Today

• Binary search trees (cont)

Assignments

• HW8 out (due Thurs)
• HW9 out today ...
• Exam 2 next week
In general, a tree forms a **heirarchy** with

- zero or more “**nodes**”
- a distinguished “**root**” node (no parents)

Nodes are arranged in “parent-child” relationships

- each node has zero or more “**children**” (child nodes)
- each node has at most one “**parent**” (parent node)
- a node *without* children is a “**leaf**” node
- a node *with* children is an “**internal**” node

A collection of trees is called a “**forest**”
A “**path**” is a sequence of nodes $n_i n_j \ldots n_k$

- the next node after $n$ in the sequence is a child of $n$
- a root–to–leaf path starts at the root and traverses children ending at a leaf
- there can be many paths within a tree

![Diagram of a tree with a path highlighted](attachment:image.png)
Parent-child relationships induce “ancestor-descendant” relationships

- The **ancestor** of a node is its parent and its parent’s ancestors
- The **descendent** of a node is its children and its children’s descendents
- The descendents of a node \( n \) lie on paths from \( n \) to leaves
- The ancestors of a node \( n \) lie on paths from the root to \( n \)

Each node can also have zero or more “siblings”

- Two nodes are siblings if they have the same parent

![Diagram showing ancestor-descendant relationships and siblings]
A “subtree” is a tree rooted at a descendent of the root node

- a subtree includes all descendents of the subtree “root” \( n \)
- we often say a subtree is rooted at \( n \)

An “empty” tree has no nodes

The “height” of a tree is the length of the longest root-to-leaf path

- the height of the above tree is 3
- the height of the subtree rooted at \( b \) is 2
- the height of the subtree rooted at \( g \) is 1
Binary Trees

In a binary tree, each node has at most 2 children.

Thus, $T$ is a binary tree if

- $T$ has no nodes (is empty); or
- $T$ has the form:

  $r$
  $\ \ \ \ \ \ \\
  T_l \quad T_r$

  - where $r$ is the root node of $T$ and $T_l$ and $T_r$ are binary trees
  - $T_l$ is the left subtree of $r$ and $T_r$ is the right subtree of $r$

Note that this definition is recursive!

- this will be helpful as we compute over trees

We can recursively define the notion of a binary tree’s height ...

- the height of a binary tree $T$ is:
  - 0 if $T$ is empty
  - 1 + max of the height of the left and right subtrees of $T$'s root
In a “**full**” binary tree

- each root-to-leaf path has the same height \( h \); and
- all internal nodes have two children

In a “**complete**” binary tree of height \( h \)

- the tree is full at height \( h - 1 \), and
- the leaf nodes are filled in from left to right
A **balanced** binary tree has for every node

- left and right subtrees that differ in height by at most 1
Binary Search Trees

A binary search tree (BST) adds additional constraints to a binary tree

- specifically, nodes are stored in sort order (by key)

The order constraint: For each node $n$ in a BST:

- if node $n_l$ is in the left subtree of $n$ then $\text{key}(n_l) \leq \text{key}(n)$
- if node $n_r$ is in the right subtree of $n$ then $\text{key}(n_r) > \text{key}(n)$