Slide 1

Administrivia

- Reminder: Homework 4 due today.
- Homework 5 on Web, due Monday after spring break.
- Midterm rescheduled for Wednesday after spring break. This means we need a firm deadline for Homework 5 (so I can make a solution available). Monday at 5pm, or Tuesday at noon?
- How many people will still be in town Friday at class time?

Slide 2

Solving Recurrence Relations, Review

- Idea is to come up with "closed-form" (non-recursive) equivalent of recursive definition of sequence. Two approaches:
  - "Expand, guess, verify".
  - Formula (equation (8) on p. 134) — but works for first-order linear recurrence relations only.
- One more example — section 2.4 problem 80.

Slide 3

Analysis of Algorithms, Overview

- Often there's more than one way to solve a given problem, i.e., more than one algorithm. Which one is "best"? Depends on what "best" means. If we mean "fastest":
- A useful measure of approximate execution time is worst-case (or sometimes average-case) execution time expressed as a function of "problem size" (e.g., for operations on array, size of array) — "time complexity" of algorithm.

  (Another measure is "space complexity".)
- Customary to skip over housekeeping operations and count only "important stuff" — arithmetic operations, comparisons, etc.

  Also customary to "round off" the estimate to an "order of magnitude" — for a problem of size N, we say an algorithm is $O(f(N))$ if execution time is $f(N)$.

Slide 4

Analysis of Algorithms, Examples

- Example — computing a sum of N numbers. How many additions?
- Example — sequential search of array of size N. How many comparisons (worst case)?
- Example — binary search of sorted array of size N. How many comparisons (worst case)?

Slide 5

Analysis of Algorithms, Longer Example

- Look at several algorithms for computing a^b, for b a positive integer. First version:

```
double exp(double a, int b) {
    double temp = a;
    for (int i = 1; i < b; ++i)
        temp *= a;
    return temp;
}
```

  First, does this work? yes, and notice we could argue that it does using a loop invariant (what?).

- How many multiplications needed?

Slide 6

Analysis of Algorithms, Longer Example Continued

- We could also express this recursively:

```
double exp(double a, int b) {
    if (b == 1)
        return a;
    else
        return a * exp(a, b-1);
}
```

  Does this work? (Yes. Why?)

- How to figure out how many multiplications? Define and solve a recurrence relation.

Slide 7

Analysis of Algorithms, Longer Example Continued

- We could also express this recursively another way:

```
double exp(double a, int b) {
    if (b == 1)
        return a;
    else {
        double temp = exp(a, b/2);
        if (b % 2 == 0)     return temp * temp;
        else                return temp * temp * a;
    }
}
```

Does this work? (Yes. Why?)

- How to figure out how many multiplications? Define and solve a recurrence relation. (To be continued.)

Slide 8

Minute Essay

- Given a simpler recurrence relation:

$$
\begin{aligned}
P(1) &= 500 \\
P(n) &= P(n-1) \times 1.1, \quad \text{for } n > 1
\end{aligned}
$$

What is a closed-form solution? (Okay to guess.)

Minute Essay Answer

- $P(n) = 500 \times (1.1)^{(n-1)}$

Slide 9