Fall 2017
CS589 PROJECT REPORT

Due Date: **November 29, 2017**
Late project: **50% penalty**
After **December 4, 2017** the project will not be accepted.

This is an **individual** project not a team project. **Identical or similar** projects will be penalized. The **hardcopy** of the project must be submitted. Notice that the Blackboard project submissions are only considered as a proof of submission on time (before the deadline). If the hardcopy is different than the electronic version submitted on the Blackboard, then **50% penalty** will be applied. If the project is submitted on the Blackboard on time, we must receive the hardcopy of the assignment by **noon on Thursday, November 30.** If the hardcopy is received after this deadline, **20% penalty** will be applied. After **December 4, 2017** the project will not be accepted.

Report

1. Model-based testing of the *Account* class
   Show that 2-transition sequence testing has been satisfied. Identify all 2-transition sequences. For each 2-transition sequence indicate which test in the test suite (TS.txt file) executes this 2-transition sequence. If a given 2-tranistion sequence is not executable, you need to explain why it is non-executable.
2. Testing default (ghost) transitions of the *Account* class
   Show that default transition testing has been satisfied. Identify all default transitions in every state. For each default transition indicate which test in the test suite executes this default transition.
3. Multiple-condition testing
   Show that multiple-condition testing has been satisfied in all methods. Identify all multiple-conditions/branches in the implementation (source-code) of the *Account* class. For each multiple-condition/branch indicate which test in the test suite (TS.txt file) executes this multiple-condition/branch. If a given multiple-condition is not executable, you need to explain why it is non-executable. Notice that if a predicate contains only a simple condition, the multiple-condition testing is equivalent to the branch testing for this predicate.
4. A Test Suit and the results of its execution
   In this section you need to provide a hardcopy of the test suite (TS.txt file). In addition, you need to execute the test suite. During the execution of the test suite, the results produced by the *Account* class need to be recorded/documented and provided in a separate file that should be posted on the Blackboard (the hardcopy of this file is not required). For each test case you need to validate the results and determine whether the *Account* class produced the correct results. If for a given test case the results are incorrect (test failed), identify the cause of incorrectness (a defect) in the source code of the *Account* class.

5. Conclusions
   In this part of the report, you should describe your experience with the implementation of the testing environment and its usage in class testing and model-based testing. Describe which activities related to class testing can be automated or partially automated.
6. Well documented source code of the *Account* class and the test driver(s).

IMPORTANT: The project executable(s) of the testing environment (a test driver(s)) and the test suite file (TS.txt file) must be prepared by students and made available for grading. The best way is to submit the project executable and the TS.txt file on the Blackboard. However, you may provide the project executable and the TS.txt file on a flash drive. If the executable is not provided (or not easily available), **20 POINTS** will be automatically deducted from the project grade.

The test suite (TS.txt file) should contain all the test cases used to test the *Account* class (as described in Sections 1, 2 and 3). A sample TS.txt file is posted on the blackboard. Notice that your test suite will be executed in order to make sure that the testing criteria specified in this project have been satisfied. In order to check whether your test suite (TS.txt file) is in a correct format, you should execute the **test suite checker** that is posted on the blackboard. The test suite checker will indicate if there are any errors/typos in TS.txt file that need to be corrected. The test suite should contain **ONLY** operations of the class *Account*, i.e., testing-oriented operations/methods should not be a part of the test suite (TS.txt file). If TS.txt file contains errors that are detected by the test suite checker, **20 POINTS** will be automatically deducted from the project grade.