CS 320

7-2 Project Two Submission

Mihir Patel

[mihir.patel@snhu.edu](mailto:mihir.patel@snhu.edu)

Southern New Hampshire University

**SUMMARY:**

The Junit test was used to examine the functionality of the three aspects to see if there were any runtime problems or other compilation difficulties that may cause the application to crash. I was able to ensure that all the software criteria in the outline were satisfied entirely by using my tests.

The testing included running a unit test to see how individual classes functioned on their own, as well as an integrated test to see how the classes functioned when they had parent and sub-classes. A regression test was also performed to establish what outcome would be noticed if there was a change in setup, such as an update to the Java IDE, and what the outcome of such a test would be. Running several sorts of tests aids in understanding the program's operation and the existence of problems. Furthermore, performing the Junit test with known variables and outputs would aid in determining whether the program was functioning properly.

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generatedTo check that there were no run time issues, I ran the test code on every part of the program. Based on my coverage %, I am certain that the code is good technically and error-free, as all parts of the code were tested.

As seen above, I checked that my code was adequate by letting the test to run and then closing the loop if the test was good. When the test is finished and no mistakes are found, it closes with a 100 percent positive result, ensuring that the code executes.

The quality of the tests produced functions that allowed the developer to identify the origins and causes of problems in the program, which might guide on resolution to have a completely working software free of errors. The code was made more efficient by using efficient concurrency controls and avoiding the usage of iteration. Clean and straightforward code aids in the creation of a well-functioning system.

**REFLECTION:**

To guarantee the stability of the software code, a Structural Unit Testing approach was applied. In the code, the Junit testing approach was used. After incorporating the Junit library in the project, the approach required generating a new Junit test case in the Java IDE and then running the test case. If the objects are equal, the test returns true; otherwise, it returns false if the given object is null or the getClass() function is not equal. As a result, the test case should be designed so that it returns a true result if the IDs of both objects are equivalent and a false result if the criteria are not fulfilled.

Experimental testing has also been used to check the code, verifying that it adhered to standards and that all the coding was functioning in accordance with my expertise.

While reviewing code, it is critical to assess mastery of key aspects such as generating operators, constructors, loops, methods, arrays, and objects, as well as the capacity to proceed beyond the basic applications. Concerns that could emerge while testing code include a lack of understanding of the fundamental core principles and how to conduct tests on code to ensure its validity.

**MINDSET:**

Throughout the development phase, I maintained a programming quality attitude. In my programming method, I employed care to ensure that all portions of my program are logical and functional at run-time. It is always crucial to have a quality mentality to guarantee that the work you deliver meets the quality level.

It was critical for me to recognize the complexity of my code to learn from it and demonstrate the skill set I can bring to the workplace during application development.

It was also critical to consider prejudice. As a programmer testing my own code, it is very simple for me to be biased in my testing, culminating in the release of my product that does not meet the criteria for testing. I had to keep in mind that I would be testing software and ignore the fact that it was my program, treating it as any other testing circumstance I would encounter.

**REFERENCE:**

Varhol, P. (2021, June 19). *Testing for bias in your AI software: Why it's needed, how to do it*. TechBeacon. Retrieved from <https://techbeacon.com/app-dev-testing/testing-bias-your-ai-software-why-its-needed-how-do-it>