Administrative Stuff

▶ assignment 3
▶ chapter 3
Methods Review. Important.
Simple Primitives

```java
public class Funcs00 {
    public static void main(String args[]) {
        int x = 10;
        int y = x;
        y++;
        System.out.println("x=" + x + "y=" + y);
    }
}
```
public class Funcs01 {
    public static void func(int y) {
        y++;
    }

    public static void main(String args[]) {
        int x=10;
        func(x);
        System.out.println(x);
        // System.out.println(y); scope error
    }
}
What do we get when we create a method?

- named hunk of code
- a scope anything declared within the method is:
  - local to the method
  - meaning that
    - it’s only visible within the method
    - can’t access it outside the method
    - this includes parameters
Why create methods?

- small simple methods easier to test
- easier to understand
- when something is hard to read, it’s easy to make a mistake
- manage complexity
- ease of code re-use
public class ByPosition {
    public static void func(int x, int y, int z) {
        x++;
        z /= 3;
        y += z;
        System.out.println("x = " + x);
        System.out.println("y = " + y);
        System.out.println("z = " + z);
    }
    public static void main(String args[]) {
        int x = 10, y = 20, z = 30;
        func(z, x, y);
        System.out.println("x = " + x);
        System.out.println("y = " + y);
        System.out.println("z = " + z);
    }
}
Same idea as first example, but we copy something back

```java
public class Funcs02 {
    public static void main(String args[]) {
        int x=10;
        int y=x;
        y++;
        x=y;
        System.out.println("x=" + x + "y=" + y);
    }
}
```
public class Funcs03 {
    public static int func(int y) {
        y++;
        return y;
    }
    public static void main(String args[]) {
        int x=10;
        x=func(x);
        // System.out.println("x=");