1039 Find the Town Judge (link)

Description

In a town, there are `n` people labeled from `1` to `n`. There is a rumor that one of these people is secretly the town judge.

If the town judge exists, then:

1. The town judge trusts nobody.
2. Everybody (except for the town judge) trusts the town judge.
3. There is exactly one person that satisfies properties **1** and **2**.

You are given an array `trust` where `trust[i] = [a_i, b_i]` representing that the person labeled `a_i` trusts the person labeled `b_i`. If a trust relationship does not exist in `trust` array, then such a trust relationship does not exist.

Return *the label of the town judge if the town judge exists and can be identified, or return* `-1` *otherwise*.

Example 1:

```
Input: n = 2, trust = [[1,2]]
Output: 2
```

Example 2:

```
Input: n = 3, trust = [[1,3],[2,3]]
Output: 3
```

Example 3:

```
Input: n = 3, trust = [[1,3],[2,3],[3,1]]
Output: -1
```

Constraints:

- `1 <= n <= 1000`
- `0 <= trust.length <= 10^4`
- `trust[i].length == 2`
- All the pairs of `trust` are **unique**.
- `a_i != b_i`
- `1 <= a_i, b_i <= n`

(scroll down for solution)

Solution

Language: cpp

Status: Accepted

```cpp
#include <vector>
using namespace std;

class Solution {
public:
    int findJudge(int n, vector<vector<int>>& trust) {
        vector<int> trustsIn(n + 1, 0);
        vector<int> trustsOut(n + 1, 0);

        for (auto& t : trust) {
            int a = t[0];
            int b = t[1];
            trustsIn[b]++;
            trustsOut[a]++;
        }

        for (int i = 1; i <= n; ++i) {
            if (trustsIn[i] == n - 1 && trustsOut[i] == 0) {
                return i;
            }
        }

        return -1;
    }
};
```