101. Symmetric Tree

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

/\*\*

\* 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:

bool checkSymmetric(vector<int> vec){

int i=0,j=vec.size()-1;

while(i<j){

if(vec[i]!=vec[j]) return false;

i++;

j--;

}

return true;

}

bool isSymmetric(TreeNode\* root) {

if(root==NULL) return true;

int childCount=1;

queue<TreeNode\*> q;

q.push(root);

while(!q.empty()){

int c=0;

vector<int> vec;

for(int i=0;i<childCount;i++){

TreeNode\* curr=q.front();

q.pop();

if(curr->left!=NULL){

vec.push\_back(curr->left->val);

q.push(curr->left);

c++;

}else{

vec.push\_back(-1);

}

if(curr->right!=NULL){

vec.push\_back(curr->right->val);

q.push(curr->right);

c++;

}else{

vec.push\_back(-1);

}

}

childCount=c;

if(!checkSymmetric(vec)) return false;

}

return true;

}

};Success

[Details](https://leetcode.com/submissions/detail/210649843/)

Runtime: 12 ms, faster than 100.00% of C++ online submissions for Symmetric Tree.

Memory Usage: 16.3 MB, less than 11.37% of C++ online submissions for Symmetric Tree.