CS108, Stanford

Handout #10
Young

CS108 HW Logistics

Handout written by Nick Parlante and Vivek Choksi

Homeworks

- The homeworks look like systems -- made of many parts
- Do not start them the night before
- We grade the homeworks mostly by their computing correct output, and we look at your source code / OOP style a little. You should aim to write reasonably clean looking code. If code has excessively bad style, we will take points off. More importantly, you should write clean code just because it's the best way to author something that is satisfying and works correctly.
- You should not change public interfaces -- often we have some grading code that goes through the public interfaces, methods, etc., and that will break if you start changing methods around. You are free to add methods etc. that do not break the public interfaces. Very often my solution has the main method and then a decomposed out private helper method to tame the complexity.
- The starter code will often include the prototypes for methods you need to write and sometimes other boilerplate code. A starter method may include a throwaway line like "return 0;" just so it compiles when you first load the project.
- For testing, your code should return the correct output when fed valid inputs and called in correct ways. We will not worry about what happens when your code is fed invalid data unless the assignment specifically says you must handle bad input.
- You code can assume that all inputs are formatted and structured correctly. Your solution should work correctly when presented with valid inputs, and we will not worry about invalid inputs (yet). If your code takes in a parameter such as a String or a List, you may assume that the String or List passed in is not null, unless null is specifically mentioned in the spec. In other words, null is not a valid String or List. The empty string and empty list are valid of course, but in Java those are different from null.
- Your code should never change the public interfaces given for homework problems. Very often, we have tests that call your code in various ways, and obviously that only works if your code keeps the exact interface as given in the starter code. You are free to add additional or helper methods -- adding extra methods will not confuse our testing code.

Office Hours

- Office hours are listed on the course page -- both the regular hours and the special hours added for each assignment. Office hours are rather empty early on when each assignment is handed in, so early is a good time to get individual coaching.
- Post questions to Piazza. When possible, public questions are preferred. However, don’t post questions publically if it will give other students excessive information on how to solve the problem. If the problem will require us to step through your code to find your bugs, then bring it to office hours. We will not debug your program for you via e-mail or on Piazza.

Submit

There are five steps in submission.

1. Comment out your printlns and other extraneous output before turning anything in.

2. Create a README file in your top-level project directory.

Create a file named exactly "README" in your project directory. At the top of that file, write your username (that is, your SUNet ID) and your real name in (last, first) form. Ideally your name will match up with the way the registrar has it when we file grades at the end of the quarter.

```
user: jsmith05 (Smith, Jane)
```

We only need your name for the first assignment. For later assignments, we only need the username. On the lines below your username, you can put any notes for the grader.

```
user: jsmith05
I honestly can't tell you how much I enjoyed this assignment. And it made me miss the
season finale of Friends. And I didn't do part b, so sue me.
```

For a team project, write the usernames of each person on the team.

```
user: jsmith05
user: kjones
user: chrisyay
```

3. Copy your project directory to a leland machine, such as myth.stanford.edu.

   Option A: Use scp from terminal.

   ```
   scp -r /path/to/local/homework/directory \
       YOURSUNETID@myth.stanford.edu:/path/to/destination/directory
   ```

   The destination directory path can be relative to ~, which denotes your home directory. Since I have created a cs108 directory in my home directory on AFS, I would use the following command to copy over the first homework:

   ```
   scp -r /path/to/hw1CodeCamp vhchoksi@myth.stanford.edu:~/cs108
   ```

   Option B: Use a graphical interface to copy your files. Stanford provides free downloads of Fetch (for Mac), SecureFX (for Windows), and OpenAFS (for either). See the website for Stanford IT Services for more details. Other programs, such as SecureCRT or MobaXterm, can also do this.

4. From terminal, log into a leland machine using ssh and navigate to your homework project directory.

   ```
   ssh -Y YOURSUNETID@myth.stanford.edu
   cd path/to/cs108/project/directory
   ```

5. From within your homework project directory, run the cleanup and submit scripts.

   ```
   /usr/class/cs108/bin/cleanup
   /usr/class/cs108/bin/submit
   ```

   The cleanup script removes .class, ~file, and other leftover files from your directory. The submit script will complain about them if they are there, and it also does not like files larger than 500k (if needed, you can go ahead and submit with those files anyway). Alternatively, you could clean up in Eclipse: in the project menu, temporarily turn off the Build Automatically option. Select, the Clean command, with the Start A Build Immediately option off. This will remove all the .class files from your project directory.

   The submit script should work on any leland machine. If you get the "Submit success!" message, it worked. The script will verify that your name is in the README, check the files, and then copy all the files and nested directories up into submit space. You may submit any number of times -- we grade the last one submitted and ignore the others.

For any problems or questions regarding submission, please post to Piazza.