University of Washington
CSE 403 Software Engineering
Winter 2024

Exam

February 9, 2024

Name: ***Solutions*** _____

CSE Net ID (username): _____

This exam is closed book, closed notes. You have **50 minutes** to complete it. It contains 33 questions and 9 pages (including this one), totaling 100 points.

Before you start, please check your copy to make sure it is complete. When you are finished, turn in all pages, together. **Write your initials on the top of *ALL* pages you turn in** (in case a page gets separated during test-taking or grading).

When you are asked for multiple answers, give answers that are as different as possible. Always give the most important answers.

Unless otherwise directed, answer in **one phrase** (when one answer line is given) or **one sentence** (when two or three answer lines are given).

If your answer to a T/F or multiple-choice answer is contingent on missing information, *briefly* note that. This should be very rare.

Please write neatly; we cannot give credit for what we cannot read.

Good luck!

1 True/False

(1.5 points each, 12 points total) Circle the correct answer. T is true, F is false.

1. **T / [F]** The Joel Test is a checklist for the quality of software product. ***The Joel test assesses a software team or process.***

2. **T / [F]** If code merges without a merge conflict, the merge does not introduce any new defects (bugs). (We define a "new defect" as one that was not in either of the parents of the merge.)

3. **T / [F]** Each commit to your version control system should pass tests. ***It is good practice to commit a failing test on a branch to ensure that it fails, before committing a fix. However, every commit you make to your* main *branch should pass its tests.***

4. **T / [F]** In requirements, an actor is a person who interacts with the system under development. ***Another (external) system can be an actor.***

5. **T / [F]** In requirements, every stakeholder is also an actor.

6. **[T] / F** In requirements, every actor is also a stakeholder.

7. **T / [F]** A mutation is a small modification to program source that introduces a defect (a bug). ***A mutant (the result of a mutation) might not be defective. An equivalent mutant is not defective.***

8. **T / [F]** Given test suite T1 with 50% code coverage and test suite T2 with 60% code coverage, T2 detects at least as many failures (bugs) as T1 does. ***This is true if T1 $\subseteq$ T2, but the question did not state that.***

2 Multiple choice

Choose the best answer.

9. (2 points) A team of 5 programmers is working on a project using git and GitHub. How many repositories exist (each of which stores its own version of the history of your project)?

   (a) 0 or more
   (b) 1
   (c) 1 or more
   (d) 5
   (e) 5 or more
   (f) 6
   [(g)] 6 or more***. One programmer might work on two different computers.***
   (h) 7
   (i) 7 or more

3 Choose all that apply

(4 points each, 20 points total) Mark all the correct answers, by circling the appropriate letters.

Note that the grading is nonlinear. For example, if you get 4 or more incorrect in question 10, 13, or 14, your score is 0.

10. Which git commands can be run without network access, assuming typical usage of Git and GitHub?

    [(a)] bisect

    (b) clone

    [(c)] commit

    (d) fetch

    (e) fork

    [(f)] merge

    (g) pull

    (h) push

11. Which of the following are true of a merge from the main branch into a feature branch?

    [(a)] It should be done prior to a merge of the feature branch into main

    (b) It should be accomplished via approving a pull request on GitHub.com

    (c) It cannot result in a merge conflict

12. Which of the following are best practices for avoiding merge conflicts?

    (a) Create a branch for each distinct task***. This actually increases the possibility of merge conflicts — but nonetheless it is a desirable practice.***

    [(b)] Pull often

    [(c)] Design your software modularly

13. Which of the following might appear in an architectural diagram?

- [x] (a) a web client
- [ ] (b) CSS
- [x] (c) a database
- [ ] (d) design patterns
- [ ] (e) Java
- [ ] (f) React
- [x] (g) a server
- [ ] (h) the user

We gave credit for "(d) design patterns" regardless of the answer. An architectural diagram doesn't contain a design pattern per se, but it may contain components and connectors that are in the shape of the canonical instantiation of a design pattern.

14. Which of the following events might trigger (start) a continuous integration workflow?

- [ ] (a) save code edits
- [ ] (b) compile
- [ ] (c) run static analysis
- [ ] (d) run tests locally
- [ ] (e) git commit
- [x] (f) git push
- [ ] (g) git pull
- [x] (h) create a pull request
- [x] (i) open an issue or bug report

4 Very short answer

Answer in one word or phrase.

15. (3 points) What architectural style is most appropriate for implementing a compiler?

    Pipe and filter

16. (4 points) Name the 7 main distinct steps in the software development lifecycle, in order.

    (a) ***Requirements***
    (b) ***Architecture***
    (c) ***Design***
    (d) ***Implementation***
    (e) ***Verification (or testing or debugging)***
    (f) ***Delivery or release***
    (g) ***Maintenance***

17. (3 points) What is the purpose of mutation testing?

    To evaluate (and/or improve) a test suite.
    More precisely, mutation testing is useful to compare two test suites, by comparing their mutation scores. The mutation score for a single test suite does not give an absolute measure of test suite quality.

18. (4 points) State three ways that a build system can speed up a build, in one brief phrase (1–2 words each is adequate).

    (a) ***Incrementalize: don't re-run a task if its inputs did not change***
    (b) ***Parallelize: run independent tasks at the same time (in different threads)***
    (c) ***Reuse: Compute a hash of a task's inputs. If the task has been run on that exact input (e.g., by a different team member), use the previous run's output without recomputing. This requires a build cache, usually in the cloud.***

19. (6 points) Requirements

    (a) Give an example of a functional requirement.
    A functional requirement maps inputs to outputs.
    "The user can search either all databases or a subset."
    "Every order gets an ID the user can save to account storage."

    (b) Give an example of a non-functional requirement.
    A nonfunctional requirement is other user-visible properties.
    "ilities": dependability, reusability, portability, scalability, performance, safety, security
    "Our deliverable documents shall be in the XYZ format."
    "The system shall not disclose any personal user information."

    (c) Give an example of a requirement that is neither functional nor non-functional.
    Anything having to do with the development process, such as constraints on programming language, frameworks, testing infrastructure, budget, software development lifecycle, etc.

20. (3 points) What are the two major concepts that an architecture defines?

    (a) ***components***
    (b) ***connectors***

21. (3 points) Give an example of a file that should *not* be committed to version control, or say "none" if all files should be committed to version control.

    Generated files, such as executables. Private information such as passwords. IDE preference files, often (though preferences about formatting, such as indentation, should be the same for all programmers and may be checked in).

22. (5 points) Alice and Bob are developing using GitHub, each one with a clone on their own laptop. Alice's repository is up to date. Alice makes an edit and saves a file. What is the smallest number of git commands that enable Bob to see Alice's changes in his own repository?

    State the actor (Alice or Bob) and the command. Leave blank any of the answer lines that you do not need.

    (a) **Alice** ***runs*** `git add`. ***(Optional; it can be combined with*** `git commit`.***)***
    (b) ***Alice*** runs `git commit`.
    (c) **Alice** ***runs*** `git pull`. ***(Optional; needed if Alice's repository is not up to date.)***
    (d) ***Alice*** runs `git push`.
    (e) ***Bob*** runs `git pull`.

    Also correct: replace "(e) Bob runs `git pull`" by
    (e) ***Bob*** runs `git fetch`.
    (f) ***Bob*** runs `git merge`.

    It is not necessary for Bob to run `git checkout`. ***Doing so puts Alice's changes in Bob's working copy, but the*** `git pull` ***already put Alice's changes in Bob's repository.***

5 Short answer

23. (4 points) What is the difference between verification and validation?

    Validation ensures that you build the right thing. It is concerned with whether the system meets its requirements and pleases its users.

    Verification ensures that you build the thing right. It is concerned with whether the code satisfies its specification.

24. (4 points) What is the difference between git and GitHub?

    Git is a program that implements a version control system. GitHub is a hosting service that runs git and acts like a remote repository. GitHub also provides services like an issue tracker and pull requests, neither of which git knows anything about.

25. (4 points) What is a software design pattern?

    A design pattern is a well-known, industry-preferred way to solve a common problem — one that arises relatively frequently when building software. Note that design principles $\neq$ design patterns $\neq$ design.

26. (6 points) Give 3 general approaches, that are as different as possible, to obtain information from customers when gathering requirements.

    (a) ***Ask them: interviews, surveys, hallway conversations***

    (b) ***Observe them: observations, shadowing***

    (c) ***Get feedback: show them use cases (note, not just*** **writing** ***them), feature list, mockups, prototypes***

27. (4 points) Define "technical debt".

    Poor development practices that will cost more to fix in the future than they would cost to fix now.

6 Git bisect

For all of this section, assume that the version control history is linear, containing n commits.

28. (2 points) What is the best-case complexity of `git bisect`, in big-O notation?

    $O(\log n)$. ***The answer is not "$O(1)$" because `git bisect` needs to test two adjacent commits to ascertain the commit that introduced the problem.***

29. (5 points) Give a modification to the `git bisect` algorithm that improves the best-case complexity without degrading the average-case or worst-case complexity.

    After testing a given commit that passes, test the previous commit before proceeding with the bisection algorithm.
    Or: ***After testing a given commit that fails, test the following commit before proceeding with the bisection algorithm.***

    "Binary search" is an incorrect answer because `git bisect` already performs binary search.
    It is not possible to precompute and cache because the assessment (e.g., running a specific test case) of each commit is done interactively by the user, or is specified by the user when running `git bisect`; git has no way to know in advance what that assessment will be.

    No credit is given below unless credit was given for this question. (We don't want to reward guessing below.)

30. (1 point) What is the new best-case complexity? $O(1)$

31. (1 point) What is the new worst-case complexity? $O(\log n)$

32. (1 point) What is the new average-case complexity? $O(\log n)$

33. (3 points) Why haven't the git maintainers implemented this improvement?

    (a) It doubles the constant factor in the average and worst case, so overall it increases the work that `git bisect` must perform.

    (b) The best case is uncommon.