OOP Gotcha Day (done after tic tac toe lab)

Print handout, write outline on board (review, references,

-Review OOP concepts

--Transition from the programmatic stuff we’ve been doing to looking at multiple objects at once

--How do these work? Separate variables, separate values…each forms its own independent\* Object.

Except for!!!

-How does this work at a low level? What does this look like? How is Java doing this? References

--Introduce static

--Scope?

--Print handout

-Ask about lab or project questions

-Frame this in terms of object-oriented programming and classes.

-Review the types of variables (primitive and reference), go over scope and static/instance/local variables, go over pass by value, go over the gotcha handout. (If time, do SimpleCanvas programming {countBlue and Brighten}.)

* Go over each gotcha.
* Variables are only references or primitives. A variable is not the same thing as an object.
* go over example on handout.
* First section is just variables. Draw memory diagram on board, showing how z is separate from x & y. I can set z independently.
* Second section: Draw pictures of the references. Define reference: in Java, a data type that refers to some object. Think of it as an arrow that points to an object.
* So when you say SimpleCanvas canvas, you aren't actually creating a new SimpleCanvas object. You are creating a **reference** to a SimpleCanvas object (that initially doesn't point anywhere).
* The object is not created until the keyword new is used with the Simplecanvas constructor.
* **IMPORTANT**: The object is not the same as the reference to the object. They can be changed independently of each other.
* Now they should understand why there are 3 references but only 2 objects.
* go over Gotcha demo 1 in code.
* Make sure to mention arrays are OBJECTS.
* When using == on reference types, Java compares the references, not their corresponding objects.
* Code demoing this is in SimpleCanvasCountBluePixels.
* All arguments in Java are passed by value, both reference types and primitive types.
* Go through gotcha demo 3.
* Mention tic tac toe---how we talked about how handleMouseClick takes the charPlayer parameter, and if it's changed inside the function, that won't be reflected back in main because it's a local variable in main.
* Technically this is the same for the board[][] array. So how are the changes to that being reflected outside of handleMouseClick? Because you're passing a reference.
* If time, talk about null pointer and null pointer exceptions. (Where Java tries to use a null value as an object)