CS305: Software Engineering

Programming Assignment 1 - Test Driven Development
Due: 14/10/2020

The objective of this assignment is to introduce you to test driven development. Specifically, you will:

1. Practice writing SRS and refine the functional and non-functional requirements for the project considered in the previous assignment.
2. Translate the SRS into software.
3. Apply a specific System-Testing technique (Category-Partition method) to test your software.

1 Software Requirements Specification

The requirements gathered during Requirements Engineering (RE) should have the following desired characteristics:

- Simple: do not mix multiple software requirements into one
- Testable: there should be at least one test case that verifies the requirement i.e. checks that the software provides the functionality without any errors.
- Organized: related requirements are grouped, priorities are indicated, and a mention of which user (customer or developer or tester) requires this functionality.

Rename `SRS_Template.md` as `SRS_document.md` and edit the file to come up with functional and non-functional requirements that are simple, testable, and organized. The SRS are to be written based on the the project described in video lectures 72 to 76. In addition, your software should print the team info on the console when passed the `-team` option as a command-line argument. When passed the `-help` option as a command-line argument, the software should output on the terminal the whole documentation (see `README.md` below).

This file is written in a markdown language[1] and all you need to do is fill in different sections with appropriate content as described.

2 TSL Generator

The goal of functional testing of a software system is to find inconsistencies between the functionality of the actual implemented system and the desired functionality as described in the system's functional specification. To detect inconsistencies we must test the system. The testing should be done in a way that maximizes the chances of finding errors and at the same time checks if all the functionality is implemented. Functional tests can be derived from SRS, design documents, or the code itself. In this assignment, we consider specification-based tests (source for the test derivation is SRS). Category-Partition method is a systematic approach to implement functional-level test cases and TSL generator is a tool that uses this approach to generate test specifications starting from the functional specification.

In this assignment, you will use TSL generator to come up with test specifications for the project described in your SRS.

3 Git Setup and Submission

The Git setup and submission instructions remain the same for all assignments. Please follow the instructions mentioned earlier in `PA0` setup to create your local PA1 repository. When you switch to the PA1 repository,

[1] You are encouraged to try and practice using the markdown syntax.

you should see all the files you will need to edit:

`SRS_Template.md` and `build.sh`.

In addition, when you submit, your repository would contain several other files as mentioned next.

3.1 What you need to submit?

- `SRS_document.md` - refines SRS document (from PA0)
- `README.md` - describes the documentation of your software system. Look at the manpages of TSL generator `./tsl -manpage` to see the format of the documentation. Your documentation should have a similar format. Specifically it should have:
  - preamble - shows software name, developer(s) info with full name.
  - Name - shows the name and brief, one-line summary of what the system is about
  - Synopsis - shows the command to execute with all possible options
  - Description - brief, one paragraph description of the project and its scope.
  - Options - supported options and brief description.
- Folder called `src` - contains
  - `build.sh` - update this file containing instructions/comands on what is expected when creating a Java JAR file called `avglength.jar`. Make sure to test your software with different types of file inputs (not just text files).
  - all Java source files required to create the JAR file (*do not include .class files*).
- Folder called `test` - contains
  - `pc.spec` - to use with the TSL generator to generate test specs. The file should partition and categorize inputs for the domain of the functionality described in your SRS. The file should use at least 1 `ERROR`, `SINGLE`, and `IF` properties. The number of test cases produced should at most be 50. *Do not include the TSL generator repository in your submission.*
  - contains test programs/scripts that implement all the test cases having high priority[2].
  - README.txt - brief description of how to test your program using the test case implementations.
  - any other supporting material useful to verify your program.

[2]high priority test cases are those that verify high priority requirements mentioned in SRS. If you miss a high priority requirement in the SRS document, then points will be deducted.