Slide 1

Administrivia

- Homework 6 on Web; due next Tuesday.
- Office hours today 4pm to 5pm.

Slide 2

Sidebar: Return Values from `main`

- Early on I (may have) said that the value returned from `main` can be a way for your program to tell the environment (command shell, for us) whether the program worked. But we've always returned 0, no matter what.
- Better: Return 0 if the program works, something nonzero if there was an error.
- Still better: Use `EXIT_SUCCESS` and `EXIT_FAILURE` (requires `#include <stdlib.h>`).

Slide 3

Arrays in C — Recap and Examples

- Basic idea is to provide something analogous to subscripted variables in math.
- Let's do some examples . . .

Slide 4

Sorting and Searching

- Something we often want to do is put things in order — similar to "alphabetizing" a list of names. Techspeak for this is *sorting*, and it can be done to anything for which you can define an ordering.
- A related problem is *searching* ("does this array contain a specified element?"). One motivation for sorting is that it makes searching much faster. (Why? Well — how would you search for something in a list, if the items are in no particular order? How does it help to know that they *are* in order? Think about searching for a particular word in a dictionary.)
- So, if you have a list of things, how would you put them in order?

Slide 5

Sorting

- Many ways to put a list of things in order. Some are simple to understand and to code, but slow. Others are somewhat more complicated, but faster.
- Simple-but-slow methods:
  - Bubble sort: Repeatedly go through the list exchanging adjacent elements that are out of order.
  - Selection sort: Find the largest (or smallest) element and put it at the appropriate end. Repeat with the next largest (smallest) element, putting it next to the end, and so forth.
  - Insertion sort: Start with one element, and "insert" subsequent elements into a sorted-list-so-far.

  All of these have running time proportional to N^2, where N is the number of things to sort. (Better algorithms have time proportional to $N \log N$.)

Slide 6

Searching

- Simple-but-slow way — *sequential search*.
- Less-simple-but-faster way, for sorted data — *binary search*. Somewhat similar to what you do when you look up a word in the dictionary.

Slide 7

Minute Essay

- Write some C code (not a complete program) to declare an array `a` of 100 integers and set each element `a[i]` to `i*10`.

Slide 8

Minute Essay Answer

- Here is one way.

```
int a[100];
int i;
for (i = 0; i < 100; ++i) {
    a[i] = i*10;
}
```

(Really it would be better to have a `#define N 100` and use `N` rather than 100 — then if you want the array to be bigger or smaller you only have to change one place.)