Computer Science 62
Lab 1

Wednesday, Jan 27, 2010

This laboratory is intended to get you started with the programs and systems in the lab, and to give you an opportunity to complete your first substantial program for the class.

Even though most of you have used Eclipse before, please follow the introductory steps below carefully. Correcting configuration errors is not impossible, but it is time-consuming and distracting.

1. **Setting things up:** You should all have your accounts and passwords. Log into one of the computers in the classroom. Among the icons in the dock at the bottom of the screen are four that will be important for the class.


Terminal


Eclipse


Safari


Emacs

   You will use Terminal and Eclipse today. Safari is a web browser (try it out!), through which you can read the course pages and documentation. Emacs is an editor that we will use later in the course.

   Open a terminal window and type the following commands. They will create a (private) folder for this class and a workspace within that folder.

```
mkdir cs062
chmod 700 cs062
cd cs062
mkdir workspace
```

(Explanation: The command `mkdir` makes a directory, and `cd` changes to that directory. The `chmod` command makes the directory private, so that others cannot see its contents. A few other commands for your information: `ls` lists the contents of the current directory, `cd ..` changes the directory to one directory lower, `rm` removes a specified file and you can type `man <command>` for any command and it will give you more information (short for "manual").

Now start Eclipse. When it asks you for a workspace, select Browse and find the `cs062/workspace` folder you just created. From the bar at the top of the screen, select `Eclipse,` then `Preferences`. Then in the `Preferences` window, expand `Java` and then `Build Path`, and then select `Classpath Variables`. Click on `New` and enter the name `BAILEY` and the path

```
/common/cs/cs062/bailey.jar
```

These steps need only be done once.

2. **Adding libraries:** Eclipse works in units called "projects." There will be one for each laboratory and each assignment in the class. To start a project, select `File/New/Java Project`. Give your project a name, like `Lab1`, but *do not click* `Finish` *yet!* Click `Next`, then `Libraries`, and then `Add Variable`. Add the variable `BAILEY`, and then click `OK` and `Finish`.

   This step adds the bailey library (which we just setup) to our project so that our project knows that code is available. This allows us to use the `import` command in our code to bring outside resources (i.e. resources that are not included with java). For most of the labs, you will need to add the `BAILEY` variable to your project.

3. Your program for lab today is to write a text-based version of the Silver Dollar Game. See Section 3.10 of *Java Structures* for a description of the game. Make sure you understand how the game is played before coding it up! Think about the methods described there and choose an appropriate data structure to model the game. This is a time for discussion in the lab; do not hesitate to ask questions or share your thoughts.

The behavior, in the console, will look like this:

```
_o____oo_oo_ Next move? 6 4
_oo____o_oo_ Next move? 2 2
Illegal move!
_oo____o_oo_ Next move? 1 1
o_o____o_oo_ Next move?
...
ooooo_______ You win!!
```

The pairs of numbers after "`Next move?`" signify the location of a coin and the number of squares that the coin is to move to the left. Remember we're computer scientists, which means we start counting at 0 not 1, so the leftmost square is location 0, not 1.

For this first exercise, we have given you a start on the code for the class `TextCoinStrip`. It can be found at `/common/cs/cs062/labs/lab1/`. To get it in to your directory, copy the file `TextCoinStrip.java` into your `src` directory. To do this, type:

```
cp /common/cs/cs062/labs/lab1/TextCoinStrip.java ~/cs062/workspace/Lab1/src/
```

(where "lab1" in the destination of the copy should be the name or your project for this lab). Then, go in to Eclipse and "ctrl+click" on the project and select "Refresh". You should now see the starting code.

Take some time to understand the code (I promise you'll be much better off spending 5-10 min. looking at the code before doing any coding yourself), then fill in the four missing methods.

- `toString` which creates the string representation of the strip,
- `isLegalMove` which determines if a move is legal,
- `makeMove` which makes a (legal) move, and
- `gameIsOver` which determines if the game is completed.

4. Before you leave the laboratory this afternoon, open Safari and navigate to the course Resources section of the class home page. At the top of the list is a link to the instructions for submitting assignments.

   For practice, submit your file `TextCoinStrip.java`. When you're creating your folder to submit, make sure to give it a name indicating

that it's your lab (e.g. "kauchakdave-lab1") so as not to confuse it with your first assignment. *This is a test. It is only a test.* You are not being graded on what you submit today. The file need not be complete or error-free. We simply want to make sure that you can successfully submit files.

5. **Extension:** This is a suggestion for those who want to go beyond today's exercise; it is not a course requirement. Let the computer assume the role of one player and experiment with strategies for the Silver Dollar Game. You may conduct your experiments in either the text-based or the graphical world. Here are some simple strategies:

    - Move the leftmost possible coin as far as possible.
    - Move the rightmost possible coin one square.
    - Find a coin capable of moving the farthest and move it the full distance.
    - A "gap" is a sequence of unoccupied squares between two coins. If possible, make a move to increase the number of gaps.