Slide 1

Administrivia

- Reminder: Homework 5 code due today.
- Homework 6 due dates posted (next week).

Slide 2

Recursion — Overview

- Basic approach:
  - Identify "base case" — something you can solve directly.
  - Figure out how to decompose non-base cases into "smaller" problems, and apply algorithm to smaller problems.
- How to think about "does it work?"
  - Does it work for base case(s)?
  - Assuming recursive calls work, does it work for other cases?
  - Does every recursive call get you at least one step closer to a base case?
- Implementation — conceptually (and usually in fact) involves a stack of calls-in-progress.
- Can be slower than iteration (though sometimes not), but can also be much easier to understand.

Slide 3

Recursion — Simple Examples

- Factorial function.
- Function to compute Fibonacci numbers (very slow!).

Slide 4

Recursion — Parsing an Arithmetic Expression

- "Fully parenthesized arithmetic expression" is one of two things:
  - A number n.
  - Something of the form

    $$(e \ op \ f)$$

    where e and f are expressions and op is one of the four arithmetic operators.
- How to evaluate one of these?
- Let's write code for that . . .

Slide 5

Recursion — More Examples

- Quicksort — pick "pivot" element, split array into elements less than pivot and elements greater than pivot, and sort recursively. Why does this work?
- Mergesort — split array (or list) into two pieces of equal size, sort recursively, merge. Why does this work?
- Filling the area inside a border.

Slide 6

Minute Essay

- None — quiz.