Trees – 3

Deletion in BST

- leaf node
- node with one child
- node with two children

Deleting a node from a Binary Search Tree

Deletion of a node is not so straightforward as is the case of insertion. It would depend on which particular node is being deleted.

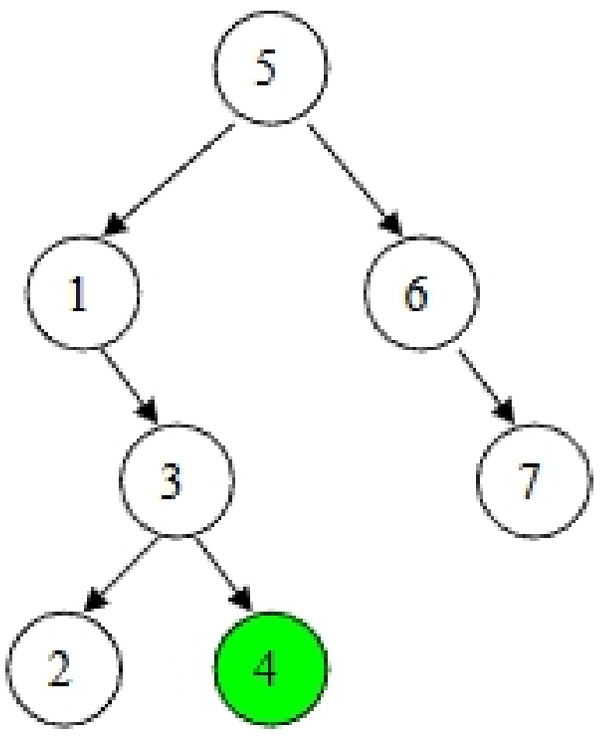
In fact, we note that there can be three separate cases and each case needs to be handled somewhat differently. The various cases are:

(1) deletion of a leaf node,
(2) deletion of an internal node with a single child *(either a left or right subtree),*
(3) deletion of an internal node with two children *(having both left subtree and right subtree.)*

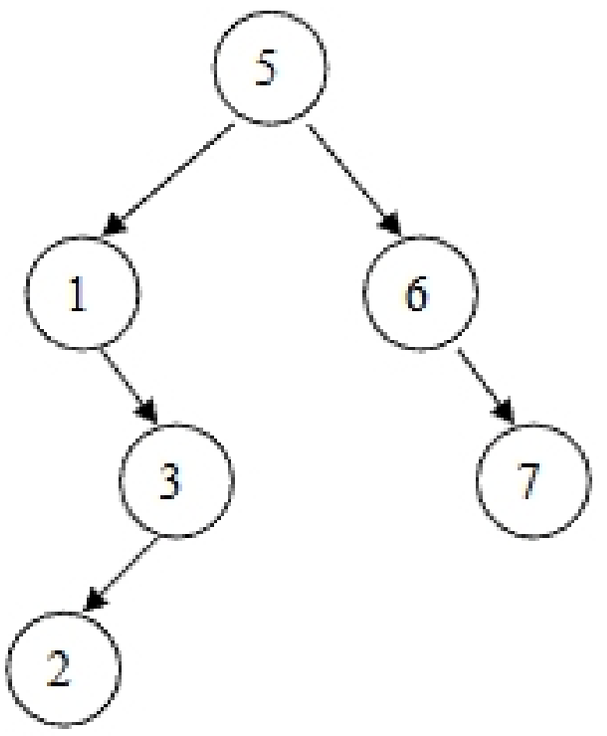
We'll examine each case separately:

Deletion of a leaf Node

Since a leaf node has empty left and right subtrees, deleting a leaf node will render a tree with one less node but which remains a BST. This is illustrated below:



A BST with a leaf node
Marked for deletion.

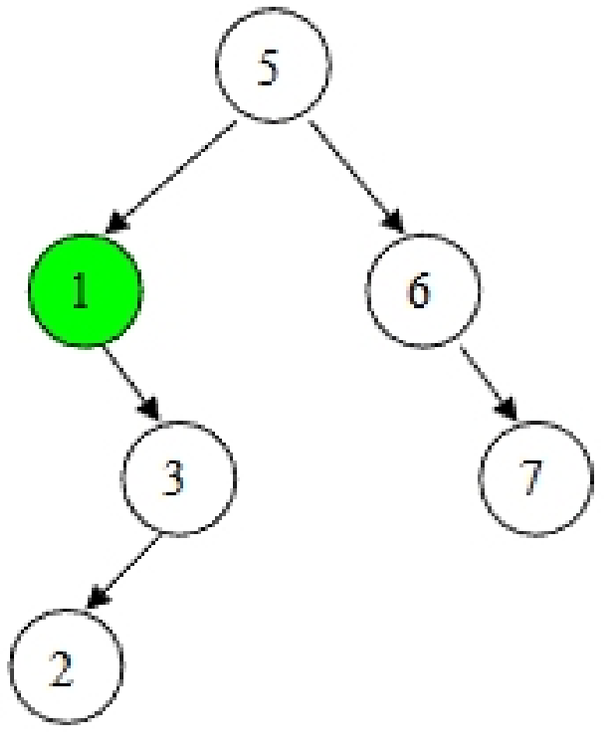


Still a BST

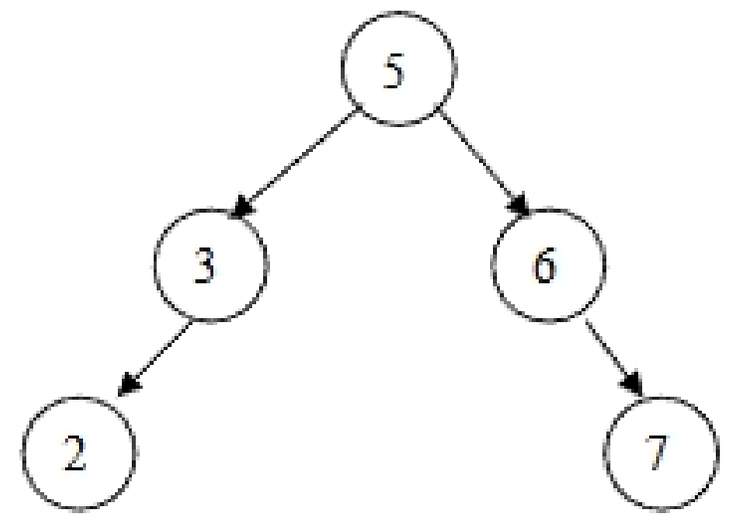
Deletion of a Node with one child

In this case, when the node gets deleted, the parent of the node must point to its left child or its right child, as the case may be.
The parent's reference to the node is reset to refer to the deleted node's child. This has the effect of lifting up the deleted node's children by one level in the tree.
An example is shown below.

A BST with an internal node having only one child marked to be deleted



The marked internal node has only a right subtree so the parent of the deleted node will now reference the deleted node's child



Note that it makes no difference if the node to be deleted has only a left or a right child. The previous example illustrated the case when the only child was a right child. The next example illustrates the case when the only child is a left child.