CS589 Fall 2021
Detailed Project Description

Due Date: **December 2, 2021**
Late project: **50% penalty**
After **December 8, 2021,** the project will not be accepted.

This is an **individual** project, not a team project. **Identical or similar** projects will be penalized. All submissions should be made on the Blackboard.

Report

1. Model-based testing of the *Account* class
   Show that Transition-Pair testing has been satisfied. Identify all transition-pairs in the EFSM model. For each-transition pair indicate which test in the test suite (TS.txt file) executes this transition-pair. In addition, provide a list of all non-executable transition pairs.

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. List all non-executable default transitions.

3. Multiple-condition testing
   Identify all multiple-conditions/branches in the 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. In addition, provide a list of all non-executable multiple-conditions/branches. Note: if a predicate contains only a simple condition, the multiple-condition testing is equivalent to the branch testing for this predicate.

4. Test Suit
   The test suite must be created and stored in TS.txt file. A sample TS.txt file for this project is posted on the Blackboard.

5. After the test suite is created, execute all test cases in the test suite. During the execution of the test cases, the results produced by the *Account* class need to be recorded/documented and provided in this section. For each test case, validate the results and determine whether the *Account* class produced the correct result or the test failed (produced incorrect result). **It is assumed that the provided EFSM represents the expected/correct behavior of the *Account* class.**

6. List of Failed Tests
   In this section, list all **failed** test cases. For each failed test case show:
   a. the incorrect outcome/output produced during the execution of the test
   b. the expected outcome/output for the test.

7. Conclusions
   Describe your experience with the implementation of the testing environment and its usage in class-testing and model-based testing. Indicate, which activities related to testing can be automated or partially automated.

Items to be submitted on the Blackboard:

1. Report
2. Well-documented source code of the testing environment and the *Account* class.
3. The executables of the testing environment (test driver(s) + *Account* class) with detailed instructions explaining how to execute the program(s).
4. Test suite file (TS.txt file).

IMPORTANT:
The project executable(s) of the testing environment (a test driver(s)) must be prepared by students and made available for grading. 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 of the Report). A sample TS.txt file is posted on the Blackboard. Notice that each test suite will be executed to make sure that the testing criteria specified in this project have been satisfied. To check whether your test suite (TS.txt file) is in the correct format, execute the **test suite checker** that will be 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.