Delete Node in a BST

450. Delete Node in a BST Solved

Medium · Topics · Companies

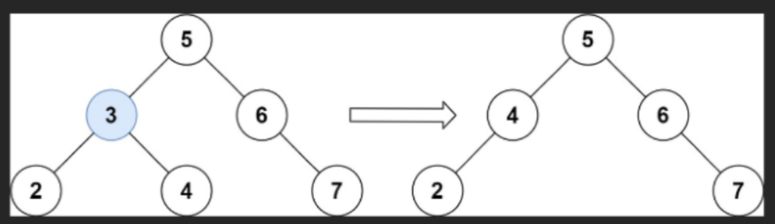
Given a root node reference of a BST and a key, delete the node with the given key in the BST. Return *the **root node reference** (possibly updated) of the BST.*

Basically, the deletion can be divided into two stages:

1. Search for a node to remove.

2. If the node is found, delete the node.

Example 1:




```
Input: root = [5,3,6,2,4,null,7], key = 3
Output: [5,4,6,2,null,null,7]
Explanation: Given key to delete is 3. So we find the node with
value 3 and delete it.
One valid answer is [5,4,6,2,null,null,7], shown in the above BST.
Please notice that another valid answer is [5,2,6,null,4,null,7]
and it's also accepted.
```

* Deleting a node is somewhat trickier because, simply removing node won't be enough

↓

We have to Conserve the BST property after deletion

BST property :

left Subtree Value's < root value < right Subtree values

how to delete a node ?

* first you have to find the node which need to delete ...

if you dont find the node that to be deleted

↓

return the input root...

nothing to do

if you found the node that to be deleted

↓

Solve the problem...

You can't do anything you are in trap

So, first the approach to find the node that to be deleted...

① Start from the root

② If key is less than the current node's value → move to left child

③ else move right (until, you find equality (or) you reach null)

Now, the main Problem

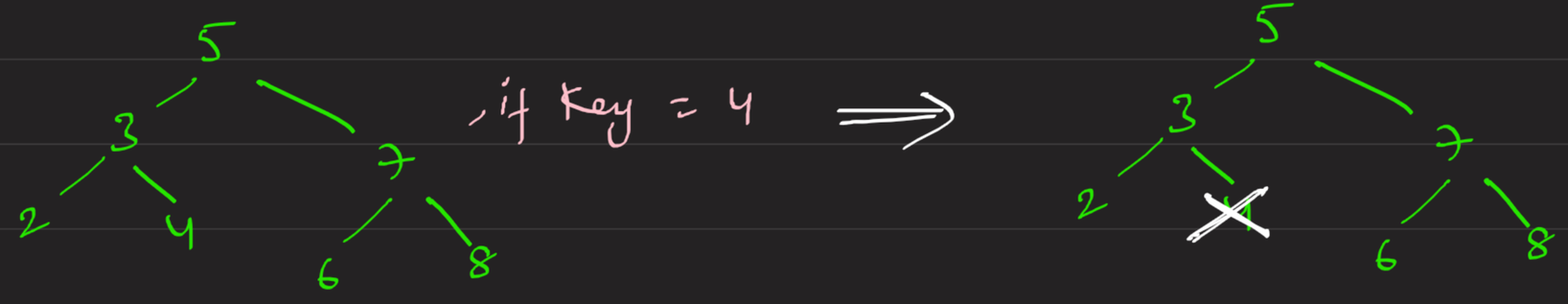
How to delete :

* Handling different cases for deletion...

Case ① : No children {leaf node}

↳ Simply make the node null

* Set the parent reference link to null



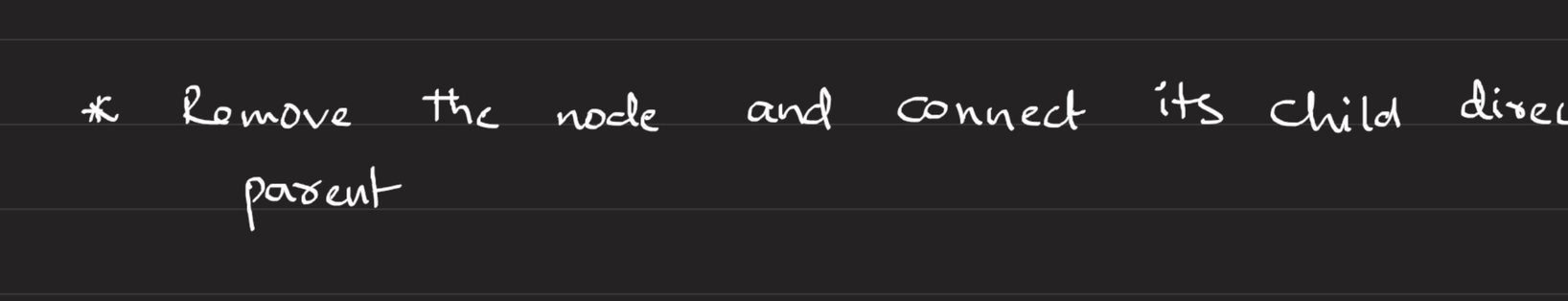
make node 3's right pointer to null

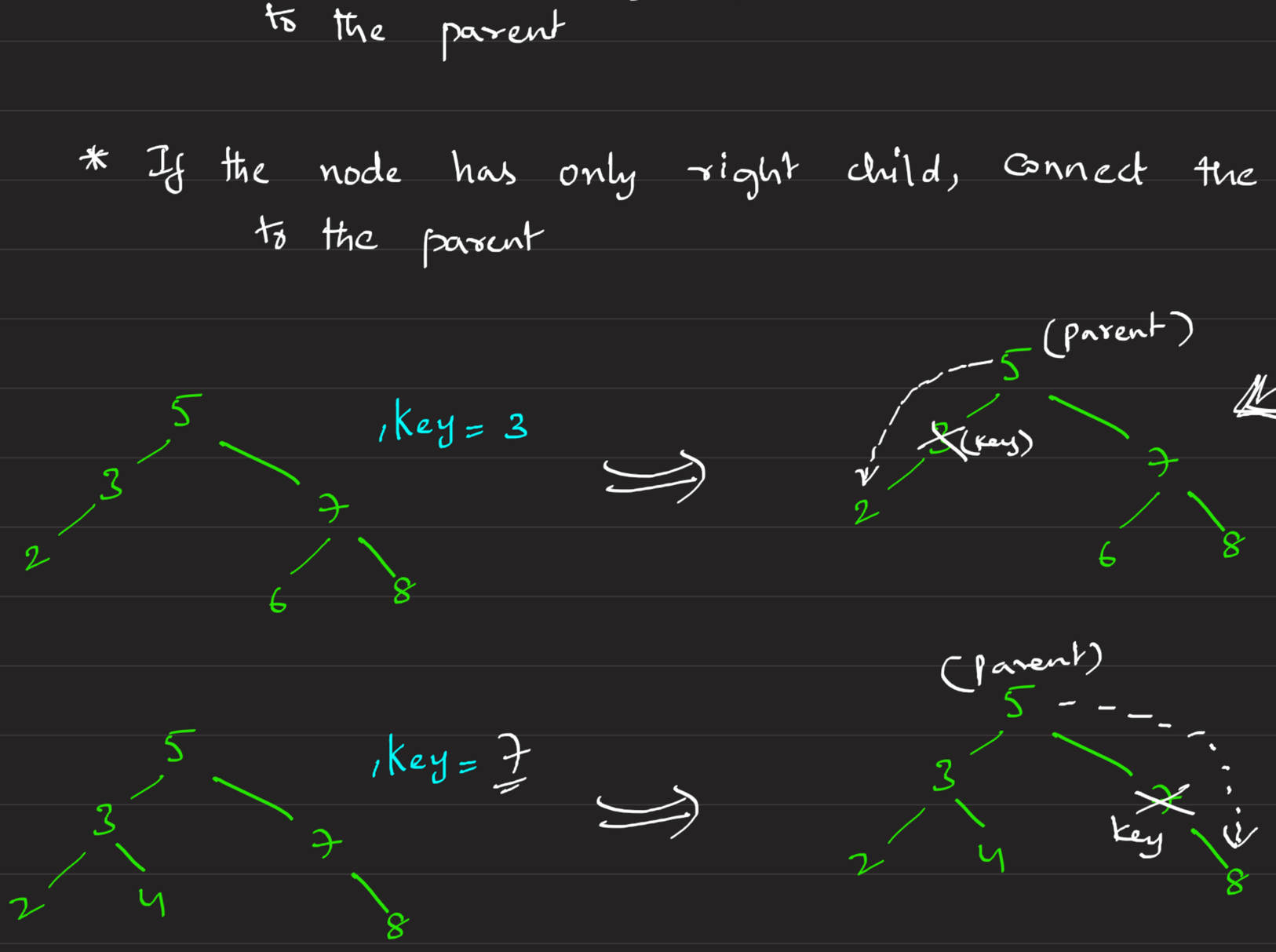
Case ② : 1 children

* Remove the node and connect its child directly to its parent

* If the node has only left child, connect the left child to the parent

* If the node has only right child, connect the right child to the parent



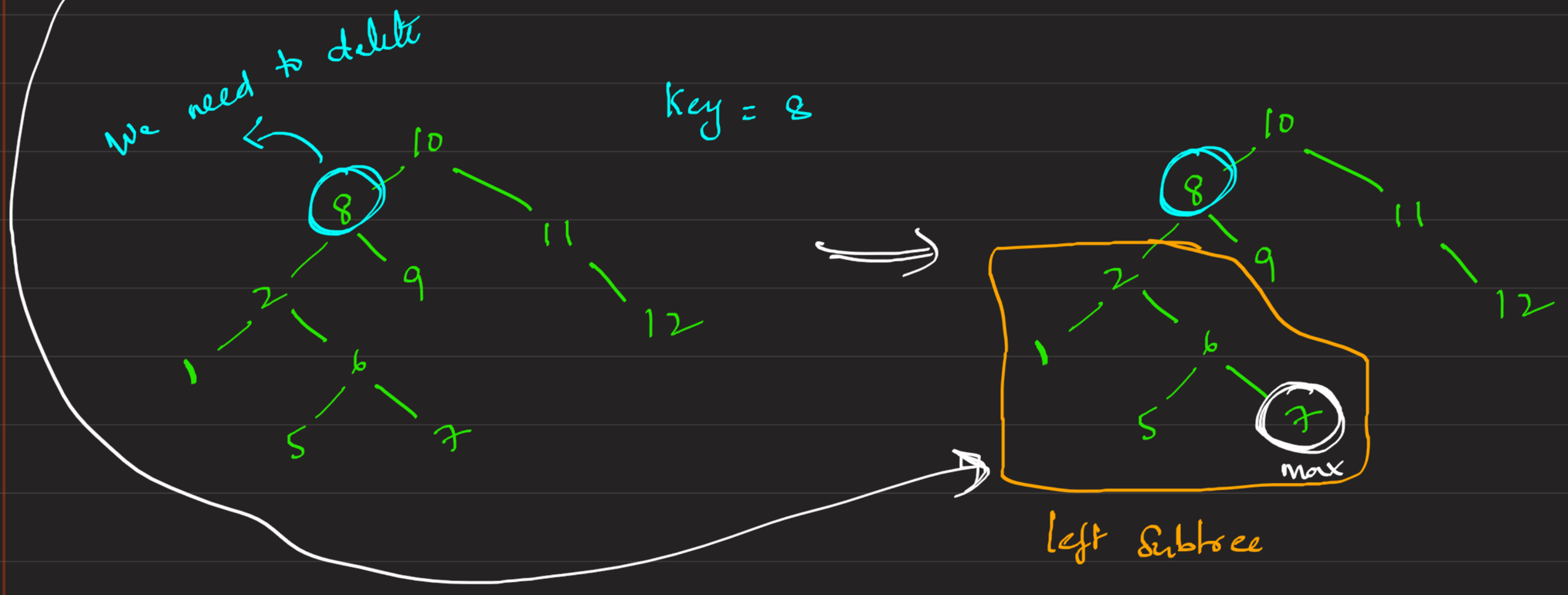


* So, key Understanding Uphill now is, we need to Stay One node above the key node to manage links

Case (3) : Node with two Children

* find the inorder predecessor (Largest node in left Subtree)

* and Set it as it points to key node's right node



And make node (7) points to keynode's left node

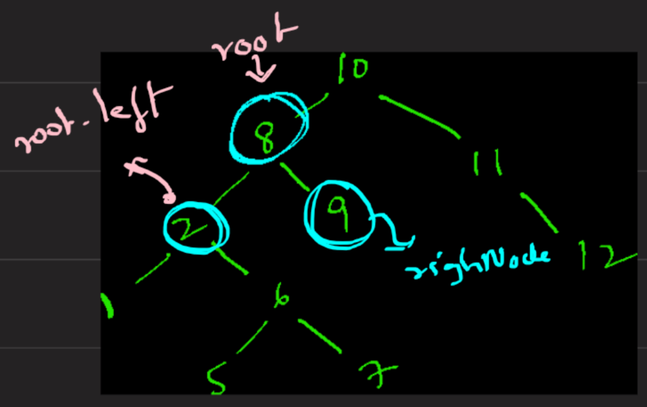
* So remember, if you want to delete a node, you need to change the link of its parent, so keep tracking parent is important

```java
public TreeNode help(TreeNode root){
    if(root.left == null && root.right == null)return null;
    if(root.left == null){
        return root.right;
    }
    else if(root.right == null){
        return root.left;
    }
    else{
        TreeNode maxLeftNode = find(root.left);
        TreeNode rightNode = root.right;
        maxLeftNode.right = rightNode;

        return root.left;

    }
```

→ Case (1)

→ Case (2)

→ Case (3)



Link maxNode of left subtree to rightNode

```java
public TreeNode find(TreeNode root){
    while(root.right!=null){
        root = root.right;
    }
    return root;
}
```

This will calculate max value in left subtree

return root.left

→ it returns node (7)

Because .

```
temp.right = help(temp.right);
```

if root.right is the node we need to delete

↓

the (root.right) link need to be modified

```
temp.left = help(temp.left);
```

if root.left is the node we need to delete

↓

the (root.left) link need to be modified

So, return the modified new head