Answers to Guide 11

* Software Quality
  + How would you define software quality? Does your definition match either of those notions listed in the introduction?
    - Software functional quality: follows functional requirements – did you build the right product?
    - Software structural quality: follow non-functional requirements – is the product built right?
  + Be familiar with Garvin’s five perspectives on quality.
    - The transcendental perspective deals with the metaphysical aspect of quality – “something toward which we strive as an ideal, but may never implement completely”
    - The user perspective is concerned with the appropriateness of the produce for a given context of use.
    - The manufacturing perspective represents quality as conformance to requirements.
    - The product perspective implies that quality can be appreciated by measuring the inherent characteristics of the product.
    - The value-based perspective recognizes that the different perspective of quality may have different important, or value, to various stakeholders.
  + How software quality and testing are related?
    - Software quality focuses on the process while software testing focuses on the end product.
    - Software quality assurance is a systematic and planned approach to identifying, defining, and modifying a set of processes that can help one attain an error-free end product.
    - Software testing is the implementation of these processes and making them a reality to deliver an error-free end product.
    - Software quality assurance speaks of creating and implementing methods and procedures to improve the overall development cycle while software testing concentrates on verifying and validating the product to discover bugs and locate defects.
  + Do you have any affinity with DiMarco’s alternate definition of software quality?
* Software Testing
  + What is the primary purpose of software testing?
    - Investigation conducted to provide stakeholders with information about the quality of the software product or service under test.
  + Be able to compare and contrast the definitions of the following sets of testing terms:
    - Concepts:
      * Software fault: occur through various sequence of processes.
        + Programmer error 🡪 defect in source code 🡪 system produces wrong results, causing a failure.
      * Programmer error: A mistake during the coding process.
      * Defect: requirements gaps resulting from unrecognized requirements that result in errors of omission by the program designer.
        + Often non-functional requirements such as testability, scalability, maintainability, usability, performance, and security.
      * Failure: The software system produces wrong results, caused by a defect (fault, bug).
    - Methods:
      * Static testing: reviews, walkthroughs, and inspections. (verification)
      * Dynamic testing: executing programmed code with a given set of test cases. (validation)
      * White testing: Verifies the internal structures or workings of a program, as opposed to the functionality exposed to the end-user.
      * Black-box testing: treats software as a “black box”, examining functionality without any knowledge of internal implementation, without seeing the source code.
    - Levels:
      * Unit testing: tests that verify the functionality of a specific section of code, usually at the function level.
      * Integration testing: any type of software testing that seeks to verify the interfaces between components against a software design.
      * System testing: tests a completely integrated system to verify that the system meets its requirements.
      * Acceptance testing: used to conduct operational readiness (pre-release) of a product, service, or system, as part of a quality management system.
    - Types:
      * Regression testing: focuses on finding defects after a major code change has occurred; seeks to uncover software regressions, as degraded or lost features, including old bugs that have come back.
      * Alpha testing: simulated or actual operational testing by potential users/customers or an independent test team at the developers’ site.
      * Beta testing: comes after alpha testing and can be considered a form of external user acceptance testing – testing performed by the customer, often in their lab environment or on their own hardware.
      * Functional testing: refers to activities that verify a specific action or function of the code – answers question of “can the user do this” or “does this particular feature work”.
      * Non-functional testing: refers to aspects of the software that may not be related to a specific function or user action, such as scalability or other performance behavior under certain constraints or security.
      * A/B testing: a method of running a controlled experiment to determine if a proposed change is more effective than the current approach.
  + According to McConnel (the cost vs. time table and references 11), how costly is a defect introduced in the requirements phase and not found until the system testing phase?
    - 10x more costly if found in system testing phase versus during requirements review.
  + Software quality folk often use the phrase “V & V”. What does it mean?
    - Verification and Validation.
      * Verification: did we build the right product?
      * Validation: did we build the product right?
* Project Management Tools & Practices
  + Familiarize with GitHub’s pull request feature.
    - I know the basics of pull requests.
    - We review our own pull requests and ensure there’s no merge conflicts before merging them ourselves.
  + Familiarize with guidelines in Google Java Style
    - Where do they put the first curly brace for general control structures, on the line or on a new line?
      * “No line break before the opening brace” – they put it on the line.
    - How wide can columns be?
      * 100 characters maximum.
    - As a minimum, where should Javadoc comments be used?
      * Every public class, and every public or protected member of such a class, with a few exceptions.
        + Exceptions include self-explanatory methods, override methods.
        + Other classes and members outside of the above criteria can use Javadoc as deemed necessary.
  + Familiarize with GitHub’s issue tracking feature.
    - For what are issue trackers commonly used?
      * Bug tracker for open source projects.
      * According to GitHub’s guide, you can also use it for request for recipes.
    - How would you integrate Git issue tracking with Trello?
      * <https://blog.trello.com/how-to-transform-trello-into-a-powerful-bug-tracker-with-the-marker-power-up>
      * The above is a guide giving a method of integrating issue tracking with Trello.
      * <https://marker.io/trello-integration-bug-tracker>
      * The above is a link to a addon for Trello that is specifically for tracking and reporting bugs.
    - Make sure that you can find the issue tracker for your team’s server and client repos
      * Yes, I know where it is.