Administrivia

• This is where I’ll remind you about upcoming deadlines, etc.

Syllabus / More Administrivia

• One purpose of the syllabus is to spell out policies, especially about:
  – Course requirements and grading.
  – Late work.
  – Academic integrity.

• Most other information will be on the Web, either on my home page (office hours) or the “course Web page”.

• Part of my job is to answer your questions outside class, so if you need help, please ask! E-mail usually works well if office hours don’t.
What I Hope You Will Get From This Class

- More things in your “bag of tricks” — shell features, shell scripts, makefiles, a text editor, etc., etc.,
  (Most of what we talk about will be applicable to all UNIX systems, not just Linux.)
- Practice in reading man pages and otherwise learning more.
- Exposure to a different operating system / user interface paradigm.

Shameless Evangelism/Ranting

- “UNIX is obsolete — history goes back to 1969!”
  You can fix a lot of bugs in 35 years, and the odds are better that what you learn will still be useful years from now.
- “It’s not user-friendly!”
  Sure it is; it’s just choosy about its friends. Designed by programmers for programmers — “expert-friendly” as opposed to “novice-friendly.”
- “Everyone knows GUIs are better!”
  For some things and some people, maybe so. But which is more expressive, pointing and gesturing or speech?
- (You don’t have to agree with me; listen and decide for yourself.)
The UNIX Philosophy

- As stated by one of its developers (Doug McIlroy):
  “Write programs that do one thing and do it well. Write programs to work together. Write programs to handle text streams, because that is a universal interface.”
- There’s more, but the emphasis is on (1) providing a set of lightweight tools that can be put together to do interesting things, and (2) providing choices to users (sometimes almost too many!)

Basic Organization / Terminology

- Kernel — heart of operating system, manages processes and files and so forth.
- Shell — program that interprets what you enter, calls (“launches”) other programs.
  This being Unix, there are several, mostly offering similar functionality but maybe with different syntax.
  Several ways to start a shell — next slide.
- Commands — internal versus external.
- Graphical environments, window managers, etc. Also several of these!
Starting a Shell

- From the console, type control-alt-$n$, where $n$ is 1, 2, \ldots 6, and log in. (To get back to the graphical virtual console, control-alt-F7.)

- From a graphical environment, start a “terminal emulator” (*xterm*, *gterm*, etc.). If your desktop has a taskbar, might be good to put a “start a terminal” icon on it. (For GNOME, right click on taskbar, then “add to panel”, “launcher from menu”, etc.)

- From a Windows system, run *putty*.

Reading The Fine Manuals

- One of the most useful things you can learn is how to learn more.
  Documentation on UNIX systems is not always perfect, and it’s not particularly novice-friendly, but usually it’s thorough.

- Places to look:
  - *man* pages. Organized into “sections” (user commands, sysadmin commands, library functions, etc.). *apropos* or *man -k* are useful.
  - *info* pages.
  - Elsewhere on the system, *locate* on Linux may help.
  - The Web, via your favorite search engine.
  - Usenet, including Google’s archives (click “Groups” from Google’s main page).
RTFM, Example

- Try `man man`.
- Of particular interest is the section `SEE ALSO`.
- Try `apropos`.
- Now you might want to know about more, or less.
- To learn more about `info`, try `info info`.

Minute Essay

- What are your goals for this course? Are there specific topics you're interested in?
- Do you have access to a Linux or UNIX system other than the department's lab machines?
- Have you bought / will you buy a copy of the recommended book? ("No" is okay — I just want to know.)