BLACK BOX TESTING AND BDD

PRESENTING YOUR TEST CASES

This document provides **one** approach you might take when presenting your black box testing strategy, BDD scenarios, step definitions and associated commentary. If you wish to adopt this approach, you may; if you wish to take an alternative approach, that's fine too.

1. INTRODUCE YOUR STRATEGY

First of all, identify the **feature or features** that you're going to test. Then introduce the strategy that you're going to adopt. This might be simple boundary value analysis, or it might be weak robust equivalence class partitioning. Perhaps you're using decision table-based approach, or maybe your work is based on state transitions. You might be using a combination of these, or you might have developed some novel approach.

2. STATE YOUR TEST CASES

You do need to state all of them, although they might not all be in the main body of your work. If there's a single input, and invalid inputs are, for whatever reason, not an issue, that's a starting point of five test cases for boundary value analysis, and you would probably just drop a table containing these into the report. However, if there are three inputs, and you're adopting a robust worst-case BVA strategy, that's a starting point of 343 test cases. Yes**, you need to identify** **all inputs and outputs for all test cases,** but here, you'll list a few in the report itself, drop the rest in an appendix and add "see appendix 1", or something similar.

Keep in mind that you do not need to rigidly stick to the number of test cases indicated by your approach. Yes, three inputs and robust worst-case BVA means 343 test cases **to begin with**, but maybe you feel that some of these are unnecessary, or that more should be added. If that's the case, justify any decisions you make at this point.

3. WRITE THE USER STORY AND SCENARIO(S)

A single user story can form part of multiple scenarios, and there's a well-established and short format:

```
Feature: Quantitative appraisal
      As a human resources manager
      I want to assign numeric scores to aspects of staff performance
      That will allow support to be accurately targeted
```

The inclusion of the user story is likely to be creditworthy, as you're actually providing justification of what you're testing, albeit in a fairly limited format.

Let's say you have 343 test cases. That's 343 scenarios, and that's probably unnecessary, even if we factor in extensive use of the appendices. If you've included *a table (for part 2)* documenting all of your test cases, listing all of your scenarios would be redundant, although you will never be penalised for stating something twice.

There should definitely be at least one scenario in the **main body of your report**, which will resemble the one presented below, and the scenarios that can be found throughout the course content:

```
Scenario: Creating an employee record
  Given an employee of name of "Pepito Perez"
  And a salary point of -1
  When the user clicks the submit button
  Then the output should be "Invalid salary point"
```

This is a test for an invalid input – there is presumably no salary point of -1. In the main body of the report, I might include this scenario, and I might include an additional scenario that deals with a salary point that actually exists. If I have other scenario formats, with more steps, fewer steps, or just different steps, these would need to be included in the report too.

If you have another 50 scenarios, and none represent a departure from the scenarios you've already provided, make use of your appendix. Listing all 50 scenarios (or 343 scenarios, as per the example above) might be overkill, but you still **need to make it clear to the marker how the test cases relate to the scenarios**.

If you do list all scenarios in full, you do not need to include the *table from part 2 (above)* as well, as this would then be completely redundant, offering nothing in addition to the scenarios.

4. WRITE YOUR STEP DEFINITIONS

There should be **one of these for each step in the scenario**, so there would be four in this case. Generally speaking, the 'Given' steps should set up the aspect of the system you're testing. This might entail creating an object or manipulating its properties. A 'When' step entails an event of some kind. A 'Then' step involves checking that the actual response matches the anticipated response, and is likely to contain a JUnit assertion. An example, in this case, would be as follows for the second step:

```
@Given("a salary point of {int}")
public void a_salary_point_of(Integer newPoint) {
    e.setSalayPoint(newPoint);
}
```

Make sure you include the annotation before the method signature. You'll need to include all of these in full, and you'll need to talk the marker through what is happening here. Essentially, you need to demonstrate *that you understand what you're doing.* Since there's no need for code to usurp word count**, you are free to include screenshots of code**, but you should ensure that it's legible when your report is viewed at 100% zoom.

5. JUSTIFY YOUR DECISIONS

By this point, you might already have done this. When you selected your black box strategy, *why boundary value analysis? Why not equivalence class partitioning?* What is it about the coursework scenario that makes your strategy suitable? *Why did you select a particular form of BVA? Did you add or remove test cases from the BVA starting point? Why those test cases?*

The list of potential questions goes on, and will vary from submission-to-submission. People who make different decisions will obviously have different things to justify.

You might have a separate justification section at the end of this section of the coursework, or you might be justifying as you go. Neither one has any inherent advantages over the other, and it really just comes down to your individual writing style.