project, as they are used more with a real time company and software team. The techniques are performance testing and security testing of the mobile application. Performance testing is a non-functional testing that will use stress and endurance testing to make sure that the increase of demand from the application can work under high demand and over an extended period. This is especially important for this software to make sure that it can run under increased traffic to add appointments. The endurance test also helps to show that we can run the software for an extended period without it crashing.

1. For each of the techniques you discussed, explain the **practical uses and implications** for different software development projects and situations.

Some of the practical uses for other software companies could be the use of apples app store or the amazon mobile app. Both applications have been under immense performance testing, as they need to be able to handle traffic at certain times and be able to always run throughout the day with limited maintenance times. They also have increased security testing as they handle immense amounts of consumer data that could be harmful for the customers if it was compromised.

1. Assess the mindset that you adopted working on this project. In acting as a software tester, to what extent did you employ **caution**? Why was it important to appreciate the complexity and interrelationships of the code you were testing? Provide specific examples to illustrate your claims.

I employed caution throughout all aspects of this project, especially since Ive learning to code day by day and I am a work in progress. I was having trouble within the set of milestones as I could not understand why my code could not work but did not really understand the importance of the test class when I began to run the code. In previous courses, I was used to just create the classes or running the code in python you necessarily did not need to run the test class for each public class. It's important to appreciate the complexity and relationships of each milestone, as each milestone helps us understand different part of the whole software and enables us the run separate junit testing for each.

1. Assess the ways you tried to limit **bias** in your review of the code. On the software developer side, can you imagine that bias would be a concern if you were responsible for testing your own code? Provide specific examples to illustrate your claims.

Bias would be a concern in any aspect of real-world applications and coding are no different. Each coder has their own bias when it comes to how they write the code, implement it, or run certain software testing techniques that could have different outcomes or may not find those certain flaws. This could skew outcomes when it comes to building software for consumers, as this could lead to unintended consequences that the company may not want.

1. Finally, evaluate the importance of being **disciplined** in your commitment to quality as a software engineering professional. Why is it important not to cut corners when it comes to writing or testing code? How do you plan to avoid technical debt as a practitioner in the field? Provide specific examples to illustrate your claims.

Being disciplined is an essential part of becoming a well-rounded software engineer, professional, and a quality human being. When you cut corners, you understand less of situations and begin to get used to cutting corners in all aspects of the things you do. When it comes to being an immaculate software engineer, some of the projects you work on will not be able to have corners cut and nor should they. You have a commitment as a professional to bring the work and abilities to a project to ensure that it comes out with all requirements met.