CSC 172 – Data Structures

Test-Driven Development
How do YOU develop?

- Design
- Build Code
- Run Main Program
- While (Find Bug)
  - Fix Bug
Finding Bugs

• Modify Code to
  – Add PRINT statements
  – Set BREAKPOINTS and examine state
  – Try different inputs and detect pattern
Found Bug

• Repeat
  – Fix Code

• Until bug squashed
Found Bug II

- Repeat
  - Fix Code
- Until bug squashed
- Remove PRINT statements
- Remove BREAKPOINTS
- Why?
  - This bug will never happen again
  - There are no related bugs
- Sure...
Testing

- Customer
- User acceptance tests
  - Product owner (your manager's manager, VP, ...)
- System testing
  - Test engineering team
- Integration testing
  - Dev team
- Unit testing
  - Developer
1. Write a test that fails

2. Make only enough code for it to pass

3. Improve code quality

REPEATE PROCESS

RED (Fail)

GREEN (Pass)

REFACTOR
Unit Testing

- Fast cycle time (5-10 minutes)
- Leads to much better code (decoupled)
- Spend a little now, save big in the future
  - Avoid technical debt
- Vastly improves quality & speed of future enhancements
- Not Unit Testing ~ Smoking
  - Eventually kills you
  - Never too late to quit!
Unit Tests

• Every class shall have a set of unit tests
• Every public method shall have 2+ tests
  – 1+ Random, typical cases
  – 1+ Boundary cases
A Test is NOT a Unit Test IF

1. It talks to the database
2. It communicates across the network
3. It touches the file system
4. It can't run correctly at the same time as any of your other unit tests
5. You have to do special things to your environment (such as editing config files) to run it.

Michael Feathers (www.objectmentor.com)
TDD Framework

- JUnit
- Fully integrated with dev environment (Eclipse, NetBeans)
- Check out www.junit.org
- Documentation of assertions: http://junit.org/apidocs/org/junit/Assert.html
- More info on TDD: https://en.wikipedia.org/wiki/Test-driven_development