2

- Fault, Error, Failure
- State of a program (wrong vs. expected)
- RIP Fault Model (fault to failure)
- Dealing with Faults
- Testing vs. Debugging

3

- QUESTION: given faulty program, identify stuff
- Ways of validating a test suite, line intersect example

4

- Static Testing
- Dynamic Testing (black-box, white-box)
- Naughty words
- When to stop testing?
- Test Case
    - Observability
    - Controllability
    - Anatomy
    - Test Set, Script
- Coverage, Coverage Level
- Test Requirements

5

- Exploratory Testing
    - When it does well
    - The process
    - Output from process

6

- CFG intro
- Steps in compilation
- Abstract Syntax Tree (AST)

7

- CFG
    - Basic block
    - If statements
    - While statements
    - For statements

8

- Large CFG examples
- Test Path Definition
- Coverage definition
- Test Case -> Test Path mapping
- Test case, test path examples
- Deterministic vs. Nondeterministic methods
- Statement coverage definition and example
- Branch coverage definition and example

9

- Complete Path Coverage criterion
- FMS definition
    - Node coverage
    - Edge coverage
    - Edge-pair coverage
    - Round trip path
    - Simple Round Trip Coverage (SRTC)
    - Complete Round Trip Coverage (CRTC)
    - Deriving FSM (best with a modelling state)
    - Advantages of FSMs
    - Disadvantages

10

- Applications of CFGs (input-space, program-based)
- Generating Inputs
    - Using Grammars vs. RegExps
    - Recognizers vs. Generators
    - Grammar Mutation Operators
        - Nonterminal Replacement
        - Terminal Replacement
        - Terminal and Nonterminal Deletion
        - Terminal and Nonterminal Duplication
    - Using Grammar Mutation Operators, and why?

11

- Fuzzing definition + back story
    - Generation-based
        - HTML5 example
        - C example
    - Mutation-based
    - Advantages + Disadvantages
- Chaos Monkey story

13 (no 12)

- How much testing is enough? Conclusions at end
- JUnit case study
    - Causes of uncovered code (too simple, empty method, etc.)

14

- Mutation Testing
    - Ground String + Mutation Operator
    - Mutant, Killing Mutants
    - Mutation score
    - Example Mutant, and at end
    - Uninteresting Mutants (stillborn, trivial, etc.)
- GOALS of Mutation Testing

15

- Strong Mutants
- Weak Mutants
- Strong Mutation Coverage (SMC)
- Weak Mutation Coverage (WMC)
- Examples of Strong and Weak Mutations
- Mutation testing workflow diagram
- Integration Mutation
- A bunch of example mutation operators
- Mutation for OO Programs
- Exercise: Killing mutants with test cases
- Syntax-Based Testing summary chart

16

- Is mutation testing any good? Case Study
- Is coverage at good means for testing?

17

- Engineering Test Suites
- Not delaying the writing of test cases

18

- Reason to test: avoiding regressions
- Test Design Principles
    - Many small tests
    - Easy to add tests
    - Unit vs. Integration
    - “TODO: write tests”
    - Avoid testing internals

19

- Selenium (what is it)
- Anatomy of Selenium JUnit Test
    - Setup/Tear-down
    - Example Test
    - Waiting
    - Page Objects

21

- Bug-Finding
- Specifications (examples)
- Coverity (brief intro)
    - Mistaken Beliefs (about how langs work)
    - Contradictions + Deviances
    - MUST-beliefs
    - MAY-beliefs
    - MUST-belief examples

22

- MUST-beliefs continued
    - Redundancy Checking
- Process for verifying MAY beliefs
    - Finding custom free fnxns
    - Finding null returning routines
    - General function pairing (assignment)
        - Support
        - Confidence

23

- Regression Testing
    - Attributes
    - Automated Regression Tests
        - Input
        - Output
    - Industrial Best Practices (Tests)
        - Unit Tests, Code Reviews, etc.th

25

- Self-checking tests
- State-based tests
- Behaviour-based tests
- State Verification Example
- Procedural Behaviour Verification Example
- Assertions
    - Alternative: External Result Ver.
- Verifying Behaviour
    - Procedural behaviour verification
    - Expected behaviour specification
- Improving Test Cases
    - Reducing Test Code Duplication
    - Avoiding logic in test cases

26

- Test doubles
    - Stubs
    - Mock Objects
    - JMock example
    - EasyMock example
- Flaky Tests
    - Dealing with them
    - Causes and Solutions
- Continuous Integration
    - Continuous Deployment

27

- More on Flaky Tests (Google blog post)
- Fixing flaky tests with Mocks/Stubs
- Airbnb testing infrastructure

28

- Code Review
    - Purpose + Formatting
    - Things to look out for
- Comments and Code Documentation
    - Method specification

29

- Reporting Bugs
- Properties of Important Bugs
- Reproducibility
- Anatomy of a Bug report (usual fields)
    - Bookkeeping fields
    - Description, etc.
- Properties of a Good Bug Report
- Bug Triage

30

- Static Code Analysis Tool: PMD
- PMD built-in rulesets (Java Examples)
- Writing your own PMD rules (XPath)
    - Example XML and rules

31

- More XPath PMD rules (Java)
- Invariants that hold wrt code
- Using Linters (JSHint) + example

Note: JSHint and PMD, are AST level tools

32

- FindBugs (bytecode level tool static analysis)
    - False Positives
- Coverity introduction
- Developer-provided specifications (Korat tool)

33

- Facebook Infer
- Infer Eradicate (detecting null ptrs)
- Infer Analyzer: Leaks (Resource + Memory)
- Infer and Tainting (security/private issues)
- Static vs. Dynamic Analysis
    - Virtual Method Call Resolution
    - Checking data structures for cycles
    - Unreachable code

34

- JML (Java Modelling Language)
- Static Java Verification: ESC/Java2
- Dynamic Java Verification (asserts)

35

- Dynamic Analyses (Analyzers)
    - Memory Error Analyzers
        - Valgrind Memcheck
        - Clang Address Sanitizer
        - Implementation Techniques
    - Race Detectors (concurrency errors)
        - Helgrind (via Valgrind)

36

- Dynamic Tool: Randoop (test generator)
- Course Summary