Lab 01

Introduction to Computer Science
CSI201
Fall 2012

SUMMARY--(DETAILED INSTRUCTIONS ARE UNDER IN-LAB BELOW)

You ***will need*** your UAlbany NetId and password, which you used on MyUAlbany to register!

First, learn a work flow to write, compile, and test Java programs, and save them on your "**S:**" drive so you can reclaim them outside of lab for further work and submission for credit.

1. You will make your first complete Java program to print your name on one line.
2. Second, make a different complete Java program to print at least three lines that express some of your thoughts.
3. Finally, make a third where you change what you said by changing the order of printing.

The rest of the lab starts you with writing programs that use one **`Turtle`** object to draw cool graphics.

1. Copy your own copy of the "Book Classes" from the place on the ITSUNIX disk we tell you.
2. Configure into DrJava the location of that copy by ***Editing*** the ***Preference*** named ***Extra Class Path***. (Unfortunately, you must restart DrJava.)
3. Write a program that makes a **`World`** (using the Book Classes) and then makes a **`Turtle`** in that **`World`**.
4. After seeing that **`Turtle`**, you will add to your program commands that command that **`Turtle`** to draw cool stuff by moving forward and turning in a carefully designed alternating sequence.
5. After your first **`Turtle`** graphics demonstration, you will make a second, differently named and behaving **`Turtle`** demonstration to draw a square.
6. We will make sure you save your **`.java`** files on your **`S:`** drive.
7. After the lab, you will, perhaps after improving and retesting them, you will upload your **`.java`** files to Blackboard for getting grade credit. Your "home directory" you see in the lab is identical to you UA "**`S:`**" drive you see in UA Library computers and can access world-wide.

OUTSIDE OF LAB (BEFORE OR AFTER)

Before or soon after the lab: Do the following things:

1. Read section 3.4 in the book, about working with **`Turtle`**s.
2. If you attended the Monday lecture, review your in-class writing on page 2 of the Project 1 assignment sheet to figure out how my code worked to draw letter E.
3. Attend, review and think about the **`Turtle`** demonstrations from lectures.
4. Start or continue on Project 1.

AFTER LAB: TO GET FOLLOWUP CREDIT

Access the 5 **`.java`** files you made in lab (**`MyNamePrinter.java`**, **`SeveralLinePrinter.java`**, **`ReorderedLinePrinter.java`**, **`TurtleTryer.java`**, **`SquareDrawer.java`**) and upload them to the Lab 01 assignment under the "Course Content" part of the Blackboard service for this course.

IN-LAB

1. Drag the mouse below the bottom of the screen near the middle, and click the boxy black (SSH to ITSUNIX) icon which is on the left of the row of icons you will see.
2. Log in with your NetID (your initials and some numbers) and your MyUAlbany password.
3. Then type in EXACTLY what you see below, which is a line of lower-case letters and slash characters with no spaces in-between. It is a UNIX shell command, so you must press ENTER at the end of the line:

   `/usr/local/depts/cs/geintro/drjava`

   After a few moments the DrJava splash screen will appear, followed by the DrJava window itself. If it offers an upgrade, click "Not Now" or "No".

MAKING YOUR FIRST JAVA PROGRAM

You will make a new so-called ***Java source file*** and save it on your "`S:`" drive for future use.

1. Make the upper-right window pane as large as you can. Use it to type and edit your Java source files like other text edit boxes.
2. Type in, correcting mis-typings as necessary, the text below **EXACTLY, watching UPPER CASE versus lower case letters**, except that instead of the professor's name, put in your own name. As you type, you will see that DrJava AUTOMATICALLY INDENTS the lines to make them look nice and systematic:

```
public class MyNamePrinter
{
   public static void main(String[] a)
   {
       System.out.println("Prof. S. Chaiken");
   }
}
```

3. After you finish typing, click the "**Save**" button. Note that DrJava offers to save your work under the name **`MyNamePrinter.java`** (It omits showing the **`.java`** just like Microsoft Word will typically not show you that Word files end with the extension **`.docx`**).

   Just click "**Save**" without changing anything.

4. After saving, click the "**Compile**" button.

   If you made typing mistakes (likely) you will see yellow highlighted messages about errors in the bottom pane. Try to fix the mistakes: GET HELP FROM THE TA or a neighbor RIGHT AWAY, so you can move quickly!

   Important: The errors that occur at this stage are called SYNTAX ERRORS. You will learn they are the EASIEST to fix and about other, very different kinds of errors before long. It will be important to know the differences between syntax and other kinds of errors.

   After each fix, click "**Save**" and "**Compile**".

   When there are no more syntax errors, you will see the message:

   `Compilation completed.`

5. After you see **`compilation completed.`** click "Run" You should see:

   `Welcome to DrJava. Working directory is` *something depending on you*
   `> run MyNamePrinter`
   `Prof. S. Chaiken`
   `>`
   Of course, your name should be printed instead of mine, and the actual name of your own working directory will be printed instead of the general description "*something depending on you*".

   Congratulations! You made and have saved, in your working directory, your first Java program for this course!

MAKING YOUR SECOND JAVA PROGRAM BY MODIFYING YOUR FIRST

1. Edit **Without Saving!!** **`MyNamePrinter`** (after the **`class`** keyword) to change it so you have:

   ```
   public class SeveralLinePrinter
   ```

2. We want you to name every program with a name that describes its purpose! DON'T SAVE IT YET! Otherwise, you will lose your work so far.
3. **REALLY IMPORTANT:** Under the "**File**" submenu, select "**Save As...**".
4. Change the ***MyNamePrinter.java*** text box **`SeveralLinePrinter.java`**. **Only then** click"**Save"**.
5. **DON'T SKIP THIS STEP(it takes under 15 seconds):** Click "Compile" and then "Run" to VERIFY that your second program ACTUALLY DOES the SAME THINGS as your first, even though it's different with a different name.

6. Now, edit your **SeveralLinePrinter.java** program so it prints several lines (at least three), all different. Be creative here, unlike me:

```
public class SeveralLinePrinter
{
  public static void main(String[]a)
  {
    System.out.println("First Line");
    System.out.println("Second Line");
    System.out.println("Third Line");
  }
}
```

7. Save, try to compile, correct syntax errors as needed, then **Save**, **Compile** and **Run**. You should see something like:

```
Welcome to DrJava.  Working directory is something depending on you
> run SeveralLinePrinter
First Line
Second Line
Third Line
>
```

MAKING YOUR 3RD PROGRAM BY REVISING YOUR 2ND

1. Apply what you practiced above to make a third complete program named **ReorderedLinePrinter.java** whose purpose is to print the SAME LINES as **SeveralLinePrinter.java** does in a DIFFERENT order or sequence.

   (What you practiced above really means: Change the name given after the **class** keyword, "**Save As**" the correctly named file, "**Compile**", "**Run**", EDIT YOUR REORDERING, "**Save**", "**Compile**" and "**Run**"; fixing any errors along the way. These steps are an example of a ***"workflow"*** and soon you will follow a workflow without worrying about it, so your mind will be freed to do much harder things!)

   When it is run, my example program prints output that looks in DrJava like:

```
Welcome to DrJava.  Working directory is something depending on you
> run ReorderedLinePrinter
Third Line
Second Line
First Line
>
```

GET YOUR OWN COPY OF THE GE BOOK CLASSES IN YOUR S: DRIVE.

(We have NOT YET explained what "class" really means, so if you feel confused, it means you are trying to think with missing factual knowledge. That's good. For now, a "class" is a bag of software.)

Type the following command **exactly** then press <Enter>. Exactly means to reproduce exactly which letters are capitalized (only the **C** in this example) and to note there are are spaces ONLY between the **unzip** token and the rest of the line. The token following **unzip** is an example of a "***full pathname***" which locates a ***file***. It consists of simple names separated by the **/**(or **** on Win.) separator.

```
unzip  /usr/local/depts/cs/geintro/bookClasses.zip
```

(Many people outside of computing professions know disks and the cloud store "*documents*". Professionals usually call those things ***files***, especially when they care about where they are located.)

You should see a stream of names as the archive gets unzipped and copied to your **C:** drive space.

Many of you will see: If so, review what you did type and try it again, with corrections.

unzip: cannot find or open You probably mis-copied the full path name above.

OR

-bash: *something***: command not found** You mistyped or miss-capitalized **unzip**

When all goes well, you will see:

```
Archive:  /usr/local/depts/cs/geintro/bookClasses.zip
   creating: bookClasses/
```

```
  inflating: bookClasses/rightArrow.gif
  inflating: bookClasses/SimplePictureProperties.txt
```
and many more lines...just watch until it's done.

ADD THE BOOK CLASSES TO YOUR CLASSPATH IN DRJAVA

From DrJava's **Edit** menu, select **Preferences**. When the Preferences window opens, click the **Extra Class Path: Add** button (under the Resources tab). In the list of files and directories shown in the new window should be your **bookClasses** directory. Click on that, and then on **Select**. Then click on **OK** at the bottom of the Preferences window. Make sure ONLY ONE extra Class Path remains.

Unfortunately, you must exit and restart DrJava for that to take effect. Select from the "**File**" menu "**Quit**" or click the upper right "X" box. If you type **`C-p`** (hold the Ctrl key down and press p) a few times, the bash shell will recall the **`drjava`** command so you don't have to retype it.

TRY OUT THE TURTLE

To get familiar with **`Turtle`** objects and their methods let's try a few simple commands. Make a new program named **`TurtleTryer`** as above. Save as **`TurtleTryer.java`**

Make this program print **`Hello from the TurtleTryer`** when it runs. Save, compile, and run it.

Underneath the **`println`** command, add the lines of Java code:

```
World world1 = new World();
Turtle turtle1 = new Turtle(world1);
turtle1.turn(90);
turtle1.forward(100);
turtle1.turn(45);
turtle1.forward(50);
```

The **`turn(int degrees)`** method commands the **`Turtle`** it is called on to turn any amount of angle in degree units. Run the program and see what the **`Turtle`** drew. Be sure these commands work properly before proceeding.

To complete this lab you will revise your TurtleTryer to make a new program named SquareDrawer so that when somebody runs it, it will make the computer draw a square.

GET CREDIT

Don't forget to show your TA what you've done before you leave so you get credit for attending and participating in the lab.

To finish and get FULL credit, go on a computer where you can access your work from this lab. is saved. If you don't know about a better way yet, go the a UA library computer, open the **`S:`** drive, and find your work there. Then, go on Blackboard and upload your five **`.java`** files for this lab. They must have different names, of course!

MORE TIPS

Instructions to access your S: drive in or out of lab: Navigate to
https://wiki.albany.edu/display/public/askit/Personal+File+Storage+Space+(S-Drive)
Get some more practice with **`Turtle`** objects and their methods by trying more at home, your dorm or the UA Library. In the Library: Start DrJava under "Java Development Tools". You'll have to download **java-sources.zip** from **http://coweb.cc.gatech.edu/mediaComp-teach#Java** , unzip and configure EXTRA CLASS PATH for bookClasses. Try some other shapes. How about a square centered at the current location? Hint: Use the **`penUp()`** and **`penDown()`** methods. Besides learning from the book what potential behavior Prof. Guzdial and Ericson programmed into their **`Turtle`**s, go to our CSI201 Web site and click (near the bottom) to see the full documentation of the bookClasses.