

## Debugging Programs

With a little of ‘this’ and object equality

Debugging—finding run-time errors in your program

Compiler/syntax errors—in Java, these are the ones javac tells you about.

maddening for beginning programmers, but really like a helpful grandmother

Run-time errors—when you run a program (java Program) and things don’t work as they should. These are the ones that can keep you up all night.

How do you debug programs now?

Debugging tools, like the one in Eclipse, can help very, very much!

Debuggers let you run your program to specific points, look at variable values during execution, and step through the program a statement at a time.

### ***Debugging Terminology***

**Breakpoints**—you can specify that the debugger stop execution at any point in the program.

**Step Over**-- When execution is stopped at a statement, you can have the debugger execute one statement at a time. If it’s a method call, the debugger will execute the entire method and return execution to the statement after the call.

**Step in**—Execute one statement, but if it’s a method call stop execution at the top of that method.

**Step return**—return execution from the method your in to the caller.

**Resume**—Tell the debugger to run the program until the next breakpoint.

***Instructor Sample***—Stepping through a (partial) Mastermind program with Eclipse

***Students—Debugging the Mastermind program.***

0. Import the files for the Mastermind program at

<http://www.cs.usfca.edu/~wolber/courses/110/javaSamples/mm/>

Create a directory called mm, save the files above into it. Then open Eclipse and create a new project, choosing to ‘create project from existing source.’

1. Step through the main program step by step, looking at variable values. To do this, set a break point at the first statement of main (double click in alley to left of code), then right-click on file to run and select Debug → Java Application
2. Now put a breakpoint at the top of the method ‘exactMatches’ of class ColorCode. Remove the breakpoint in main by double clicking on the breakpoint icon. Re-debug the program and step through the execution of ‘exact matches’.

Can you find ‘this’? Show your instructor or TA when you’ve found it.

Can you figure out what data members the Java Library class ArrayList has?

Is there a ‘this’ in the main?

Put a breakpoint in ColorCode.toString, and debug. Can you find out who is really calling toString?

The Mastermind program doesn’t work. Continue debugging to find out why.

***Basic Debugging Strategy***

Start in the main

Step over statements until you find the method call causing the problem

Then step into that method (or set a breakpoint in it) and step through it.

You may have to dig further into methods called by that method.

***Final Thought***

You are in control. You write the program. The debugger allows you to see what’s happening at all points in the program. No more mystery! No more guessing!

(Well, maybe some)