Testing & Testing Tools

Emphasis on JUnit & ECLEmma

Why Testing

- Finding obvious bugs
- Finding non-obvious bugs (?)
- Increasing confidence for the system

What is a Test?

- Consists of three parts
  - Input(s)
  - Test program
  - Oracle (i.e., expected output)
- Run test program on input(s) to generate an output
- Assert output == oracle

Testing Frameworks

- JUnit for Java
  - Comes default with Eclipse
- PHPUnit for PHP
  - Comes with PEAR

- Easily runs your tests
- Contains commonly used assertions
- Contains constructs to setup and clean common test code

Coverage Criterion

- Function
- Line (Statement)
- Decision (Edge)
- Condition
- Decision & Condition
- Path
- Loop
- Does 100% <X> coverage imply correctness?

Testing for User Interaction

- How?
  - You don't really (with minor subtlety)
- Build your system so that logic is in the backend (do NOT do computation in the UI)
  - UI transmits data (input) from the user to your backend
  - Backend does the computation and transmits data (output) back to UI for visualization)

Testing for User Interaction II

- Shouldn't we test user interaction at all, then?
  - No, but most of UI test will be mostly manual
  - Try to have as few as these possible
- Validate user input with UI tests
  - Make sure that user cannot enter input that would crash the system
    - e.g., can user enter string to a text box that should only get integers?
- Test UI transitions and different visualizations important for your system

Reminders

- ZFR due today @11 pm
- ZFR demos due tomorrow during class
  - I have to skip class tomorrow, so my teams should set up meeting with me so that I can see their UI
- Beta release due on May 15th

- Any questions on ZFR?