Software Testing

Quality Assurance

Quality Assurance - is a process that assures that all software engineering processes, methods, activities , and work items are monitored and comply with the defined standards

Important

QA is very important, and could lead to death or injury if not properly done

Challenges of QA

- How can we ensure that the specifications are correct?
- How can we ensure a system meets its specifications?
- How can we ensure a system meets the needs of the users?
- How can we ensure a system does not behave poorly?

Verification and Validation

QA includes verification and validation to ensure the quality of software products
Verification: the process of evaluating the system or components to determine if they are built correctly

- Code Review
- Testing
  - Setting testing
  - Functionality testing

Validation: checks whether the software conforms to the customers expectations and requirements

Code Review

- A constructive review of a fellow developers code. A required sign-off from another team member is permitted to check in changes or new code
- Common industry practice
- Made easier by tools such as GitHub
  - Integrate with configuration management systems
  - highlight changes
  - allow traversing back into history

- **Who**: Original developer and reviews, sometimes in person.
- **What**: Reviewers give suggestions for improvement on logical and/or structural level to conform to previously agreed upon set of standards
- **When**: When code author has finished a coherent system change that is otherwise ready for checkin
  - Should not be too large or too small
  - Before committing the code to repository or incorporating it into the new build

> **Note**
>
> Issue Rate
> Unit - 25%
> Function - 35%
> Integration - 45%
> Design - 55%
> code inspection - 60%

Software Testing

- Software testing is the process of evaluating and verifying that a software product does what it is supposed to do
- Variety of software testing:
  - Testing the settings
  - Testing functionality
- Testing can demonstrate the presence of the bugs, but not their absence
- No matter how much testing is done, bugs can still hide in the code
- Test coverage: measure used to describe the degree to which the source code of a program is executed when a particular test suite runs
  - Higher coverage -> lower chance of bug in software going undetected

Types of Testing

- **Test To Specifications**: (black box), use the specifications to select test case, ignoring the actual code. Checking the input and the output.
- **Test to code**: (white box), test code in detail, ignoring the specifications, use the code to select test cases
- **Main Goal**: Demonstrate the software can be depended upon, given diverse tests
- **Worst Way** - Random Testing
  - There is no time to test all the tiniest fraction of all possible test cases.
  - We need to chose cases carefully in order to avoid repetition, systematically.

Test Case

- Select a small manageable set of test cases
    - Maximize the chances of detecting a fault
    - Minimize waste
    - Until we reach acceptable dependability for the software
- First **black box** then **white box** testing
- **Equivalence Testing:** Any one member of an equivalence class is as good a test case as any other member of the equivalence class
- **Boundary Value Analysis**: Select test cases on and just to the side of the boundary of equivalence classes
    - Increases probability of finding a fault

White Testing

- Independent from other tests
- One test class per independent
- Test each feature once
- Include trivial cases and uncommon cases
- Test for success and failure
- The purpose of testing is to expose faults in **Interaction between integrated units**