1724. Checking Existence of Edge Length Limited Paths II Premium

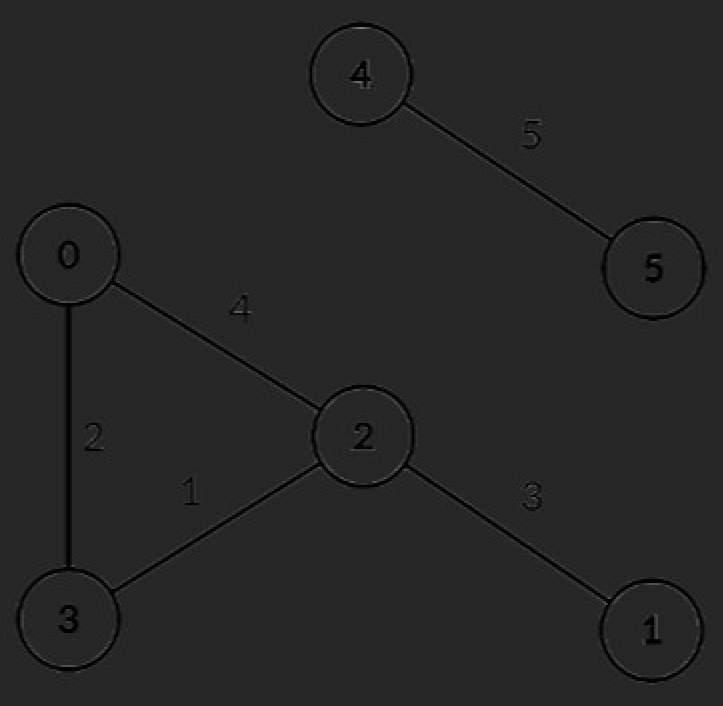
Hard Topics Companies Hint

An undirected graph of `n` nodes is defined by `edgeList`, where `edgeList[i] =` $[u_i, v_i, dis_i]$ denotes an edge between nodes u_i and v_i with distance dis_i. Note that there may be **multiple** edges between two nodes, and the graph may not be connected.

Implement the `DistanceLimitedPathsExist` class:

- `DistanceLimitedPathsExist(int n, int[][] edgeList)` Initializes the class with an undirected graph.
- `boolean query(int p, int q, int limit)` Returns `true` if there exists a path from `p` to `q` such that each edge on the path has a distance **strictly less than** `limit`, and otherwise `false`.

Example 1:



```
Input
["DistanceLimitedPathsExist", "query", "query", "query", "query"]
[[6, [[0, 2, 4], [0, 3, 2], [1, 2, 3], [2, 3, 1], [4, 5, 5]]], [2, 3, 2], [1, 3, 3], [2, 0, 3], [0, 5, 6]]
Output
[null, true, false, true, false]

Explanation
DistanceLimitedPathsExist distanceLimitedPathsExist = new DistanceLimitedPathsExist(6, [[0, 2, 4], [0, 3, 2], [1, 2, 3], [2, 3, 1], [4, 5, 5]]);
distanceLimitedPathsExist.query(2, 3, 2); // return true. There is an edge from 2 to 3 of distance 1, which is less than 2.
distanceLimitedPathsExist.query(1, 3, 3); // return false. There is no way to go from 1 to 3 with distances strictly less than 3.
distanceLimitedPathsExist.query(2, 0, 3); // return true. There is a way to go from 2 to 3 to 0 with distance < 3: travel from 2 to 3 to 0.
distanceLimitedPathsExist.query(0, 5, 6); // return false. There are no paths from 0 to 5.
```

Constraints:

- $2 <= n <= 10^4$
- $0 <=$ `edgeList.length` $<= 10^4$
- `edgeList[i].length == 3`
- $0 <= u_i, v_i, p, q <= n-1$
- $u_i != v_i$
- `p != q`
- $1 <= dis_i, limit <= 10^9$
- At most 10^4 calls will be made to `query`.

Seen this question in a real interview before? 1/5

Yes No

Accepted 3.2K | Submissions 6.2K | Acceptance Rate 50.9%

Topics

Union Find Graph Minimum Spanning Tree

Companies

0 - 6 months

Google 2

Hint 1

Find the minimum spanning tree of the given graph.

Hint 2

Root the tree in an arbitrary node and calculate the maximum weight of the edge from each node to the chosen root.

Hint 3

To answer a query, find the lca between the two nodes, and find the maximum weight from each of the query nodes to their lca and compare it to the given limit.

Similar Questions

Checking Existence of Edge Length Limited Paths	Hard
Number of Good Paths	Hard
Minimum Score of a Path Between Two Cities	Medium

Discussion (5)