Objectives

- Coverage
  - Strengths, Limitations
  - Tools: EclEmma

1

How to Implement an Effective Solution

- Understand the problem
- Understand external constraints
- Design an effective solution to the problem
- While designing the solution, design some ***tests*** to verify that the problem is solved (and remains solved)
- Code the effective solution to the problem
- Teach other team members about your solution to the problem

2

How to Implement an Effective Solution

- Understand the problem (interact with ***people***)
- Understand external constraints (interact with ***people***)
- Design an effective solution to the problem
- While designing the solution, design some ***tests*** to verify that the problem is solved (and remains solved)
- Code the effective solution to the problem
- Teach other team members about your solution to the problem (interact with ***people***)

Probably interacting with people while designing, testing, and coding too

Some Approaches to Testing Methods

- Typical case
  - Test typical values of input/parameters
- Boundary conditions
  - Test at boundaries of input/parameters
  - Many faults live "in corners"
- Parameter validation
  - Verify that parameter and object bounds are documented and checked
  - Example: pre-condition that parameter isn't null

➥ All black-box testing approaches

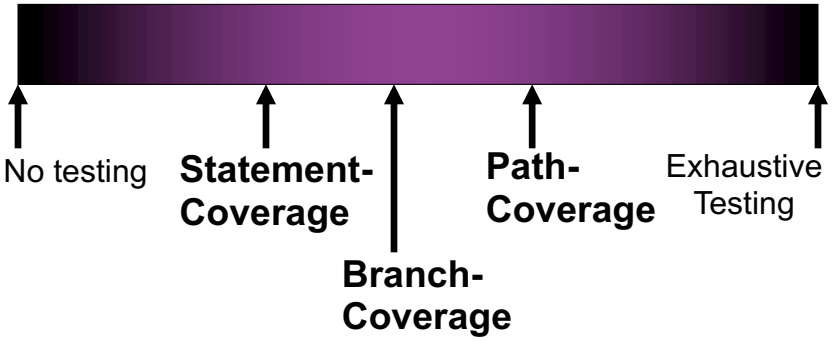
Characteristics of Good Unit Testing

- **Automatic**
  - Since unit testing is done frequently, don't want humans slowing the process down
  - Automate executing test cases and evaluating results
  - Input: in test itself or from a file
- **Thorough**
  - Covers all code/functionality/cases
- **Repeatable**
  - Reproduce results (correct, failures)
- **Independent**
  - Test cases are independent from each other
  - Easier to trace fault to code

Review

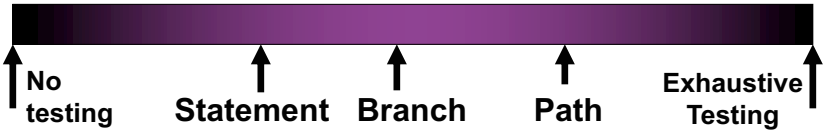
1. What is code coverage?
2. What are examples of code coverage criteria?
3. Compare the strengths/limitations of each of those criteria
4. How could you use code coverage/coverage criteria?
5. Can we use code coverage in the testing project?

Testing Continuum



7

Comparison of Coverage



Coverage Criterion	Advantages	Disadvantages
Statement		
Branch		
Path		

8

Comparison of Coverage

Coverage Criterion	Advantages	Disadvantages
Statement	Practical	Weak, may miss many faults
Branch	Practical, Stronger than Statement	Weaker than Path
Path	Strongest	Infeasible, too many paths to be practical

9

How Can We Use Coverage Criteria?

10

Uses of Coverage Criteria

- "Stopping" rule → sufficient testing
  - Avoid unnecessary, redundant tests
- Measure test quality
  - Dependability estimate
  - Confidence in estimate
- Specify test cases
  - Describe additional test cases needed

11

Coverage Criteria Discussion

- Is it always possible for a test suite to cover all the statements in a given program?
  - No. Could be infeasible statements
    - Unreachable code
    - Legacy code
    - Configuration that is not on site
- Do we need the test suite to cover 100% of statements/branches to believe it is adequate?
  - 100% coverage does not mean correct program
  - But < 100% coverage does mean testing inadequacy

12

True/False Quiz

- A program that passes all test cases in a test suite with 100% path coverage is bug-free.
  - **False.**
  - Examples:
    - The test suite may cover a faulty path with data values that don't expose the fault.
      - Towards Exhaustive Testing
    - Errors of omission
      - Missing a whole if

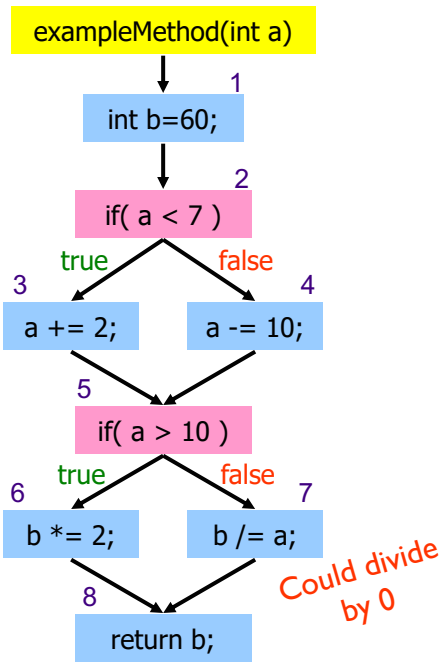
Example

Test Suite:

3-7: a=3
4-6: a=30
3-6: a=6
4-7: a=9

But, error shows up with

3-7: a=0
4-7: a=10



Example

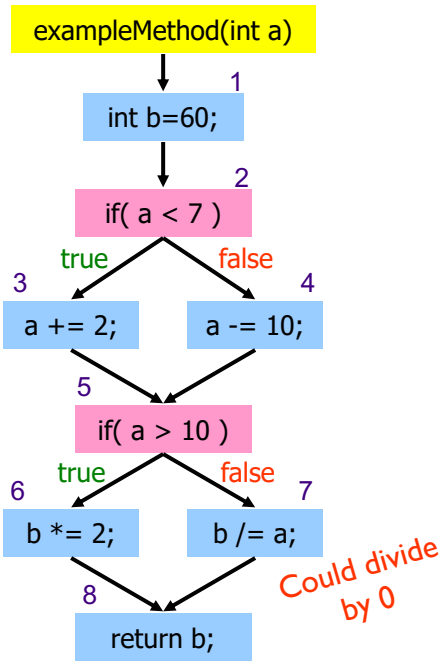
Test Suite:

3-7: a=3
4-6: a=30
3-6: a=6
4-7: a=9

But, error shows up with

3-7: a=0
4-7: a=10

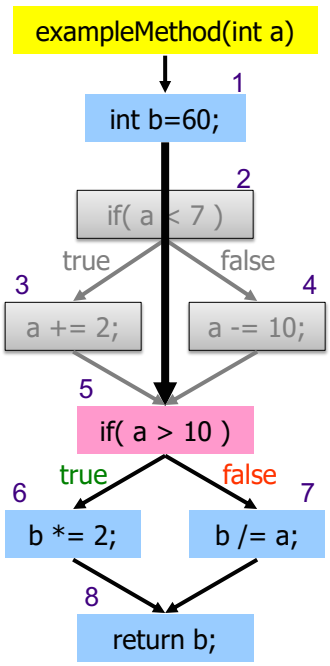
Also an error of omission:
- Should have statement 7 within an if statement that checks value of a



Omission Example

Consider if the first `if` block wasn't in the code.

You could cover all the paths, but you're missing a crucial condition.



True/False Quiz

- When you add test cases to a test suite that covers all statements so that it covers all branches, the new test suite is more likely to be better at exposing faults.
  - **True**.
  - You're adding test cases and covering new paths, which may have faults.

17

Which Test Suite Is Better?

Statement-adequate Test Suite

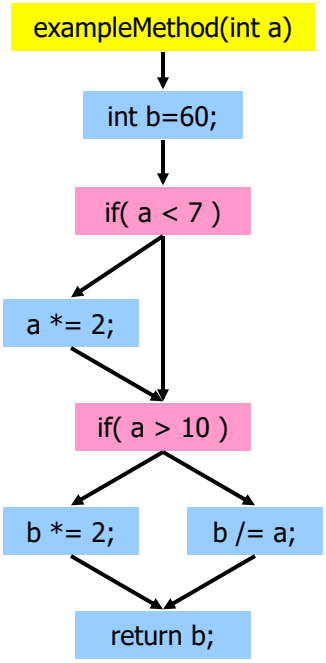
Branch-adequate Test Suite

- Branch-adequate suite is not *necessarily* better than Statement-adequate suite
  - Statement-adequate suite could cover buggy paths and include input value tests that Branch-adequate suite doesn't

18

Example

- TS1 (Statement-Adequate):
  - a=0, 6
- TS2 (Branch-Adequate):
  - a=3, 30
- Statement-adequate will find fault but branch-adequate won't
  - Covers the path that exposes the fault



Software Testing: When is Enough Enough?

- Need to decide when tested enough
  - Balance goals of releasing application, high quality standards
- Can use program coverage as "stopping" rule
  - Also measure of confidence in test suite
  - Statement, Branch, Path and their tradeoffs
  - Use coverage tools to measure statement, branch coverage
- Still, need to use some other "smarts" besides program coverage for creating test cases

No Silver Bullet

- Recall the Fred Brooks' quote:
  - "There is no single development, in either technology or in management technique, that by itself promises even one order-of-magnitude improvement in productivity, in reliability, in simplicity."
  - Known as "no silver bullet"
- Test coverage is one tool that will help us improve the quality of our code, but it will not solve everything

21

COVERAGE TOOLS

22

Coverage Tools

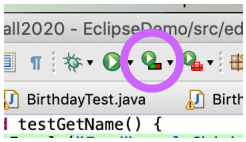
- Coverage is used in practice
- Don't need to figure out coverage manually
- Available tools to calculate coverage
  - Examples for Java programs: Cobertura, Clover, JCoverage, **Emma**
  - Measure statement, branch/conditional, method coverage

Eclipse Plugin: EclEmma for Coverage

- Eclipse can be extended through ***plugins***
  - Provide additional functionality
- EclEmma Plugin
  - Records executing program's (or JUnit test case's) coverage
  - Displays coverage graphically
- Built into Enterprise Edition of Eclipse
  - What you were supposed to install
  - If you got the regular version of Eclipse, you'll need to install the EclEmma plugin

Demonstration

- Execute test with coverage



Note: Coverage and Testing Project

- You won't be able to leverage coverage for the testing project
  - Even if you write code for the Car class, it's not _my_ code.
- Challenge of test-driven development (TDD)
  - Common practice in industry

More Testing Tools, Frameworks

- Mockito
  - Mock objects before have other code
  - Allows you to test in isolation, e.g., mock the payment system so you focus on your code
- Cucumber
  - Behavior-driven development
  - Language parser: Gherkin
- Many more

Looking Ahead

- Testing Project due
  - Added an FAQ
    - Updated as I get more questions
  - Monday – before class: Tests due
  - Wednesday – 9:59 a.m.: Individual analysis due
- Monday:
  - Eclipse debugger
  - Design in the Small