5-2 Journal: Software Testing Techniques

Michael Wood

SNHU

Course Number: CS-320

Instructor: Tanisha Jacks

Due: 04/06/2025

5-2 Journal: Software Testing Techniques

Throughout all my coding, I essentially just utilized two kinds of testing. I used unit and boundary testing. The input ranges for the functions are used for boundary testing. Since I didn't believe it was a concern for these assignments, I didn't test the minimums anyway. testing the null and maximum values. Since none of the inputs could be NULL in these situations, some sort of character would need to be entered for the system to function properly. Following that, tests were conducted on the typical values, which were somewhat higher than the maximum permitted values.

According to my research, boundary testing works incredibly well for rapidly identifying defects. This kind of testing aids in guaranteeing dependability and quality. This kind of testing helps cut down on overall effort and time because it may examine a large range of input values with fewer test cases. When done properly, boundary testing ensures that the product reacts as expected.

Unit testing was also performed by testing individual components throughout the code. By using unit testing, it is faster and more efficient than trying to test the entire code at once. The unit testing that I implored was boundary check and error handling. The major benefit of doing unit testing is early detection of issues, improved code quality, and helps developers and coders increase their confidence because they can get validation of their code or software functioning the way it was designed.

The following techniques were not used:

Developers can use Exception Testing to catch bugs in their programs before they affect users. An exception is an unexpected condition that could cause a program to stop running suddenly because of errors. A synchronous exception would be divided by zero, but an asynchronous exception would be a disk failure. I recently discovered that the Visual C++ compiler lets you choose an exception handling model using its own flags. This model determines how Structured Exception Handling (SEH) works.

Integration testing is a vital part of software testing, integration testing ensures that all of the application's modules and components work together as intended and that data is properly exchanged between them. Finding issues or faults that develop because of component combinations and interactions is the main objective. Usually done after unit testing but before system testing, this kind of testing aids in finding and fixing integration problems at an early stage of development. There are different approaches to integration testing such as Big-Bang Integration, Bottom-Up Integration, Top-Down Integration, and Mixed Integration testing.

Exploratory testing is a form of software testing in which the tester actively investigates the application without the use of predetermined test cases. This method utilizes the tester's intuition, abilities, and knowledge to enhance the quality of software and identify defects. It is frequently implemented in Agile environments and prioritizes learning, investigation, and discovery.

Regression testing helps avoid broken functionality caused by newly added code; regression testing is a must. To ensure the software maintains its functionality following updates or modifications, it is necessary to re-run previously conducted tests. Software applications must be tested to guarantee they work as intended under realistic conditions. Performance testing falls under this category. System performance under a specific workload can be evaluated using this testing technique, which measures sensitivity, reactivity, and stability.

Software testing includes an essential component known as acceptance testing, which ensures that the software is in accordance with the requirements of the business and the demands of the users. The primary objective of this test is to determine whether the system satisfies the needs of the business and to determine whether it is suitable for delivery.

In conclusion, the utilization of a variety of techniques is required to generate software that is free of errors and that functions in the manner that was intended. Since every technique possesses its own set of advantages and disadvantages, it is essential to make use of a variety of procedures. The selection of the technique that is utilized needs to be in accordance with the requirements of the project, the stages of development, and the objectives of the team.