# Lab 03
# User environment: tonight we dine in shell

Hands-on Unix system administration DeCal

2012-09-17, due 2012-09-24

Your answers should only contain commands or brief responses. Please save trees, don't print out your computer's output.

If you don't know what a command does, skim the manpage, e.g. `man tee`.

1. What is the meaning of `A < B > C`? What about `D | E >> F`? What is the difference between how `C` and `F` are treated if the files already exist?

2. What is the difference between `echo "$PATH"` and `echo '$PATH'`?

3. Create a file using `touch` with a single space as the filename. Now remove it using `rm`, without using quotes in the command.
   Hint: use the escape character.

4. Create a file using `touch` with a single dash (`-`) as the filename. Now remove it using `rm`.
   Hint: you will not be able to remove the file using quotes or the escape character because rm will think that you're passing an option. Recall that each directory has a special hidden subdirectory which points to itself.

5. Create a file using `touch` that contains newlines in the filename. (To remove it, try using tab-completion. You don't need to provide this in your answer.)
   Hint: use quotes.

6. Print "Error, $world not found!" on standard error (stderr).
   Hint: use echo to print the message to stdout (be sure to properly escape or quote the dollar sign) and redirect stdout to stderr. If you're unsure how to redirect stdout to stderr, a search online should quickly provide you with answers.

7. Print your machine's hostname on stdout while simultaneously *appending* the hostname to a file.
   Hint: redirection using `>` will not work. You will need to pipe your output to `tee`.

8. Display the last line of `/etc/group`.
   Hint: use `tail`.

9. Print `/etc/passwd` in numeric order of UID (third column).
   Hint: use `sort` on OCF. You will need to take a look at `-n`, `-k`, and `-t` options.

10. File `alice` is supposed to contain only one a single line of input but has not been properly sanitized. Save only the alphanumeric characters of `alice` in a file `bob`.
    Hint: use `tr` with `-c` and `-d` options.

11. Using `find` and `xargs` separated by the null character, concatenate all `.conf` files in `/etc` and save the output in a file `conf_files`, while hiding error messages.
    Hint: to separate records with `NUL`, you will have to provide the `-print0` option to `find` *and* the `-0` option to `xargs`. To hide errors, redirect stderr to `/dev/null`.

12. Write a regular expression that matches hyphenated Social Security numbers.

**Extra for Experts™!**

The Fall 2008 lecture on shell scripting may be a useful reference for this problem. You can find it at: `http://decal.ocf.berkeley.edu/2008-fall/intermediate/week06_slides.pdf`.

Write a shell script, `piglatin`, that converts English into Pig Latin, one word at a time. Assume that sentences only contain letters and spaces — don't worry about numbers and other special cases. You can either use standard input (look up the shell command `read`), or command line arguments, whichever you think is easier. That is, your program should have one of the following interfaces:

- `echo "The quick brown fox jumped over the lazy dog" | piglatin`
- `piglatin The quick brown fox jumped over the lazy dog`

The Pig Latin algorithm: For each word $w$,

- if $w$ starts with a consonant, move its first letter to its end and append "-ay."
- if $w$ starts with a vowel, just append "-ay."

And some tips:

- If you plan using a `read` loop, be warned that `read` will read a whole line into a variable. To split it into words, you'll want to change `bash`'s Internal Field Separator.
- You can check the first character of a word using `case`. For example (note that your script has to handle uppercase letters as well):