102. Binary Tree Level Order Traversal

Add to List

Question | Editorial Solution | My Submissions (/problems/binary-tree-level-order-traversal/submissions/)

Total Accepted: **148678** Total Submissions: **399848** Difficulty: **Easy** Contributors: **Admin**

Given a binary tree, return the *level order* traversal of its nodes' values. (ie, from left to right, level by level).

For example:

Given binary tree `[3,9,20,null,null,15,7]`,

```
    3
   / \
  9  20
    /  \
   15   7
```

return its level order traversal as:

```
[
  [3],
  [9,20],
  [15,7]
]
```

Subscribe (/subscribe/) to see which companies asked this question

Hide Tags | Tree (/tag/tree/) | Breadth-first Search (/tag/breadth-first-search/)

Hide Similar Problems | (M) Binary Tree Zigzag Level Order Traversal (/problems/binary-tree-zigzag-level-order-traversal/) | (E) Binary Tree Level Order Traversal II (/problems/binary-tree-level-order-traversal-ii/) | (E) Minimum Depth of Binary Tree (/problems/minimum-depth-of-binary-tree/) | (M) Binary Tree Vertical Order Traversal (/problems/binary-tree-vertical-order-traversal/)

Have you met this question in a real interview? Yes No

Discuss (https://discuss.leetcode.com/category/110) | Top Solutions | Pick One (/problems/random-one-question/)

C++

```cpp
/**
 * 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<vector<int>> levelOrder(TreeNode* root) {
        vector<vector<int>> res;
        if(root==NULL) return res;
        queue<TreeNode*> Q;
        Q.push(root);
        while(!Q.empty()){
            int size = Q.size();
            vector<int> cur(size,0);
            for(int i=0;i<size;i++){
                TreeNode* tmp= Q.front();
                Q.pop();
                cur[i]=tmp->val;
                if(tmp->left!=NULL) Q.push(tmp->left);
                if(tmp->right!=NULL) Q.push(tmp->right);
            }
            res.push_back(cur);
        }
        return res;
    }
};
```

Custom Testcase

Contribute Testcase

Run Code | Submit Solution

Notes