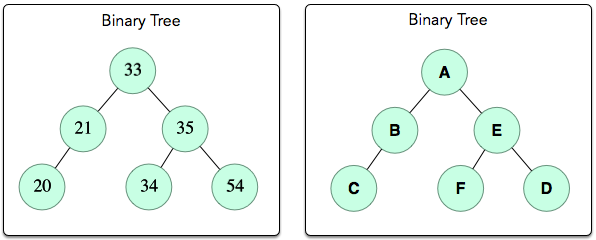
Fundamentally, a binary tree is composed of nodes connected by edges (with further restrictions discussed below). Some binary tree, t, is either empty or consists of a single root element with two distinct binary tree child elements known as the left subtree and the right subtree of t. As the name binary suggests, a node in a binary tree has a maximum of 2 children.

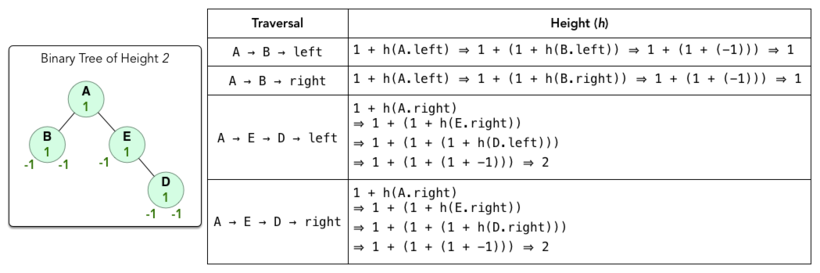
The following diagrams depict two different binary trees:



Here are the basic facts and terms to know about binary trees:

1. The convention for binary tree diagrams is that the root is at the top, and the subtrees branch down from it.
2. A node's left and right subtrees are referred to as children, and that node can be referred to as the parent of those subtrees.
3. A non-root node with no children is called a leaf.
4. Some node a is an ancestor of some node b if b is located in a left or right subtree whose root node is a. This means that the root node of binary tree t is the ancestor of all other nodes in the tree.
5. If some node a is an ancestor of some node b, then the path from a to b is the sequence of nodes starting with a, moving down the ancestral chain of children, and ending with b.
6. The depth (or level) of some node a is its distance (i.e., number of edges) from the tree's root node.
7. Simply put, the height of a tree is the number of edges between the root node and its furthest leaf. More technically put, it's (i.e., one more than the maximum of the heights of its left and right subtrees). Any node has a height of 1, and the height of an empty subtree is -1. Because the height of each node is 1+ the maximum height of its subtrees and an empty subtree's height is -1, the height of a single-element tree or leaf node is 0.

Let's apply some of the terms we learned above to the binary tree on the right:

1. The root node is A.
2. The respective left and right children of A are B and E. The left child B of is C. The respective left and right children of E are F and D.
3. Nodes C, F, and D are leaves (i.e., each node is a leaf).
4. The root is the ancestor of all other nodes, B is an ancestor of C, and E is an ancestor of F and D.
5. The path between A and C is A->B->C. The path between A and F is A->E->F. The path between A and D is A->E->D.
6. The depth of root node A is 0. The depth of nodes B and E is 1. The depth of nodes C, F, and D, is 2.
7. The height of the tree, height(t), is 2. We calculate this recursively as . Because this is long and complicated when expanded, we'll break it down using an image of a slightly simpler version of t whose height is still 2:   In the diagram above, the height of t is 2 because that is the maximum height of t's left and right subtrees.

Binary Search Tree

A Binary Search Tree (BST), t , is a binary tree that is either empty or satisfies the following three conditions:

1. Each element in the left subtree of t is less than or equal to the root element of (i.e., max(leftTree(t).value <= t.value).
2. Each element in the right subtree of is greater than the root element of (i.e., min(rightTree(t).value > t.value).
3. Both leftTree(t) and rightTree(t) are BSTs.

You can essentially think of it as a regular binary tree where for each node parent having a leftChild and rightChild , leftChild.value <= parent.value <= rightChild.value In the diagram above, the binary tree of integers on the left side is a binary search tree.