Slide 1

Administrivia

- Reminder: Homework 9 due today.
- Homework 10 on the Web. Due last day of class. This one is a bit long and not easy, but I think it does something interesting.
- Quiz 6 rescheduled for next Wednesday. Likely topic is `structs`. Starting on the homework may help you prepare.

Slide 2

One More Way to Get Input

- Getting input from standard input or a text file is kind of a pain in C. One way that some sources recommend is to read it a line at a time with `fgets` and then do your own "parsing", possibly using `sscanf`. An advantage of this method is that it lets you fairly easily discard bad input (e.g., if you want to prompt again).
- (Example: Program to read grades and print average. Also uses a `struct` and could be extended to sort the grades.)

Slide 3

Non-Text ("Binary") Files in C

- Up to this point we've dealt only with text files. But surely not all files are plain text? right, so how to work with them?
- In C, can have "binary files". Open as usual, but include `b` in "mode" parameter (e.g., `"rb"` rather than just `"r"`). What's different about this? on UNIX systems, nothing really, but on other systems, lines can be separated with two-character sequences, and processing of text streams automatically converts to/from single `'\n'`. Opening files as binary avoids that.

Slide 4

Binary Files in C, Continued

- Open with `fopen` and close with `fclose` as usual, except include `b` in mode when opening.
- Read with `fread`; write with `fwrite`. (Look at `man` pages for description of parameters.)
- Probably worth noting that while text files are relatively portable (many tools deal with converting between different conventions for line separators), binary files probably aren't.
- (Example.)

Slide 5

Non-Sequential File Access

- We've also been reading and writing files sequentially, and often that's what you want, but not always — consider for example a big database of some sort, where it would be nice to somehow represent the data in some way that avoids having to read the whole file to find one piece of information. Exactly how to do that might be complicated, but to do it at all, need some way to read file non-sequentially — "random access".
- In C, functions `fseek` and `ftell` make this possible.
- (Also note `rewind`.)

Slide 6

Minute Essay

- Anything noteworthy about Homework 9?
- (And best wishes for a good holiday!)