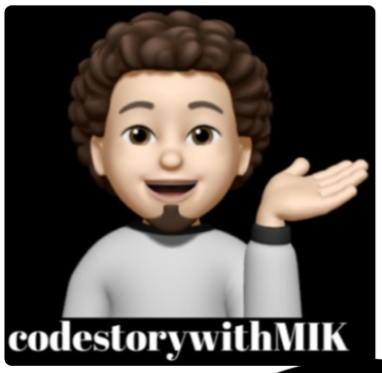
Hash Map/Set

video - 22



Leetcode-
1733

Medium

codestorywithmik

CSwithMIK

codestorywithMIK

+ Tech ...

Motivation :-

You still have few months left this year. Make the most of it. 2-3 months can make you an entirely different person if you stay consistent.

MIK...

1733. Minimum Number of People to Teach

Medium | Topics | Companies | Hint

On a social network consisting of `m` users and some friendships between users, two users can communicate with each other if they know a common language.

You are given an integer `n`, an array `languages`, and an array `friendships` where:

- There are `n` languages numbered `1` through `n`,
- `languages[i]` is the set of languages the i^{th} user knows, and
- `friendships[i] = [u_i, v_i]` denotes a friendship between the users u_i and v_i.

You can choose **one** language and teach it to some users so that all friends can communicate with each other. Return *the* ***minimum*** *number of users you need to teach.*

Note that friendships are not transitive, meaning if `x` is a friend of `y` and `y` is a friend of `z`, this doesn't guarantee that `x` is a friend of `z`.

(x, y) (y, z)

Example 1:

u1 u2 u3

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

Explanation: You can either teach user 1 the second language or user 2 the first language.

Example 2:

Input: n = 3, languages = [[2],[1,3],[1,2],[3]], friendships = [[1,4],[1,2],[3,4],[2,3]]
Output: 2
Explanation: Teach the third language to users 1 and 3, yielding two users to teach.

Constraints:

- `2 <= n <= 500`
- `languages.length == m`
- `1 <= m <= 500`
- `1 <= languages[i].length <= n`
- `1 <= languages[i][j] <= n`
- `1 <= u_i < v_i <= languages.length`
- `1 <= friendships.length <= 500`
- All tuples `(u_i, v_i)` are unique
- `languages[i]` contains only unique values

Thought Process

n = 3

languages = [(2), (1, 3), (1, 2), (3)]

user = 1, 2, 3, 4

friendships = [(1,4), (1,2), (3,4), (2,3)]

lang.

u = 2 → {1, 3}

v = 3 → {1, 2}

sadUsers = { 1, 4, 2, 3 }

lang. | count of user.

2 → 1+1

3 → 1+1

1 → 1+1

4 − 2 = 2

Teach language 2

* One language pick such that I have to teach minimum users.

→ (User 1, User 2) → Hindi / English → sad

→ (User 1, User 3) → Hindi / English → sad

2 users taught

User 2 → Hindi

User 3 → Hindi

User 1 → English.

1 user teach

sadUsers = { user1, user2, user3 }

Hindi → User 1

English → user2, user3

maxTalkedLangu = English. (2)

3 users — 2 users

= 1 user (user1).

=

SadUsers.length() — mostTalkedLa..

=

Story to code

(1) Find SadUsers { } → (ui, uj)

(2) Check most known language among sadUsers.

(3) sadUsers.length() — mostKnownLanguageCount;