Last Time

- ArrayLists
  - Trying to get the best of both worlds between arrays and linked lists
    - $O(1)$ insertion and retrieval
    - $O(n)$ removal
  - Need to occasionally resize the array to make it bigger for more data or smaller to keep from wasting space
    - Doing this occasionally is called amortization
Trees

- A data type that stores nodes hierarchically.
Definitions

• A **tree** is a set of nodes storing elements such that the nodes have a **parent-child** relationship that satisfies the following:
  – If the tree is nonempty, it has a special node, called the **root**, that has no parent.
  – Each node different from the root has a unique **parent** node, and every node with a parent *w* is a **child** of *w*.

• Two nodes that are children of the same parent are **siblings**.

• A node is **external** (or a **leaf**) if it has no children, and **internal** if it does have children.
More Definitions

• A tree is *ordered* if there is a meaningful order among the children of each node (some node is first, some other node is second, etc).
Even More Definitions

• An **edge** of a tree is a pair of nodes \((u, v)\) such that \(u\) is
  the parent of \(v\), or \(v\) is the parent of \(u\).

• A **path** is a sequence of nodes such that any two
  consecutive nodes in the sequence form an edge.

• A **subtree** rooted at node \(v\) is the tree consisting of all
  the **descendants** of \(v\) in the tree, including \(v\) itself.

• The **height** of a tree is the maximum distance from the
  root to a leaf node. The distance from a node to the
  root is the **depth** of that node.

• All nodes at the same depth are said to be on the same
  **level** of the tree.
More and More Definitions

• A **binary tree** is an ordered tree with the following properties:
  – Every node has at most two children.
  – Each child node is labeled as being either a **left child** or a **right child**.
  – A left child precedes a right child in the sibling order.

• A binary tree is called **proper** or **full** if each node has either zero or two children. A binary tree is **improper** if it is not proper.

• The left child is the root of the **left subtree**; the right child is the root of the **right subtree**.
Why Bother with Trees?
Any Questions?