Lab 1

Due 14 February

Handout 2
CSCI 136: Spring, 2005
7 February

Java, Unix, Silver Dollar Game

1 Getting Started

During lab today, you will learn how to use the Java environment on the CS Lab Mac systems.

1. Go through the Unix tutorial handout. This will teach you how to log in and out of the machines, use basic Unix commands, and edit files with Emacs.

2. Identify the function of and experiment with these Unix Commands:

   ```
   ls          cd          cp          mv          rm          mkdir       pwd
   man         history     cat         more        grep        head        tail
   ```

   Identify the function of and experiment with these Emacs Commands:

   ```
   C-x C-s     C-x C-c     C-x C-f     C-x C-w     C-g         C-a         C-e
   C-d         C-_         C-v         M-v         C-s         C-r         M-%
   ```

   Learn these commands- you will use them often. Hints can be found in the Unix and Emacs web pages on the course website.

3. Make a directory in your Unix account for CS 136 work (perhaps "`136`" or "`cs136`" might be reasonable). Make a subdirectory `lab1` in this new directory for files related to this lab.

4. To set up your account to run the correct version of Java, you should enter the command

   ```
   source /Network/Servers/cortland.cs.williams.edu/Volumes/Courses/cs136/bin/136
   ```

   each time you log in. Rather than type this command every time, you can make it happen automatically by adding it to the file `.local_bashrc` in your home directory.

5. Write, compile, and run a Java program under Unix that prints the first ten odd numbers. Call it `Odd.java`. Turn in your source code for `Odd.java` using the `turnin` utility. To hand in the file `Odd.java`, type "`turnin -c 136 Odd.java`" at the Unix prompt.

2 Lab Program

This week, we will write the Silver Dollar Game at the end of Chapter 3. You should come to lab with a written design of the program.

For your design, you should first decide on an internal representation of the coin strip. Make sure your representation supports all of the operations necessary, ie. testing for a legal move, printing the board, testing for a win, moving pieces easily, etc. You should think about alternative designs and be able to justify your decisions. You may read ahead a little to Vectors if you like, but the lab can be implemented just as easily with arrays.

Once you have decided on a representation, write down the set of operations supported by your data structure. In other words, what are the public methods of `CoinStrip`, and what do they do?

When you are finished with the program, answer the thought questions at the end of the lab. Put your answers in a comment at the top of your file, which you should name `CoinStrip.java`. Turn in `CoinStrip.java` using the `turnin` utility— type `turnin -c 136 CoinStrip.java`.

As in all labs, you will be graded on design, documentation, style, and correctness. Be sure to document your program with appropriate comments, including a general description at the top of the file. Also use comments and descriptive variable names to clarify sections of the code which may not be clear to someone trying to understand it.

This program is due by 11:59pm on Monday, 14 February.