101. Symmetric Tree ★

Question | Editorial Solution | My Submissions (/problems/symmetric-tree/submissions/)

Total Accepted: **132627** Total Submissions: **369584** Difficulty: **Easy**

Given a binary tree, check whether it is a mirror of itself (ie, symmetric around its center).

For example, this binary tree `[1,2,2,3,4,4,3]` is symmetric:

```
    1
   / \
  2   2
 / \ / \
3  4 4  3
```

But the following `[1,2,2,null,3,null,3]` is not:

```
    1
   / \
  2   2
   \   \
   3    3
```

Note:

Bonus points if you could solve it both recursively and iteratively.

Subscribe (/subscribe/) to see which companies asked this question

Show Tags

Have you met this question in a real interview? Yes No

Discuss (https://leetcode.com/discuss/questions/oj/symmetric-tree)

Pick One (/problems/random-one-question/)

C++

```
/**
 * Definition for a binary tree node.
 * struct TreeNode {
 *     int val;
 *     TreeNode *left;
 *     TreeNode *right;
 *     TreeNode(int x) : val(x), left(NULL), right(NULL) {}
 * };
 */
class Solution {
public:
```

Send Feedback (mailto:admin@leetcode.com?subject=Feedback)

Notes

```
12      bool checkSymmetric(TreeNode* p, TreeNode* q) {
13          if (!p && !q)
14              return true;
15          else if (!p || !q)
16              return false;
17          return (p->val == q->val) && checkSymmetric(p->left, q->right) && checkSy
18      }
19
20      bool isSymmetric(TreeNode* root) {
21          return (root)? checkSymmetric(root->left, root->right) : true;
22      }
23  };
```

Notes

Custom Testcase

Run Code | Submit Solution