Software Testing 2020

Justin Pearson

October 26, 2020

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 flipped lectures on aspects of software testing;
2. A TDD exercise that should be done in week 46. Instead of having a lab session the lab assistants will be available online that week to help with the lab.
3. Group work on test design;
4. An exam (2021-01-07).

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 2020

1. An exercise on test driven development (TDD) for author name parsing in BibTex. This is to be done individually.
2. Project work on test case design for some Python library. This is done in groups of 5 or 6.
3. An exam.

Information on the exercise can be found either linked from the student portal, studium or via `http://user.it.uu.se/~justin/Hugo/courses/softwaretesting/`.

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 deliverable is the written report.

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 (including prime paths) 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 end of week 45 you must form a group. This will be handled via Studium. If you are happy to be assigned in a random group then email `it-testing-course@lists.uu.se`. If you do not join a group or do not email me to be assigned a random group then I will assume that you are not following the project.
- In week 46 you will meet virtually your assigned lab assistant; before the meeting you should pick a python library (see `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 discuss with your lab assistant.
- In week 48 you should have a brief meeting with your assigned lab assistant to discuss progress and show a draft report.
- In week 50 or in the new year, you again meet with your lab assistant to discuss your draft of your report. You must submit a draft of your report to your assistant before your meeting.
- Jan 15, 2021 is the deadline for the final report. This should be submitted via Studium.

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?

Unsuitable Python libraries

This is not an exhaustive list of libraries, just some libraries that have not worked well in the past.

- numpy `https://pypi.org/project/numpy/`: This has been a tempting library for many students in the past, but it is hard to come up with good test cases, especially when you are trying to do path coverage.
- Almost anything from the python standard library, especially implementations of basic data types.
- The Python date and time library, is covered by the above item, but it is very hard to come up with good test cases.

Ideas for libraries

This is not a prescriptive list, but if you are looking for inspiration some of these libraries have worked quite well in the past, or they are libraries that we suggest you look at:

- beautifulsoup `https://pypi.org/project/beautifulsoup4/` is a web scraping library. This is interesting, because you have to work out how to generate and send test cases to the library.
- PyGreen `https://pypi.org/project/pygreen/` PyGreen is a static site generator. Any static site generator or a tool to convert markdown to some other format.
- Panda `https://pandas.pydata.org/` is a data analysis tool for Python.
- Diplomacy `https://pypi.org/project/diplomacy/` is an implementation of the board game Diplomacy. Any similar project will be interesting especially if you try to come up with black-box test cases that cover the game requirements.

Remember that you must discuss the choice of the library with your lab assistant. The library should not be too simple, nor should be too complex. You are in Sweden so the library should be lagom.