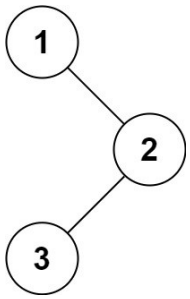
94. Binary Tree Inorder Traversal

Given the `root` of a binary tree, return *the inorder traversal of its nodes' values*.

Example 1:




```
Input: root = [1,null,2,3]
Output: [1,3,2]
```

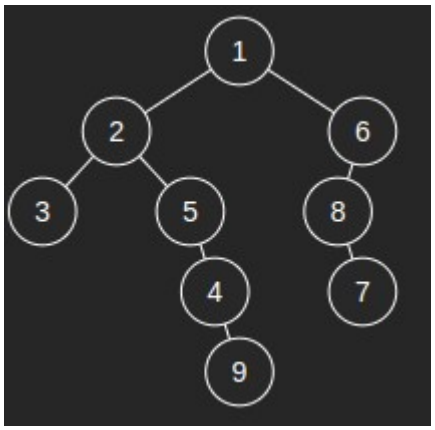
Example 2:

```
Input: root = []
Output: []
```

Example 3:

```
Input: root = [1]
Output: [1]
```

Example 4:




```
Input:
[1,2,6,3,5,8,null,null,null,null,4,null,7,null
,9]
Output: [3,2,5,4,9,1,8,7,6]
```

Constraints:

- The number of nodes in the tree is in the range `[0, 100]`.
- `-100 <= Node.val <= 100`

Follow up: Recursive solution is trivial, could you do it iteratively?

94. Binary Tree Inorder Traversal

```
/*
    Morris
    Time complexity: O(n)
    Space complexity: O(1)
    n: #nodes in the binary tree
*/

typedef std::vector<int> vi;

class Solution {
    public:
        vi inorderTraversal(TreeNode* root) {
            vi ans;
            TreeNode* cur=root;
            while(cur){
                TreeNode* ptr=cur->left;
                if(!ptr){
                    ans.push_back(cur->val);
                    cur=cur->right;
                }
                else{
                    while(ptr->right) ptr=ptr->right;
                    ptr->right=cur;
                    TreeNode* temp=cur;
                    cur=cur->left;
                    temp->left=nullptr;
                }
            }
            return ans;
        }
};
```