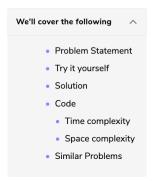
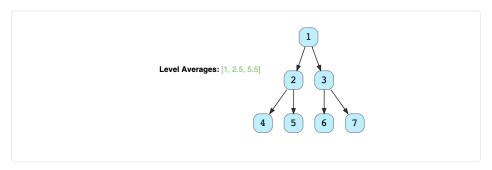
Level Averages in a Binary Tree (easy)



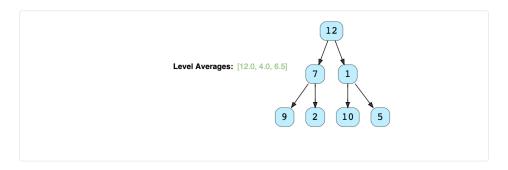
Problem Statement

Given a binary tree, populate an array to represent the averages of all of its levels.

Example 1:



Example 2:



Try it yourself

Try solving this question here:

```
public static void main(String[] args) {
    TreeNode root = new TreeNode(12);
    root.left = new TreeNode(7);
    root.left = new TreeNode(1);
    root.left.left = new TreeNode(9);
    root.left.right = new TreeNode(2);
    root.right.left = new TreeNode(2);
    root.right.left = new TreeNode(5);
    root.right.right = new TreeNode(5);
    List<Double> result = LevelAverage.findLevelAverages(root);

Run

Run

Ran

Reset C3
```

Solution

This problem follows the Binary Tree Level Order Traversal pattern. We can follow the same BFS approach. The only difference will be that instead of keeping track of all nodes of a level, we will only track the running sum of the values of all nodes in each level. In the end, we will append the average of the current level to the result array.

Code

Here is what our algorithm will look like; only the highlighted lines have changed:

```
Python3
                       G C++
         java.util.*;
   class TreeNode {
     TreeNode left;
     TreeNode right;
     TreeNode(int x) {
   class LevelAverage {
     public static List<Double> findLevelAverages(TreeNode root) {
       List<Double> result = new ArrayList<>();
       if (root == null)
         return result;
       Oueue<TreeNode> queue = new LinkedList<>():
       queue.offer(root);
       while (!queue.isEmpty()) {
       int levelSize = queue.size();
         double levelSum = 0;
         for (int i = 0; i < levelSize; i++) {</pre>
           TreeNode currentNode = queue.poll();
           levelSum += currentNode.val;
Run
                                                                                              Reset []
```

Time complexity

The time complexity of the above algorithm is O(N), where 'N' is the total number of nodes in the tree. This is due to the fact that we traverse each node once.

Space complexity

The space complexity of the above algorithm will be O(N) which is required for the queue. Since we can have a maximum of N/2 nodes at any level (this could happen only at the lowest level), therefore we will need O(N) space to store them in the queue.

Similar Problems

Problem 1: Find the largest value on each level of a binary tree.

Solution: We will follow a similar approach, but instead of having a running sum we will track the maximum value of each level.



! Report an Issue ? Ask a Question