Slide 1

Administrivia

- Reminder: Homework 3 due 5pm today.
- Homework 4 and Homework 5 on Web; due March 3 and March 6 respectively. Not accepted later than March 6.
- Longer-term reminders: Quiz 3 next Wednesday; midterm the following Wednesday.
- (Yes, we're going to move a little more quickly for the next week and a half.)

Slide 2

Some Words of Advice

- General advice about the class: In general, you may find it easier to follow what we do in class if you keep up with the reading.
- Specific advice if confused about proofs: Study as many examples as you can — this is probably the best, maybe the only, way to learn this material.

Slide 3

Recursion and Recursive Definitions

- Idea of recursion closely related to idea of induction — "build on previous smaller cases".
- First look at recursive definitions. To define something recursively:
  - Define one or more "base cases".
  - Define remaining cases in terms of other ("smaller") cases.

Slide 4

Recursive Definitions — Sequences

- A silly example:

$$\begin{aligned} S(1) &= 1 \\ S(n) &= S(n-1) \times 10, \quad \text{for } n > 1 \end{aligned}$$

  Try writing down some terms.

- Another example:

$$\begin{aligned} S(1) &= 1 \\ S(2) &= 1 \\ S(n) &= S(n-2) + S(n-1), \quad \text{for } n > 2 \end{aligned}$$

  Try writing down some terms. Anyone recognize this one?

Slide 5

Recursive Definitions — Sets

- Example — could define the set of "integer arithmetic expressions" like this:
  - Integers are expressions.
  - If E and F are integer arithmetic expressions, so are $(E + F)$, $(E - F)$, $(E \times F)$, and (E/F).

  Examples?

  Notice that this allows us to generate only "sensible" expressions. Notice also that it's a bit more restrictive than we might like.
- We could write similar definitions for the wffs of propositional and predicate logic.

Slide 6

Recursive Definitions — Operations

- Example — factorial.
- Example — multiplication of non-negative integers, defined in terms of addition.
- Example — (integer) division of a non-negative integer by a positive integer, defined in terms of subtraction.

Slide 7

Recursive Algorithms

- Recursive definitions of sequences or operations often can be turned into recursive algorithms with little effort.
- Examples — function to compute n-th Fibonacci number, function to do division by repeated subtraction.
- Efficiency considerations:
  - In terms of computer time/memory usage, recursion is almost always worse than iteration — but not always, and sometimes not much worse.
  - In terms of human effort to get program running correctly, recursion may be much better.

Slide 8

Reasoning About Recursive Algorithms

- A recursive algorithm "works" if:
  - It works for the base case(s).
  - For other cases, it works *assuming* the recursive calls work.
  - The recursion eventually stops — recursive calls are always "smaller", and eventually reduce to base cases.
- We could formalize this as a proof by induction.

Slide 9

Minute Essay

- Consider the following recursive definition of a sequence:

$$\begin{aligned} S(1) &= 1 \\ S(n) &= 10S(n-1)+1, \quad \text{for } n > 1 \end{aligned}$$

What are $S(1), S(2), \ldots S(5)$?

Slide 10

Minute Essay Answer

- The first few terms:

$S(1) = 1$

$S(2) = 11$

$S(3) = 111$

$S(4) = 1111$

$S(5) = 11111$