More trees

Unlike linked list, which are linear, trees are not. We can’t just loop down, from head to tail to visit every node.

We need to move forward and then go back to do the other possibility.

So, we will need to use (either explicitly or implicitly) a stack. Recursion naturally uses a stack

Let’s write a recursive algorithm to print each node in the tree. We’ll use pseudo code for now.

In contrast to preorder, we can do postorder, which prints the node after its children

Preorder = top down, postorder = bottom up

Postorder is used for deallocating all nodes in the tree.

Height

The height of a tree is defined as the length (measured in nodes) of the longest path from the root to a leaf.

Single node: height = 1

Empty tree: height = 0

Binary tree

A binary tree is a tree in which each node has at most two children.

Let’s compute bounds on the height of a binary tree. Given a binary tree of height h, what are the minimum and maximum number of nodes it can contain?

MIN: one node per level, so for height h, we have h nodes h

MAX: every level is full 2^h -1 or O(2^h)

O(h) <= n <= O(2^h)

Given a binary tree with n nodes, what are its minimum and maximum heights?

MAX: one per level O(n)

MIN: every level is full O(logn)

O(logn) <= h <= O(n)