CSC230 Lab 1

Due: Sept 3rd, 11:59pm

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

Please try your best to finish the lab in class, and submit it to CANVAS. Use command "jar" to zip all your .java files and .class files into lab1.jar. Submit lab1.jar to CANVAS. To use command "jar", type the following command at prompt

*jar –cvf lab1.jar **

The above command will put all files (that is meaning of *) under current directory into lab1.jar. If you want put several selected files from the directory, list the filenames at * place. For example, you want to put two files, foo and bar, into lab1.jar. You can type the following command

jar –cvf lab1.jar foo bar

PART I: Getting started

---

1. Please login the computer by using your TCNJ email username and password. If the login process does not work, please restart your computer and try again.
2. Open a terminal in the computer. You can type Unix commands inside the terminal now. For a Mac machine, you can find terminal icon in the bottom of the window. If you want to do the work on your own Linux machine, like Ubuntu. You can do this by selecting the "Dash Home" icon at upper left, typing "terminal" in the search bar, then clicking on the "Terminal" program. There's also a keyboard shortcut: hold down the Ctrl and Alt keys while typing "t".

3. Warning: do not, at this time or any point in the future, delete the .login, .bashrc, and .emacs files in your directory. If you would like to customize your account, do so by adding new commands to the end of these files, not replacing the files.

PART II: Compile-time errors (1 point)

After logging in, please do the following steps.

1. Create a directory for lab1 by typing

   mkdir lab1

   Download file Names.java to directory lab1.

2. Change into your lab1 directory:
   cd lab1

3. Run Emacs by typing
   emacs

   Then load the program file Names.java into Emacs using C-x C-f. "C-x" is read "control x," and is typed by holding down the control (Ctrl) key while typing "x". Type "Names.java" (you might need to type the full pathname lab1/Names.java; try both if necessary), then hit Enter.

   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.

4. Compile the program within Emacs using:
   ESC !

   This will allow you to run a shell command:
   javac -g

   "javac" is the name of the Java compiler. "-g" is a switch to tell the compiler to generate the information the debugger will need. You need to finish the command line by supplying the name of the file to be compiled:
   javac -g Names.java

   You may instead type this command in a different terminal, assuming you're in (your copy of) the lab1 directory.

5. 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.) Use the control sequence C-x ` to jump directly to errors found by javac. (Be sure to use the single opening quote, not the apostrophe.) Figure out what's wrong, correct the errors, save the file using "C-x C-s", and compile the code.

PART III: Run-time error

---

1. 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. In a shell (an xterm or an Emacs buffer called *shell*), type

   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. To find this, start Firefox

   Double-click on the URL window and type the URL for the class Web page.

   http://docs.oracle.com/javase/7/docs/api/

   This link points to "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.

   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.

PART IV: Emacs help
---
1. Load the file "roster.txt" in Emacs. You will see an unsorted list of names, with each line in the form SURNAME, GIVEN NAMES.

You should use the help facilities of Emacs (apropos and info) to find out how to sort all these names by surname. Then sort the lines of the file. If you are unfamiliar with the help facilities type "C-h ?", and Emacs will guide you through them.

Save the result using "C-x C-s".

If you have the course reader, take a few minutes to look through it now so you know exactly what information you can look up in it. Note that the Emacs Quick Reference Guide is in the very front of the reader. It will be handy to keep your reader by you whenever you're in the lab.

Hint:

- You need to set a region. To do that, move the cursor to the end of the file. Use C-@ to set the mark. Then, move the cursor back to the beginning of the file.
- Check M-x works or not. If it does not, the meta key setting of your Mac machine is not right, do the following
  Go to Preferences > Settings > Keyboard > Use option as meta key.
- Use M-x sort-lines

Wrap up

When you're done, jar three files to lab1.jar

Jar –cvf lab1.jar Names.java Names.class roster.txt

Submit lab1.jar to Canvas.

Make sure you logout before you leave!

If you cannot finish lab in class, please save all your files to cloud. Once you logout the computer in the lab, your files on the computer will be GONE. Please save them before you logout.