DC3270 WRITING GUIDE

COURSEWORK

This document provides guidance on **how you might approach writing the coursework**. Nothing in this document constitutes a requirement, and you are under no obligation to follow it. Your work will be assessed against the descriptors in the assessment guide **only**.

THE OVERVIEW (30%)

Although this will probably appear first in your work, being an introduction of sorts, you should consider writing it last. That's because you're going to write about, among other things, **who's doing what**. That's much easier if the 'what' is already written in the other sections.

1a. Why testing is important? (7.5%)

This is likely with no illustrations or appendix references necessary. The following prompts may help:

- Why is testing necessary in general? (be brief here, but it's a good place to start)
- Why is testing necessary in this particular situation? A good answer to this question will also deal with what the potential consequences of not testing could be. Yes, the system wouldn't work, but so what? What consequences of failures would exist in a broader context?
- **Which** specific **test techniques** are key here, and **why** are they important? Again, this is a much easier question to answer if you've already written up the tests, and you can refer to them. Perhaps "The scenario test described in section 5b is critical to this system, because…"

1b. When do we test, and who is involved? (7.5%)

You might see fit to include diagrams of development approaches (waterfall and agile), particularly if these will help you to reduce your word count.

- When does testing take place? Be specific here; although 'everywhere' is true, we need more here. Which tests take place at which points in which development approaches?
- How does that translate to the scenario? When do the tests that you've planned for fit in? Why do they fit in there?
- In terms of involvement of people, consider developers, testers, people who are both developers and testers, managers, clients and end users.
- **Justify** **whenever you can**. Why would a developer perform some of your tests and not others?

1c. How can development aid testing? (15%) (Testing and software Architecture)

This is very open-ended, and there's content from the module that could be included as well as beyond it. You might think about how classes might be structured to facilitate testing. you could discuss cohesion and/or coupling. There might be a particular design pattern that would aid testing in this situation. You don't need to be comprehensive here (which would be impossible given the word count), but you need to be as specific as possible, and you need to talk about this specific application, rather than applications in general.

BLACK BOX TESTING AND BDD (25%)

This is the largest individual section of the coursework, as well as being the section most likely to require figures and appendix references. The following represents an approach you might take. For the benefit of sound illustration, let's use worst-case boundary value analysis. That's not an endorsement of worst-case BVA for the coursework – we just need an example. The following would be a sound approach:

1. Outline the feature(s) of the system you're going to test with this technique
2. Explain why black box testing is appropriate for this feature
3. Explain why worst-case BVA is appropriate for this feature, possibly including explanations of why other prospective candidates, such as robust worst-case BVA, were not selected
4. Present a BDD scenario that would be essential when testing this feature
5. Justify everything about this BDD scenario – why do we need it at all? What benefit of the system are we testing here? Why are you using the specific test value you've written?
6. Present a method, with annotation, that would implement the BDD scenario. This will probably be in Java, and calls will be made to methods that don't exist within classes that don't exist. **Make reasonable assumptions** about program structure and explain any assumptions that aren't obvious. Your assumed code does not need to match the code you describe in section 1c.
7. Refer the reader to the appendix, which **should include a complete set of test cases** for worst-case BVA for this feature. The appendix is likely to contain a table that includes inputs and expected outputs
8. Ensure any decisions you make are justified, and add more test cases in the main body of the assignment as you see fit. Each test case you present should give us something new. You might wish to include the upper bound and the lower bound, as these can help you to illustrate your decisions using fewer words. The vast majority of test cases, however, will be found only in the appendix.

You are free to use screenshots of code rather than copying the code directly into your work. Dedicate the bulk of your word count here to describing what you would do and explaining why you would do it.

WHITE BOX TESTING (15%)

In order to lay out your white box testing approach, you'll need pseudocode, which can also be in the form of a screenshot if you wish to maximise the available word count. Let's say, hypothetically again, you're going to test a particular feature using condition coverage

1. Outline the feature(s) you're going to test with this technique
2. Present your pseudocode that will be used to illustrate your testing approach
3. Explain why condition coverage is appropriate – what's wrong with other approaches in this situation?
4. Identify all of your test cases (the appendix is at your disposal if needed), and justify these test cases. Your test cases need to match the strategy you've selected as well as your pseudocode. Ensure all tests are necessary and minimise gaps in coverage. If you include **redundant tests**, or if you miss a statement/branch/condition, **you will be penalised** for this.

QUADRANTS 3 AND 4 (15% EACH)

These are grouped together, because they are likely to be presented in a similar way. You should aim for two types of test from each quadrant, since you can only score a maximum of 59 for a quadrant that is only addressed with one test. Given the restrictive word count, you are advised against discussing more than two testing approaches per quadrant.

This is likely to be just prose, but you may use images and appendix references as you see fit. Some testing approaches are more likely than others to require such supporting evidence. The following pointers should help you in putting together each of the approaches:

- Which feature(s) would you test
- Describe the test in terms of when it would take place, who would be involved, what resources might be needed and exactly how the test would be conducted; be as specific as you can, and relate everything to the scenario.
- Justify any decisions you make. Why have you chosen this approach at all? Why do you need those particular resources? Why did you, for instance, load test with ten thousand simultaneous connections rather than twenty thousand?
- Address any shortcomings in the testing approach you've selected.

GENERAL POINTERS

- Be as specific as possible – assume someone will actually conduct your tests
- Link everything to the scenario – if you're going to talk in general about, say, usability testing, that's fine initially, provided that you then talk about usability testing of the scenario application
- Justify every decision that you make. The 'why' is as important as the 'what'
- Label your appendices (appendix A, appendix B…) and images (figure 1, figure 2…), and refer to them in the main body of your work
- **Use references** **where applicable**, particularly when justifying a decision in any of the sections above, and provide a references list using Harvard Referencing (see https://www.citethemrightonline.com/)
- Read the assessment guide that's incorporated into the coursework brief. This tells you where all of the marks come from.