

CSC108H lab – week 4

This document contains the instructions for the week 4 CSC108H lab. The following two rules apply for the rest of the term, and will not be repeated in later lab handouts:

- To earn your lab marks, you must actively participate in the lab. *You don't need to finish in the time allotted, you just need to try hard.*
- The navigator must not touch the keyboard or mouse. For the rest of the term, if the navigator does type when they are not supposed to, the navigator will get a zero for the lab.

1 Objectives

1. Practice writing constructors.
2. Practice with JUnit.
3. Practice using Strings.

2 Starting up

Sit down with your partner. The rest of these instructions call you two `s1` and `s2`. Pick which one is which. `s1` should log in and start up DrJava, and be the first driver.

3 Account and AccountTester

Download classes `Account` and `AccountTester` (right-click on the links and choose “Save target as...”) from the Labs page on the course website:

<http://www.cs.toronto.edu/~campbell/108/05w/labs.shtml>

Open them up in DrJava and then read them and discuss them with your partner. All the `Account` method bodies are “stubs”: they have just enough code in them to get it to compile, but they are meaningless.

Click the `CompileAll` button.

3.1 JUnit: the first two constructors

Finish the first two constructors, compiling as you finish each one.

- One takes the owner as a `String`, the initial balance as a `double` and the minimum balance as a `double` (this has three parameters), and
- One takes the owner as a `String`, the initial balance as a `double` and has a minimum balance of zero (this has only two parameters).

3.2 JUnit: toString

Once everything compiles click the `Test` button. It will have lots of failures. One of the problems is method `toString`: it currently returns `null`. Fix it so that it returns the expected value.

3.3 JUnit: String parsing

We have started the third constructor. This one takes all the information in a single `String` with the three pieces separated by commas. Finish it. You'll need to use `String` methods `substring` and `indexOf`. You will also need this: `Double.parseDouble(String)`, which converts the argument to a double.

Test it.

3.4 JUnit: testGetters

`testGetters` fails because the getter methods in `Account` need fixing. Fix them so that `testGetters` passes.

Notice that the `assertEquals` method calls that involve `getBalance` and `getMinBalance` have *three* arguments. The last argument is a tolerance because double calculations are sometimes inaccurate. Here, `assertEquals` is checking whether the first two arguments are *exactly* the same.

3.5 JUnit: the rest

Finish working through the testers and writing `Account` code.