69. Binary Tree Level Order Traversal

* [Description](http://lintcode.com/en/problem/binary-tree-level-order-traversal/" \l "description)
* [Notes](http://lintcode.com/en/problem/binary-tree-level-order-traversal/#note)
* [Testcase](http://lintcode.com/en/problem/binary-tree-level-order-traversal/#testcase)
* [Judge](http://lintcode.com/en/problem/binary-tree-level-order-traversal/#judge)

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

Have you met this question in a real interview?

Yes

**Example**

Given binary tree {3,9,20,#,#,15,7},

3

/ \

9 20

/ \

15 7

return its level order traversal as:

[

[3],

[9,20],

[15,7]

]

<http://lintcode.com/en/problem/binary-tree-level-order-traversal/#>

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package javaapplication1;

import java.util.\*;

class TreeNode {

public int val;

public TreeNode left, right;

public TreeNode(int val) {

this.val = val;

this.left = this.right = null;

}

}

public class JavaApplication1 {

public static List<List<Integer>> levelOrder(TreeNode root) {

// write your code here

if(root == null)

return new ArrayList();

ArrayList<ArrayList<Integer>> lista =

new ArrayList();

ArrayList<Integer> aux =

new ArrayList();

// Create an empty queue for level order tarversal

Queue<TreeNode> q =new LinkedList();

// Enqueue Root and initialize height

q.add(root);

while(true)

{

// nodeCount (queue size) indicates number of nodes

// at current level.

int nodeCount = q.size();

if(nodeCount == 0)

break;

// Dequeue all nodes of current level and Enqueue all

// nodes of next level

while(nodeCount > 0)

{

TreeNode node = q.peek();

//System.out.print(node.val + " ");

aux.add(node.val);

q.remove();

if(node.left != null)

q.add(node.left);

if(node.right != null)

q.add(node.right);

nodeCount--;

}

//System.out.println();

lista.add(aux);

aux = new ArrayList();

}

List<List<Integer>> res = new ArrayList(lista);

return res;

}

public static void main(String[] args) {

// TODO code application logic here

int[] tree = {3,9,20,15,7};

}

}