Slide 1

Administrivia

- About your Linux accounts, the mystery of the missing e-mails has been resolved: All of you already *had* accounts (created the first time you registered for an ENGR course), and — well, passwords were sent when they were created and presumably ignored because not needed.
- The approved way to deal with forgotten passwords is to go to the ITS help desk and ask them to reset your password, making sure they know this is for the CS machines (and not your regular ITS account). They'll ask for your ID.
- Our student ACM chapter will be doing peer tutoring for intro courses, including this one, 5pm–9pm M/T/W/R in this room. Details in e-mail soon. Excellent resource for getting help with homeworks!

Slide 2

More Administrivia

- Homework 1 on the Web, linked from the "lecture topics and assignments" page. Due next Wednesday (11:59pm).
- Notice that usually you will turn in homework by e-mail. Please send from your Trinity e-mail and put in the Subject line which assignment *and which course*, to be sure I put it in the right folder for grading.

  If you have *questions* about the homework, be sure to say that in the Subject line — and it doesn't hurt to add "urgent".

Slide 3

Minute Essay From Last Lecture

- Many very interesting questions! Several about compiler/linker versus interpreter, which I think is useful background but may only start to make sense as we start to with the relevant tools.
- Also some questions about various languages. Why so many? I say because (1) ideas about "good" languages evolve and (2) people don't all agree about "good" (and indeed it can depend on application area).

Slide 4

Getting Started with Linux

- When you log in, you should get a graphical desktop, which should be navigable with what you know from using other graphical environments (though some details are different).
- In Linux, we talk about files and directories; the idea is the same as Windows's files and folders, though again some details are different.
- The graphical system should give you a way to get a terminal window. Once you have that . . .

Slide 5

Getting Started With the Command Line

- What you get when you start a terminal window is a "command shell", similar to Windows' "MS-DOS prompt".

  Rather than pointing and clicking, you type the name of the program you want to run, plus whatever arguments (parameters) it needs.

- (Why would you want to use a command line? because for some things it's arguably more efficient, and it's "scriptable" in ways that GUIs typically aren't.)
- Let's try some commands . . . (Don't worry if this goes by quickly — you should plan anyway to spend some time outside class trying out what we do in class and what's in the "reading" (video lectures).)

Slide 6

Some Commands

- `pwd` shows the current directory. (When you give a filename, it's relative to this directory unless you give a full pathname.)
- `ls` lists the current directory. Add `-l` to get more information.
- `cd foo` changes to directory `foo`. Just `cd` goes back to your home directory. Try `cd Local` and then `ls`.
- `mkdir foo` creates a director `foo`. Might be useful to create one for your files for this class (call it csci1312, maybe, or CS1).
- `passwd` changes your password. (Not a command you'll want often, but maybe now!) Notice that this command does *not* echo what you type.

Slide 7

Useful Command-Line Tips

- The shell (the application that's processing what you type) keeps a history of commands you've recently typed. Up and down arrows let you cycle through this history and reuse commands.

  (Pedantic aside: "The shell" here means the one you're most likely to be using. There are other programs with similar functionality you could use instead.)

- The shell offers "tab completion" for filenames — if you type part of a filename and press the tab key, it will try to complete it.
- To learn more about command `foo`, type `man foo`. This is reference information rather than a tutorial, but usually very complete. `man -k foo` will give you a list of commands having something to do with `foo`.

Slide 8

Remote Access

- One of the strengths of a command-line enviroment is that it works well for "remote access" (using the computer when you aren't sitting in front of it).
- To do this from another UNIX-like computer, use `ssh`. `scp` and `sftp` can be used to copy files.
- From a Windows computer, install either Cygwin (UNIX-like toolkit) or PuTTY (terminal emulator). I'll send e-mail with details, but be advised that one option — which will work even from off campus — is to connect to Trinity's VDI system, which has PuTTY installed.

Slide 9

Text Editors

- Many, many text editors, and people have favorites. Notepad is an example from the Windows world.
- I use and will teach in this class `vi`: It's found on every UNIX/Linux system I know of, and is very powerful, though it takes some getting used to. (`vi` on our Linux machines is actually `vim`, a more featureful "clone" of the original `vi`.)
- Other popular Linux text editors include `emacs`, `pico`, and `gedit`. Advice: Give `vi` a real try first, but if using it is just too painful, use something else!

Slide 10

`vi` Basics

- `vi` has two *modes* — insert mode (where what you type goes into the file) and command mode (where you can type commands to copy, move, delete, save, etc.).
- You start an editing session by typing, e.g., `vi hello.txt`. It starts in command mode. Enter insert mode by typing `i`. Exit by pressing `ESC`. Move around with the arrow keys. (Try entering some text.) Backspace deletes a single character.
- Save by typing `:w`; exit by typing `:q`.
- *Highly recommended:* `vimtutor` brings up an interactive tutorial. (Homework 1 asks you to try it.)

Slide 11

`vi` Tips — Errors/Mistakes

- `u` means "undo" the previous action (insertion, deletion, paste). Repeat to undo multiple actions.
- `:q!` exits without saving. Useful if you make a complete mess of things.

Slide 12

More Commands

- Now that we have a way of creating files, we can try out some other basic commands.
- `cat` to show contents of a file. `more` or `less` to show it a screenful at a time.
- `cp` to copy one file to another. `mv` to move or rename a file. For both, `-i` to warn about overwrites.
- `rm` to delete a file. (Note — no recycle bin, so use with caution! or `-i` to prompt.) For this and `cp` and `mv`, `-v` shows what's being done.
- Other useful/interesting commands in video lectures and next time.

Slide 13

UNIX Filesystem Basics

- Unlike in Windows (and Mac, usually), UNIX filesystems are case-sensitive (so hello and Hello are different files).
- Files have two levels of ownership — "owner" (user) and "group". Groups allow sharing files with some but not all users.
- File access is controlled by "permissions". Three levels (owner, group, and everyone else), three types of access (read, write, execute).
- `ls -l` shows permissions. `chmod` changes them.

Slide 14

Minute Essay

- If you've used a command-line environment, how does this one compare to what you've used before?
- Anything surprising or startling today?