Project Two

Dennis Renfro

**Summary:**

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

**specific evidence.**

My testing approach was aligned to the software requirements as the Junit tests tested the methods that

were implemented in the code. An example of this is the contact class stated the first name and last

names cannot be longer than 10 characters. The ContactTest Junit test had to implement this through

assertions to test that the first name or the last name had too many characters.

**Assertions.assertThrows(IllegalArgumentException.class,()->(new Contact(**

**“12”,”David”,”Brown”,”1234567890”,”327 Oak St”).**

The task class stated the task Id could not be longer than 10 characters. The TastTest Junit test had to

implement this through the method,

**void test TaskIdTooLong() in order to throw an illegal argument if the task was greater than 10**

**characters.**

**Defend the overall quality of your Junit tests. In other words, how do you know your Junit test were**

**effective based on the coverage percentage?**

When I started learning about Junit testing and developed my first test, I was new but interesting. As

time progressed and with practice, I could see how the quality of the tests was improving. To have an

effective test, the methods have to be implemented in the test cases. The coverage percentage was

lower in the task test than the contact test but I could see improvements in the coverage percentages in

both tests. The coverage percentages showed me how close the tests were aligned to the functions I had

written and I realize the importance of Junit testing for ensuring code correctness. For instance, for the

client class, the test inputted a contact ID, first and last name, phone number, and address. This was to

ensure that no errors were populated unexpectedly and confirmed that the getter methods for each

variable worked correctly. Next was the test of each variable with an input that did not meet the criterial

such as a phone number longer than 10 digits in order to set up an error to populate. If an error was not

thrown, the test would fail.

**Describe you experience with Junit tests.**

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

I haven’t had any prior experience working with Junit tests. This was my first interaction with it.

This is what I did to ensure my code was technically sound. I created variables that met the

requirements

of the Junit tests and I only had to call the variable instead of writing the value for each test. This allows

for more rapid changes and updates. I added comments to improve readability of the code. For example,

in the Task Test class

**“void testTask(){//Test that variables that meet the requirements don’t throw errors.”**

This helped to ensure I was able to quickly read over the code to make sure everything was working

properly. I tested the code through Junit testing and static testing.

**How did you ensure that your code was efficient? Cite specific lines of code from your tests to**

**illustrate.**

I ensured my code was efficient I condensed the code combining two if statements linked by an or

symbol for readability such as this:

Instead of if(contactId==null{

if{contactId.length>10{

throw IllegalArgumentException

I combined them,

If (contactId==null||contact.length>10) {

throw new IllegalArgumentException

**Reflection**

**Testing Techniques**

**What were the software testing techniques that you employed in this project? Describe their**

**characteristics in specific details.**

I used two main software testing techniques in this project. The first one was static testing. Static testing

involves checking the code without actually running the code. I checked the code over and over, line by

line for completeness and correctness. I checked to ensure the code was well commented and had good

readability. The second one was Junit testing where I created a separate test file for Junit testing. I wrote

tests that would implement methods from a class in order to get the expected output.

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

**characteristics using specific details.**

There are other software testing techniques that I did not use for this project. They are security testing,

acceptance testing, functional and non-functional testing, system testing, integration testing and

performance testing. Security testing tests the software to identify vulnerabilities and ensure it meets

security requirements. Acceptance testing tests the software to ensure it meets the customer’s or end-

user’s expectations, Functional testing test the functional requirements of software to ensure they are

met and non-functional testing tests requirements such as performance, security, and usability, system

testing tests the complete software system to ensure it meets specified requirements, integration testing

test the integration of different components of the software to ensure they are integrated or work

together as a system, and performance testing which test the software to determine its performance

characteristics such as scalability, speed and stability. There are also other forms of testing.

**Mindset**

**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.**

As a software developer, I employed a lot of caution. I was aware that if I did not implement the

methods correctly, the software requirements would not be fulfilled and I would have errors. For

example, I knew my unit tests had to verify that exceptions were thrown when the client code attempted

to send a null for ID, ID>10 characters, in the Contact class, just one example. My constructor and setters

had to throw exceptions and it took a lot of time checking for errors to ensure correctness. When testing

coverage in the Junit tests, it was important to understand that the code that was shaded red in the test

was not called on and therefore not executed.

**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?**

With the complexity of code, it is easy to see how bias can become an issue for the software developer

who is also the tester of the code. Caution is important to software developers as they develop test.

Being able to understand the relationships between the methods as they are called and how they relate

to other methods and variables within the code is important. One method may call another method.

The software developer should be well disciplined to ensure they are sticking with the set standards for

coding and not be distracted by time constraints or lost profits for example. Bias could lead to straying

away from the software requirements when the software developer is also the tester of the code. There

should always remain a commitment to quality and industry standards from the beginning to the end of the project.