Introduction to UNIX and the Shell

Hands-On UNIX System Administration DeCal

Lab 1 — 27 August 2012

Introduction

Today we went over the shell, a few common UNIX commands, text streams, substitution, and `vi`. In this lab, you'll get some practice with these tools while learning more about the shell.

This lab is due in hard copy at the beginning of next class (that is, by 6:10 PM on the 10th). You can type up your answers in a text file and use `lpr` to print it (your cs198 account comes with 150 free print pages); if you want to use a word processor, try `soffice`. Please **make sure your submission includes your name and your cs198 and/or OCF account name**. If you get stuck, ask for help!

1 The basics

In a sentence or two, precisely describe what each of the following commands does. (You may need to look them up in the `man`ual; some commands are shell builtins that don't have manual entries, but you can always try them out!)

Note: The commands below are not necessarily in order of which you run them.

1. `cd .` Hint: You might want to run the next command before this one. (What is the difference between . and ..?)
2. `cd ..`
3. `cat /etc/hosts /etc/motd`. Fun fact: You'll often see constructions such as `cat README | grep garply`. This is often unnecessary, as many commands will accept files as arguments (`grep garply README`), and using commands this way will win you a Useless Use of Cat award.
4. `ls -a`, `ls -l` and `ls -d`.
5. `cd /tmp`.
6. `mkdir -p foo/bar/baz`.
7. `man mkdir`.

8. `rmdir *`. Hint: `rmdir`'s behavior is not immediately obvious. Be careful.

9. `less ∼cs198-8/longfile`.

10. `find ∼ -name '.*' 2>/dev/null`. Hint: Break down each part of the command first.

11. `touch file`.

12. `pwd`.

13. `cp file /tmp`. Hint: Pay attention to /tmp. This is a bit subtle.

14. `mv file /tmp/newfile`.

15. `echo the foo baz`. Note there are no quotes.

16. `uname -a`. What kind of information does this command print out?

17. `history`. This command should be pretty obvious.

18. `bash` or `tcsh` or `zsh`. Specifically, what do each of these commands do?

19. `apropos download`.

2 Extra for Experts$^{\text{TM}}$!

2.1 Editing text

UNIX text editors are not the most user-friendly tools around — some, like `vi`, don't even let you enter text without reading a manual first! Vim (Vi IMproved) has a tutorial, which you can access by running `vimtutor`; you don't need to work through the whole thing, but at least get comfortable with the basics. ^[is equivalent to the Escape key, which will save you a bit of effort.

The shell doesn't (by default[1]) use modal editing, but supports some useful keyboard shortcuts that are worth committing to memory:

- **^U** — delete all text from your cursor to the beginning of the line
- **^K** — delete all text from your cursor to the end of the line
- **^W** — delete the last word (delimited by spaces)
- **^A** — jump to the start of the line
- **^E** — jump to the end of the line

Another useful keyboard shortcut, while not strictly related to editing text, is **^L**, which clears your display.

[1] Many programs use GNU Readline to handle text input; you can configure readline to use Emacs or vi keybindings, depending on which you prefer.