# EECS 481 - Exam #2

Winter 2020 - Software Engineering

## Instructions (read carefully, please!)

1. Please obtain your own editable copy via Google Docs (requires UM login):
    a. https://docs.google.com/document/d/1XVFoPFWCLwzEEbT1qhR86TkjvZPl74nDChLWC4yBQHk/copy
2. The expected time for this take-home exam is two hours.
    a. Expected time-and-a-half for the exam is three hours (e.g., for SSD).
3. The exam is **optional**. If you do not submit a valid exam file by the deadline, or if your last submission has “do not grade” instead of your name, you are skipping the exam.
    a. If you elect to skip the exam, your course grade will be out of 82 instead of out of 100. See the course forum for details.
    b. If you do not elect to skip the exam, we will mark your exam and it will count for 18% of your grade.
4. The exam file is due **Thursday, April 16th at 11:59pm Eastern** via Gradescope.
    a. You may resubmit your exam file as often as you like. We will grade the last submission file (unless you mark it “do not grade”, in which case we will not grade your exam at all). Your PDF must be 12 pages matching this layout.
5. You must use a word processor to **type your answers** by editing the exam file. You may not hand write the exam and scan it in.
6. You must type your answers only in the **framed answer boxes**.
7. You must **not change the size** or position or margins of the framed answer boxes.
8. You must use the same **11 point Arial** typeface for all of your answers. If your answer for a question does not fit in its provided box, edit and simplify your answer.
9. If you leave a non-extra-credit answer box **blank or type “skip”** in it, you will receive one-third of the points for that subquestion.
    a. Because the exam is optional and you can skip questions, grading on answers you do include will be **quite strict** compared to other assignments.
10. The exam is **open book**, open notes, open computer, and open Internet.
11. You must **work alone** to complete the exam. You may quote or refer to text from the readings or Stack Overflow, for example, but you must turn in your own work and may not collaborate with other humans.
12. On certain questions you will be required to craft examples. You will **not receive credit** for class-, reading- or Internet-derived examples. In other words, examples must be personal in some aspect (whether real, figurative or imaginative).
    a. Coincidental overlap with other students will be investigated manually.
13. We will use prose **plagiarism detection** software and investigate reports manually.
14. Please use the **one stickied Piazzas forum thread** for all public exam clarifications.
    a. As with in-person exams, we can’t say much beyond strict clarifications.

# Q1. Logistics and Extra Credit (3 points)

**Q1a (1 pt.).** If you wish your exam to be graded and you understand the instructions from the previous page, type your name in the box below. If you previously submitted an exam file but no longer wish your exam to be graded, type “do not grade” in the box below and re-submit.

| |
|---|
| |

**Q1b (2 pts.).** Type an “X” to the left of every claim that is true about this exam. (Because this is a remote exam and many students are concerned about exam integrity, the instructions for this exam are quite specific. Refer to the Instructions above.)

| | |
|---|---|
| | I can resize answer boxes on this exam to make my answers fit. |
| | I can use material from the Internet on this exam. |
| | I can change the font on this exam to make my answers fit. |
| | I can write examples from the lectures for short answer questions on this exam. |
| | This exam will be graded more strictly than previous assignments. |

**Q1c (2 pts extra credit).** Suppose or refute the claim that Delta Debugging could be used to replace the input minimization algorithm in “Minotaur: Adapting Software Testing Techniques for Hardware Errors”.

| |
|---|
| |

**Q1d (2 pts extra credit).** Argue in favor of one recommended guideline from “Hiring is Broken: What Do Developers Say About Technical Interviews?” and argue against another.

| |
|---|
| |

**Q1e (2 pts extra credit).** Identify a single case study company from Beck et al.'s "Industrial Experience with Design Patterns" that you think is most indicative of modern practice and support that claim.

**Q1f (2 pts extra credit).** What is one thing you liked about this class? What is one thing you would change for next semester?

# Q2. Delta Debugging (22 points)

**Q2a (10 pts.).** Consider Delta Debugging (as defined on Slide 42 from the lecture) applied to the set {0, 1, …, 9}. Suppose Delta Debugging queries Interesting() on the following sets, in order:

{0, 1, 2, 3, 4}
{5, 6, 7, 8, 9}
{0, 1, 5, 6, 7, 8, 9}
{2, 3, 4, 5, 6, 7, 8, 9}
{0, 2, 3, 4, 5, 6, 7, 8, 9}
{1, 2, 3, 4, 5, 6, 7, 8, 9}
{0, 1, 2, 5, 6, 7, 8, 9}
{0, 1, 3, 4, 5, 6, 7, 8, 9}
{0, 1, 2, 3, 5, 6, 7, 8, 9}
{0, 1, 2, 3, 4, 5, 6}
{0, 1, 2, 3, 4, 7, 8, 9}
{0, 1, 2, 3, 4, 7}
{0, 1, 2, 3, 4, 8, 9}
{0, 1, 2, 3, 4, 7, 8}

Assume Interesting() is monotonic, unambiguous and consistent. What was the final result (i.e., the returned one-minimal subset) from that run of Delta Debugging?

**Q2b (4 pts.).** Consider Delta Debugging (as above) with Interesting(X) returning true iff the sum of the numbers in X is greater than 9. What will Delta Debugging return on the set {0, 1, …, 7}?

**Q2c (5 pts.).** Type X left of each claim that is true for the “greater than 9” situation from Q2b:

| | |
|---|---|
| | Interesting() is monotonic. |
| | Interesting() is unambiguous. |
| | Interesting() is consistent. |
| | Delta Debugging returned a one-minimal subset. |
| | Delta Debugging returned a minimal subset. |

**Q2d (3 pts.).** Consider Delta Debugging on {0, 1, 2, 3}. Consider a definition for Interesting(X) such that:

1. There is a set P = {1, 3} such that Interesting(P) is true but P is not one-minimal.
2. There is a different set Q such that Interesting(Q) is true and Q is one-minimal.
3. There is a different set R that is one-minimal but not minimal and Interesting(R) is true.
4. Delta Debugging (from Slide 42 of the lecture) on {0, 1, 2, 3} returns R.

Give a definition for Interesting(X) that accepts the smallest number of sets possible such that all four properties above are satisfied.

# Q3. Localization and Profiling (14 points)

**Q3a (8 pts.).** Create a small Python procedure harrold(x,y) with the following properties:

1. It accepts two integer arguments x and y.
2. It invokes a “divide(_, _)” function that fails when the second argument is zero.
3. It has another line that is simply “safe_function() # false alarm” to mark a false alarm.
4. The positive test {x=1,y=2} does not trigger the bug (i.e., does not divide by zero).
5. The negative tests {x=3,y=4} and {x=5, y=6} do trigger the bug (i.e., divide by zero).
6. A fault localizer, such as Tarantula or Ochiai, ranks the false alarm line as more suspicious than the line with the real bug on the tests {x=1,y=2} and {x=3,y=4} {x=5,y=6}.
7. Your example cannot be taken from class or the readings or the Internet and should be distinct from other student examples if possible.

**Q3b (6 pts.).** You are the interviewer at a company. Type an interview question that centers around profiling. The question must include one opportunity for the applicant to fail to demonstrate technical mastery and also one opportunity for the applicant to fail a behavioral aspect. You must provide a hypothetical situation or framing for the question that is not from the class or the readings or the Internet and is distinct from other student answers if possible.

# Q4. Design Patterns & Maintainability (14 points)

Consider the following three maintenance design goals (A-C) and three design patterns (X-Z).

| | |
|---|---|
| A. Design for Code Comprehension | X. Observer Pattern |
| B. Design for Change Documentation | Y. Singleton Pattern |
| C. Design for Testability | Z. Template Method Pattern |

In all parts of this question, any examples you devise may not refer to course or Internet material and must be distinct from other student examples if possible.

**Q4a (7 pts.).** Choose one design goal (A-C) and one pattern (X-Z). Argue that the pattern specifically supports the design goal. Call out at least two properties of the pattern and two aspects of the goal. Devise a brief example setting in which the pattern supports the goal.

**Q4b (7 pts.).** Choose a different design goal (A-C) and a different pattern (X-Z). Argue that there is at least one example instance in which the pattern, even when applied correctly, specifically hinders the goal. Be concrete about the metric or fashion by which the goal is hindered.

# Q5. Requirements and Elicitation (15 points)

**Q5a (6 pts.).** Consider a requirement elicitation and software development situation involving

1. Requirements involving a quality property of interest are incorrectly captured
2. Stakeholders do not believe the system satisfies that property
3. That system failure is caught by testing but not by static analysis
4. The resolution involves refining the requirements document and the source code

Describe and detail a situation that satisfies the above criteria. You may not use an example from course material or the Internet, and must be distinct from other students if possible.

**Q5b (9 pts.).** Your software company plans to outsource its technical interview activities to a human resources subcontractor that specializes in assessing candidates. The subcontractor will inspect many applicants and identify those that meet your needs. The subcontractor wants a requirements document describing desired candidates. Identify one quality property and one functional property associated with modern skills-based interviews. For each, identify an error or mistake in a hypothetical RE (Requirements and Elicitation) conversation with the subcontractor and how you would correct it. Finally, phrase each as correctly as you can using terminology and best practices from RE. You may not use an example from course material or the Internet, and must be distinct from other students if possible.

| | |
|---|---|
| Quality Property | |
| RE Mistake & Correction | |
| Correct Requirement | |

| | |
|---|---|
| Functional Property | |
| RE Mistake & Correction | |
| Correct Requirement | |

# Q6. Expertise and Productivity (18 points)

**Q6a (8 pts.).** In Chi et al.'s "Expertise in Problem Solving", a claim was made about how experts and novices cluster or categorize problems differently. In class and in the reading, physics problems were considered. You will consider test suite quality metric problems instead. Briefly describe four example problems (A, B, C and D) associated with test suite quality metrics. Your examples must not be isomorphic to each other or inverses of each other (e.g., you cannot base one on line coverage and another on lines not visited). Then explain why novices would cluster {A,B} together and {C,D} together, while experts would cluster {A,C} together and {B,D} together. Your explanations should highlight your mastery of both expertise and the nuances of test suite quality. You may not use examples from course material or the Internet (although you may, and likely will, use metrics from the course material, etc.), and must be distinct from other students if possible.

| | |
|---|---|
| A | |
| B | |
| C | |
| D | |
| Why Novices Cluster {A,B} and {C,D} | |
| Why Experts Cluster {A,C} and {B,D} | |

**Q6b (5 pts.).** In Huang et al.'s "Distilling Neural Representations of Data Structure Manipulation using fMRI and fNIRS", a paper not assigned for this class, the following result is reported: "The brain works measurably harder for more difficult [as measured by Big-Oh notation] software engineering problems (in terms of cognitive load). Moreover, the regions activated suggest a greater need for effortful, top-down cognitive control when completing challenging [array and list data structure] manipulation tasks." Support or refute the claim that this result is aligned with the findings in Siegmund et al.'s "Measuring Neural Efficiency of Program Comprehension".

**Q6c (5 pts.).** In "The Costs and Benefits of Pair Programming", by Cockburn and Williams, the following example is considered: "A team leader given four junior designers to design a graphics workstation, was also given a private office. After a few weeks, he felt uncomfortable with the distance to his team, and moved his desk to the floor with the other designers. Although the distractions were great and his main focus was not teaching the other designers, he was able to discuss with them on a timely and casual basis. They became more capable, eventually reducing the time he had to spend with them and giving them skills for their next project." Briefly describe an example software development situation (not from class, distinct from others, etc.). Then, with respect to that situation, support or refute the claim that pair programming or mentoring is effective process investment in maintainability.

# Q7. Language and Repair (14 points)

**Q7a (8 pts.).** Consider the Eraser analysis (from Savage et al.'s "Eraser: A Dynamic Data Race Detector for Multithreaded Programs"). It may or may not report a race condition and a real race condition may or may not actually be present. For each such combination, describe a situation involving a multi-language Java-and-C project that would result in that outcome. In each case, whether or not the report is made or not and whether it is correct or not should be described in terms of the multi-language aspect, not on other aspects of dynamic analyses (e.g., assume a high quality test suite, etc.). You should not use course examples or examples from the Internet directly, and must be distinct from other students if possible.

| | |
|---|---|
| True Positive | |
| False Positive | |
| True Negative | |
| False Negative | |

**Q7b (6 pts).** Support or refute the claim that "it is a better business decision, with respect to repair cost and repair quality, to use automated program repair in conjunction with automated test input generation than it is to use automated program repair alone". Your argument should speak to these concepts and include an example situation that you devise. Your example should not come from class or the Internet and must be distinct from other students if possible.