

MCS – 253P ADVANCED PROGRAMMING AND PROBLEM SOLVING

LAB 6 Program(Print Binary Tree)

Aswin Sampath


saswin@uci.edu

53844684

Date: 08/11/2023

Question:

655. Print Binary Tree

Medium  433  421  

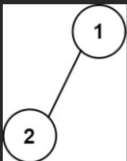
 Companies

Given the `root` of a binary tree, construct a 0-indexed `m x n` string matrix `res` that represents a **formatted layout** of the tree. The formatted layout matrix should be constructed using the following rules:

- The **height** of the tree is `height` and the number of rows `m` should be equal to `height + 1`.
- The number of columns `n` should be equal to $2^{\text{height}+1} - 1$.
- Place the **root node** in the **middle** of the **top row** (more formally, at location `res[0][(n-1)/2]`).
- For each node that has been placed in the matrix at position `res[r][c]`, place its **left child** at `res[r+1][c-2height-r-1]` and its **right child** at `res[r+1][c+2height-r-1]`.
- Continue this process until all the nodes in the tree have been placed.
- Any empty cells should contain the empty string `""`.

Return the constructed matrix `res`.

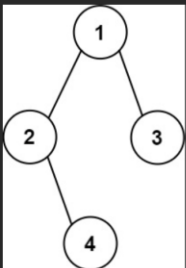
Example 1:



Input: `root = [1,2]`

Output:
`[["", "1", ""],`
`["2", "", ""]]`

Example 2:



Input: `root = [1,2,3,null,4]`

Output:
`[["", "", "1", "", "", ""],`
`["", "2", "", "", "3", ""],`
`["", "", "4", "", "", ""]]`

Code

```
1 class Solution {
2 public:
3     int findHeight(TreeNode* root){
4         if(!root) return 0;
5         if(root && root->left && root->right) return 1 + max(findHeight(root->left), findHeight(root->right));
6         if(root->left) return 1 + findHeight(root->left);
7         return 1 + findHeight(root->right);
8     }
9
10    void assignMatrix(vector<vector<string>> &ans, TreeNode *root, int r, int c, int height){
11        if(!root) return;
12        ans[r][c] = to_string(root->val);
13        if(root->left) assignMatrix(ans, root->left, r+1, c - pow(2, height-r-1), height);
14        if(root->right) assignMatrix(ans, root->right, r+1, c+pow(2, height-r-1), height);
15    }
16
17    vector<vector<string>> printTree(TreeNode* root) {
18        int m = findHeight(root);
19        int height = m-1;
20        int n = pow(2, m)-1;
21        vector<vector<string>> ans(m, vector<string>(n, ""));
22        int middle = n/2;
23        assignMatrix(ans, root, 0, (n-1)/2, height);
24        return ans;
25    }
26 };
```

Output:

Problem List < > ⇅ Dynamic Layout Premium ⌚ 0

Aswin
Nov 08, 2023 20:20

Details + Solution

C++

Runtime 0 ms Beats 100% Memory 8.6 MB Beats 66.30%

Click the distribution chart to view more details

Related Tags

Select related tags 0/5

```
/**
 * 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) {}
 * };
 */
```