CS114: Lecture 3
Shells

Have you done HW1? (due this afternoon)

Miscellany

- If you ssh from a terminal, you won't see the password as you type it
  - This is actually more secure
  - Also you can do ssh `netid@empire.csuglab.cornell.edu`
  - Instead of `ssh -l`*`netid`* `empire.csuglab.cornell.edu`
- To be able to *chmod*, you just need to be the owner
- *"aquí root, soy yo"*

A note on editors

- How to quit
  - Vi Escape-Colon-q-Exclamation-point
    - This quits without saving
    - To save, Escape-capital-Z-capital-Z
  - Emacs Control-X Control-C
- For this class, use nano/pico unless you already know vi/emacs

What's a shell, again?

- A program that you use to run other programs
- Many choices
  - DOS: COMMAND.COM
  - Windows: cmd.exe
  - UNIX:
    - csh / tcsh (“C shell”)
    - sh (“Bourne shell”)
    - bash (“Bourne Again Shell”)
    - ksh (“Korn shell”)
    - Zsh
    - ...

Starting a shell

- One is started for you when you login (`tcsh`)
- To start another, just run it:
  - `bash`
  - `tcsh`
  - Etc.
- To exit, type `exit`
- To use a different shell at login...
  - `chsh` ought to work but it doesn't
    - File a ticket

Shell substitutions

- `echo` *`stuff`*
  - Prints *stuff* to standard output
- Print out files in the current directory
  - `echo *`
- Print all files beginning with 'b'
  - `echo b*`
- Print all text files
  - `echo *.txt`
- Print all text files whose name is one character
  - `echo ?.txt`

More shell substitutions

- Print all files that begin with f or b
  - `echo [fb]*`
- Print all files that begin with an uppercase letter
  - `echo [A-Z]*`
- Print all music files
  - `echo *.{mp3,wma,m4p,m4a}`

Wildcards in general

- * = any sequence of characters (might be empty)
- ? = one character
- [...] = one character from this set
- {... , ... } = either the first thing or the second thing

A quiz

- Delete all jpgs in my home directory:
  - `rm ~/*.jpg`
- List all files that have 'pig' in their name and end with a number
  - `ls *pig*[0-9]`
- Print out a*b (“a” asterisk “b”)
  - `echo a\*b`

Command substitution

- `` ` *command* ` `` is replaced by the output of *command*
  - These are “backticks” - not single quotes
- List permissions on the `man` command
  - `ls -l `which man``

Shell variables

- Just like variables in Java, C, MATLAB, ...
  - All string-typed*
- Names start with $
- Two sorts
  - Shell variables (local to this shell)
  - Environment variables (passed to programs that you run)
- Important variables
  - `$PATH` (environment variable)
  - Prompt (`$prompt` in C-shell, `$PS1` in Bourne)

Using shell variables

- Print my path
  - `echo $PATH`
- What's the full path to `foo/bar`?
  - `$PWD` is the current directory
  - `echo $PWD/foo/bar`
  - How would I do this using the `pwd` command?

Setting shell variables

- C-shell family (csh, tcsh)
  - `set a=hello`
    - Now `$a` is `hello`
  - `setenv PATH /path/to/prog:$PATH`
    - Now we can find prog
- Bourne shell family (sh,ksh,bash)
  - `PATH=/path/to/prog:$PATH`
  - `export PATH`
- With no arguments, `set` & `setenv` print all variables

Quoting

- `mkdir 'My Documents'`
- `mkdir 'My Pictures'`
- `set type=Documents`
- `ls 'My $type'`
- `ls “My $type”`

- " does not interpolate variables, “” does.

Customizing shells

- All shells read and execute one or more files when they start
  - tcsh reads .cshrc (and .login if login shell)
  - bash reads .bashrc or .profile
- So, put your `setenv`s in your ....rc
  - Except don't.
  - I'll post instructions on the course website

Aliases

- Names for common options
  - `l` for `ls -l`
- Include options you don't want to type
  - `rm -i`
- Common typos
  - `dc` for `cd`
- `alias` w/o args lists all aliases