CIS 201 Labs: Lab 2
Chapter 1 Lab: A time and place for structure

Some of the examples in this laboratory are based on problems and examples from the text Building Java Programs.

Objectives

In this lab you will work on

- finding syntax errors in programs using compiler output
- writing structured programs with more than one method
- diagnosing logic errors
- using strings that contain special characters

Directions

1. **Read this lab description carefully as you go along. Make sure you try ever exercise.**
2. **There are 7 checkpoints ✓ in this lab**
3. **If you need help with any problem, raise your hand. Call us over to check you off.**

Getting Started

1. Log into your lab account and open a terminal window. Or open java on your local machine
2. Only if you login to lab computer, change directories so that you are in your 201directory. If this directory does not exist, create it using the mkdir command.
3. Now copy the lab materials to your account. If you are using your local machine copy the materials from moodle. If you are using lab machine, you will need to use the `-r` option on the Unix cp command in order to copy not only files, but also directories. To copy, enter

```
cp -r /home/student/Classes/201/Labs/Lab02 .
```

Don't forget the space and the dot . at the end of the command. Also, ignore any message indicating that you can't copy the Solutionsubdirectory.

After running this copy command, do an ls;you will see a new directory called Lab02. What you copied was actually a directory, plus all of its contents. Everything you need for the lab exercises today is contained in this new directory.

4. Change directories into this new subdirectory and list the contents.

Getting Rid of Syntax Errors

6. Open the program file Buggy.java

   Without closing kate, go back to your terminal window and try to compile this program. There are lots of syntax errors in it. Use the compiler's error messages to help you fix all of the errors in kate. After you have fixed any errors, save the file, go back to your terminal window, and recompile the program. You may need to do this step more than once until you have fixed all of the errors.

   ✓ **1** Show us this program after you have all the errors fixed and the program compiles and runs.

A Program With a Problem

7. Open the program file Runaway.java. Compile and run this program. You will notice that it has a bit of problem. It doesn't seem to stop. Actually, the program will stop (in less than a minute). It will cause a *stack overflow* (something you will learn more about in CS II). You won't see it, but the program crashes to a halt with an error message and a trace that looks like this:

```
...
All work and no play makes Jack a dull boy.
Jack and Jill went up the hill to fetch a pail of
water.
All work and no play makes Jack a dull boy.
Jack and Jill went up the hill to fetch a pail of
water.
Exception in thread "main" java.lang.StackOverflowError at
            java.nio.Buffer.(Buffer.java:176)
            at java.nio.CharBuffer.(CharBuffer.java:259)
            at java.nio.HeapCharBuffer.(HeapCharBuffer.java:52) at
            java.nio.CharBuffer.wrap(CharBuffer.java:350)
            at sun.nio.cs.StreamEncoder$CharsetSE.implWrite(StreamEncoder.java:378) at
            sun.nio.cs.StreamEncoder.write(StreamEncoder.java:136)
            at java.io.OutputStreamWriter.write(OutputStreamWriter.java:191) at
            java.io.BufferedWriter.flushBuffer(BufferedWriter.java:111) at
            java.io.PrintStream.write(PrintStream.java:458)
            at java.io.PrintStream.print(PrintStream.java:602) at
            java.io.PrintStream.println(PrintStream.java:739) at
            Runaway.jack(Runaway.java:15)
            at Runaway.jill(Runaway.java:22)
            at Runaway.jack(Runaway.java:16)
            ...
```

The problem with this program is not syntactic. Remember, it did compile and run. The problem has to do with the logic in the program itself - called *logic error*. Logic errors are introduced by programmers.

8. Right now, you and your partner look at Runaway.java and diagnose the logic error. Draw a diagram showing the calling hierarchy to help with this.

   ✓ **2** For this checkpoint, call us over, show us your diagram, and explain what the problem is with this program.

Strings that Contain Special Characters

9. Now you are going to write a new program from scratch in a new file called SingleString.java.

10. In the file SingleString.java that you are now editing, create a class called — well, what do you think it should be called? — to print the following output using a single System.out.println statement with a single String in it.

    This "output" was produced using a single string.

    Remember that we talked about special characters in class. Besides \"and \\ there was \t and \n. Will any of these help you?

    ✓ **3** For this checkpoint, your output must match the above output *exactly*. Call us over to see your program when you are ready.

Now For Some Practice-It

11. Login to your Practice-It account (there is a link on our Moodle home page). Do **Exercise 1.4 Difference.**

    ✓ **4** Show that you have succesfully completed Exercise 1.4.

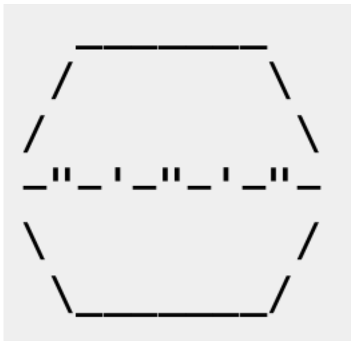
12. Do Practice-It **Exercise 1.5 MuchBetter**

    ✓ **5** Show that you have successfully completed Exercise

13. Do Practice-It **Exercise 1.8 Stewie2**.

    ✓ **6** Show that you have successfully completed Exercise

14. Write a complete Java program in a class named egg.java that displays the following output:



```
  _______
 /       \
/         \
_"_'_"_'_"_
\         /
 \_______/
```

✓ 7 Show that you have successfully completed this program