CS 100
Lab 6: shell commands

Reminder: find a *new* partner for this lab (**50% grade reduction** for working with a person ever worked before).
You are allowed to use 1 (one) computer per team (**50% grade reduction** for using more than 1 computer per team including laptops and tablets).

Grading: 11% each program, 12% attendance.

This lab will require getting to know and practicing shell commands (grep, awk, sed, ps, etc.). Submission details below. Do your best to accomplish the following tasks:

Note: These may take more than a single command and involve piping, wild cards, redirection, and multiple lines. Do not forget to look at the man pages for more information about options for commands.

Complete 8 out of 9 exercises for the full credit.

The Commands:

1. Use the ps command to list all of the current processes running on the system along with user information. Note the possible variations in argument syntax ps accepts.

2. Print the processes; stop one and then wake it back up.

3. List all processes which have been running for longer than an hour.

> **Hint**: The following command will print processes running from one to two hours; this is NOT the correct solution. You might have an easier time finding a correct solution using awk...
>
> ```
> ps -eo pid,cmd,etime | grep 01:..:..
> ```

4. Search for any files below your home directory which you are not the owner of.

5. Print all but the first and last line of a file.

6. Schedule a process to run at 2am which will list all of the files modified in the last 24 hours inside the user's home directory.

> **Note**: You can schedule a job with at, but the system here won't run it. Showing that it is scheduled is enough.
>
> > **Edit**: I'm not even sure the system here will schedule it; basically, if you can get it to the point where it says "Can't open /var/run/atd.pid to signal atd. No atd running?", that is

good enough.

7. Consider a directory which contains all of your source code for a single project. This may include sub-directories. For a given function name, find the file and line where it is declared. Can this be done to differentiate overloaded functions?

> **Note**: You may assume that there are no classes of the same name as the given function (this could make lines containing "FunctionName object = new FunctionName();" fail).

8. Create a compressed tar file with multiple files and sub-directories. Extract a single sub-directory. Insert a single file into the archive. Un-compress the archive into a new folder.

9. Given a text file with a list of **file names** (assume these files exist in current directory), append a string to the beginning of each file name and append a string to the end of each file name (do it NOT in the file treating the file names as strings, but actually rename files in the folder). Between each string and file name there should be a white space.

Submission:

Create a text file containing the list of commands necessary for each of the problems, and submit it to iLearn. And of course, demo your code to the TA.

Useful Links:

Literals in Bash: http://wiki.bash-hackers.org/syntax/quoting
sed tutorial: http://www.grymoire.com/Unix/Sed.html
awk tutorial: http://www.grymoire.com/Unix/Awk.html
grep tutorial: http://www.panix.com/~elflord/unix/grep.html