Final Exam Review

1. In a few sentences, informally describe the meaning or relevance of following terms with respect to verification and validation:

   - Verification
   - Validation
   - Software metric
   - Defect
   - Failure
   - Defect classification
   - Static analysis
   - Software metric
   - Model checking
   - Theorem proving
   - Specification language

2. What is black box testing?

3. What is the single fault assumption, and how does it affect our testing?

4. How does the role of coverage metrics differ in black-box testing and white-box testing?

5. What is a basis path set?

6. What are two static checks we can perform using a data flow (def-use) graph?

7. What is said to be the difference between verification and validation? Give an example activity for each.

8. Give the equivalence partitions generated using the specification below.

9. Give a test case for the partition below which would only be created using (weak, strong) (normal, robust) boundary value analysis.

10. Give the decision table representing the specification below. Make sure to mark impossible rules.

11. Consider the method and test cases given below. What is the “highest” level of coverage (condition, decision, condition/decision, modified condition/decision, multiple condition) achieved?

12. Give a set of inputs for the Boolean expression below which would achieve (condition, decision, condition/decision, modified condition/decision) but not (condition, decision, condition/decision, modified condition/decision, multiple condition).

13. What is the difference between a general control flow graph and a DD path graph? Why are both acceptable to use for analyzing path coverage? Which one is preferred and why?

14. In a few sentences, summarize how data flow testing and control flow (basis path) testing are similar and how they are different.

15. What is the difference between a tool like PyLint and a tool like FindBugs?
16. What is the relationship between a(n) \{defect, error, fault, failure, problem\} and a(n) \{defect, error, fault, failure, problem\}?

17. How are equivalence partitioning and boundary value analysis related? Can I have, for instance, strong robust equivalence partitioning with weak normal boundary value analysis?

18. In “theory”, equivalence partitioning by itself should be a sufficient test case generation strategy. Why do we also have boundary value analysis?

19. Draw a control flow graph for the given Java method.

20. Generate a basis path set for the given Java method.

21. Draw a data flow (def-use) graph for the given Java method.

22. List the elements of the sets below, along with a subpath for each element.

23. Give a good set of pre-conditions and post-conditions for the following method definition. You may describe your conditions in any clear formal notation.

24. What is one benefit of model checking over theorem proving?

25. What is one benefit of theorem proving over model checking?

26. Give two reasons a formal specification can be useful, particularly given the relatively high cost of implementation over other ways of specifying desired behavior.

27. For the hypothetical failure sketched below:
   (a) What is the defect?
   (b) Of the two parties mentioned in the scenario, who would you say is most to blame for the failure?
   (c) Justify your answer to the previous part. In particular, provide a paragraph describing what mistakes your blamed party made, and another paragraph describing how the mistakes of the other party were not as “egregious”.

28. When might you prefer using (equivalence partitioning, decision tables) over (decision tables, equivalence partitioning) for a black box testing strategy?

29. Give two benefits of using (decision tables, equivalence partitioning) to create test cases.

30. Several standards for safety-critical software (such as DO-178C) require organizations only use black-box testing. Defend this choice.

31. Generate a set of test inputs that would realize all basis paths for the given Java method.

32. Generate a set of test inputs that would provide (all-defs, all-c-uses/some-p-uses, all-p-uses/some-c-uses, all-uses) coverage for the given Java method.

33. Argue that proper use of (black-box testing, white-box testing, static analysis techniques, specification languages) would have prevented the hypothetical failure described below.