Learning Objective 4

Evaluate the Limitations of a Given Testing Process, using Statistical Methods where Appropriate, and Summarise Outcomes

4.1 Identifying Gaps and Omission in the Test Process

- The main gap in the testing process is that not every single requirement outlined in the LO1 document has been tested for, and as such it cannot be guaranteed that the untested requirements have been met by the system. However, for the available timeline and thus scope of testing the system, it would not have been feasible to achieve this outcome efficiently and effectively. Focus has therefore been given to the 6 highlighted requirements that have been discussed throughout the process.
- Another limitation resides in some of the scaffolding, or lack of consistency through some test suites. In the LngLatHandlerTest.java file, parameters such as coordinates to test with have been manually produced and hard coded into the tests. Conversely, in the OrderValidatorTest.java suite time has been dedicated to efficiently generate random, valid data for each iteration of every test, giving a more thorough and realistic analysis of the involved components. Giving the LngLatHandlerTest.java the same time would provide more reliable and robust results of the tested functions, especially giving the fact that these requirements were highlighted to be of equal priority within the PizzaDronz system. Given more time, this would be a feasible, and arguably necessary, approach to testing.
- Also within the LngLatHandlerTest.java class, there are limitations in the style of some unit level tests. A few required the exact same equation to be implemented into both the test and actual code line by line, to mitigate certain errors such as numerical accuracy that Java entails. This makes for slightly less reliable testing, as it is essentially verifying that a line is equal to itself. If there were more time, then it would be advisable to consider more concrete methods of implementing both the test and functioning equations.
- Due to time constraints, the tests are not accompanied by much documentation such as comments outlining the functionality of tests. Although most are straightforward and self-explanatory, it is still good coding practise to implement these. However, their omission does not take away from the quality and reliability of the tests themselves.

4.2 Identifying Target Coverage/Performance Levels for the Different Testing Procedures

Functional Tests

100% Test Passing is the target. All tests for a requirement must be passed in order for the requirement to be met within the system. Therefore, complete test passing will verify the correct behaviour of the system, and indicate that all (tested) requirements have been met.

Structural Tests
100% Code Coverage is the target. Covering every method will indicate that everything that does something with the system has encountered at least one test and therefore had its behaviour verified. A line of code must be executed to verify that it does not contain any errors; omitted code cannot be guaranteed to be functioning correctly. Covering every part of the code can therefore guarantee that it has been tested for errors, and that it does not contain any should the functional tests pass.

Performance Tests
The program must execute in under 60 seconds, every time.

4.3 Discussing how the Testing carried out Compares with the Target Levels

All targets have been met by the code and implemented tests. This is discussed the LO3 document. Structural tests have been deemed to have a 100% completion because the single method omitted from testing is functionally identical to its sister method, that was rigorously tested. This specific method was pre-implemented into the code by the School of Informatics, but it is not called in normal execution of the PizzaDronz system. The tests relating to LngLatHandler.java and OrderValidator.java were manually assessed to meet 100% coverage, although there is no IDE calculation to show this. However, AppTest.java meets the 100% test coverage across every single class and method in the system, which validates their correctness.

4.3 Discussion of what would be necessary to achieve target levels

The target levels have been met, therefore no changes need to be made.