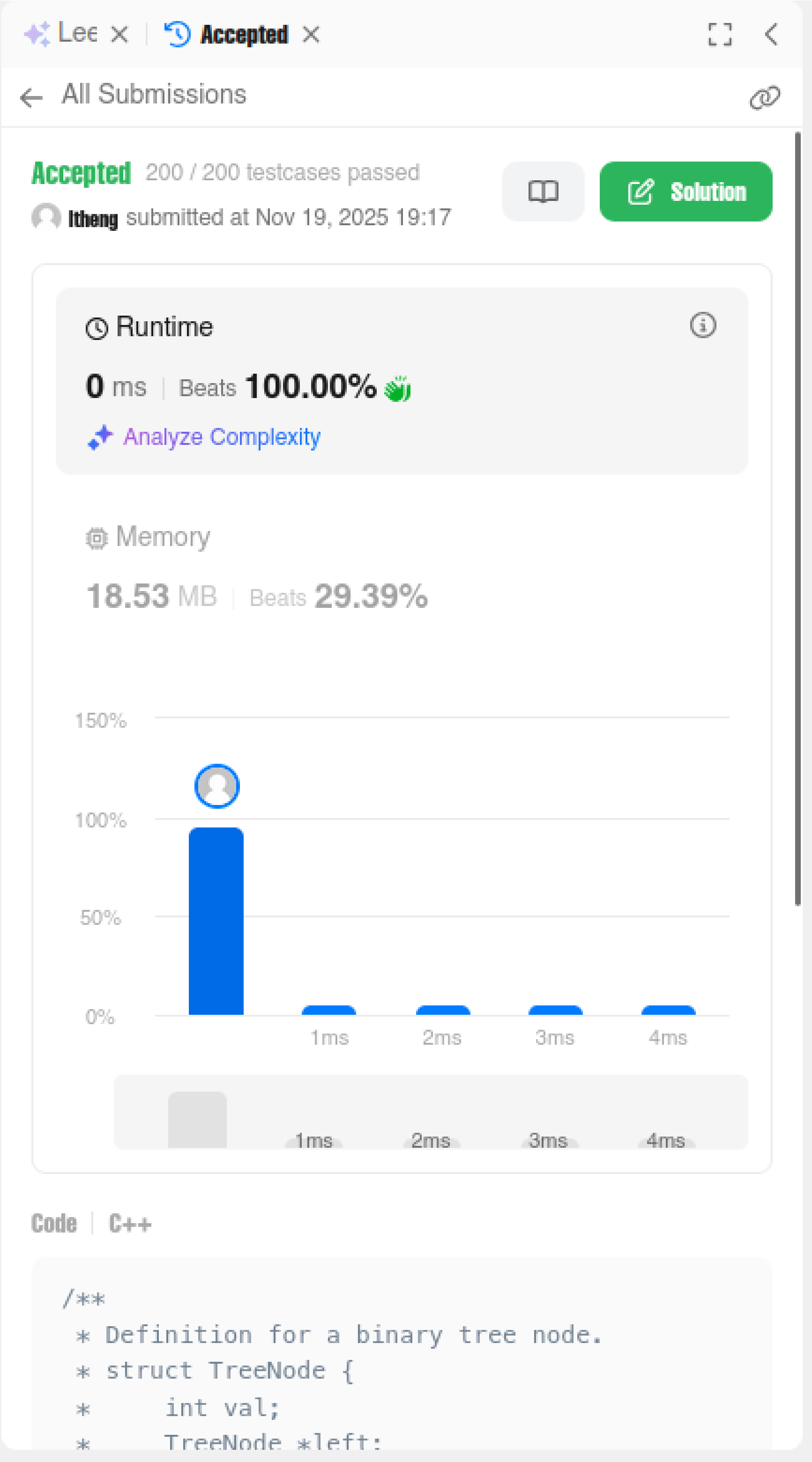
第三周练习

张睿恒 2024302182001

完成题目：101-简单、104-简单、102-中等、110-中等、124-困难、106-困难

T1 Leetcode101

```
/**
 * 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) {}
 * };
 */
class Solution {
 public:
  bool isSymmetric(TreeNode* root) {
    return isSymmetric(root, root);
  }

 private:
  bool isSymmetric(TreeNode* p, TreeNode* q) {
    if (!p || !q)
      return p == q;

    return p->val == q->val &&                //
           isSymmetric(p->left, q->right) &&  //
           isSymmetric(p->right, q->left);
  }
};
```



All Submissions

Accepted 200 / 200 testcases passed

ltheng submitted at Nov 19, 2025 19:17

Runtime

0 ms | Beats 100.00%

Analyze Complexity

Memory

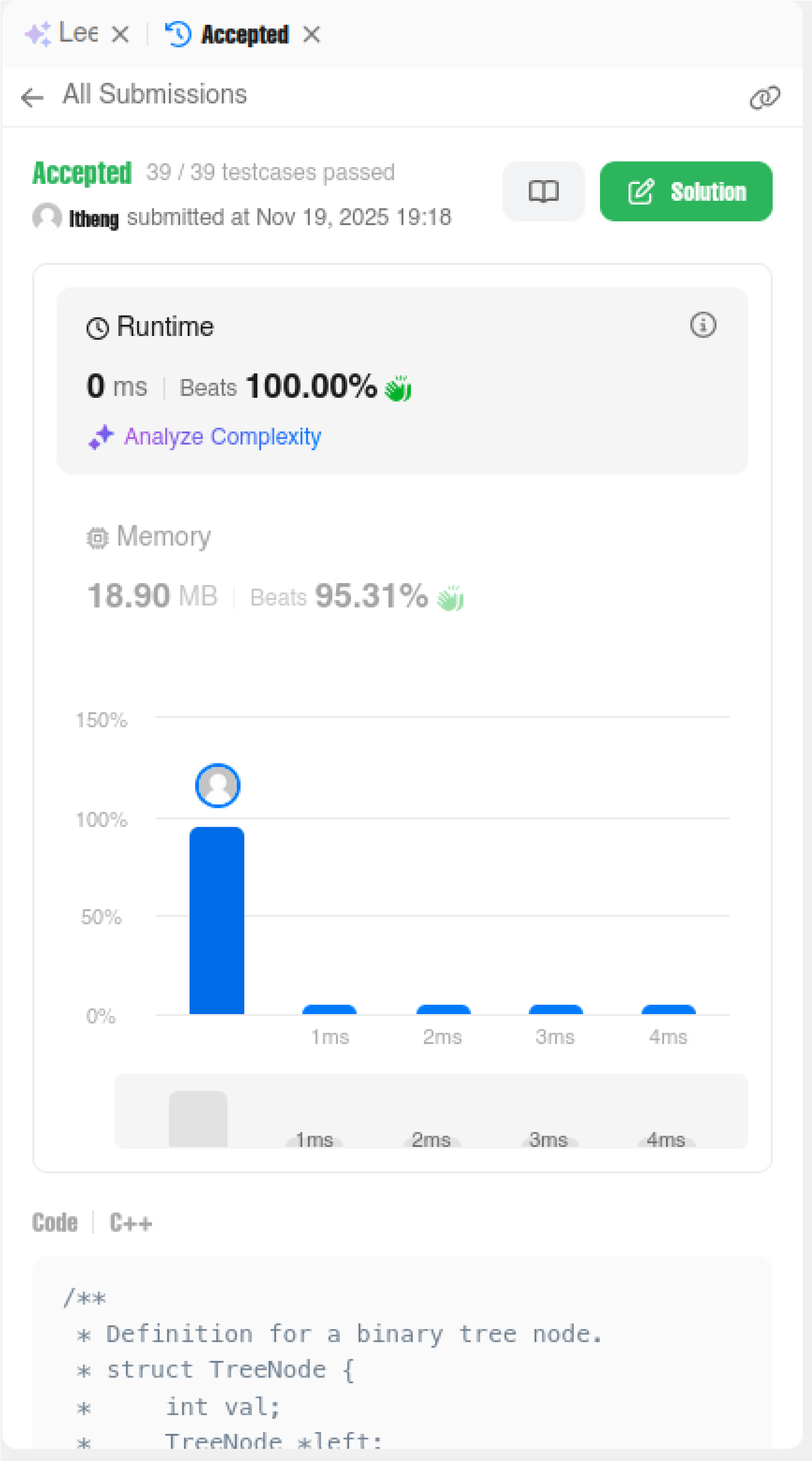
18.53 MB | Beats 29.39%

Code | C++

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
```

T2 Leetcode 104

```cpp
/**
 * 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) {}
 * };
 */
class Solution {
 public:
  int maxDepth(TreeNode* root) {
    if (root == nullptr)
      return 0;
    return 1 + max(maxDepth(root->left), maxDepth(root->right));
  }
};
```



Accepted 39 / 39 testcases passed

ltheng submitted at Nov 19, 2025 19:18

Runtime

0 ms | Beats 100.00%

Analyze Complexity

Memory

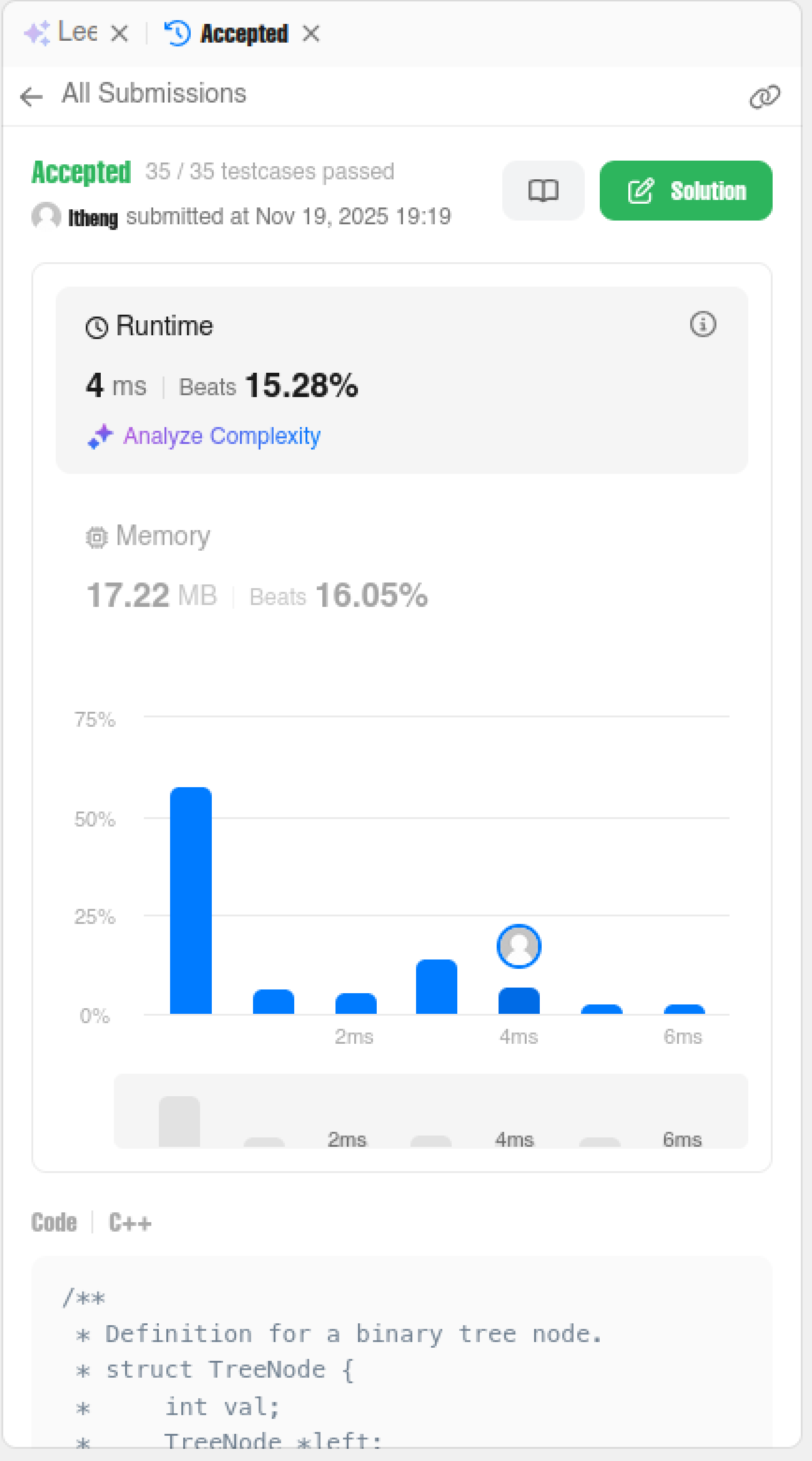
18.90 MB | Beats 95.31%

Code | C++

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
```

T3 Leetcode102

```cpp
/**
 * 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) {}
 * };
 */
class Solution {
 public:
  vector<vector<int>> levelOrder(TreeNode* root) {
    if (root == nullptr)
      return {};

    vector<vector<int>> ans;
    queue<TreeNode*> q{{root}};

    while (!q.empty()) {
      vector<int> currLevel;
      for (int sz = q.size(); sz > 0; --sz) {
        TreeNode* node = q.front();
        q.pop();
        currLevel.push_back(node->val);
        if (node->left)
          q.push(node->left);
        if (node->right)
          q.push(node->right);
      }
      ans.push_back(currLevel);
    }

    return ans;
  }
};
```



Accepted 35 / 35 testcases passed

ltheng submitted at Nov 19, 2025 19:19

Runtime

4 ms | Beats 15.28%

Analyze Complexity

Memory

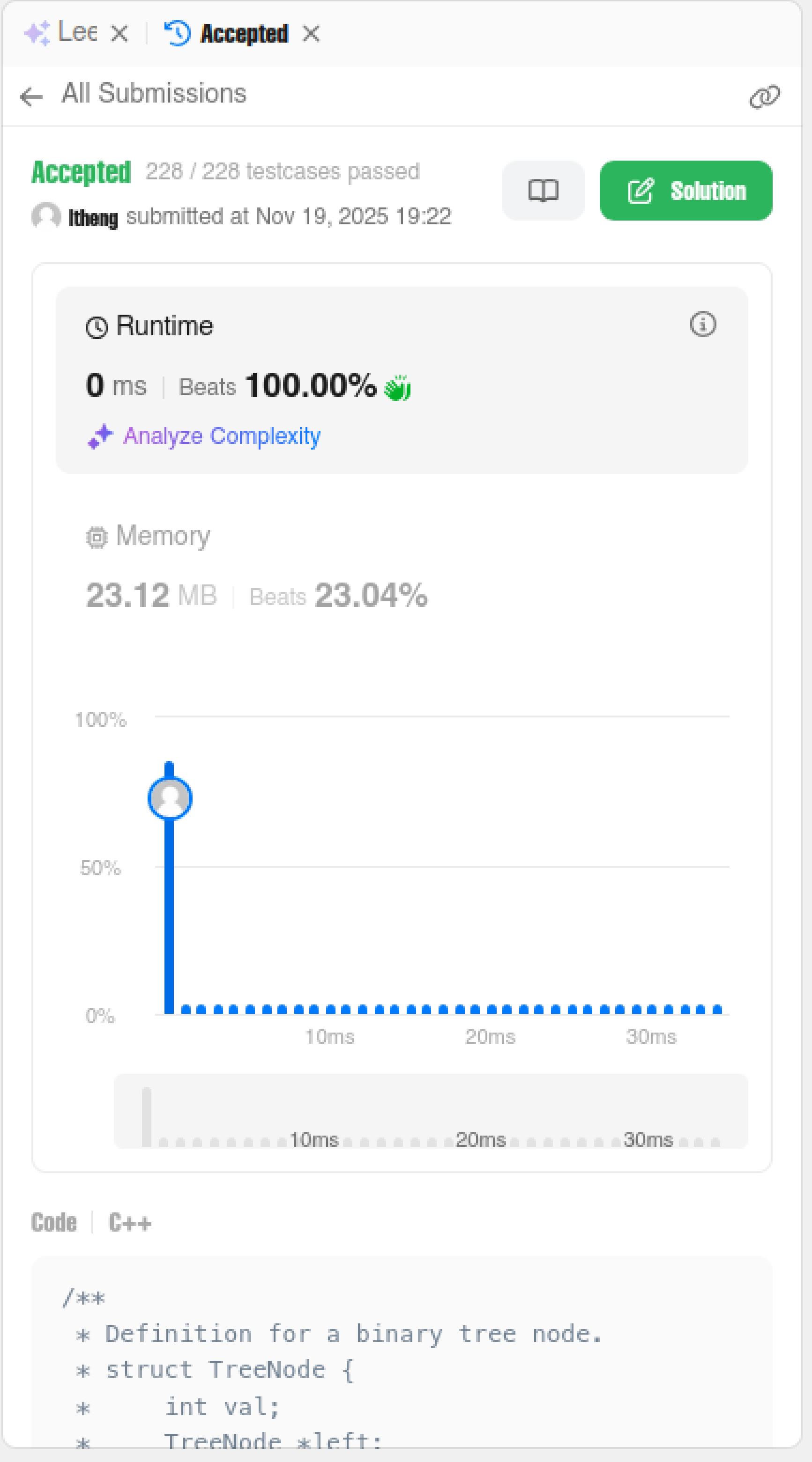
17.22 MB | Beats 16.05%

Code | C++

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
```

T4 Leetcode110

```
/**
 * 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) {}
 * };
 */
class Solution {
public:
  bool isBalanced(TreeNode* root) {
    if (root == NULL) return true;
    return abs(maxDepth(root->left) - maxDepth(root->right)) <= 1 and
           isBalanced(root->left) and isBalanced(root->right);
  }

private:
  int maxDepth(TreeNode* root) {
    if (root == NULL) return 0;
    return 1 + max(maxDepth(root->left), maxDepth(root->right));
  }
};
```



Accepted 228 / 228 testcases passed

ltheng submitted at Nov 19, 2025 19:22

Runtime

0 ms | Beats 100.00%

Analyze Complexity

Memory

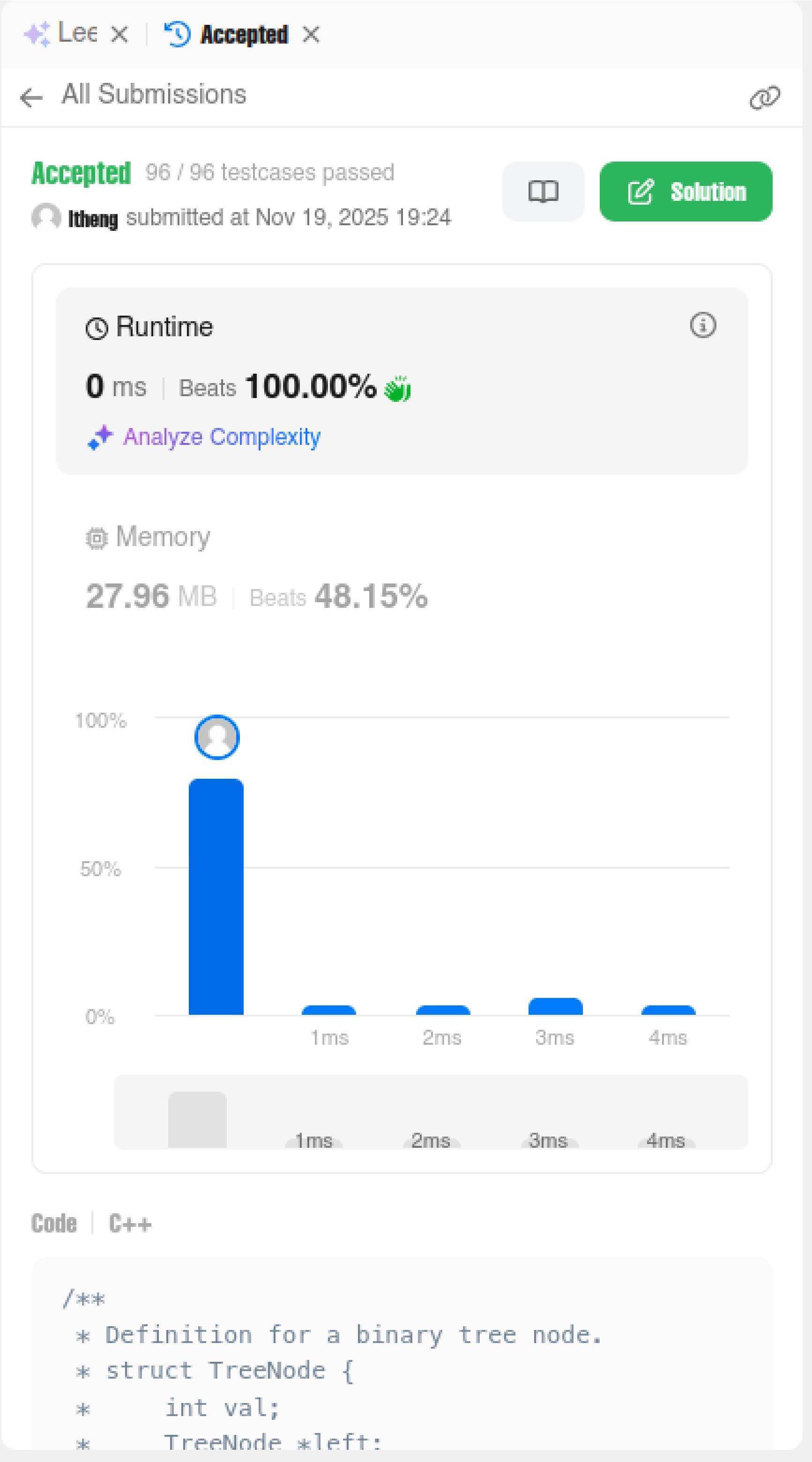
23.12 MB | Beats 23.04%

Code | C++

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
```

T5 Leetcode124

```
/**
 * 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) {}
 * };
 */
class Solution {
 public:
  int maxPathSum(TreeNode* root) {
    int ans = INT_MIN;
    maxPathSumDownFrom(root, ans);
    return ans;
  }

 private:
  int maxPathSumDownFrom(TreeNode* root, int& ans) {
    if (root == nullptr)
      return 0;

    const int l = max(0, maxPathSumDownFrom(root->left, ans));
    const int r = max(0, maxPathSumDownFrom(root->right, ans));
    ans = max(ans, root->val + l + r);
    return root->val + max(l, r);
  }
};
```



All Submissions

Accepted 96 / 96 testcases passed

ltheng submitted at Nov 19, 2025 19:24

Runtime

0 ms | Beats 100.00%

Memory

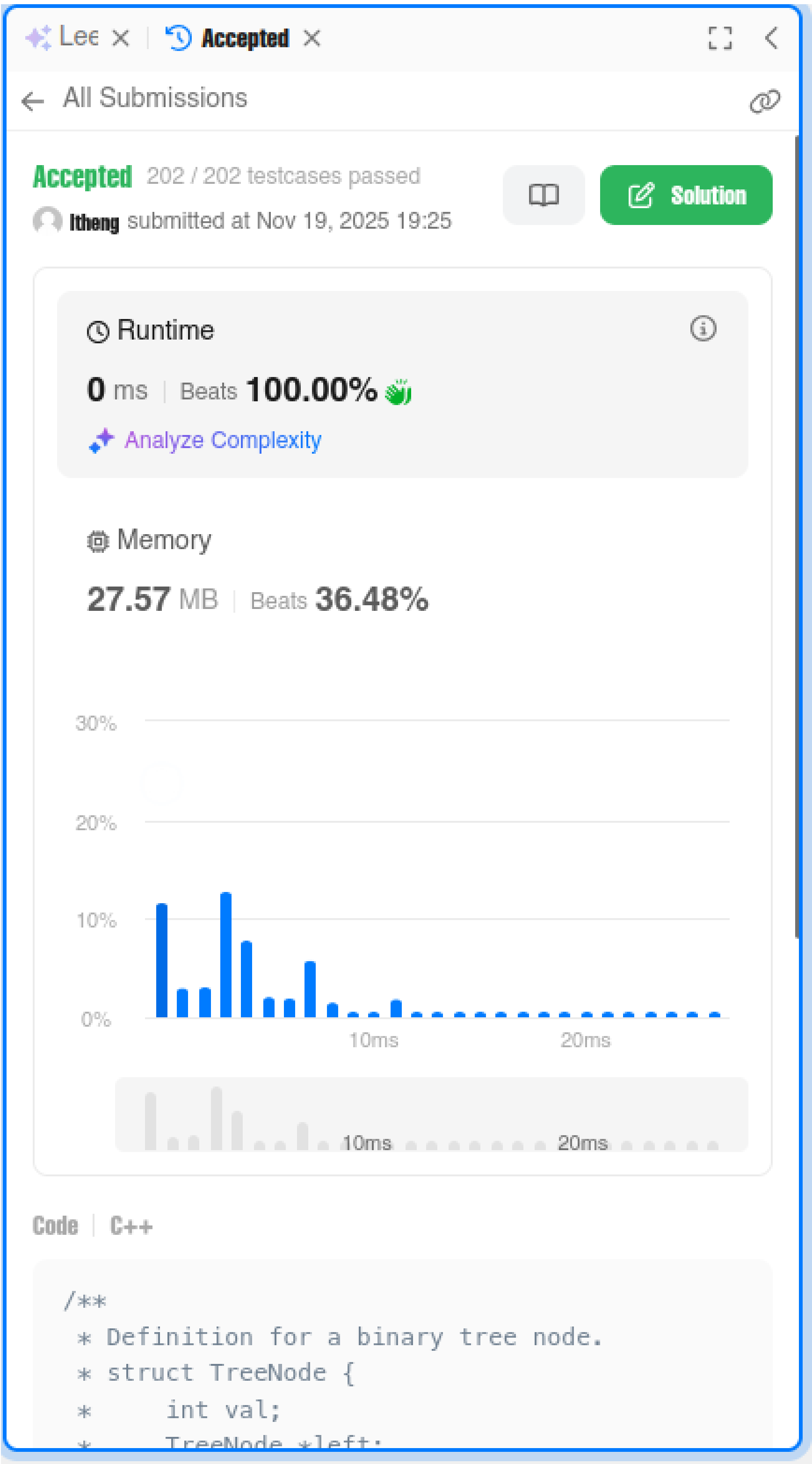
27.96 MB | Beats 48.15%

Code | C++

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
```

T6 Leetcode106

```
/**
 * 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) {}
 * };
 */
class Solution {
 public:
  TreeNode* buildTree(vector<int>& inorder, vector<int>& postorder) {
    unordered_map<int, int> inToIndex;

    for (int i=0;i<inorder.size();++i) inToIndex[inorder[i]] = i;

    return
build(inorder,0,inorder.size()-1,postorder,0,postorder.size()-1,inToIndex);
  }

 private:
  TreeNode* build(const vector<int>& inorder,int inStart,int inEnd,const vector<int>&
postorder,int postStart,int postEnd,const unordered_map<int,int>& inToIndex){
    if (inStart > inEnd) return NULL;

    const int rootVal = postorder[postEnd];
    const int rootInIndex = inToIndex.at(rootVal);
    const int leftSize = rootInIndex - inStart;

    TreeNode* root = new TreeNode(rootVal);
    root->left = build(inorder, inStart, rootInIndex - 1, postorder, postStart,
                       postStart + leftSize - 1, inToIndex);
    root->right = build(inorder, rootInIndex + 1, inEnd, postorder,
                        postStart + leftSize, postEnd - 1, inToIndex);
    return root;
  }
};
```



Accepted 202 / 202 testcases passed

ltheng submitted at Nov 19, 2025 19:25

Runtime

0 ms | Beats 100.00%

Analyze Complexity

Memory

27.57 MB | Beats 36.48%

Code | C++

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
```