Computer Science COMS W4156 Advanced Software Engineering Fall 2017 - Second Exam

December 7, 2017

Do not open the exam until the proctor tells you to do so. You may not use any books or notes. You may not use a calculator or any other device beyond a pen, pencil and eraser. Please write each answer in the corresponding space, continuing on the blank backs of pages if needed. Read through the entire exam before beginning to answer questions. Question 3 is long, with some intermediate pages to provide plenty of space for answers. It is not necessary to use all the space. The exam consists of 11 pages, with the last page saying only "(this page intentionally left blank)".

Name:

UNI (also put your UNI at the top of every page, since the pages will be separated during grading):

Problem No.	**Max Points**	**Points Scored**
1	10	
2	20	
3	30	
Total	60	

Problem 1 – Multiple Choice
(10 minutes, 10 questions, 1 point for each correct answer)

Circle the letter that represents the **best** answer to each of the following questions.

1. What is the purpose of *mutation analysis*?
   a. Validate your design
   b. Validate the changes between sequential commits
   c. Validate your test suite
   d. All of the above
   e. None of the above

2. What should you *always* do before integrating third-party source code into your own project?
   a. Exploratory testing to determine the behavior of the parts you expect to use
   b. Apply tools that automatically generate class diagrams from existing code
   c. Push it into your version control repository
   d. All of the above
   e. None of the above

3. Which of the following is a *bug*?
   a. Software doesn't do something requirements say it should do
   b. Software does something requirements say it shouldn't do
   c. Software does something that requirements don't mention
   d. All of the above
   e. None of the above

4. Which of the following *best* describes regression testing?
   a. Zero bug bounce
   b. Test-to-pass
   c. Using a mocking framework
   d. All of the above
   e. None of the above

5. Which of the following might be *input* to a method invoked during a unit test?
   a. An object instance of that method's class
   b. Global data shared by all instances of the method's class
   c. Return values from external API calls made by the method
   d. All of the above
   e. None of the above

6. Which of the following are likely to be included among the *outputs* of a method invoked during a unit test?
   a. Sub-boundary conditions
   b. Teardown
   c. Parameters to external API calls made by the method
   d. All of the above
   e. None of the above

7. What is a reasonable set of boundaries for *range* equivalence classes?
   a. Root node, internal node, leaf node
   b. min-1, min, min+1, max-1, max, max+1
   c. Data entered using a Dvorak keyboard
   d. All of the above
   e. None of the above

8. Which of the following should *never* be included in the issues of an issue tracker?
   a. References to specific commits in the version repository
   b. A specific sequence of steps to reproduce a bug
   c. Assignment to a specific developer
   d. All of the above
   e. None of the above

9. Which of the following should always be done *before* a code inspection meeting?
   a. Assigning a reader and a recorder
   b. Delta debugging
   c. Minimizing the truck factor
   d. All of the above
   e. None of the above

10. What is the *most significant difference* between acceptance testing and system testing?
    a. Acceptance testing validates the application in a customer's production environment whereas system testing is normally done prior to deployment
    b. System tests are normally executed from inside an interactive debugger whereas acceptance tests are usually run from the end-user's user interface
    c. System testing guarantees that a bug will never re-occur
    d. All of the above
    e. None of the above

Problem 2 – Vocabulary
(20 minutes, 5 questions, 4 points for each correct answer)

Explain the following terms in a few sentences and/or small drawings.

1. User input validation

2. Condition coverage

3. Integration testing

4. Greybox testing

5. Test oracle

Problem 3 – Mini-Project
(30 minutes, 2 questions, 15 points each)

Recall the mini-project from the first exam. You are still working on software to manage task boards for agile teams. The primary features that the system needs to provide are:

- An administrator needs to be able to specify who are the members of each team and provide each team with a workspace.
- Team members should be able to make changes only in their own workspace.
- Team members need to be able to set up and manipulate task boards in their workspace.
- Task boards support To Do, In Progress, Completed and Overflow status categories.
- UPDATED! Task boards support Items, each of which consists of the description of a user story or another task to perform, a priority, a time estimate in "story points", who is assigned to do the task, and a status category (initialized as To Do).
- NEW! The Items of tasks boards automatically record when they move from To Do to In Progress and again when they move to Completed or Overflow, to track how much work time has elapsed between In Progress and Completed (or the time spent so far in the case of Overflow).

The actual questions to answer are on the following pages, parts A and B.

Part A – Unit Testing (15 minutes, 15 points)

You are a member of the development team for the task board system. Your pair has been assigned to implement the Item class. Your team uses test-driven development, which means you and your pair partner need to invent unit tests for the Item class *before* starting to code the class. Describe at least five TDD unit tests in prose or pseudo-code, and explain why you chose those test cases. Discuss any equivalence classes and boundaries that seem relevant to your test cases. Make any assumptions that seem reasonable about the other classes in the system, such as the Task Board class. Do not be concerned with how Items are stored persistently; a different pair is working on the database interface, for now your Items just need to exist as objects in memory. There is no single correct answer.

Continue your answer for part A on this page if necessary (you can also use the backs of pages).

Part B – System Testing (15 minutes, 15 points)

In this scenario, you are a member of a separate testing team, not the development team. The features sketched in the mini-project overview have all been completed by the development team. Your testing team's job now is to test the task board at the system level prior to the first demo for the customer. Make any assumptions that seem reasonable about the system's user interface. Describe at least five specific examples of system level tests, written as prose or pseudo-code, and explain why you chose those test cases. Discuss any equivalence classes and boundaries that seem relevant to your test cases. There is no single correct answer.

Continue your answer for part B on this page if necessary (you can also use the backs of pages).

(this page intentionally left blank)