## 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:**

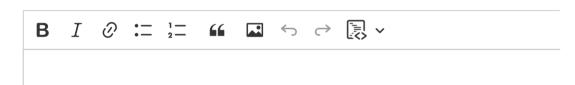
 $1 \le N \le 10^4$ 

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 Disgus comments.

## Discussions **2**



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;
```

```
C++ (g++ 5.4)
                  Test against custom input
         void solve(Node* root, map<int, vector<int>>& level, int vLevel){
106
107
             if(root == NULL)
108
                 return;
109
             solve(root->left, level, vLevel + 1);
110
111
             level[vLevel].push_back(root->data);
             solve(root->right, level, vLevel + 1);
112
113
         }
114
         vector <int> zigZagTraversal(Node* root)
115
             // Code here
116
117
             map<int, vector<int>> level;
118
             vector<int> ans;
             solve(root, level, 0);
119
120
             for(auto node: level){
                 if(node.first % 2)
121
122
                     reverse(node.second.begin(), node.second.end());
123
                 ans.insert(ans.end(), node.second.begin(), node.second.end());
124
125
126
             return ans;
         }
127
128
```

♂ Average Time: 30m

Your Time: 35m 36s



<u>-ÿ</u>.