



Overview of Trees

A quick overview of trees, its types, and some important formulas to compute height and number of nodes in a tree.

We'll cover the following



- Binary Trees
- Binary Search Trees
- Red Black Trees
- AVL Trees
- 2-3 Trees

Binary Trees#

Definition: A tree where each vertex has two children at most.

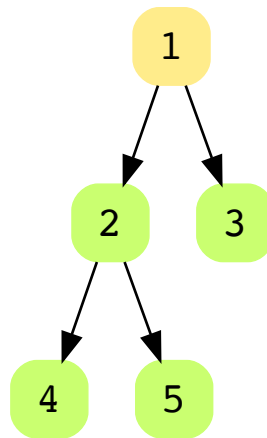
Types: *Perfect, Full, Complete, Skewed*

Total number of nodes: $2^{(h+1)} - 1$

Total number of leaf nodes: 2^h or $\frac{(n+1)}{2}$

Height: $\log_2(n + 1) - 1$

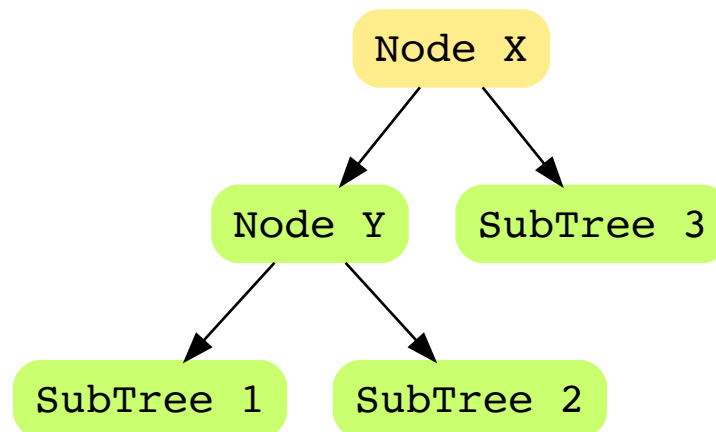




Binary Search Trees#

Definition: Every node has a value greater than/equal to all the node values in its left sub-tree and has a value less than all the node values in its right sub-tree. Mathematically,

$$Keys(SubTree1) < Keys(Y) < Keys(Subtree2) < Keys(X) < Keys(SubTree3)$$



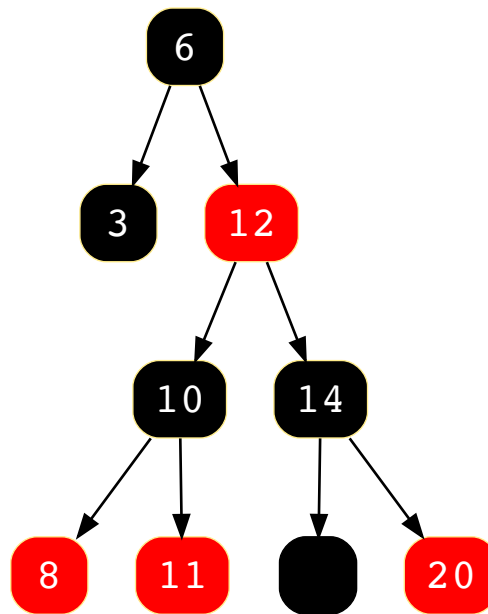
Red Black Trees#

Definition: A tree where every node is colored as red or black and no two adjacent nodes have *red* color and root and null nodes are considered *black*



Height: $h \leq 2\log_2(n + 1)$

Minimum number of nodes: $(h + 1) + 2 \left(\sum_{i=0}^{h-1} 2^i - 1 \right)$, where h is the height of the tree, to: $\text{floor}(h/2)$



AVL Trees#

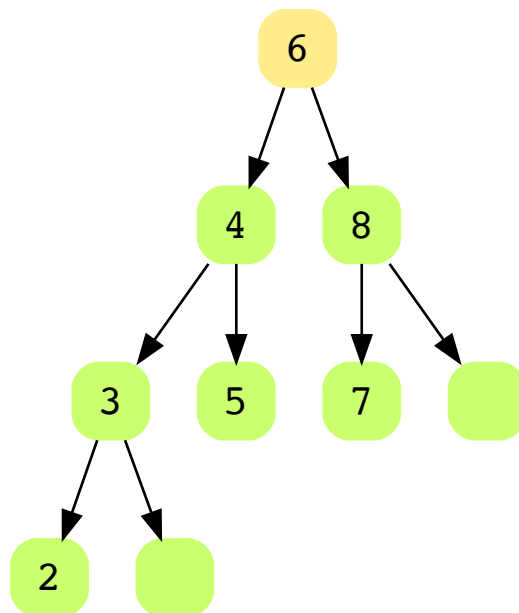
Definition: For each node, height of left and right *subtree* differ by at max one

Minimum number of nodes: $N(h) = 1 + N(h - 1) + N(h - 2)$

Maximum number of nodes: $N - 1 + 2^{\log(N-1)+2}$

Height: $O(\log_2 n)$





2-3 Trees#

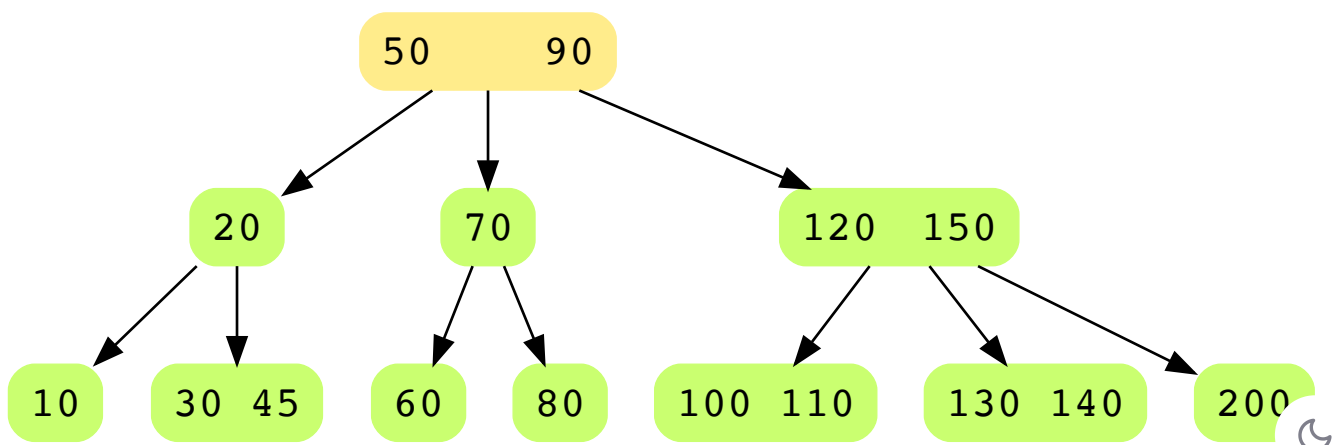
Definition: A balanced and ordered tree where each node can have two keys (X and Y) and three children at max such that:

$LChild.Key < X < MChild.Key < Y < RChild.Key$

Maximum number of nodes: 3^h

Height: $\log_4(n + 1) - 1 < h < \log_2(n + 1) - 1$

Types: 2-3-4 Trees



Interviewing soon? We've partnered with Hired so that companies apply to you instead of you applying to them. [See how](#) ⓘ

[← Back](#)[Next →](#)

2-3-4 Trees

Challenge 1: Find minimum value in Bi...

[Mark as Completed](#)[Report an Issue](#)