15-313 Sample Questions for the Final Exam 2015

The practice exam is intended to give you an overview of the style of questions we may ask on the final. It is *quite a bit* longer than the actual final. Note that this practice final contains only questions referring to the second half of the course. *For example questions on requirements and architecture, see the practice midterm.*

- Please write concise, careful answers. Short and specific is much better than long, vague, and rambling, and grading will reflect this. You have enough time to write clear and readable responses. Bullet points and phrases are acceptable. Spend time to consider how best to present your answers, including citing examples to make your points more concretely.
- Clearly indicate and write your answers in the space provided for each problem. We cannot give you points for answers we cannot find or read.
- The point value of each problem is indicated. We planned the exam to allocate approximately one minute per point.
- If you do not know the answer to a question, you may leave it blank and receive the indicated number of "No-nonsense" points (abbreviated NN for the rest of this exam). This may be an improvement over the zero (0) points that will be awarded a truly incorrect answer. The number of NN points varies per question. If you start to answer a question and then change your mind, you may invoke this rule by crossing out your answer and prominently writing "LEAVING THIS BLANK." Note that we will not read partial answers to questions on which you invoke this rule.
- You may consult **one sheet of paper with printed notes or two sheets with handwritten notes** (both sides). You may not use *or look at* a calculator, cell phone, laptop, or any other electronic or wireless device during the exam.

  --Christian, Claire, Zack, and Gabriel.

Most questions refer to the following scenario (you may remember it from the first lecture!):

You are part of a team at Apple that aims for the next big thing. Just as the iPhone revolutionized the cell phone market with a new user interface, your team is planning to make immersion interfaces for virtual reality *broadly available (think Google glasses and haptic gloves). The company will provide body scanners in all major cities to allow users to create realistic avatars. Three key application are planned from the start: a social network where avatars can interact realistically, a communication infrastructure for everything from chats to business meetings and entire conferences, and a virtual shopping mall where users can virtually try on clothes and shoes for purchase.*

The project has backing from the company's top management and must remain top-secret. The project is considered high risk, high gain, and will require the coordinated effort of several software and hardware teams over at least the next 3 years. You are part of a group of 30 developers tasked with developing the operating system and some demonstration applications for the new hardware and infrastructure (including glasses, gloves, etc). The organization within this group has not yet been decided, but as one of the more experienced staff members, you have significant influence on all aspects of the process.

Decisions under constraint [15p, 4 NN]

Approximately 4 months into the project, you and your team are making good progress on a first prototype. This prototype is still quite rough, but it already provides an exciting glimpse into what might be possible. Despite extensive planning, however, you're growing worried about a number of problems you've begun to notice. You observe:

- *The requirements are so vague that you're having a difficult time making time estimates of any kind. The marketing team constantly has new ideas for the product to include in the first publicly-showcased prototype.*
- *There is no hard deadline or budget limit for this project (yet), but you are eager to show strong early results to ensure that it remains a high company priority. It is also challenging to keep the project secret for long, and there are rumors that competitors are working on similar products.*
- *Because the company's headquarters did not grow with the last round of hirings, your team has very limited space and is distributed over three different buildings. Most of your employees work in open floor plans (cubicles) shared with other teams. Some team members prefer to work from home due to the noisy environment.*
- *Most of your developers are young and highly motivated. However, most of them have more experience with operating system details and algorithms than with software development methods and large-scale projects. They are also keen to experiment, constantly trying new tools from the latest startup. As a result, they spend a lot of time exploring and experimenting with new tools and languages, which often leads to unstable solutions that need to be completely rewritten or even abandoned.*
- *You testing team is happy with the software they receive and find very few bugs. However, when you hand your prototype (whatever is usable right now) to people from other teams, they struggle to figure out how to use it, and complain that the system crashes constantly.*
- *The project heavily depends on various sensors (eye trackers, motion detectors, etc). Several of those sensors are newly-developed and their reliability is as yet not fully established.*
- *The final system is safety-critical. Actually, the tech press has indicated that Congress is considering regulating VR hardware that interfaces with human operators as medical devices, which would require a comprehensive (and expensive!) certification process. Regardless, hacks of the system would create major publicity problems, and failures leading to fatal accidents could ruin the company's reputation.*

Consider how you would proceed in this situation as a project manager. You cannot perfectly achieve all goals, and you will have to make potentially painful decisions regarding tradeoffs. Note that different managers may make different reasonable and justifiable decisions. We will not grade your specific decision, but rather how well you justify it in the context of the scenario. Remain within the realism of the scenario and avoid deus-ex-machina solutions.

Identify and rank the top three *problem areas* on which you should focus your attention and time (such as: meeting particular deadlines, establishing precise requirements, motivating developers, or ensuring the absence of security vulnerabilities). For each of your top three picks, justify why they are (most) important as compared to the others. Note that problem areas (or corresponding risks) do not need to align with individual bullet points outlined above, but should originate from the scenario.

- Top Priority:

  Justification:

- Second Priority:

  Justification:

- Third Priority:

  Justification:

Suggest and justify two actions that you should take immediately (corresponding to your priorities):

-

-

Risk [9p; 3p each; 1 NN each]

For each of the following phases of the lifecycle, identify an important risk (in the scenario) and suggest at least one technique to mitigate the risk through process interventions:

- *Architecture and design*
  Risk:

  Mitigation strategy:

- *Development*
  Risk:

  Mitigation strategy:

- *Testing and quality assurance*
  Risk:

  Mitigation strategy:

Soundness and Completeness [8 pt total, 2 pt each; 2 NN pt if entire question left blank]

In the context of a static or dynamic analysis tool:

1. What is a false positive, and why is it undesirable for a tool to produce many of them? Provide an example.

2. Under what circumstances is it acceptable for an analysis tool to produce false positives?

3. What is a false negative, and why is it undesirable for a tool to produce many of them? Provide an example.

4. Under what circumstances is it acceptable for an analysis tool to produce false negatives?

Static and dynamic analysis

For all parts of this question, refer to the following example code:

```
1· Connection cn;
2· ResultSet rs;
3· try {
4·   cn =
5·     ConnectionFactory·getConnection(/*…*/); // exn possible
6·   do {
7·     StringBuffer qry = /*···do some work …*/;
8·     rs = cn·prepareStatement(qry·toString())
9·           ·executeQuery(); // exn possible
10·  /*···do I/O-related work with rs··· */
11·  rs·close();  // exn possible
12·  done = /* ···some condition···*/
13· } while(!done);
14·} finally {
15·  cn·close();
16·}
```

ResultSet and *Connection* resources that are created must eventually be closed on all paths.

Part 1: Draw a control flow graph for the example code. Leave space to annotate the graph with the results of an analysis. You may assume that the code will not throw any exceptions except for the methods marked (*getConnection*, *executeQuery*, *ResultSet·close*). **[4 pt; 1 NN]**

Part 2: Does an analysis to determine whether opened resources are always eventually closed require a control- or data-flow analysis? Why? **[3 pt, 1 NN]**

Part 3: Static analysis uses abstraction to reduce the possible values of interest to a tractably-low number, such that the entire space of possibilities can be explored. List the abstract states/values you will be tracking per variable, location, or path (depending on what kind of analysis you are performing). Then explain how you will interpret those abstract values to check the specification. (That is, based on the values, when will your analysis issue an error message?) **[4 pt, 1 NN]**

a. States:

b. Interpretation:

[Parts 4--6 are worth 8 points total, 2 NN if you leave all blank.]

Part 4: Given the possible abstract states you listed for part 3, how will you update the analysis values based on what you might see in a basic block in a control-flow graph (put differently: what is the *transfer function*)? You may answer with a textual description, with a formula, or with (pseudo) code.

Part 5: How will you update the abstract state at nodes that have more than one successor (like an if-then-else) (put differently: how do you *split* the state)? You may answer with a textual description, with a formula, or with (pseudo) code.

Part 6: How will you update the abstract state for nodes that have more than one predecessor, for example, the statement after an if-statement or the first statement in a while loop (put differently: how do you *join* the state, or otherwise handle control flow)? You may answer with a textual description, with a formula, or with (pseudo) code.

Part 7: Annotate the program points in the CFG you drew for part 1 with the state *after the analysis has terminated*. If you take notes for intermediate values of the state, cross them out and/or put a box around the final value so we can find it for grading. **[6 pt; 2 NN]**

Part 8: Does your analysis terminate on programs with loops? If so, why? If not, why not? A short textual argument is sufficient, no formal proof required. **[6 pt; 2NN]**

Part 9: Does your analysis identify any possible cases in the example code in which resources are opened but never closed? If so, where/under what conditions? **[3 pt; 1 NN]**

Part 10: Now that you've designed a static analysis for the open/closed resource problem, sketch a dynamic approach to detect the same type of problem. You do not need to give a specific implementation (such as an aspect-oriented program), just an outline. What type of information will you track, where? What are you abstracting in the dynamic version of this analysis (i.e., which information do you track and which information do you not track)? **[4 pt; 1 NN]**

Part 11: What are the limitations (disadvantages) of each of the two approaches for detecting *this type of resource management error*? What are the advantages? Based on the advantages and disadvantages, which approach (if any) do you recommend for detecting this type of error in your organization in the scenario, and why? **[8 pt total; 2 NN, must leave entire question blank]**
Advantages of static analysis:

Disadvantages of static analysis:

Advantages of dynamic analysis:

Disadvantages of dynamic analysis:

Recommendation:

Testing and metrics [8 p; 4 p each; 1 NN p each, must leave entire subquestion blank]

Pick two quality attributes from the following list: *reliability, maintainability, robustness, evolvability*. For each, give a definition and describe why it is or may be relevant to the system from the scenario. Then name, define, and justify the use of a *specific* testing technique that can be used to help evaluate it. Give a criterion or useful metric you could use to evaluate when the *testing* activity has been performed to a sufficient degree (i.e., when to stop testing).

Quality attribute 1:
Definition and relevance:

Testing technique name, definition, and justification:

When to stop (criterion/metric):

Quality attribute 2:
Definition and relevance:

Testing technique name, definition, and justification:

When to stop (criterion/metric):

Comparing techniques

There are many different quality-assurance techniques with different capabilities and tradeoffs. For each of the following QA techniques think about their unique strength and identify an example of a defect that this technique could detect better than all other listed techniques. Explain with one or two sentences why this technique is better suited for detecting this kind of defect than the other techniques. **[12 p; 2 p each; 3 NN if you leave the whole question blank]**

A. Modern code reviews (tool-based, passaround-style inspection of commits)

B. Fuzz testing

C. Dynamic analysis

D. Heuristic static analysis (with false positives and false negatives)

E. Sound static analysis (finds all defects of a class but may report false positives)

F. Formal verification

Quality assurance and process

Discuss each of the following QA practices by explaining advantages and disadvantages in the context of the scenario. Decide whether you would adopt the practice in the scenario. **[9p; 3p each; 1 NN each]**

1. Requiring unit, integration, or smoke tests on commit.

   Advantage(s):

   Disadvantage(s):

   Decision:

2. Requiring all commits to the main repository be reviewed by one or more other developers.

   Advantage(s):

   Disadvantage(s):

   Decision:

3. Testers should be part of every development team.

   Advantage(s):

   Disadvantage(s):

   Decision:

Quality assurance and process, 2

How would you evaluate whether a static analysis tool is worthwhile in your development environment or company? **[4 p; 1 NN]**

QA agree/disagree [20 p; 4p each; 1 NN each]

1. Symbolic execution tracks concrete values of all variables in a program.

2. Manual testing is expensive and should always be automated.

3. Dynamic analysis can ensure the absence of concurrency bugs.

4. With static analysis, inspection is not necessary.

5. A test-driven development strategy ensures that all code is sufficiently tested.

Process practices [20p; 5p each, 1 NN each]

When deciding how to organize the project, you consider several practices that you could integrate into your process. For each practice, briefly define how the practice would affect your process, discuss advantages, disadvantages, and make and justify a decision whether your team should adopt the practice. Make sure that your decision is grounded in the scenario.

Code reviews on every commit (modern tool-based code reviews, passaround style)
Description:

Advantages:

Disadvantages:

Decision and justification:

On-site customer
Description:

Advantages:

Disadvantages:

Decision and justification:

Collective Code Ownership
Description:

Advantages:

Disadvantages:

Decision and justification:

Extensive documentation
Description:

Advantages:

Disadvantages:

Decision and justification:

Working in Groups [8p; 4 points each; 1 NN each]

Three months into the project, you realize that your team is not as productive as you had hoped. You observe the problems below. In the context of the scenario, suggest a specific agile-style mitigation strategy each and explain how it helps.

Process costs: Team members spend time communicating, coordinating, planning, assigning roles, merging results, etc.

Agile technique:

How it addresses the problem:

Social loafing: Individual level of effort diminishes with group size.

Agile technique:

How it addresses the problem:

Process Monitoring [12p]

Your company's CEO has announced a big keynote 5 months from now where he wants to present a prototype of the immersion hardware to the press. This will be a pivotal point for this product and the company is prepared to invest heavily in advertisement. You are fearing the following problem: *"The developers report good progress, but you don't see any concrete results. About two weeks before the keynote you realize that the product would not be anywhere near finished at the keynote and could not be handed to journalists."*

As a project lead, what can you do to understand the *progress* and avoid any last-minute surprises? Name three concrete steps you could take and explain how they could address uncertainty in your project's schedule (in the scenario).

-

-
-

Open source [3p, 1 NN]

Characterize at least two forms of freedom and one restriction the GNU Public License provides for free software as the Linux kernel:

Freedom 1:

Freedom 2:

Restriction: