Final Exam Stuff

CS170 Final Exam: April 30th, 6:30-9:00 PM, White Hall 208
(Except for those with conflicts, who should have received an e-mail with the date, time, and location for their CS170 Final)

CS170 Review Session (note the new time and room!!):
  Wednesday April 29th, 3:00-5:00 PM, MSC w201
  This will be a joint review session for all sections given by Dr. Swenson and Clarissa Garvey

Final Exam Study Guide now available via course calendar page!!
Lecture 32:
Constructors,
Visibility modification vs. static, 
and Scope of variables

Apr 21 2015
Default constructor vs. no-argument constructor

• The *default* constructor is the constructor provided by the JVM
  – The default constructor is provided ONLY if NO constructor is included in the class definition;
  – To reiterate if a class includes a constructor *of any kind*, the default constructor is NOT provided.
  – This means that, for example, if you include a one- or two- argument constructor for your class, but not a no-arg constructor, then no no-argument constructor exists for your class

• A *no-argument* constructor is one that takes zero parameters.
  – The default constructor is a no-argument constructor
  – But when you call a no-argument constructor, you’re only using the default constructor if no constructors have been written in the class
Revisiting \texttt{this}

- Using \texttt{this} with constructors (explicit constructor invocation)

- Using \texttt{this} with data fields

```java
public class Rectangle {
    private int x, y;
    private int width, height;

    public Rectangle() {
        this(0, 0, 1, 1);
    }

    public Rectangle(int width, int height) {
        this(0, 0, width, height);
    }

    public Rectangle(int x, int y, int width, int height) {  
        this.x = x;
        this.y = y;
        this.width = width;
        this.height = height;
    }
    ...
}
```
class Complex {
    private double re, im;

    // A normal parametrized constructor
    public Complex(double re, double im) {
        this.re = re;
        this.im = im;
    }

    // copy constructor
    Complex(Complex c) {
        System.out.println("Copy constructor called");
        re = c.re;
        im = c.im;
    }
}
Comparing two objects: 

== vs. .equals()

In Java, we can distinguish two kinds of equality.

- Object reference equality: when two object references point to the same object. (==)
- Object value equality: when two separate objects happen to have the same values/state. (.equals(), which does not exist by default, but can be provided in the class definition.)

What code should go in the body of the method

static boolean equals(Rectangle r)?
Public/private vs. static

• The key words **public** and **private** change the visibility of a variable or method
  – They determine which classes can see and therefore use/access the given method/variable
  – **private** hides from all other classes
  – **public** exposes to all other classes outside

• The key word **static** creates a variable or method that belongs to the class as a whole, instead of to an individual object. Methods/variables declared **static** are called **class methods/variables**, while methods/variables NOT declared static are called **instance methods/variables**
Class vs. Instance methods and variables

Class methods cannot access/use instance methods or instance variables!!
Why not?
Scope
(one of the topics for Thursday’s quiz!)

• Instance (non-static) variables have scope within all instance methods
• Class (static) variables have scope in the whole class
• Local and parameter variables have scope within their method

Example:
Exam2Prob3