Software Testing 2018

Justin Pearson, Jo Gay and Vasileios Loukas

November 4, 2019

1 Introduction

The purpose of this document is to make clear how you will be examined, and what is required of you in the course. This course essentially consists of four components:

1. A series of lectures on aspects of software testing;
2. Group work on test design;
3. A lab on test driven development;
4. An exam.

The exam is graded U,3,4,5. The group work and the project is pass or fail.

If you only learn one thing about testing during this course, then you will fail, but if you learn the following idea "You should have a reason for each test. For each test you should be able to explain what you are trying to test for." then you will at least understand why we study testing. When we examine you, and when you produce tests, we expect you to be able to explain the reasons behind your tests.

2 Deliverables and requirements 2018

1. A lab on test driven development (TDD) for BibTex. This is done in groups of 2 or 3. It will be marked at the lab. There is no hand-in.
2. Project work on test case design for some python library. This is done in groups of 4.
3. An exam.

Information on the lab can be found either linked from the student portal or via `http://user.it.uu.se/~justin/Teaching/Testing/index.html`. Information on the exam will come later. It is safe to say that if I mention a concept on the slides, then I might ask an exam question on it. It is also safe to say, that my slides do not contain all the information that you need. You will probably have to read the book to get enough information to pass the exam.

Project Work

The idea of the project is to pick some python library, and write some test cases for it. You will have to write both black-box and white-box tests and tests that provide coverage of selected parts of the code. The final deliverables will be a written report and a 10 minute presentation.

Using your chosen library, your tasks are as follows:

⇒ Black box testing of the API. You are to produce test cases that cover the API.

⇒ White box testing. In agreement with your lab assistant you are to pick some areas of code in your library to cover. Together with the python library `https://coverage.readthedocs.io/en/v4.5.x/` you are to produce test cases that not only provide statement coverage but some sort of path coverage.

⇒ For at least one function or method of the library you should construct a control flow graph and apply the coverage criteria that have been covered in the course to the code.

⇒ You need to document and motivate your design of the test cases in a written report.

Schedule for the project

- By the start of week 46 you must form a group. This will be handled via the student portal.
- In week 47 you will meet your assigned lab assistant; before the meeting you should pick a python library (for example look at the Python Standard Library or `https://pypi.python.org/`). You must agree with your lab assistant that you can work on that library. We have to decide if the library is too trivial or too complicated. If you are having trouble finding a good library to test then you should try to meet your lab assistant in week 46.
- In week 48 you should have a brief meeting with your assigned lab assistant to discuss progress.
- On 4/12 (week 49) there is a compulsory presentation, where you are to present your library and present some of your test cases and the reasoning behind them. Your presentation is of work in progress.
- In week 49 or 50 you again meet with your lab assistant to discuss your draft of your report.
- Jan 7, 2019 is the deadline for the final report. This should be submitted via the student portal.

What should your report contain

- Description of the python library with examples of use. This should be written (in English) for somebody who has not read the documentation.
- Outline of your testing strategy, describing which parts of the library you have performed black box and white box testing on.
- A control flow graph for at least one function, with a description of how your tests for this function are related to the graph.
- Documentation of all test cases. When documenting your test cases you have to document what your tests are designed to do. This can be done in comments in your code, for each test case or group of test cases.

Grading criteria for the project

- Clarity of presentation. It is important that your document can be read without reference to the documentation of the library. This means that you will have to re-explain things already in the documentation.
- Quality of the test cases. What sort of coverage do they provide? Do they test enough functionality? Have a range of different techniques been used?
- Documentation of the test cases. Have you clearly stated what you are testing, and why, for each test or group of tests?