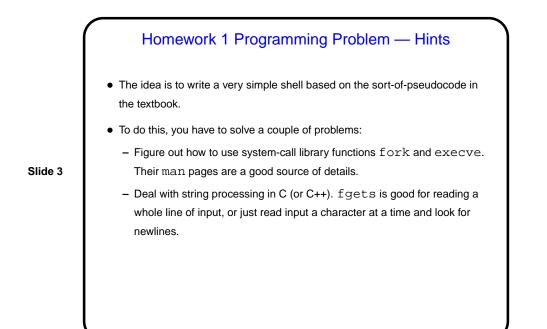
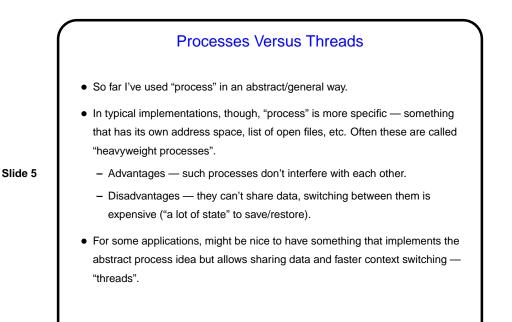
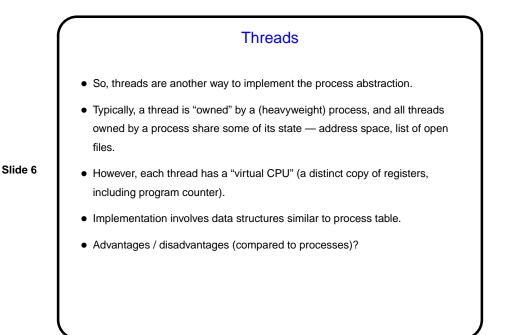


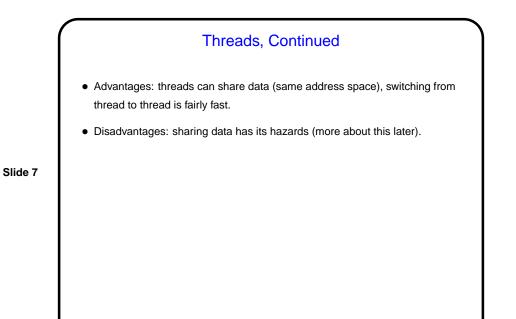
Minute Essay From Last Time
"CPU" is kind of ambiguous these days. Usually I use it to mean "processing element" (processor/core).
Maximum of one running process per CPU (in this sense). Minimum number depends on whether there are ready processes. (And if you're pedantic, maybe there is some small interval during a context switch in which there is no process running on the CPU involved.)



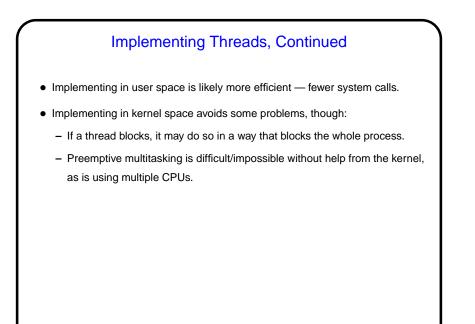
C Programming Advice
I strongly recommend always compiling with flags to get extra warnings. There are lots of them, but you can get a lot of mileage just from -Wall. Add -pedantic to flag nonstandard usage.
If you want to write "new" C (including C++-style comments), add -std=c99.
If typing all of these gets tedious, consider using a simple makefile. Create a file called Makefile containing the following: CFLAGS = -Wall and then compile hello.c to hello by typing make hello.







Implementing Threads
Two basic approaches — "in user space" and "in kernel space" Various hybrid schemes also possible.
Basic idea of "in user space" — operating system thinks it's managing single-threaded processes, all the work of managing multiple threads happens via library calls within each process.
Basic idea of "in kernel space" — operating system is involved in managing threads, the work of managing multiple threads happens via system calls (rather than user-level library calls).
How do they compare?...



Slide 9

Adding Multithreading • If you've written multithreaded applications - moving from single-threaded to multithreaded not trivial: - Figure out how to split up computation among threads. - Coordinate threads' actions (including dealing properly with shared variables). • Similar problems in adding multithreading to systems-level programs: - Deal properly with shared variables (including ones that may be hidden). - Deal properly with signals/interrupts.

