Chapter 1. Effective & Systematic Software Testing

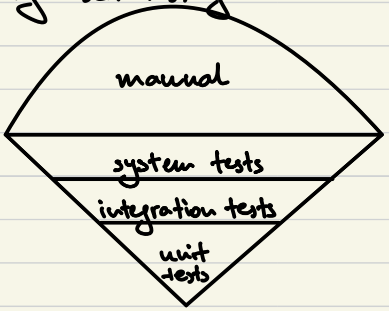
1. Systematic testing

array
- → null
- → empty
- → 1 element
- → many elements

} validations

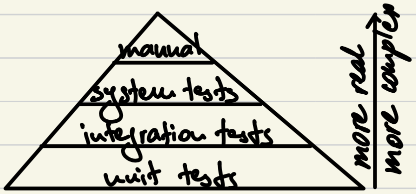
domain testing
structural testing
example-based testing
property-based testing

2. Effective software testing



→ requirement analysis
→ TDD (test-driven development) cycles
→ implement testable units
→ domain, boundary, structural testing
→ integration, system testing
→ automated testing - test case generalization, mutation testing, static analysis
→ release the feature

3. Principles of software testing



→ Exhaustive testing is impossible
→ Knowing when to stop testing
→ Variability is important (pesticide paradox)
→ Bugs happen in some places more than others (defect clusters)
→ Testing will never be perfect
→ Context is king
→ Verification is not validation (absence-of-errors fallacy)

4. Testing pyramid

→ unit testing - fast, easy control, easy write, lack reality, bugs not caught
→ integration → tests a component of the system and some external component
→ system → tests overall functionality of the project (slow, hard to write)