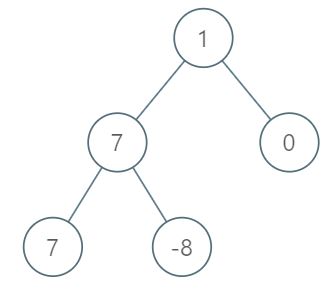
**1161. Maximum Level Sum of a Binary Tree :-**

Medium Accepted: 185.2K Submissions: 272.1K Acceptance Rate: 68.1%

Given the root of a binary tree, the level of its root is 1, the level of its children is 2, and so on.

Return the **smallest** level x such that the sum of all the values of nodes at level x is **maximal**.

**Example 1:**



**Input:** root = [1,7,0,7,-8,null,null]

**Output:** 2

**Explanation:**

Level 1 sum = 1.

Level 2 sum = 7 + 0 = 7.

Level 3 sum = 7 + -8 = -1.

So we return the level with the maximum sum which is level 2.

**Example 2:**

**Input:** root = [989,null,10250,98693,-89388,null,null,null,-32127]

**Output:** 2

**Constraints:**

* The number of nodes in the tree is in the range [1, 104].
* -105 <= Node.val <= 105

**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:

    int maxLevelSum(TreeNode\* root) {

        int ans = 1, sumcurrent = root->val, level=2;

        queue<TreeNode \*> q;

        if(root->left)

            q.push(root->left);

        if(root->right)

            q.push(root->right);

        while(q.empty()==false){

            int count = q.size(), sumlevel=0;

            while(count--){

                TreeNode \*front = q.front();

                q.pop();

                sumlevel += (front->val);

                if(front->left)

                    q.push(front->left);

                if(front->right)

                    q.push(front->right);

            }

            if(sumlevel > sumcurrent){

                ans = level;

                sumcurrent = sumlevel;

            }

            ++level;

        }

        return ans;

    }

};

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

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