Problem 261: Graph Valid Tree

Problem Information

Difficulty: Medium
Acceptance Rate: 49.64%
Paid Only: Yes
Tags: Depth-First Search, Breadth-First Search, Union Find, Graph

Problem Description

You have a graph of `n` nodes labeled from `0` to `n - 1`. You are given an integer n and a list of `edges` where `edges[i] = [ai, bi]` indicates that there is an undirected edge between nodes `ai` and `bi` in the graph.

Return `true` _if the edges of the given graph make up a valid tree, and_ `false` _otherwise_.

Example 1:

![](https://assets.leetcode.com/uploads/2021/03/12/tree1-graph.jpg)

Input: n = 5, edges = [[0,1],[0,2],[0,3],[1,4]] **Output:** true

Example 2:

![](https://assets.leetcode.com/uploads/2021/03/12/tree2-graph.jpg)

Input: n = 5, edges = [[0,1],[1,2],[2,3],[1,3],[1,4]] **Output:** false

Constraints:

* `1 <= n <= 2000` * `0 <= edges.length <= 5000` * `edges[i].length == 2` * `0 <= ai, bi < n` * `ai != bi` * There are no self-loops or repeated edges.

Code Snippets

C++:

```
class Solution {
public:
bool validTree(int n, vector<vector<int>>& edges) {

}
};
```

Java:

```
class Solution {
public boolean validTree(int n, int[][] edges) {

}
}
```

Python3:

```
class Solution:
def validTree(self, n: int, edges: List[List[int]]) -> bool:
```