Slide 1

Administrivia

- Homework 1 to be on Web tomorrow. Due next Monday.

Slide 2

Variables and Expressions — Review/Recap

- In order to do anything useful we usually (though not always!) need some variables. In C, variables must be *declared* before being used. (Contrast with Python?) Declaration specifies name and type. (Contrast with Scala?)
- Once you have variables, you can assign values to them, using *expressions* that range from simple constants to complex math-y formulas involving constants and/or other variables.

Slide 3

Statements in C

- C programs are made up of *statements* (usually collected inside *functions*).
- Statements come in several types:
  - Null (`;`).
  - Expression (*expression* `;`).
  - Return (`return` *expression* `;`).
  - Compound (conditional execution, various kinds of loops).

Slide 4

Functions in C

- Functions in C are conceptually much like functions in other procedural programming languages. (Functions in object-oriented languages are similar but have some extra capabilities.)

  I.e., a function has a *name*, *parameters*, a *return type*, and a *body* (some code).
- One distinction in C is that you aren't supposed to use a function before you tell the compiler about it, either by giving its full *definition* or by giving a *declaration* that specifies its name, parameters, and return type. The function body can be later in the same file or in some other file.

Slide 5

Conditional Execution

- Also as in other procedural languages, C has syntax for saying that some code should be executed only if some condition holds.
- Syntax is

  `if (` *boolean-expression* `)`

  statement1

  `else`

  statement2

  where *statement1* and *statement2* can be single statements or blocks enclosed in curly braces.
- You can build up chains of conditions by making the statement after `else` another `if`, and you can omit the `else` and following statement. (The ideas here should be very familiar; only the syntax should be new.)

Slide 6

Example — Finding Roots of a Quadratic Equation

- As an example of all of this, let's write a function that finds and prints the root(s) of a quadratic equation of the form

  $$ax^2 + bx + c = 0$$

  using the familiar(?) formula

  $$x = \frac{-b +/- \sqrt{b^2 - 4ac}}{2a}$$
- And then to test this function we'll write a main program that calls it with different inputs. (How to get input from a human user? defer that until after we talk about pointers.)

Slide 7

Minute Essay

- Try writing a simple C function that takes one `int` parameter and prints whether the parameter is positive, negative, or zero.

Slide 8

Minute Essay Answer

- Here is one answer:

```
void silly(int x) {
    if (x < 0)
        printf("negative\n");
    else if (x > 0)
        printf("positive\n");
    else
        printf("zero\n");
}
```