A1 possible test cases

---

CS1110 lecture 5  14 Sept 2010
Testing: the class Object; toString; static variables & methods

Specifications and headers for methods in class Worker, plus test cases
** Constructor: a worker with last name ("if none), SSN s, **
* Precondition: n is not null, in 0..999999999 with no leading zeros.
* so SSN 012-34-5678 should be given as 123456789
public Worker(String n, int s, Worker b) { 
  ** = worker's last name */
  public String getName() {
    ** = worker's boss (null if none) */
  public Worker getSSN4() {
  ** = last 4 SSN digits without leading zeros. */
  public int getSSN4() {
  ** = last 4 SSN digits without leading zeros. */
  public void setSSN4(int s) {
  ** = boss's boss (null if none) */
  public Worker getBoss() {
  ** = boss (null if none) */
  public void setBoss(Worker b) {
  ** = boss to */
  public void setBoss(Worker b) { 

A testMethod to test a constructor (and getter methods)
Filer> new JUnit test case ... [save in same directory as Worker.java] imports junit.framework.TestCase, with key methods.

```java
/** Test constructor and getters*/
public void testConstructor() {
  Worker w1 = new Worker("Obama", 123456789, null);
  assertEquals("Obama", w1.getName());
  assertEquals(123456789, w1.getSSN4());
  assertEquals(null, w1.getBoss());

  Worker w2 = new Worker("Biden", 2, w1);
  assertEquals("Biden", w2.getName());
  assertEquals(2, w2.getSSN4());
  assertEquals(w1, w2.getBoss());
  System.out.println(w1.toString());
}
```

Every time you click button Test in DrJava, all “testX” methods are called, in some order.
Please put your name, netID, and "Quiz 2" on a piece of paper, circle your last name, and then answer these questions.

1. What is the purpose of a constructor?
2. How do you evaluate a new expression, such as `new myClass()`?

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, ```public class C {...}```
is equivalent to ```public class C extends Object {...}```

Method toString() — kind of an “ultra-getter”

Convention: c.toString() returns a String representation of folder c, giving info about the values in its fields. So we need to override the Object toString() to be able to talk about the subclass’s fields.

When a String is expected (or in the Interactions pane), the expression c is evaluated as c.toString() (the “lowest” one).

```/** = 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
```
Should the following function be static?
(A) yes (B) no

/** = "a and b are not null and the last 4 digits of their SSNs are the same", */
/** (This notation means that the function yields * the truth value of the quoted statement. * So, a and b are allowed to be null! */

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