Software Carpentry / Shell Novice Lecture Notes

Craig Squires

December 2, 2017

Overview:

Setup

Introducing the Shell

Navigating Files and Directories

Working With Files and Directories

Pipes and Filters

Loops

Shell Scripts

Finding Things

Setup:

- Download and install the data
- How to open a shell
  - Linux
  - Mac OS
  - Windows

Introducing the Shell:

- GUI vs CLI/TUI (compare)
- Nelle's Pipeline
- my UNIX story: early '90s (the adventure)

Navigating Files and Directories:

- whoami (whereami (errors))
- pwd (paths, relative and absolute, / vs \, names)
- ls (switches (parameters vs arguments)):
  - -F, -R, -j, -a, -t, -r, 'man ls' vs -h, |more or less, how many?
  - digression: grep (-v ":", -v "^$")|wc
  - lots of commands: which ls, ls /usr/bin |wc
  - jump ahead: find . -type f, find . -type d
- cd (., .., ~)
- tab completion and history and **text** ("Ctrl-r", "!!", "!$",)

Working With Files and Directories:

- mkdir ('thesis', naming things: please no spaces!)
- less and nano (editor wars: vi, emacs, Gedit)
- touch (empty files, file update time)
- rm (deletion is forever..., rm -rf *, rm -i)
- mv (be careful of silent overwrites! -i)
- cp (also accepts multiple files if last arg is a directory)

Pipes and Filters:

- wc ('molecules': wc *.pdb, *t*ne.pdb, *t??ne.pdb, -l, -w, -c)
- wildcards: *, ?, [abc]
- redirection: wc -l * > lengths.txt
- echo: echo "hello" > hellos.txt, echo "hello" >> hellos.txt
- cat: cat hellos.txt, cat lengths.txt
- head:wc -l * |head -n 1
- tail:wc -l * |tail -n 1
- sort:wc -l * |sort -n
- each process has 'channels': STDIN, STDOUT, STDERR
- "|" and ">" and "<" connect these channels
- cat data/animals.txt |head -n 5 |tail -n 3 |sort -r |uniq
- cat data/salmon.txt |uniq, sort data/salmon.txt |uniq (sort -u)
- cut -d , -f 2 data/animals.txt (|sort |uniq -c |wc -l)
- sed and awk

Loops:

- for/in/do/done: 'creatures': for [i or x or file] in [list or fork] do [task] done
- how to get lists:
  - wildcards ("*", "?", "[]")
  - forks: $(commands: 'ls', 'find', 'cut')
- dry runs (you could damage things...): use echo "to show what would be done"
- nested loops: problem: in data/pdb
  - find how many atoms, and how many atoms of each type, in each file

Shell Scripts:

- ’bash filename’ (forking and environments)
- arguments: $1 $2 etc. (”$@”)
- comments
- extract a script from history (use the number of atoms problem)
- file magic and permissions (”hash bang slash bin slash bash” (shebang))

Finding Things:

- grep
- find
- ”$()” or ”“”
- man command