CS 361 – Fall 2017 – Final Review

Collaboration and Version Control Systems (VCS)

1) Name some ways VCS systems help in software development. Give examples
2) What is the difference between Distributed and Centralized VCS? Name an example of each.
3) What are the advantage of DVCS over Centralized VCS
4) Know the GitHub workflow: what is a good policy to contribute?
5) What is: branching, forking, cloning, origin, master, HEAD

OO Concepts

1) What is encapsulation, inheritance, polymorphism. Explain with an example
2) What is Single Responsibility principle. Explain with an example
3) What is meant by Information Hiding. Explain with an example

Software Process

1) Know the different SDLC phases
2) Name 2 types of software processes. Explain their pros and cons
3) What are the main principles for Agile Programming (hint Agile manifesto)
4) What is the scrum process?

Requirements:

1) Why do we need requirements?
2) Name one technique to gather (elicit) requirements?
3) Name 2 characteristics of a good requirement
4) What is the difference between Functional and Non-Functional requirement?
5) What are the 3 Cs of a user story?
6) What is INVEST?
7) What is an epic?
8) Be prepared to write user stories following the above principles

Architecture Patterns

1) What is meant by an architectural style? Why do we need it?
2) Be able to describe and sketch diagrams of the architectural styles presented in class, and discuss advantages and disadvantages of these.
3) Given a system description, be able to suggest appropriate architectural patterns for realizing it, and argue for their appropriateness.

Designing and evaluating UI

1) What are Nielson's 5 usability goals?
2) What is meant by affordance?
3) Be able to identify the implications of the 4 Psychological Principles behind UI design
    a. Users see what they expect to see
    b. Users have difficulty focusing on more than one activity at a time

    c. It is easier to perceive a structured layout
    d. It is easier to recognize something than recall it
4) What is a persona? Why is it useful when designing your application?
5) What is paper prototyping? Why would you use it?
6) Name one way of evaluating your paper prototype
7) What is the difference between Hi-fidelity and low-fidelity prototype

Design Patterns

1. Why do we want to use design patterns?
2. What are the 3 Design Pattern Categories? What differentiates them?
3. Know a design pattern from each category. Know their pros and cons.
4. Be prepared to propose a design pattern for a given problem or identify the design pattern from piece of code

Code Smells, Refactoring

1. What is technical debt?
2. What is a code smell?
3. What are the 5 types of code smells?
    a. Know the code smells for each type (and why it is bad)
4. What is refactoring?
5. Name one time when it is needed
6. Know the different types of refactoring. Be prepared to perform a refactoring for a code smell (in a piece of code)
7. Why is code review needed?
8. What are the problems of doing a code review via GH pull requests?

Diagrams:

1. Be able to read, interpret, modify, and create UML activity, use-case, class, sequence, and state chart diagrams
1. Given a system description in English or user stor(ies), be able to use any of the foregoing models to represent those systems

Quality and Testing

1) What are the 4 main aspects of a dependable system?
2) What are some of the software qualities that you would want in your system? How will you measure them?
3) What is the difference between Robustness and Reliability; efficiency and scalability; flexibility and reusability;
4) What is the difference between verification and validation?
4) What is a “fault”? What is a “failure”? What is an “error”?
5) Why do we test? What are some things we can reasonably hope to gain from it?
6) Know the basic differences between the two overall testing approaches we’ve discussed (white-box and black-box).
7) Know about choosing inputs (paying attention to boundary conditions)

8) What is equivalence partitioning?
9) Know the different methods test adequacy criteria: statement vs. branch vs. path coverage.
10) Given a program, be able to construct test suites that are adequate according to these criteria for that program.
11) What is JUnit testing? What will you test using JUnit testing?

OSS

1. What is FSF?
2. What is the difference between FSF vs. Open source vs. code available “just” publicly