Practice(https://practice.geeksforgeeks.org/home/) 168 (/leaderboard)

Problem Editorial Submissions Doubt Support C++ (g++ 5.4) Test against custom input

```
   8    8   6
  /  \
 10   9
Output:
7 7 9 8 8 6 9 10
```

Your Task:
You don't need to read input or print anything. Your task is to complete the function **zigZagTraversal()** which takes the root node of the Binary Tree as its input and returns a list containing the node values as they appear in the Zig-Zag Level-Order Traversal of the Tree.

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

Related Interview Experiences

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

Discussions

+1

karansinghyoyo77 Premium 3 days ago

```
class GFG  // If this help plzz upvote
{
    //Function to store the zig zag order tr
    ArrayList<Integer> zigZagTraversal(Node
    {
         ArrayList<Integer> ans= new ArrayLi
        if(root==null){
            return ans;
        }
        Queue<Node> q=new LinkedList<>();
        Stack<Integer> s=new Stack<>();
        boolean reverse=false;
        q.add(root);
        while (q.isEmpty()==false){
            int count=q.size();
            for (int i = 0; i <count ; i++)
                Node curr=q.poll();
                if(reverse==true){
                    s.push(curr.data);
                }
                else {
                    ans.add(curr.data);
                }
                if(curr.left!=null){
                    q.add(curr.left);
                }
                if(curr.right!=null){
                    q.add(curr.right);
                }

            }
            if(reverse==true){
                while (s.isEmpty()==false){
                    ans.add(s.pop());
                }
            }
            reverse=!reverse;
        }
        return ans;

    }
```

```
void solve(Node* root, map<int, vector<int>>& level, int vLevel){
    if(root == NULL)
        return;

    solve(root->left, level, vLevel + 1);
    level[vLevel].push_back(root->data);
    solve(root->right, level, vLevel + 1);
}
vector <int> zigZagTraversal(Node* root)
{
    // Code here
    map<int, vector<int>> level;
    vector<int> ans;
    solve(root, level, 0);
    for(auto node: level){
        if(node.first % 2)
            reverse(node.second.begin(), node.second.end());

        ans.insert(ans.end(), node.second.begin(), node.second.end());
    }
    return ans;
}
```

Average Time: 30m
Your Time: 35m 36s

Compile & Run Submit