1135. Connecting Cities With Minimum Cost

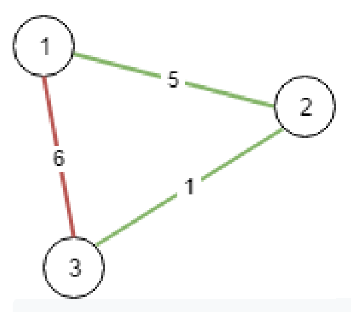
Medium 741 40 Add to List Share

There are `n` cities labeled from `1` to `n`. You are given the integer `n` and an array `connections` where `connections[i] = [`x_i`, `y_i`, `$cost_i$`]` indicates that the cost of connecting city x_i and city y_i (bidirectional connection) is $cost_i$.

Return *the minimum **cost** to connect all the* `n` *cities such that there is at least one path between each pair of cities*. If it is impossible to connect all the `n` cities, return `-1`,

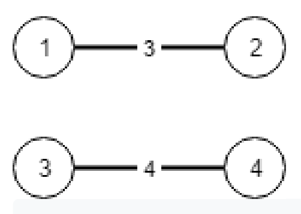
The **cost** is the sum of the connections' costs used.

Example 1:



```
Input: n = 3, connections = [[1,2,5],[1,3,6],[2,3,1]]
Output: 6
Explanation: Choosing any 2 edges will connect all cities so we choose the minimum 2.
```

Example 2:



```
Input: n = 4, connections = [[1,2,3],[3,4,4]]
Output: -1
Explanation: There is no way to connect all cities even if all edges are used.
```

Constraints:

- `1 <= n <=` 10^4
- `1 <= connections.length <=` 10^4
- `connections[i].length == 3`
- `1 <=` x_i, y_i `<= n`
- x_i `!=` y_i
- `0 <=` $cost_i$ `<=` 10^5

Accepted 45,034 | Submissions 74,892

Seen this question in a real interview before? Yes No

Companies i

0 ~ 6 months 6 months ~ 1 year 1 year ~ 2 years

Amazon | 20

Related Topics

Union Find Graph Heap (Priority Queue) Minimum Spanning Tree

Hide Hint 1

What if we model the cities as a graph?

Hide Hint 2

Build a graph of cities and find the minimum spanning tree.

Hide Hint 3

You can use a variation of the Kruskal's algorithm for that.

Hide Hint 4

Sort the edges by their cost and use a union-find data structure.

Hide Hint 5

How to check all cities are connected?

Hide Hint 6

At the beginning we have n connected components, each time we connect two components the number of connected components is reduced by one. At the end we should end with only a single component otherwise return -1.

Java Autocomplete

```java
class Solution {
    public int minimumCost(int n, int[][] connections) {
        
    }
}
```