Problem 1361: Validate Binary Tree Nodes

Problem Information

Difficulty: Medium
Acceptance Rate: 43.98%
Paid Only: No
Tags: Tree, Depth-First Search, Breadth-First Search, Union Find, Graph, Binary Tree

Problem Description

You have `n` binary tree nodes numbered from `0` to `n - 1` where node `i` has two children `leftChild[i]` and `rightChild[i]`, return `true` if and only if **all** the given nodes form **exactly one** valid binary tree.

If node `i` has no left child then `leftChild[i]` will equal `-1`, similarly for the right child.

Note that the nodes have no values and that we only use the node numbers in this problem.

Example 1:

![](https://assets.leetcode.com/uploads/2019/08/23/1503_ex1.png)

Input: n = 4, leftChild = [1,-1,3,-1], rightChild = [2,-1,-1,-1] **Output:** true

Example 2:

![](https://assets.leetcode.com/uploads/2019/08/23/1503_ex2.png)

Input: n = 4, leftChild = [1,-1,3,-1], rightChild = [2,3,-1,-1] **Output:** false

Example 3:

![](https://assets.leetcode.com/uploads/2019/08/23/1503_ex3.png)

Input: n = 2, leftChild = [1,0], rightChild = [-1,-1] **Output:** false

Constraints:

* `n == leftChild.length == rightChild.length` * `1 <= n <= 104` * `-1 <= leftChild[i], rightChild[i] <= n - 1`

Code Snippets

C++:

```
class Solution {
public:
bool validateBinaryTreeNodes(int n, vector<int>& leftChild, vector<int>&
rightChild) {

}
};
```

Java:

```
class Solution {
public boolean validateBinaryTreeNodes(int n, int[] leftChild, int[]
rightChild) {

}
}
```

Python3:

```
class Solution:
def validateBinaryTreeNodes(self, n: int, leftChild: List[int], rightChild:
List[int]) -> bool:
```