CMSC 123 Lab 1

Goal: This lab will give you practice editing and running Java programs, and accessing online Java documentation.

There are several things you will need to check off so you can get credit for this lab. You can have them checked off all at once when you are finished, but don’t wait until the last minute--another lab may start as soon as yours ends, so ask to be checked off when there is still plenty of time. Checkoffs will not be done outside of your lab time.

**PART I: Getting started**

The rest of this lab is a TEAM assignment with two-person teams. Introduce yourself to the people to your left and right. Choose a partner.

If you’re the odd person out in the lab, go ahead and begin, but if someone new arrives, partner up with them and help them catch up. You are not permitted to go solo unless everyone else present is already partnered up. This rule will hold throughout the semester, though you can change partners between labs (and you must if your regular partner doesn’t show up).

**PART II: Compile-time errors (1 point)**

You and your partner may find it easier to work on one computer for a while. However, you should each be separately capable of demonstrating your knowledge of the lab when it’s time to check off.

*Mac Help: You will be needing the following simple commands to navigate the directory:*

*cd [directoryname] Go to the directory of directoryname*

*ls Show directory contents*

1. Download [lab1](https://drive.google.com/file/d/0BxWgfE7ZstaYNW5VMVVlTjl3aEU/view?usp=sharing) file, entitled Names.java, to your computer. Save it to a folder Documents/yourstudentnumber/cmsc123/lab/01/.
2. Bring up Sublime Text application. Open the file Names.java. Names.java is a simple Java program for performing various string operations on a name. It almost works, but you need to make some changes so that it compiles and runs correctly.
3. Bring up the terminal window. Go to the directory of Names.java and type

javac Names.java

"javac" is the name of the Java compiler.

1. You will find that javac cannot compile the program. The code contains two syntax errors which you are quite likely to make when you will write programs. (The second error won’t rear its head until you fix the first one.)

**PART III: Run-time error (1 point)**

Once javac is able to compile the code, it will create a file called Names.class in the same directory. You can run this program using the Java interpreter.

java Names

Java automatically appends the ".class" file extension on Names.class. If you accidentally type "java Names.class", Java will look for a file called "Names.class.class".

The program has an error. How can you tell there is something wrong? This type of error, which occurs at run-time, tends to be significantly more difficult to correct than compile-time errors. (It’s still somewhat easier than discovering an error in which the program appears to finish without problems, but is doing some computation incorrectly.) The error message may be hard to read at first, but it will allow you to answer certain questions: What is the method (i.e., procedure) that generated an error? What is the line number within the file Names.java? The error is in one of the methods in the String class, which is a standard Java library. Your textbooks contain some documentation of the Java library, but the best source is the online documentation.

Search in the web: "The Java 2 standard libraries API." There are two libraries that you will be using early in the semester--the java.lang package and the java.io package. Documentation on String and other standard data types is found in java.lang, so go there and find the String class. There is a lot of information, not all of which will make sense right now, but you should be able to find a description of the problematic String method.

Be prepared to show how you found this information and how you figured out from it what was wrong with the program. When you think you have found the error, correct it, save the file, recompile it, and execute it to see if the problem is solved.

*Aside: You may think that the file produced by javac is named Names.class because the input file is named Names.java. Not so--the name of the .class file is based on the class name IN Names.java. To experiment with this, change the line "class Names {" to "class Silly {" and recompile using javac.*

**Check-off**

When you’re done, show your work, and explain how you accessed the Java String documentation and found the bug.

1 point: Show that your Names program works.

1 point: Briefly explain how you discovered and fixed the bugs, and show how you accessed the Java String documentation.