Slide 1

Administrivia

- Homework 9 due date extended to December 10 (Monday during finals).

  No homework accepted past December 12. (You can still get partial credit for all assignments, as long as you haven't looked at sample solutions.)

- About getting help with Homework 9 (and others, if you're behind):

  ACM tutoring ends on the last day of class.

  I'm tentatively planning to have some office hours next week (times TBA by e-mail), but it's probably best not to count on that. But I do respond pretty well to e-mail!

Slide 2

Computer Representation of Data of Numeric Data, Revisited

- Many (most?) languages strictly define sizes of data types. C defines only minimum ranges. Why?? to allow implementations to do whatever is most efficient, while allowing programmer to make *some* assumptions.
- Example program `sizes.c` gives different answers on a 32-bit system!

Slide 3

C Programming using Non-Standard Features and Libraries

- C's standard library is pretty limited, in keeping with the idea that the language should be implementable on a very wide range of platforms of varying capabilities.
- So if you want to write completely standard and portable C, there are a lot of things you can't do.
- However, a lot of real-world uses of C require going outside the standard (e.g., programming those embedded systems, where you have to interact with hardware in low-level way).
- And there are a *lot* of non-standard libraries, some platform-specific, that do interesting or useful things . . .

Slide 4

C Non-Standard Features and Libraries

- Multithreading available with the OpenMP extensions. Fairly cross-platform.
- Parallel programming over a network available with MPI ("Message-Passing Interface"). Also fairly cross-platform.
- Parallel programming using GPU available via OpenCL. Somewhat cross-platform.
- Many other things — multithreading, GUIs/graphics, networking, etc. — are typically done with platform-specific libraries, though some cross-platform libraries exist.

Slide 5

C Non-Standard Features and Libraries — UNIX/Linux

- Text-based GUI-ish features available (most systems?) via the `ncurses` library.
- "Real" GUIs and graphics available via a lot of libraries. "X11" is very low-level; other "toolkits" available.
- Networking available via "sockets".

Slide 6

Just For Fun — "Extreme" ASCII Art?

- Some of you may have heard of "ASCII art"? a truly over-the-top example, from quite a while ago, can still be found, via

  `telnet towel.blinkenlights.nl`

  (to interrupt control-`]` then "quit" or control-d — although this doesn't seem to work in a terminal window??)

  (For a while recently the site wasn't working. Seems okay now?)
- (What some people choose to do with their time can be — interesting?)

Slide 7

Quotes of the Day/Week/?

- From a key figure in the early days of computing:

  "As soon as we started programming, we found to our surprise that it wasn't as easy to get programs right as we had thought. Debugging had to be discovered. I can remember the exact instant when I realized that a large part of my life from then on was going to be spent finding mistakes in my own programs." (Maurice Wilkes: 1948)

- From someone in a discussion group for the Java programming language:

  "Compilers aren't friendly to anybody. They are heartless nitpickers that enjoy telling you about all your mistakes. The best one can do is to satisfy their pedantry to keep them quiet :)"

Slide 8

Course Topics — Recap

- Basic C programming, for people who already know how to write programs in some other language. Especially useful (I think!) for those who start in a very abstract/high-level language.
- Review of the Linux/UNIX command-line environment and command-line development tools.
- Review of basics of computer arithmetic and data representation. A little more about floating-point representation.

Slide 9

Why Learn C? (For Java/Python/Scala Programmers — Recap)

- Scala and Python (and Java, less so) provide a programming environment that's nice in many ways — lots of safety checks, nice features, extensive standard library. But they hide a lot about how hardware actually works.
- C, in contrast, has been called "high-level assembly language" — so it seems primitive in some ways compared to many other languages. What you get (we think!) in return for the annoyances is more understanding of hardware — and if you do low-level work (e.g., operating systems, embedded systems), it may well be in C.

Slide 10

Minute Essay

- None really — just sign in.
- Best of luck with your finals, and best wishes for a good holiday!