COSC 236

**LAB 1 – Getting to Know DrJava\***

We will be making use of the DrJava program development environment this semester. This tool was developed especially for programmers new to Java. In order to understand how the DrJava environment works, and to learn some of its capabilities, we will execute some simple Java programs.

**Task 1 – Configuring DrJava**

By going to Edit 🡪 Preferences 🡪 Display Options, you can customize aspects of the DrJava environment. One feature that you should turn on is automatic line numbering. Do this by clicking the checkbox on (under Display Options). Now lines numbers should appear for all of the lines in the edit window.

**Task 2 – Setting the Language Level**

DrJava allows different “language levels” of Java to be use. We are going to use “full Java.” Therefore, under the “Language Levels” menu, select Full Java.

**Task 3 – Running a First Program**

Type (do not copy and paste) the “Hello World” program given below:

public class HelloWorld

{

public static void main(String[] args)

{

System.out.println(“Hello World!”);

}

}

Save the file on your H (network drive), create a folder and call it COSC236, you can save all your files related to this course there and access it from anywhere on campus (and home through vpn). Note that it will automatically be saved with a .java extension, so you don’t have to type the extension when you enter the file name (**make sure the file name matches exactly the class name!**). Note: it also allows the extension to be entered without creating two extensions.

Now compile the file using the compile button. If you click on the compiler output tab on the bottom (if not already showing), you should see a message that the compilation was successful. To run the program, click on the run button.

(If you get a compilation error on the line with the double quotes, it is because you copied and pasted the program from a Microsoft Word documents, and the “smart quotes” of Word are confusing the compiler. In this case, delete and re-enter all the double quotes.)

**Task 4 – Running a Program with Command Line Arguments**

Now, enter, save and compile this “Hello There” program that uses a command line argument.

public class HelloThere

{

public static void main(String[] args)

{

System.out.println(“Hello There “ + args[0]);

}

}

When you run this program, it is expecting a command line argument (e.g., java HelloThere John). Since we are no longer running Java from the command line, we can use the Interactions window (tab at bottom of DrJava) to run the program,

> java HelloThere John

Hello There John (output)

What do you conclude is the purpose of the String[] args part of main?

**Task 5 – Using the System.out.println statement**

Write a JAVA program that prints out to the screen the following output (exactly as shown!), you can call the class Lab1Task5.

Hello, world!

This program produces

four lines of output

**Task 6 – More on using the System.out.println statement**

**Write a JAVA program that displays the following pattern:**

**\***

**\*\*\***

**\*\*\*\*\***

**\*\*\*\*\*\*\***

**\*\*\*\*\***

**\*\*\***

**\***

You can call the class Lab1Task6.

**Deliverable:**

Submit your solutions (attach the .java files) **through Black Board.**

\* Modified from Chuck Dierbach.