Guidelines for Graded unit Project

Development stage: (implementing the planned solution)

Coding Standards (actual Program code)

1. Font to be Consolas or Courier new
2. Font size 14pt minimum
3. Internal comments
4. Code for methods do not break over pages
5. Page numbering
6. Indentation
7. External links
8. Working Prototype
9. Database Functioning
10. Database documented
11. Files Format documented
12. Images documented
13. Sound (optional)
14. Usability
15. Extra external features used(e.g. GPS, maps, e.t.c)

- ✓ All global variables should have a comment, to the right of the declaration, describing what the variable will be used for.
- ✓ All methods should have a short comment after the method header describing the purpose of the method.
- ✓ All difficult sections of code should have a short comment nearby clarifying the purpose of the code.

Development stage: (Testing the planned solution)

(Testing the solution / prototype)

It is important that tests are described before they are carried out. The description of every test case should include

- the date of the test
- the identity of the component under test
- the purpose of the test
- the conditions of the test
- the input data
- the expected results

It should contain sufficient detail for anyone to duplicate later. Do not experiment, always know the expected results before testing. The description of test-cases should start as a module is being designed when the programmer will be most aware of possible trouble spots.

Testing is not carried out in a 'trial and error' manner. It must be logically planned as a set of test cases each of which are designed to examine some particular facet of the software. The results of each test (the **actual** computer result) must be evaluated with the **expected results**, and if successful will lead to debugging else to the next test case.

Guidelines for Graded unit Project

Your documentation of testing will contain two different sections:-

1 the testing strategy i.e. the method used to build the program while at the same time conducting tests to uncover defects.

2 testing the completed program:-

- the test-cases you carried out to ensure correct performance as an integrated system: that it functions as specified and produces correct results as functioned.

That is, an exhaustive range of black-box tests, and a few white-box tests designed to check problematic areas.

Your tests should include, but not restricted to the following:

- Check Application loads
- Start page initialised
- Pictures load within acceptable time
- Files load within acceptable time
- Page navigation functions as expected
- Confirm Application closes down properly
- Saved data can be retrieved
- Edited data can be saved
- Each page operates as expected
- Other relevant tests

Black Box Testing

The technique of testing without having any knowledge of the interior workings of the application is Black Box testing. The tester is oblivious to the system architecture and does not have access to the source code. Typically, when performing a black box test, a tester will interact with the system's user interface by providing inputs and examining outputs without knowing how and where the inputs are worked upon.

Advantages:

1. Well suited and efficient for large code segments.
2. Code Access not required.
3. Clearly separates user's perspective from the developer's perspective through visibly defined roles.
4. Large numbers of moderately skilled testers can test the application with no knowledge of implementation, programming language or operating systems.

Disadvantages:

1. Limited Coverage since only a selected number of test scenarios are actually performed.
2. Inefficient testing, due to the fact that the tester only has limited knowledge about an application.
3. Blind Coverage, since the tester cannot target specific code segments or error prone areas.
4. The test cases are difficult to design.

White Box Testing

White box testing is the detailed investigation of internal logic and structure of the code. White box testing is also called glass testing or open box testing. In order to perform white box testing on an application, the tester needs to possess knowledge of the internal working of the code. The tester needs to have a look at the source code and find out which unit/chunk of the code is behaving inappropriately.

Advantages:

As the tester has knowledge of the source code, it becomes very easy to find out which type of data can help in testing the application effectively.

1. It helps in optimizing the code.
2. Extra lines of code can be removed which can bring in hidden defects.
3. Due to the tester's knowledge about the code, maximum coverage is attained during test scenario writing.

Disadvantages:

1. Due to the fact that a skilled tester is needed to perform white box testing, the costs are increased.
2. Sometimes it is impossible to look into every nook and corner to find out hidden errors that may create problems as many paths will go untested.
3. It is difficult to maintain white box testing as the use of specialized tools like code analysers and debugging tools are required.

Grey Box Testing

Grey Box testing is a technique to test the application with limited knowledge of the internal workings of an application. In software testing, the term "the more you know the better" carries a lot of weight when testing an application.
Mastering the domain of a system always gives the tester an edge over someone with limited domain knowledge. Unlike black box testing, where the tester only tests the application's user interface, in grey box testing, the tester has access to design documents and the database. Having this knowledge, the tester is able to better prepare test data and test scenarios when making the test plan.

Advantages:

1. Offers combined benefits of black box and white box testing wherever possible.
2. Grey box testers don't rely on the source code; instead they rely on interface definition and functional specifications.
3. Based on the limited information available, a grey box tester can design excellent test scenarios especially around communication protocols and data type handling.
4. The test is done from the point of view of the user and not the designer.

Disadvantages:

1. Since the access to source code is not available, the ability to go over the code and test coverage is limited.
2. The tests can be redundant if the software designer has already run a test case.
3. Testing every possible input stream is unrealistic because it would take an unreasonable amount of time; therefore, many program paths will go untested.