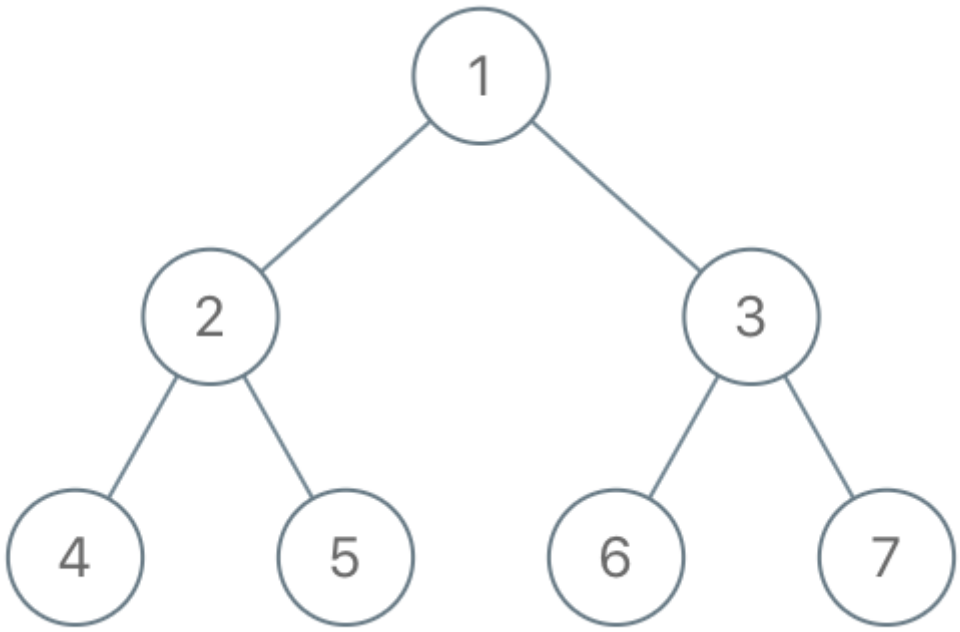
1110. Delete Nodes And Return Forest

Given the `root` of a binary tree, each node in the tree has a distinct value.

After deleting all nodes with a value in `to_delete`, we are left with a forest (a disjoint union of trees).

Return the roots of the trees in the remaining forest. You may return the result in any order.



Input: root = [1,2,3,4,5,6,7], to_delete = [3,5]
Output: [[1,2,null,4],[6],[7]]

```
def delNodes(self, root: TreeNode, to_delete: List[int]) ->
List[TreeNode]:
        to_delete = set(to_delete)
        res = []
        if root.val not in to_delete:
            root = self.dfs(root,to_delete,res)
            res = res + [root]
            return res
        else:
            root1 = self.dfs(root.left,to_delete,res)
            root2 = self.dfs(root.right,to_delete,res)
            root.left = None
            root.right= None
            if root1:
```

```
                res = res+[root1]
            if root2:
                res = res+[root2]

            return res

    def dfs(self,root,to_delete,res):
        if root is None:
            return None
        root.left = self.dfs(root.left,to_delete,res)
        root.right = self.dfs(root.right,to_delete,res)
        if root.val in to_delete:
            if root.left!=None:
                res.append(root.left)
            if root.right!=None:
                res.append(root.right)
            root.left,root.right = None,None
            return None
        return root
```