CMSC 131
Object-Oriented Programming I

Testing/Debugging

Dept of Computer Science
University of Maryland College Park

This material is based on material provided by Ben Bederson, Bonnie Dorr, Fawzi Emad, David Mount, Jan Plane
Announcement (Not Related to CMSC 131)

• Make sure you do not miss your advising appointment. There are thousands of students and if you miss your appointment you might not get the course you want
• Be aware of which day and time you register. Being late by one minute could translate on you missing the class/section you want
• Some advising information available at:
  • http://www.cs.umd.edu/~nelson/advising/
Debugging

• Process of finding and fixing software errors
  • After testing detects error
• Goal
  • Determine cause of run-time & logic errors
  • Correct errors (without introducing new errors)
• Similar to detective work
  • Carefully inspect information in program
    • Code
    • Values of variables
    • Program behavior
• It is a skill you need to develop
  • After all, there is no such thing as the “Company TA” 😊
Debugging Approaches

• Classic
  • Insert debugging statements
  • Trace program control flow
  • Display value of variables

• Modern
  • IDE (integrated development environment)
  • Interactive debugger
Interactive Debugger

- Capabilities
  - Provides trace of program execution
  - Shows location in code where error encountered
- Interactive program execution
  - Single step through code
  - Run to breakpoints
- Displays values of variables
  - For current state of program
Terminology

• **Break Point**
  • Drop a marker into the code so when it runs the execution will stop at that point
  • Allows you to not have to go step by step through things you believe are correct

• **Step Over**
  • Takes one step in the current method
  • If that step is a method call, it performs that whole method call and steps to the next line in the current method

• **Step Into**
  • Takes one step in the current method
  • If that step is a method call, it steps into that method so that you can then step through it before getting to the next line in the method you were in
Eclipse

- Perspective
  - Debug Perspective
  - Java Perspective
- Run
  - Debug As...
  - Run As...
- Know if it is still running
  - Watch the red square – click it to stop it
- **Example:** AuxMath.java
  - Let’s set a break point in maximum
  - Let’s see the contents of parameters
  - Let’s see the different stack entries
  - Let’s step over and step into
JUnit → Testing framework for Java

With JUnit testing you define a class where each method represents a test.

To create the test method define public void methods that have the @Test annotation.

```java
@Test
public void checkingEvenValues() {
    ....
}
```
JUnit

- Inside of a test method you can have typical Java code where assertions are used to specify the expected results for the test
- Common assertions:
  - `assertTrue` → verifies that the argument expression is true. If the argument expression is false the test fails; otherwise execution is successful
  - `assertEquals` → takes two arguments (expected value and actual value). If the values are not equal the test fails; otherwise execution is successful
- **Example:** `JUnitExample.java`
- You can define auxiliary methods (e.g. private methods) that support the set of tests you are developing
- You can have multiple assertions in a JUnit test. Once the first one fails the whole test is considered to had failed
- Pick descriptive names for your test methods
JUnit

- To create a JUnit .java test file in Eclipse:
  - File→New→JUnit Test Case→New JUnit 4 test
  - If the JUnit library is not part of the project you will get a message indicating whether it should be added; just add it
  - Pick a class name (e.g., Myclass or any name you prefer)
  - If you don't provide an assertion a test is not considered to fail
  - The use of static fields can lead to results from one test to propagate to the next
  - You can use the Eclipse debugger with JUnit tests
  - You can add output statements to see results generated by your tests
    - Helpful to identify problems with project tests
  - Additional information at: