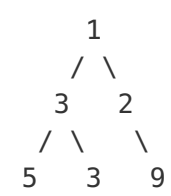
Find Largest Value in Each Tree Row

You need to find the largest value in each row of a binary tree.

Example:

Input:



Output: [1, 3, 9]

Solution 1

Alright, two binary tree level order traversal problems in one contest. This time, mission is to find the `max` of each level...

```
public class Solution {
    public int[] findValueMostElement(TreeNode root) {
        List<Integer> res = new ArrayList<>();
        if (root == null) return new int[0];

        Queue<TreeNode> queue = new LinkedList<>();
        queue.add(root);

        while (!queue.isEmpty()) {
            int size = queue.size();
            int max = Integer.MIN_VALUE;
            for (int i = 0; i < size; i++) {
                TreeNode node = queue.poll();
                max = Math.max(max, node.val);
                if (node.left != null) queue.add(node.left);
                if (node.right != null) queue.add(node.right);
            }
            res.add(max);
        }

        int[] result = new int[res.size()];
        for (int i = 0; i < res.size(); i++) {
            result[i] = res.get(i);
        }

        return result;
    }
}
```

written by shawngao original link here

Solution 2

```
def findValueMostElement(self, root):
    maxes = []
    row = [root]
    while any(row):
        maxes.append(max(node.val for node in row))
        row = [kid for node in row for kid in (node.left, node.right) if kid]
    return maxes
```

written by StefanPochmann original link here

Solution 3

```
    public int[] findValueMostElement(TreeNode root) {
        Queue<TreeNode> queue = new LinkedList<TreeNode>();
        List<Integer> res = new ArrayList<Integer>();
        queue.add(root);
        int queueSize = root == null ? 0 : 1;
        while (queueSize > 0) {
            int largestElement = Integer.MIN_VALUE;
            for (int i=0;i<queueSize;i++) {
                TreeNode cur = queue.poll();
                largestElement = Math.max(cur.val, largestElement);
                if (cur.left != null) queue.add(cur.left);
                if (cur.right != null) queue.add(cur.right);
            }
            res.add(largestElement);
            queueSize = queue.size();
        }
        int[] resArray = new int[res.size()];
        for (int i=0;i<res.size();i++) resArray[i] = res.get(i);
        return resArray;
    }
```

written by compton_scatter original link here