Slide 1

Administrivia

- Reminder: Homework 5 due next week.
- Grades for Homework 4 mailed. Summary of grades up to Monday coming soon.
- Sample solutions for homeworks will be online. I will send mail.

Slide 2

A Very Little About "Random" Numbers

- Homework 4 asked you to work with the library functions `srand()` and `rand()`. Belatedly, a few words about what they do . . .
- First, what we mean by "random" is (I think!) an interesting question with no obvious answer. What's often wanted is something that can't be predicted, and it's not clear we can get that with a system that's deterministic. Further, even if we could, we might not want that, since we often want to be able to repeat a test.
- So, often what we really want is a "pseudo-random number generator" — something that generates a sequence of numbers that looks random but is repeatable given some reproducible starting point.
- Early researchers apparently thought more-complex algorithms would give better results, but — not necessarily. Very simple algorithms can give quite good results.

Slide 3

A Very Little About "Random" Numbers, Continued

- Lots of uses for "random" sequences (e.g., so-called "Monte Carlo" methods for simulating things), so many libraries include function(s) to produce them.
- Typical library provides some way to set the starting point (the "seed") and then a function that when called repeatedly produces the sequence — `srand()` and `rand()` in standard C. Mostly these produce a large range of possible values. (Why is this good?)
- Some libraries also provide functions to map the full range to a smaller one (e.g., to simulate rolling a die). C doesn't, but there are some semi-obvious approaches. The problem on Homework 4 asks you to do a simple comparison of two of them.

Slide 4

Loops and Arrays — Another Example

- Homework 5 asks you to complete a starter sort program, as a way of getting more practice with `for` loops and arrays.
- Code should be pretty readable, but review?

Slide 5

Pointers and I/O — Tips and Examples

- Homework 5 also asks you to do some work with character data. Two things to note:
- It's easier in C (possibly in contrast to your first language) *not* to read input in big chunks. Here, a character at a time works fine and avoids potential problems.
- The suggested method for modifying characters uses pointer arithmetic, so — an example of that is in the sample program `simple-strings.c`.

Slide 6

Minute Essay

- None really — unless questions?