Slide 1

Administrivia

- Reminder: Homework 3 due today. Remember that I want two things:

  Your source code, in a form I can compile (so, plain text).

  Your pledge and essay, in something I can easily read. Plain text preferred, PDF okay; anything else I won't read.

  If you're submitting a not-final draft, please mark it as such — in file name or in pledge/essay.

- Homework 4 posted; due in two weeks. Two problems, both a tad harder than the problems in previous assignments. I advise not waiting until the last minute!

Slide 2

More Administrivia

- Note that I don't usually reply much to video-quiz responses — it's just too overwhelming — though on occasion I will. (My thinking is that for questions with some notion of a right answer, the slide after question(s) will have my answer(s), so you should have some idea whether what you said was right.)

Slide 3

Recap of Video Lectures

- Loops in C (`while` and `for`).

  Most people's answers to quiz question about printing powers of 2 fine. Note that I recommend *not* using `pow()` to compute integer powers of integers: It converts to/from `double`, with possibly loss of precision, and may be less efficient.

- Arrays in C.

  Why no checking of array indices . . . Partly an efficiency measure, but also no way to do it in general without storing length with array.

- A little about "random" numbers, plus a digression about TeX.

- Questions?

Slide 4

`main()` Revisited

- If no access to command-line arguments needed, declare `main()` as

  `int main(void)`

  Note that in C this is subtly different from

  `int main()`

- Return value from `main()` should be zero if program "worked" (whatever that means in context), something else if it didn't. Appropriate values for the "something else" somewhat implementation-dependent. Nice touch to `#include stdlib.h` and use

  `EXIT_SUCCESS` and

  `EXIT_FAILURE`.

Slide 5

Homework 4

- Second problem gives many students trouble.
- Outline of what you're supposed to do:

  Generate N "samples" using `srand()` and `rand()`.

  Map each to range from 0 to $B - 1$, where B is a number of "bins", and count how many fall into each "bin".

  Starter program has code to do both mappings, so no need to struggle with that.

Slide 6

"Practice Problems" from Previous Lecture

- (Review problems.)
- Anyone come up with good solutions?
- Or should we/I do these in class?

Slide 7

Minute Essay

- If you haven’t watched the recordings for the past two weeks, is there a problem I should know about?