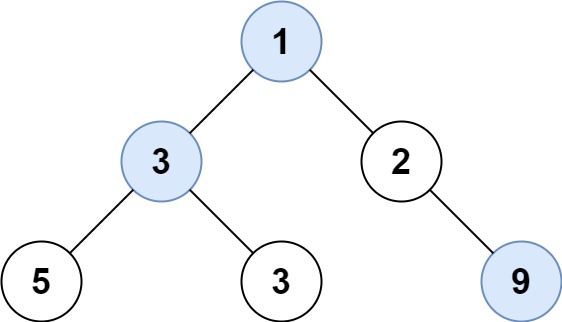
**515. Find Largest Value in Each Tree Row: -**

Medium Accepted: 274.1K Submissions: 421.2K Acceptance Rate: 65.1%

Given the root of a binary tree, return *an array of the largest value in each row* of the tree **(0-indexed)**.

**Example 1:**



**Input:** root = [1,3,2,5,3,null,9]

**Output:** [1,3,9]

**Example 2:**

**Input:** root = [1,2,3]

**Output:** [1,3]

**Constraints:**

* The number of nodes in the tree will be in the range [0, 104].
* -231 <= Node.val <= 231 - 1

**Code: -**

/\*\*

 \* Definition for a binary tree node.

 \* struct TreeNode {

 \*     int val;

 \*     TreeNode \*left;

 \*     TreeNode \*right;

 \*     TreeNode() : val(0), left(nullptr), right(nullptr) {}

 \*     TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}

 \*     TreeNode(int x, TreeNode \*left, TreeNode \*right) : val(x), left(left), right(right) {}

 \* };

 \*/

class Solution {

public:

    void dfs(TreeNode \*root, vector<int> &ans, int level){

        if(!root)   return;

        if(ans.size() <= level)     ans.push\_back(root->val);

        else        ans[level] = max(ans[level], root->val);

        dfs(root->left, ans, level+1);

        dfs(root->right, ans, level+1);

        return;

    }

    vector<int> largestValues(TreeNode\* root) {

        vector<int> ans;

        dfs(root, ans, 0);

        return ans;

    }

};

**T.C: - O(N)**

**S.C: - O(H)**

**H = height of tree**