Last Time
- Binary Search Trees (cont'd)
- Operations: print, lookup, min/max, succ/pred, insert, delete

Today
- HW5 due
- Last week in review
- Associative Arrays/Maps
- Binary Search Trees
- delete method (finish)
- Complexities
- Balanced Search Trees (intro)

Next Time
- Balanced Search Trees
- Red-Black Trees
(Last) Week in Review

● Trees (general)
● Terminology
● Implementing trees
● Traversals
● Binary Trees
● Types and heights
● Priority Queues
● Heaps
● Concept
● Insert/Delete
● Complexities
● Binary Search Trees (BSTs)
● Concept, Examples
● Implementation
● Operations: print, lookup, min/max, succ/pred, insert, delete
Associative Arrays/Key-Value Maps/Dictionaries

Concept:

Examples:

Operations:

Possible implementations:
Lowest Common Ancestor (LCA) of two nodes

In a BST:

In a (regular) binary tree:

Using only constant additional storage:
Deleting from a BST

High-level algorithm:

Example
BST operations complexities

Assume the number of nodes is n, arranged in a tree of height h.

print():

lookup(E):

min/max:
succ(E)/pred(E):

insert(E):

delete(E):
Height of an “average” BST (with $n$ nodes):

**Final BST example**

Starting with an empty BST, insert 7, 14, 18, 23, 1, 11, 20, 29, 25, 27.
Now delete 18 and 23:

**Balanced Search Trees**

Goal

How?

Rotations
Types