1110. Delete Nodes And Return Forest

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
 *     TreeNode *right;
 *     TreeNode() : val(0), left(nullptr), right(nullptr) {}
 *     TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}
 *     TreeNode(int x, TreeNode *left, TreeNode *right) : val(x), left(left), right(right) {}
 * };
 */

/*
    Time complexity: O(n)
    Extra space complexity: O(2n)

    [1,2,3,4,5,6,7,8,9,null,null,10,11,null,null,null,12,13]
*/
class Solution {
   public:
        std::vector<TreeNode*> ans;
        bool is_to_delete[1001]={false};

    public:
        void dfs(TreeNode*& root,bool is_root){
            if(!root) return;

            dfs(root->left,is_to_delete[root->val]);
            dfs(root->right,is_to_delete[root->val]);

            if(!is_to_delete[root->val] && is_root) ans.push_back(root);
            if(is_to_delete[root->val]) root=nullptr;

        }

        std::vector<TreeNode*> delNodes(TreeNode* root, std::vector<int>& to_delete) {
            for(auto& to_del_val: to_delete) is_to_delete[to_del_val]=true;
            dfs(root,true);
            return ans;
        }
};
```