COMP 4004 Assignment 1 due October 10, 2017

Premise: You have been given (see folder for A1 on the 4004 website) two files from two totally different sources. The first is a pdf containing a use case diagram and an *incomplete* list of use cases for a system involving first and foremost a single librarian using a single terminal. This document is to be taken as the specification of the desired functionality but **not** of the way the desired system should actually work: It is too centered on the librarian.

The second file is a zip containing a simple working library system (which is to be discussed on September 21st). It may contain bugs and/or may miss functionality with respect to the first document. It was developed independently from these use cases!

Goal: We want a simple **networked** library system that supports several users borrowing and renewing books, and paying fees from different terminals (using the simple menu driven approach of the provided system), as well as a **single** librarian who deals with all administrative facets of the system.

PART 1: You are to use a TDD approach to implement a system that covers all the functionality described in the supplied use case document. Through this approach, you will obtain a test suite at the unit level for your system. That is, you must develop the system using the TDD approach of writing a small test first, and then the code to make this test succeed. You are working essentially at the unit level and should expect lots of tests. You are encouraged to reuse some of the supplied code. It is crucial you maintain traceability between your code and your unit tests. You are to use Java/Eclipse, JUnit, and git, sharing access to your repository with BOTH TAs. For each TDD iteration, you must submit the test(s) that initially failed for this iteration, and the code that made them work. Consequently, through the history of your commits to the repository, we will be able to verify you indeed did follow a TDD approach.

PART 2: Acceptance testing
You must develop a series of user stories to cover the system at the acceptance level. That is, you must cover all functionality suggested by the provided use cases by developing story descriptions (*including rules*) **and** their corresponding tests. We are *not* using conversations but instead you are to add rules directly to your stories. Your story tests must be developed using JUnit and must be traceable back (i.e., refer) to steps (which you should number) of your user stories. Please note that you must address inter-scenario relationships (which you will have to figure out by yourself).

Your culearn submission will consist of a single zip file containing:

- all source code for the system **and** its tests (reuse the structure of the supplied code). You should organize your tests into unit ones and story ones (and this should be also clear in the comments in the code for these tests).
- all compiled code (again look at the supplied zip)
- a rtf/odt/doc document describing the scenarios (ie stories) of the required system and their corresponding tests. It is essential that traceability between story tests and steps of scenarios be captured. You must **also** include the pdf of your document.
- any information required for the TAs to access your git repository: be sure to include an email address so that they contact you if they have problems.
- one or more screen captures of all your unit tests running in JUnit
- one or more screen captures of all your story tests running in JUnit

Aim for as few screen captures as possible.