ECE 322 Lab Assignment 5

Integration Testing #2 (incremental testing)

Overview

The objective of this lab is to become familiar with incremental integration (Bottom up and Top down) white box testing tools. This lab makes of use of of Python , unittest python package for integration testing. This lab complements the previous lab of integration testing. The application under test is the same as in the previous lab(lab 4), but testing technique is incremental integration testing rather than nonincremental integration testing.

Incremental Testing Integration Testing:

This general approach of integration testing combines the next module to be tested with the set of previously tested modules before running tests. Generally done in either a bottom up or top down method

- Bottom up: Test the lowest level modules in isolation, then incrementally add higher and higher level modules.
- Top down: Test the highest level modules in isolation stubbing out lower level functionality, incrementally add lower modules.

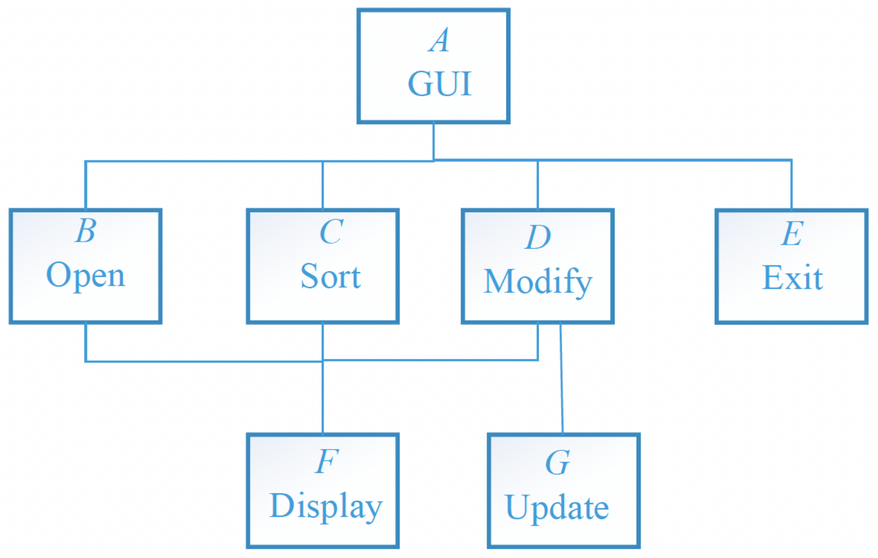
Task (100 marks)

Preparation: Prepare test cases for the tasks you will be working on in the lab session. Make sure you have Pycharam and unittest installed on your account, or an equivalent Pycharam IDE. , unittest python package is still used as mocking framework in this lab. The code for this lab is available on the class website and can be imported into Pycharam as an existing project.

Consider a simple database system which has been constructed in a modular fashion: Module A invokes Module B Module C, Module D, and Module E. Module D invokes Module F, and Module G. Modules B and C also invoke Module F.

Entries in the database are composed of two elements, a String name and a String Phone number. Module A is a Command Line Interface which processes command strings and delegates functionality to sub modules:

1. Open a data file (Module B)
2. Sort records (Module C)
3. Modify a record (Module D)
4. Exit (Module E)



Files for this database should contain one entry per line, and elements of an entry should be comma separated. Sort function sorts records by first name. The modify function (Module D) additionally uses the Display function (Module F) and the update function (Module G).

Your task is to prepare and run test cases, stubs, and drivers for:

- Bottom Up testing
- Top Down testing

Unit tests should cover the full functionality of each module; there should be at least one test for each piece of functionality and method. Your test cases should strive for, at a bare minimum, statement coverage. You do not need main function of the provided code as it implements a simple command loop.

Only the **public** methods of Module A need to be tested directly.

Your test cases **must be broken into two distinct test suites**, one for each type of incremental integration testing (two test suites total). The reuse of individual tests is encouraged; however, be aware that test order and setup is key to correct integration testing. Your test suites should be easily run against the provided application code for verification.

Run the test cases and record your observations and results in your report, as in previous labs test cases must be presented in your report as a test case table with meaningful descriptions, expected and actual results. Failed test cases must be highlighted.

Comment on the effectiveness of integration testing. Which type of integration testing do you think is best? Which would be the most appropriate for a test driven development environment? Which would be the most appropriate for library development? Which is the easiest to maintain? Consider these and similar questions in your discussion of integration testing to show you have a strong understanding of the underlying concepts and motivations of this testing technique.

Your report should include

- Two test case tables, one for each type of incremental integration testing. Test case reuse is okay where applicable; however results should be fully listed in each table for clarity.
- A discussion of your results regarding the application under test. This should include a discussion of any errors you found in the code, and fixes to make your tests pass
- A discussion of the above questions regarding integration testing technique.

Lab report:
Must be typed, no handwritten versions will be accepted. Follow the general format as included in the lab guideline. Submit all code via eclass. Your report should include:

- Your write-up
- The results of your test cases, in the form of test case tables
- Any conclusions you have drawn from these test cases regarding the applications under test. This should include your discussion focusing on the questions raised in each section
- Your test code as Python unittest files to be easily executed against the provided application code

Clearly indicate all failed test cases, and explain the cause of these failures.