UNI:

Computer Science COMS W4156 Advanced Software Engineering Fall 2016 - Second Exam

December 6, 2016

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 13 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	20	
2	30	
3	50	
Total	100	

Problem 1 – Multiple Choice (2 points for each correct answer; -1 point for each wrong answer or if multiple answers are selected; 0 if blank)

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

1. How is acceptance testing *different* from system testing?
    a. System testing tests from the GUI whereas acceptance testing uses a test runner
    b. It's not acceptable to start system testing until after all acceptance tests have passed
    c. Project Velocity is recalculated only after acceptance testing
    d. All of the above
    e. None of the above

2. What should you *always* do when integrating third-party source code into your own project?
    a. Conduct security penetration testing on the entire codebase
    b. Rename all the classes and methods you will use
    c. Push it into your version control repository
    d. All of the above
    e. None of the above

3. What should you do at the beginning of *every* iteration (after the first)?
    a. Revise priorities and time estimates for the remaining user stories
    b. Fit user stories into your iteration using the project velocity from the previous iteration
    c. Set up a fresh task board
    d. All of the above
    e. None of the above

4. What should you do at the end of *every* iteration (except the last)?
    a. Subtract 0.1 from your project velocity
    b. Allocate one day to staff training and software and hardware updates
    c. Allocate five days to spike bug fixing
    d. All of the above
    e. None of the above

5. What is the *most significant difference* between testing conducted by developers vs. non-developer testers?
    a. Non-developer testers usually perform system level testing rather than unit testing
    b. Non-developer testers only inspect the source code without executing it
    c. Developers should never test their own code
    d. All of the above
    e. None of the above

6. Which of the following is an example of a bug that could be found by considering *sub-boundary conditions*?
    a. A text box intended to allow only alphanumeric characters also allows the ';' character
    b. A conditional uses '>' when it should use '>='
    c. a[0] is left uninitialized when the rest of array a is initialized
    d. All of the above
    e. None of the above

7. What is the *most significant difference* between exploratory testing and extinction testing?
    a. Exploratory testing is used to figure out what is the functionality of an application whereas extinction testing is a technique for removing (extinguishing) bugs
    b. Exploratory testing is much faster
    c. Exploratory testing is a term often used in software engineering whereas I made up the term extinction testing
    d. All of the above
    e. None of the above

8. How does invalid input *differ* from valid input?
    a. Valid input is a term often used in software engineering whereas I made up the term invalid input
    b. Invalid input should result in an error message understandable by the user
    c. Invalid input requires a special keyboard
    d. All of the above
    e. None of the above

9. Which of the following should occur *during* a code inspection meeting?
    a. Take notes about any problems found
    b. Write a report describing any problems found
    c. Fix all the problems found
    d. All of the above
    e. None of the above

10. Which of the following is *useful* for isolating and fixing a bug?
    a. Devising the simplest test case that reproduces the bug
    b. Using an interactive debugger
    c. Recording as many details as possible in a bug tracking tool
    d. All of the above
    e. None of the above

Problem 2 – Vocabulary (3 points each x 10 = 30 points)

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

1. Test-driven development

2. Branch coverage

3. Regression

4. Refactoring

5. Project Velocity

6. Iteration Review

7. Spike (in an agile software process)

8. Static whitebox testing

9. Dynamic whitebox testing

10. Test-to-Pass

Problem 3 – Mini-Project (50 points)

Recall the mini-project from the first exam. You are still developing software to manage team projects for a variant of this course, where instead of using trello, github projects, etc. the course will use its own task board system. The primary features that the system needs to provide are:

- The teaching staff needs to be able to specify who are the members of each team and provide each team with a private workspace.
- The teaching staff needs to be able to read the contents of every team's workspace, while the team members need to be able to read and write only in their own workspace.
- Team members need to be able to create, read, update and delete task boards in their workspace.
- 

Task boards include items representing the user stories to implement and other tasks to perform, who is assigned to do them, and their status (e.g., not started yet, in progress, completed).
- 

The task board needs to be integrated with (at least) a version control repository and an issue tracker. Assume that these other systems have well-documented APIs.

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

Part A – Code Inspection (20 points)

The features intended for the first release of the task board have been coded by your team (4 members). The codebase consists of about 25k lines of code split among about 50 different files. Your team is tasked with inspecting this code. Explain how you would proceed. Consider whether you might need more than one code inspection meeting, what roles your team members should play in a code inspection meeting, what should be done before a meeting, what should be done during a meeting, and what should be done after a meeting.

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

Part B – Testing (20 points)

The features intended for the first release of the task board have been coded and inspected by your team (4 members), and all problems found during inspection have been fixed. Your team's job now is to test this code (i.e., the developers are testing the code here, not a separate testing team). Explain how you would proceed. Consider both unit and system testing (unit testing probably should have been done before code inspection but in this case it wasn't). Include at least two specific examples, written as prose or pseudo-code, of system level test cases.

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

Part C – Greybox Testing (10 points)

Now consider greybox testing of your task board system (if you included greybox testing in your answer to part B, that shows us that you did not read all the questions before starting to write your answers). Explain how you would test the low-level interactions among any major subsystems within the task board and with other systems such as the version repository and issue tracker.

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

(this page intentionally left blank)