Deletion in BST

∴ There are 3 cases in deletion of a node in BST.

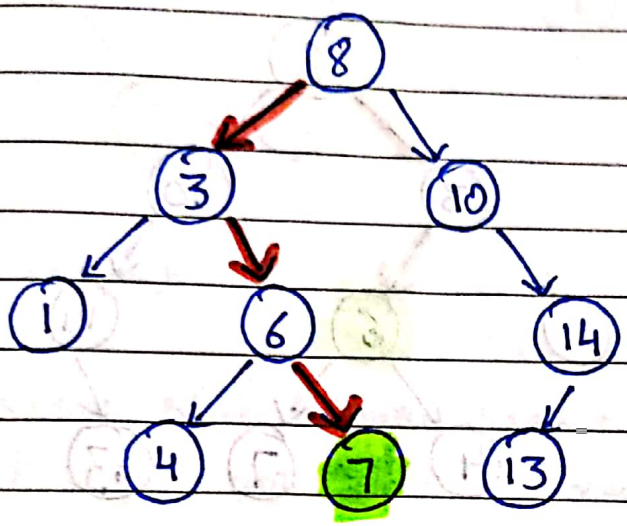
1) The node is a leaf node.
2) The node is a non-leaf node.
3) The node is the root node.

(1) DELETING A LEAF NODE:

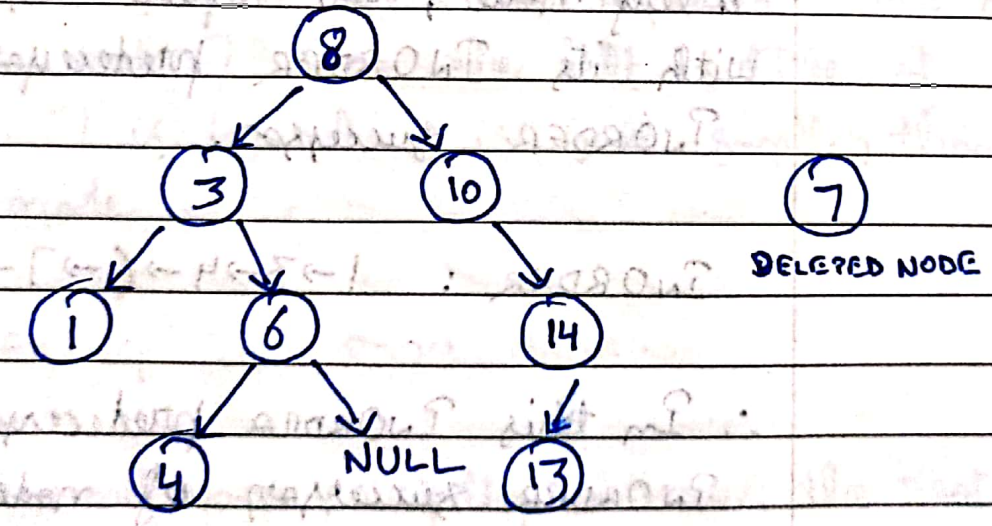
→ Deleting a leaf node is the simplest case in deletion in binary search trees where the only thing you have to do is to search the element in the tree,

and remove it from the tree, and make its parent node point to NULL.

1. Search the node:



2. Delete the node:



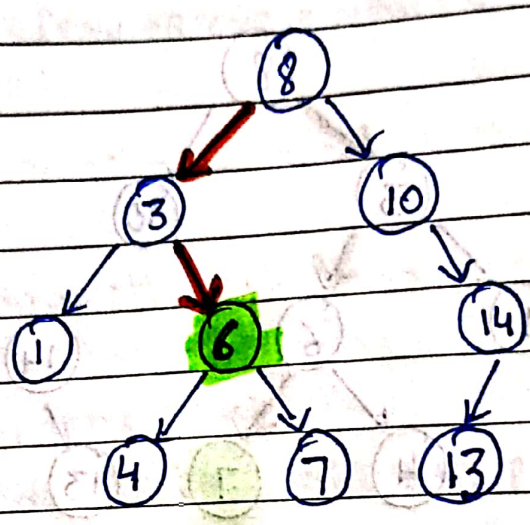
② DELETING A NON-LEAF NODE:

Now suppose the node is not a leaf node, so you cannot just make its parent point to NULL, and get away with it. You have to deal with the children of this node. Let's try deleting node 6 in the

Page No.
Date

BST.

① So, the first thing you would do is to search element 6.

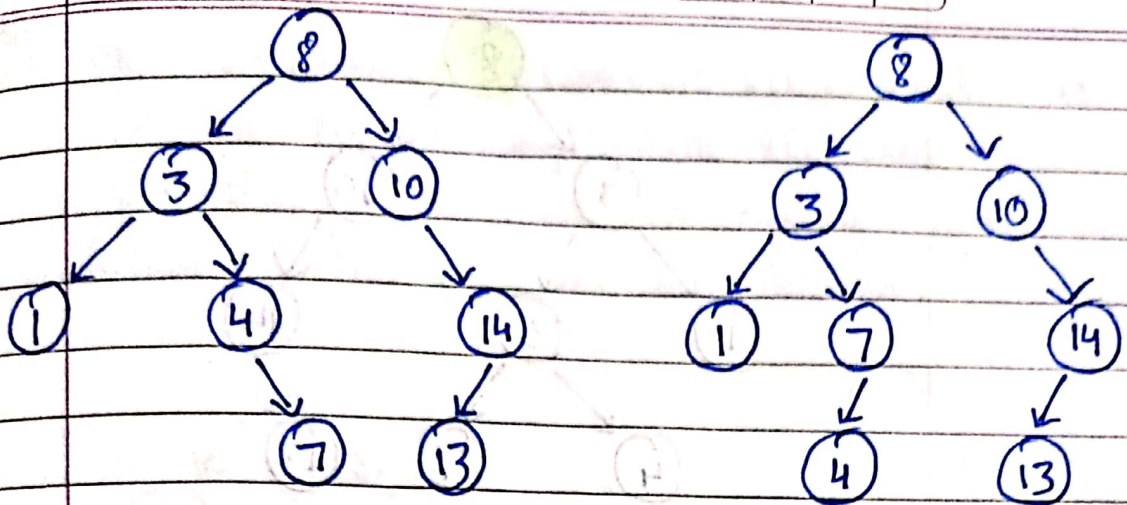


② When you delete a node that is not a leaf node, you replace its position with its INORDER predecessor or INORDER successor.

INORDER : 1→3→4→6→7→8→10→13→14

∴ In this INORDER predecessor and the INORDER successor of node 6 are 4 and 7 respectively. Hence, you can substitute node 6 with any of these nodes, and the tree will still be a valid Binary search tree.

Refer to how it looks below:

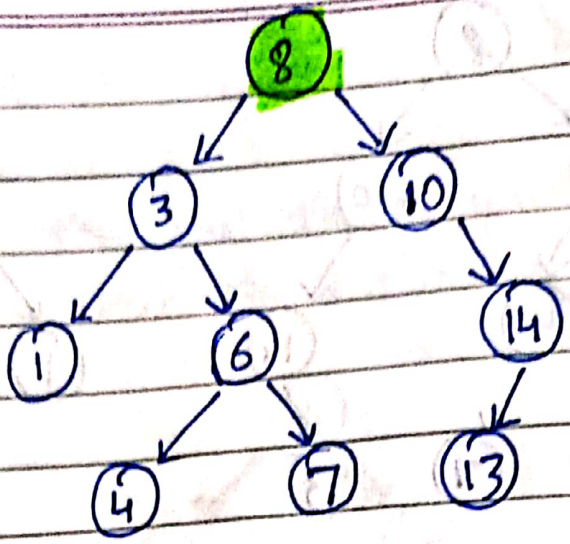


(1) (2)

∴ Both are still binary search tree. In the first case, we replaced node 6 with node 4. And the right subtree of node 4 is 7, which is still bigger than it. And in the second case, we replaced node 6 with node 7. And the left subtree of node 7 is 4, which is still smaller than the node.

③ DELETING THE ROOT NODE:

1) Now, if you carefully observe, the root node is still another non-leaf node. So, the basics to delete the root node remains the same as what we did for a general non-leaf node. But since the root node holds a big size of subtree along with, we have separate case for it.

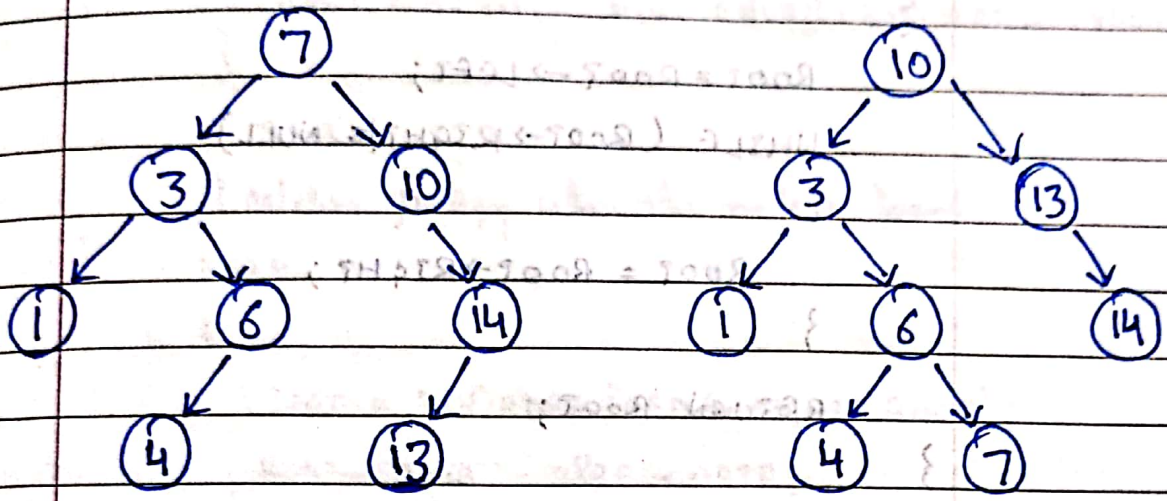


2) So, the first thing you do is write the INORDER traversal of the whole tree. And then replace the position of the root node with its INORDER predecessor or INORDER successor.

1 → 3 → 4 → 6 → 7 → 8 → 10 → 13 → 14

∴ the INORDER predecessor and the INORDER successor of the root node 8 are 7 and 10 respectively. Hence you can substitute node 8 with any of these nodes, but there is a catch here. So, if you substitute the root node here, with its INORDER predecessor 7, the tree will still be a binary search tree, but when you substitute the root node here, with its INORDER successor 10, there still becomes an empty position where node 10 used to be. So, we still placed

the INORDER successor of 10, which was 13 on the position where 10 used to be. And then there are no empty nodes in between. The finalizes our deletion.



(1) (2)

∴ There are a few steps:

1) First, search for the node to be deleted.
2) Search for the INORDER predecessor and successor of the node.
3) Keep doing that until the tree has no empty nodes.

And this case is not limited to the root nodes, rather any nodes falling in b/w a tree. Well, there could be a case where the node was not found in the tree, so, for that, we would revert the statement that the node could not be found.