Reading for next two lectures: Executing method calls, if-statements, the return statement in a function, local variables. Chapter 2 except 2.3.8 and 2.3.9. This reading will clearify some concepts, such as method parameters, that we've had to gloss over so far.

A1 possible test cases

Specifications and headers for methods in class Worker, plus test cases

Test cases "nend", "rhythm" reveal vagueness in the spec!

Developing test cases first, in "critiquie" mode, can prevent wasted work.

Class Object: The superest class of them all

A minor mystery: since Worker doesn't extend anything, it seems that it should have only the methods we wrote for it. But it has some other methods, too.

Java feature: Every class that does not extend another one automatically extends class Object. That is,

```java
public class Class { ... }
```

is equivalent to

```java
public class Class extends Object { ... }
```
A **static** variable appears not in each folder, but as a single entity in the file drawer. It can be used to maintain information about multiple folders.

Declaration:
```
private static int numberOfWorkers; // no. of Worker objects created
```

Reference the variable by `Worker.numberOfWorkers`.

Should the following function be static?
(A) yes (B) no

```java
public [static?]? boolean clashingSSNs(Worker a, Worker b) {
    return a != null && b != null
    && a.getSSN4() == b.getSSN4();
}
```

A **static** method is also in the drawer, not in individual folders. Make a method static if it doesn’t need to be in all the folders because:
- it wouldn’t reference the contents of the “containing” folder; equivalently, its actions/results would be the exactly the same no matter which folder it were in.
- `getName()` should not be static, but the following should:

```java
public static boolean moreThan100() {
    return numberOfWorkers > 100;
}
```

**Method toString()**

Contribution: `c.toString()` returns a representation of folder `c`, giving info about the values in its fields.

When a String is expected (or in the Interactions pane), the expression `c` is evaluated as `c.toString()`.

```
** = e.g., “Obama, XXX-XX-6789, boss null”.
“Biden, XXX-XX-2, boss Obama” [see posted code for full spec]
```

```
public String toString() {
    return ...; } // bad scrunched style, slides are tiny
```

** = “there are more than 100 workers”. */

```
public static boolean moreThan100() {
    return numberOfWorkers > 100;
}
```

```
/** = e.g., “Obama, XXX-XX-6789, boss null”,
“Biden, XXX-XX-2, boss Obama” [see posted code for full spec]
```