# 94. Binary Tree Inorder Traversal

## 题目描述：https://leetcode.com/problems/binary-tree-inorder-traversal/

> 给定一棵二叉树，求这棵二叉树的中序遍历。

## 解题思路：

1. 迭代
2. 采用stack

## 代码v1.0 Recursive：

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
 *     TreeNode *right;
 *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
 * };
 */
class Solution {
public:
    void inorder(TreeNode *root, vector<int> &res){
        if(root->left) inorder(root->left, res);
        res.push_back(root->val);
        if(root->right) inorder(root->right, res);
    }
    vector<int> inorderTraversal(TreeNode* root) {
        vector<int> res;
        if(root == NULL)
            return res;
        if(root->left) inorder(root->left, res);
        res.push_back(root->val);
        if(root->right) inorder(root->right, res);
        return res;
    }
};
```

## 代码v2.0 Iterative:

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
 *     TreeNode *right;
 *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
 * };
 */
class Solution {
public:
    vector<int> inorderTraversal(TreeNode* root) {
        stack<TreeNode*> s;
        vector<int> res;
        if(root == NULL)
            return res;
        s.push(root);
        while(!s.empty()){
            TreeNode *t = s.top();
            s.pop();
            if(t!=NULL){
                if(t->left == NULL && t->right == NULL){
                    res.push_back(t->val);
                }
                else{
                    if(t->right) s.push(t->right);
                    TreeNode *p = new TreeNode(t->val);
                    s.push(p);
                    if(t->left) s.push(t->left);
                }
            }
        }
        return res;
    }
};
```