What is a Binary Search Tree (BST)?

This lesson will introduce Binary Search Trees and their properties

We'll cover the following

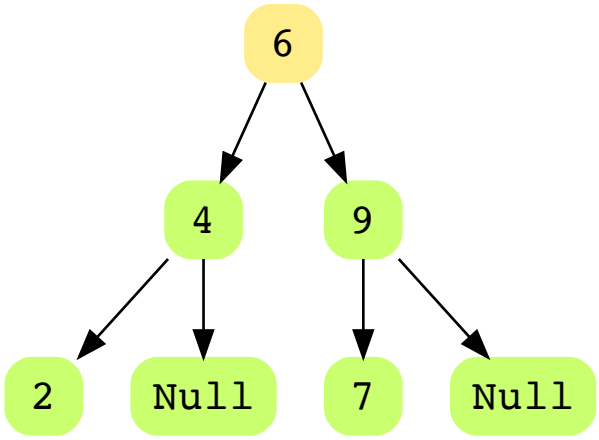
Introduction#

Binary Search Trees (BSTs) are a special kind of binary tree where each node of the tree has key-value pairs. These key-value pairs can be anything, like $(username, bank)$ or $(employee, employeeID)$. For all the nodes in a BST, the values of all the keys in the left sub-tree of the node are less than the value of the nodes themselves. All the keys in the right subtree are greater than the values of the node. This is referred to as the BST rule.

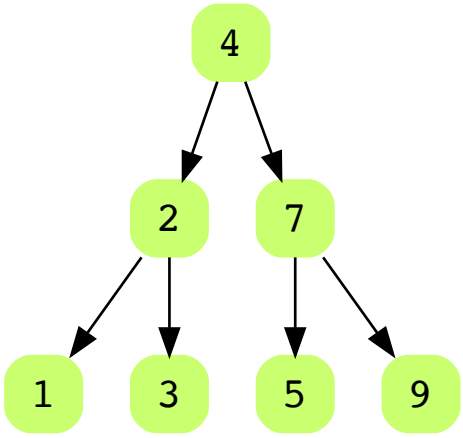
$$NodeValues(leftsubtree) <= CurrentNodeValue < NodeValues(rightsubtree)$$

Examples#

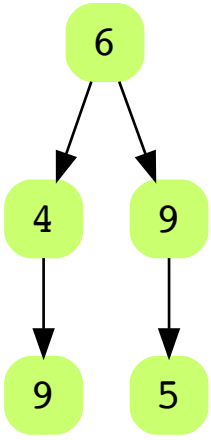
Let's see a few examples of BSTs



Valid BST



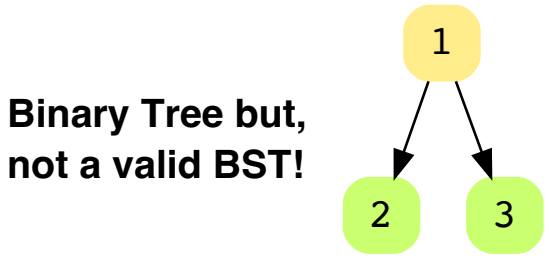
Valid BST



Invalid BST because 5 is less than 6 and is in 6's right sub-tree

Binary Search Trees vs. Binary Tree

Not every binary tree is a Binary Search Tree. As a binary search tree is a type of binary tree, so some people assume that every binary tree might be binary search tree. But that's not the case because for a binary tree to qualify as a binary search tree, it needs to follow the BST property. Let's see an example to understand why every binary tree is not also a binary search tree?



1 of 2

Explanation

In the figure above the first one is a Binary Tree because each node has maximum of two children. But it is not a Binary *Search* Tree, because it does not follow the BST rule according to which Node **2** cannot be a leftChild of Node **1** since it's greater than **1**. So to convert it into a binary search tree we need to make Node **1** left child of Node **2**.

Now that we have studied Binary Search Trees and how they are different from regular Binary Trees, let's implement basic tree traversals in the next few lessons.

companies apply to you instead of you applying to them. how

Back

Skewed Binary Trees

Next

Implementing a Binary Search Tree in ...

Completed

Report an Issue