Deletion in a Binary Search Tree

In this lesson, we are going to learn how nodes are deleted in binary search trees. We will take a look at a few node deletion scenarios and what to do in each one.

We'll cover the following

Introduction

In this lesson, we are going to study how a node is deleted in a BST. In general, to delete a node in a BST, you will search for it and, once found, you'll make it `None` by making the left or right child of its parent `None`. However, to make things simpler, we've identified six possible cases involved in BST node deletion. We'll tackle each one separately.

1. Deleting in an empty tree
2. Deleting a node with no children, i.e., a leaf node.
3. Deleting a node which has one child only
   I. Deleting a node which has a right child only
   II. Deleting a node which has a left child only
4. Deleting a node with two children

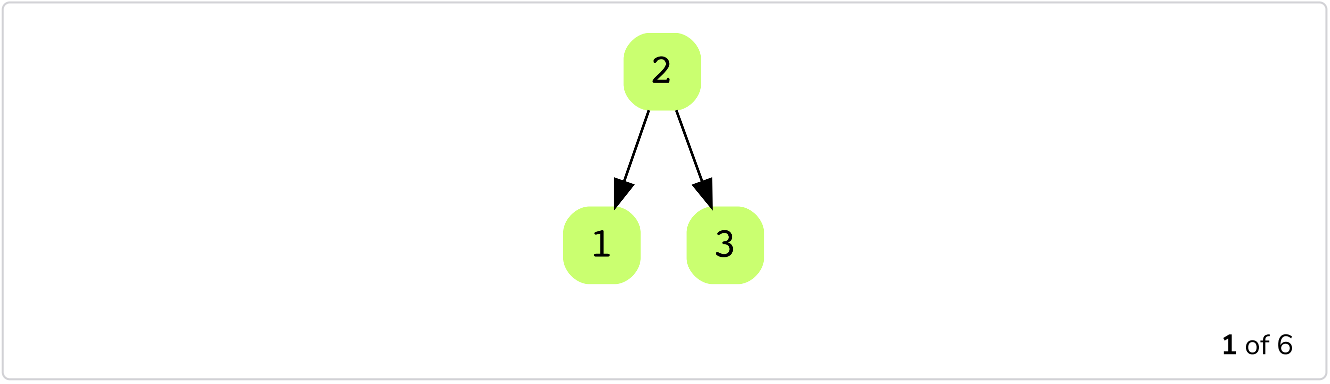
Let’s look at each of these in detail,

1. Deleting an empty tree

If the given starting node is `Null` then do nothing and return `False`. This is an edge case for error handling.

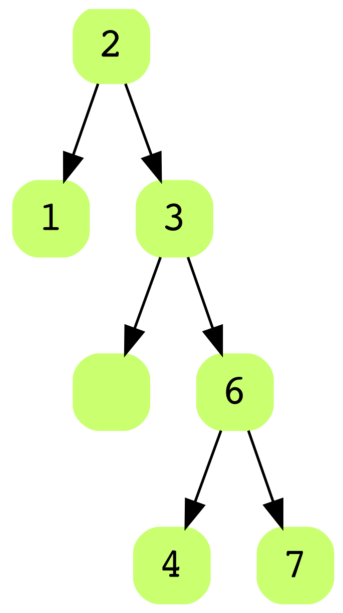
2. Deleting a Leaf Node

When the node to be deleted is a leaf node in a Binary Search Tree, we simply remove that leaf node. We do this by making the parent node’s left or right child (whichever one the leaf node was) `None`. Have a look at the following example for a demonstration



3. Deleting a node which has one child

We search for the node, once the node is found we check if and how many children it has. If it has only one child, we check the parent node to see if the current node is the left or right child and then replace its child node with the current node. It will be easier to understand visually so take a look at the following illustration



1 of 7

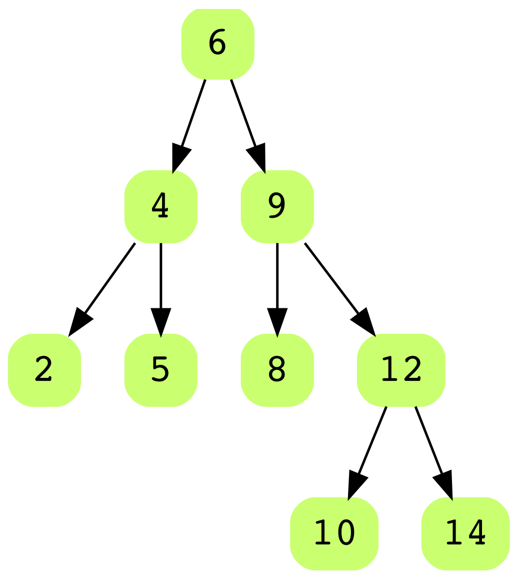
4. Deleting a node with two children

1. From the given node to be deleted, find either the node with the smallest value in the right sub-tree or the node with the largest value in the left sub-tree. Suppose you want to find the smallest value in the right sub-tree; you do this by moving on to every node's left child until the last left child is reached.

2. Replace the node to be deleted with the node found (the smallest node in the right sub-tree or the largest node in the left sub-tree).

3. Finally, delete the node found (the smallest in the right sub-tree).

Take a look at the following animation for a visual of this algorithm!



1 of 10

In this lesson, we covered the cases for deleting nodes in Binary Search Trees. Let's dive into coding the delete function in Python in the next lesson!

Back
Searching in a Binary Search Tree (Im...

Next
Deletion in a Binary Search Tree (Impl...

Completed

Report an Iss