Probl... Editor... Subm... Doubt ...

Reverse Level Order Traversal

Easy Accuracy: 47.34%
Submissions: 70803 Points: 2

Given a binary tree of size N, find its reverse level order traversal. ie- the traversal must begin from the last level.

Example 1:

```
Input :
        1
      /   \
     3     2

Output: 3 2 1
Explanation:
Traversing level 1 : 3 2
Traversing level 0 : 1
```

Example 2:

```
Input :
       10
      /  \
     20   30
    / \
   40  60

Output: 40 60 20 30 10
Explanation:
Traversing level 2 : 40 60
Traversing level 1 : 20 30
Traversing level 0 : 10
```

Your Task:
You dont need to read input or print anything. Complete the function **reverseLevelOrder()** which takes the root of the tree as input parameter and returns a list containing the reverse level order traversal of the given tree.

You get marks only for the first correct submission i.e. after the problem is marked as Solved. Viewing the full solution.

Expected Time Complexity: O(N)
Expected Auxiliary Space: O(N)

Constraints:
1 <= N <= 10^4

View Bookmarked Problems (https://practice.geeksforgeeks.org/explore/?problemType=bookmark)

Company Tags

Topic Tags

Related Courses

We are replacing the old Disqus forum with the new Discussions section given below.
Click here (https://practice.geeksforgeeks.org/comments/reverse-level-order-traversal/1/?rel=https://practice.geeksforgeeks.org/problems/reverse-level-order-traversal/1) to view old Disqus comments.

Discussions

sibajit1176be20

0

This comment was

namangoel801 1 w

0

```
EASIEST SOLUTION
SMALL CHANGE IN NOI

class Inpo
```

C++ (g++ 5.4) Test against custom input

```
114 }; */
115 vector<int> reverseLevelOrder(Node *root)
116 {
117     // code here
118
119     queue<pair<Node*, int>> traverse;
120
121     vector<int> ans;
122     map<int, vector<int>> levels;
123     traverse.push(make_pair(root, 0));
124     while(!traverse.empty()){
125         if(traverse.front().first->left != NULL) traverse.push(make_pair(traverse.front().firs
126         if(traverse.front().first->right != NULL) traverse.push(make_pair(traverse.front().fir
127         levels[traverse.front().second].push_back(traverse.front().first->data);
128         traverse.pop();
129
130     }
131     for(int i = levels.size() - 1; i >= 0; i--){
132         ans.insert(ans.end(), levels[i].begin(), levels[i].end());
133     }
134     return ans;
135 }
```

Output Window

Problem Solved Successfully

Test Cases Passed: **110** / 110

Total Time Taken: **0.32**/1.62

Correct Submission Count: **2**

Attempts No.: **2**

Compile & Run Submit