75 Days of Code Day 39 1161. Maximum Level Sum of a Binary Tree

Type : BST / bfs

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

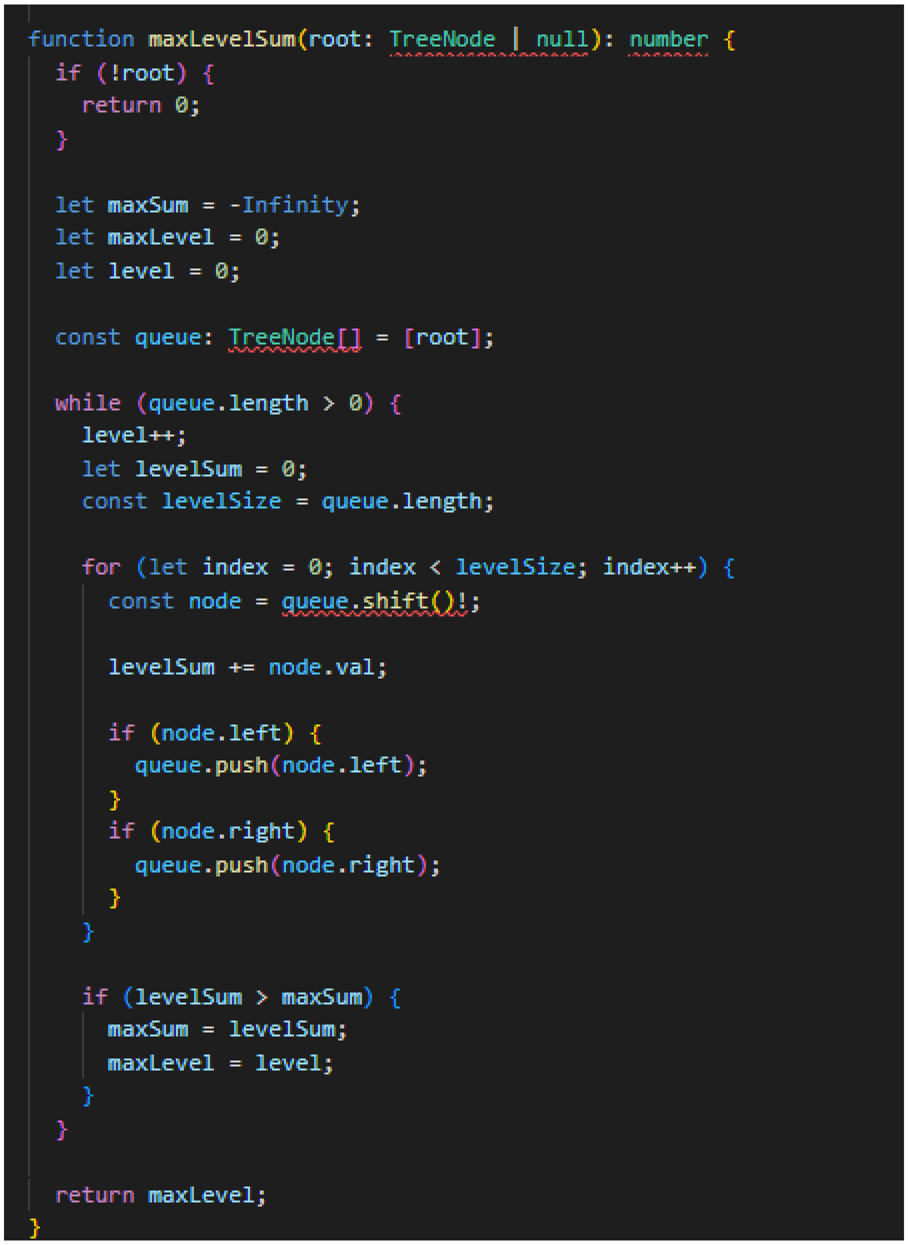
Constraints:

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

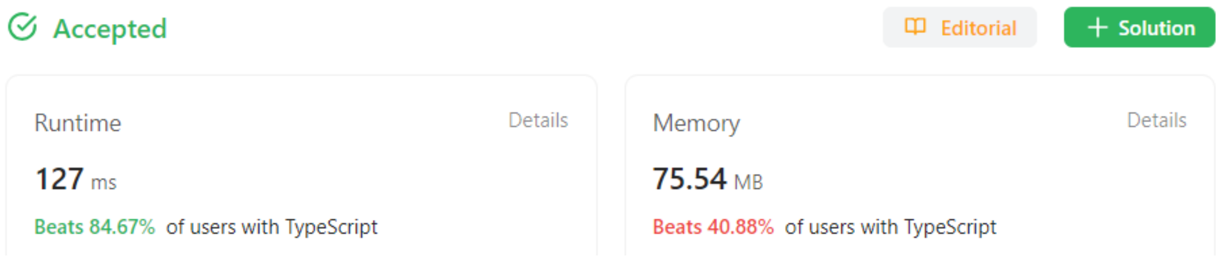
Shubham Agrahari

Solution using BFS

1. Use Queue to enqueue (insert at first) and dequeue(delete at first)
2. Loop for each element inserted (which acts as a level)
3. Compare and put the max level



```
function maxLevelSum(root: TreeNode | null): number {
  if (!root) {
    return 0;
  }

  let maxSum = -Infinity;
  let maxLevel = 0;
  let level = 0;

  const queue: TreeNode[] = [root];

  while (queue.length > 0) {
    level++;
    let levelSum = 0;
    const levelSize = queue.length;

    for (let index = 0; index < levelSize; index++) {
      const node = queue.shift()!;

      levelSum += node.val;

      if (node.left) {
        queue.push(node.left);
      }
      if (node.right) {
        queue.push(node.right);
      }
    }

    if (levelSum > maxSum) {
      maxSum = levelSum;
      maxLevel = level;
    }
  }

  return maxLevel;
}
```



Accepted

Runtime — Details

127 ms

Beats 84.67% of users with TypeScript

Memory — Details

75.54 MB

Beats 40.88% of users with TypeScript

Shubham Agrahari

Shubham Agrahari