GRAPHS...

video-44

"let's make it easy too"

Hard

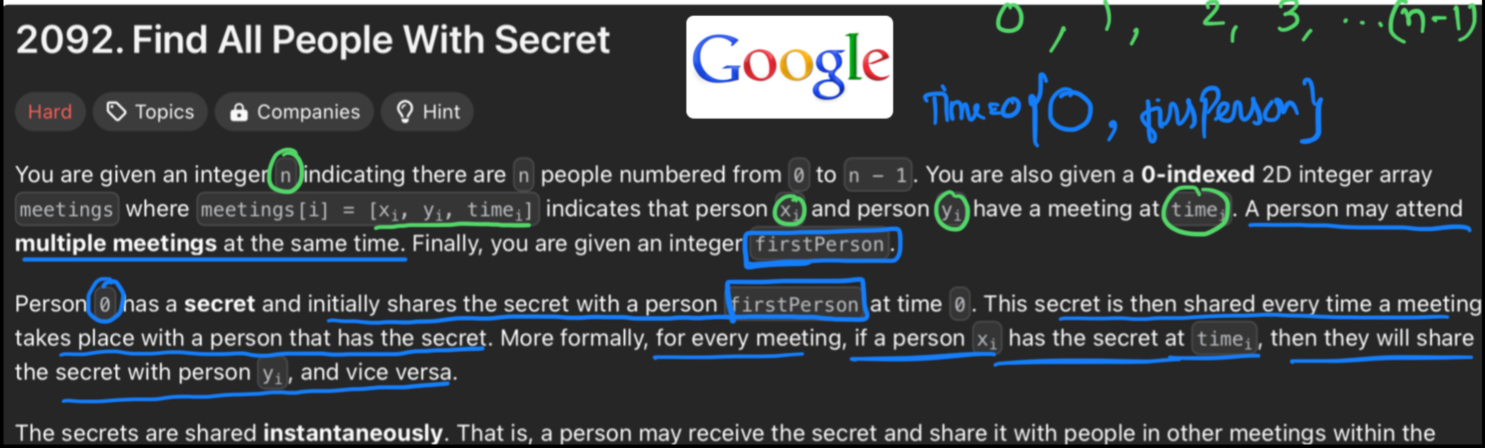
Leetcode 2092

If you have tried my "Graph Concepts & Qns" playlist, these Qns, will seem very easy. Do try it once ;)

Facebook, Instagram → codestorywithMIK

(Twitter) → CSwithMIK

codestorywithMIK → WhatsApp



0, 1, 2, 3, ...(n-1)

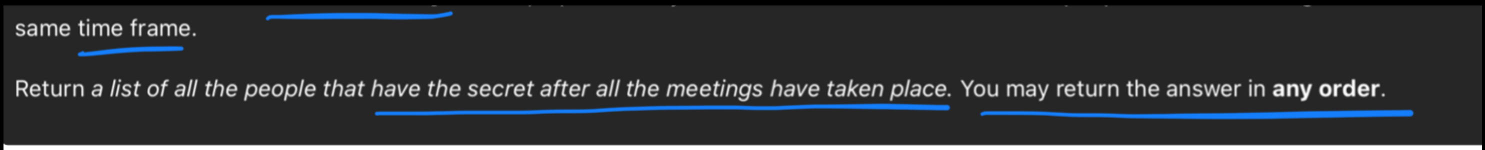
Time=0 {0, firstPerson}

2092. Find All People With Secret

You are given an integer `n` indicating there are `n` people numbered from `0` to `n - 1`. You are also given a **0-indexed** 2D integer array `meetings` where `meetings[i]` = [x_i, y_i, $time_i$] indicates that person x_i and person y_i have a meeting at $time_i$. A person may attend **multiple meetings** at the same time. Finally, you are given an integer `firstPerson`.

Person `0` has a **secret** and initially shares the secret with a person `firstPerson` at time `0`. This secret is then shared every time a meeting takes place with a person that has the secret. More formally, for every meeting, if a person x_i has the secret at $time_i$, then they will share the secret with person y_i, and vice versa.

The secrets are shared **instantaneously**. That is, a person may receive the secret and share it with people in other meetings within the



same time frame.

Return a *list of all the people that have the secret after all the meetings have taken place.* You may return the answer in **any order**.

Example :- n = 6

0 1 2

p1 p2 t p2 p3 t p1 p5 t

meetings = [(1, 2, 5), (2, 3, 8), (1, 5, 10)]

firstPerson = 1

Person.

time = 0 → 0, 1

time = 5 → 2

time = 8 → 3

t = 10 → 5

Output :- {0, 1, 2, 3, 5}

n = 5

0 1

meetings = {(4, 2, 5) (3, 2, 2)}

firstPer = 3

Time = 0

0, 3

2, 4

① Sort array in Asc. of meetings.

Most Basic Thought

n = 6

meetings = [(1, 2, 5), (2, 3, 8), (1, 5, 10)]

firstPerson = 1

set

0, 1

2, 3

5

Time :	0	5
Person :	0	

1

n = 5

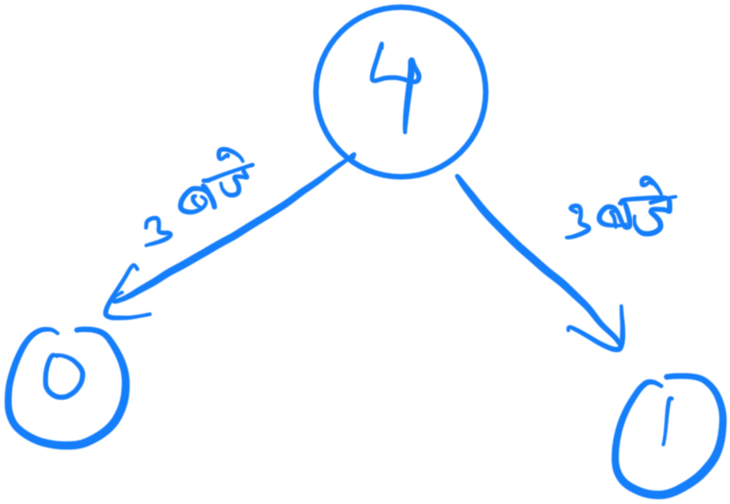
if time equal.

[(1, 4, 3) , (0, 4, 3)]

firstPerson = 3

Time :	0	3
Person :	0 3	

0, 3
4, 1



time → m1, m2, ...

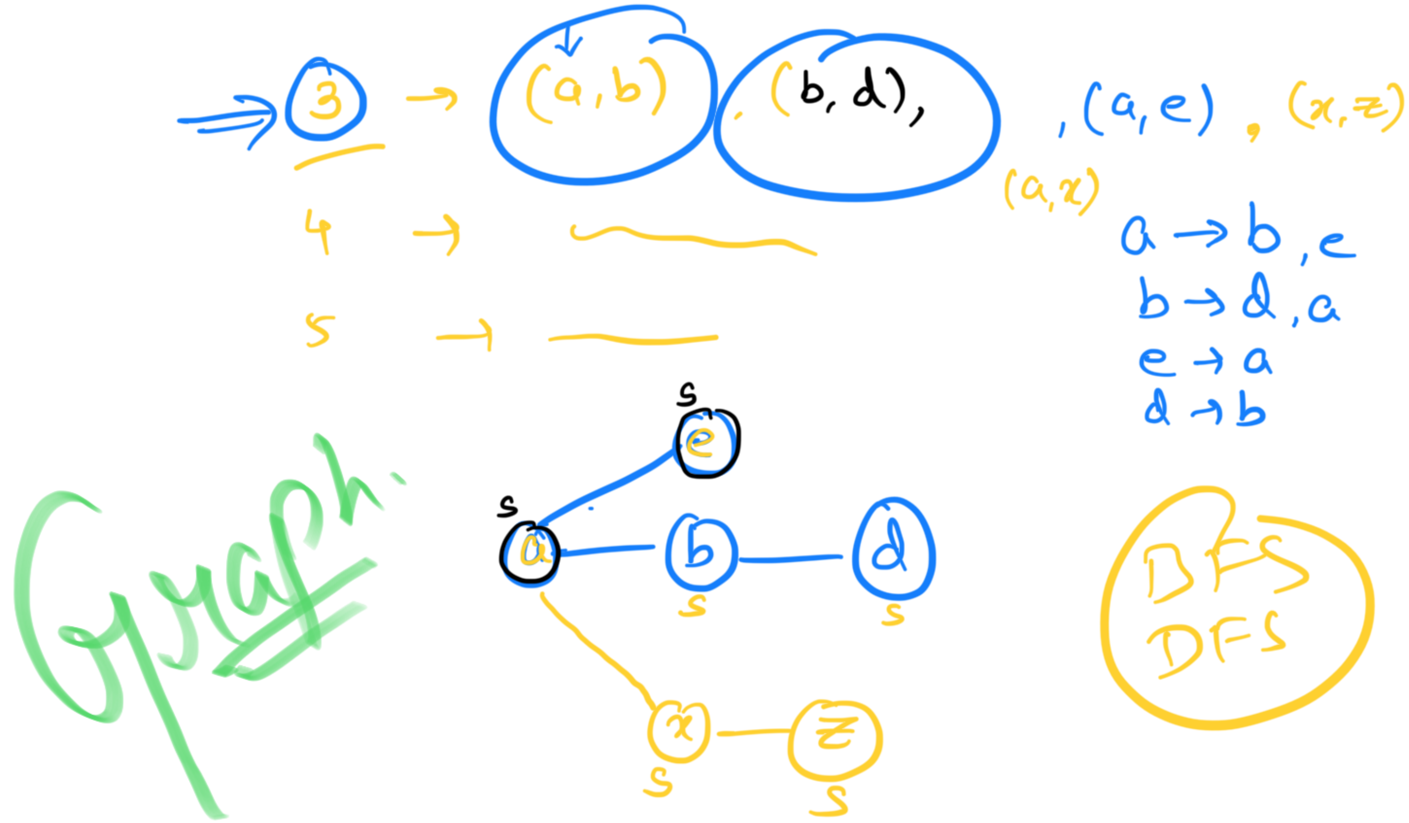
Key

{ (P1, P2), (P1, P4), (P5, P6) }

time → meetings   map ordered

- time → meetings

- ~~Sort based on time (ascending)~~



-: Story Points :-

P = pair<int, int>

① time → meetings (ordered map)

```
map < int, vector< P >> timeMeetings ;
KnowsSecret [0] = True; KnowsSecret[firstPer] = True;
```

② Traverse in timeMeetings map.

3 → {(P1, P2), (P1, P3), (P4, P5)}

↳ Make a graph (adjList) ⇐ P1 → P2 , P3
P2 → P1
P3 → P1

↳ if Person (Pi) KnowsSecret (que.add (P1)).

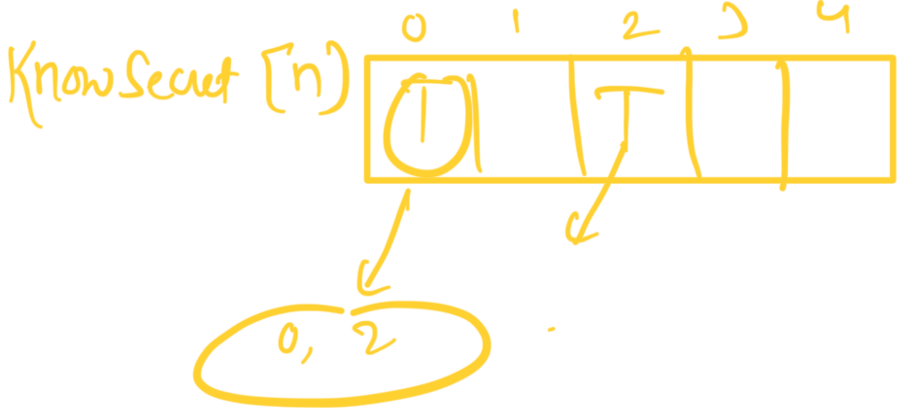
↳ Avoid adding duplicate person.

Graph.

③ Now simply do BFS traversal.

```
while (!que.empty) {
        Person = que.front();
        que.pop();

        for ( nextPers : adj [Pers]) {
            if( KnowsSecret [nextPi] == False) {
                que.push (nextPer).
                KnowsSecret [nextPer] = True;
            }
        }
    }
}
```



Time & Space:-

tim → metn

for (timemeetin) //O(M)

{

//BFS → $O(\underset{\text{meeti.}}{M} + \underset{\text{no of Pers.}}{N})$

}

$\approx O(M * (M+N))$

Space :- $O(M+N)$

O(M)

O(N)   KnowsSecr

Approach - 2 ...

Problem with Approach-1 :-

time → meetings map
meetings → adj
adj → BFS

n = 6

meetings = [(1, 2, 5), (2, 3, 8), (1, 5, 10)]

firstPerson = 1

adj

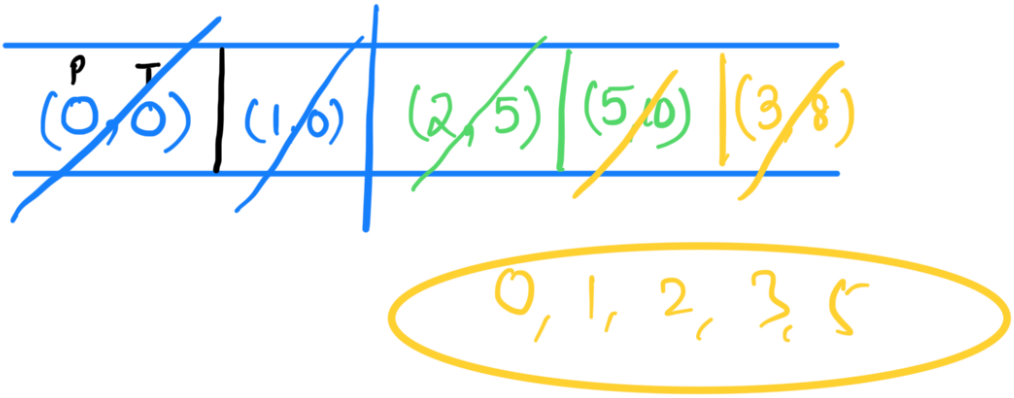
Person		P T
1	→	(2, 5) , (5, 10)
2	→	(1, 5) , (3, 8)
3	→	(2, 8)
5	→	(1, 10)

KnowsSecret

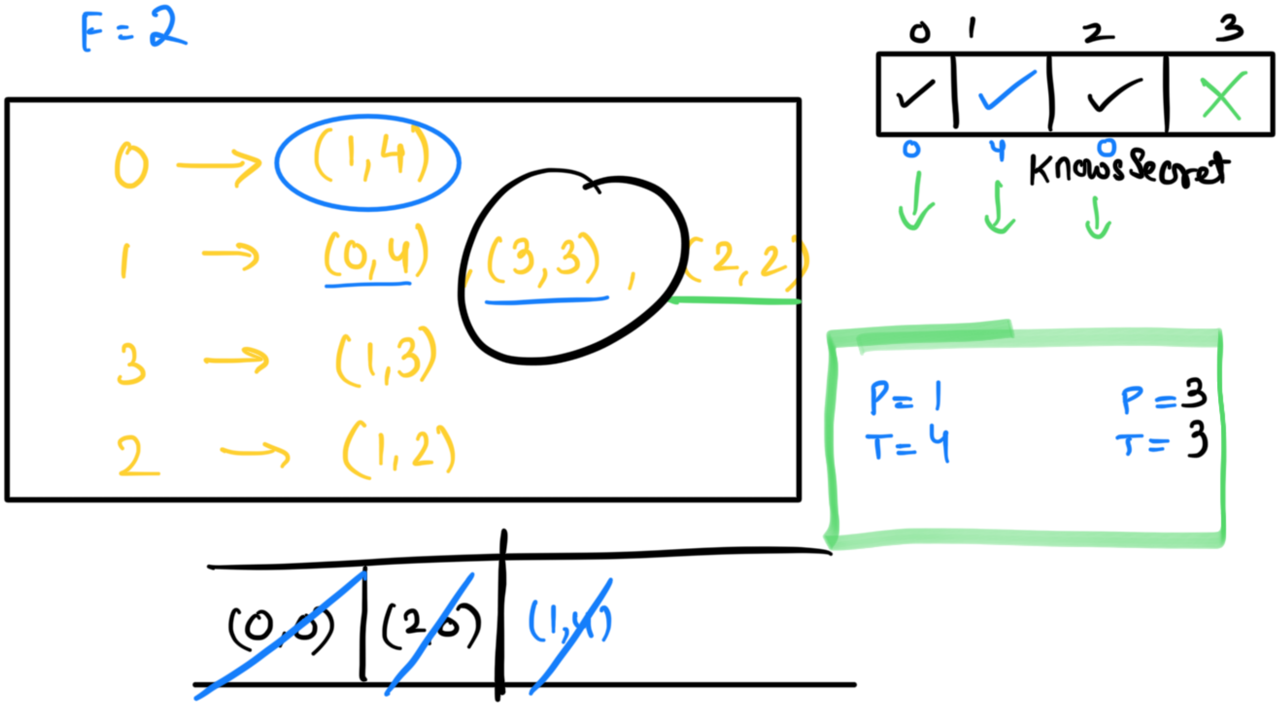
0	1	2	3	4	5
✓	✓	✓	✓	X	✓

P = 3
T = 8

P = 2
T = 8



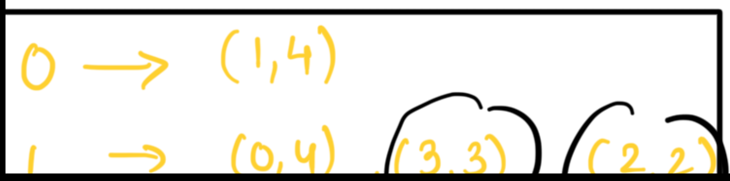
Example:



n=4

meetings = [{0, 1, 4}, {1, 3, 3}, {2, 1, 2}]

F = 2



Knows Secret
0 1 2 3
(0)
~~(4)~~
(2)
(2)
3

3 → (1,3)

2 → (1,2)

P= 1
T= 2

P=
T=

~~(0,0)~~	~~(2,0)~~	~~(1,4)~~	~~(1,2)~~	~~(3,3)~~	~~(2,2)~~

Story Points:-

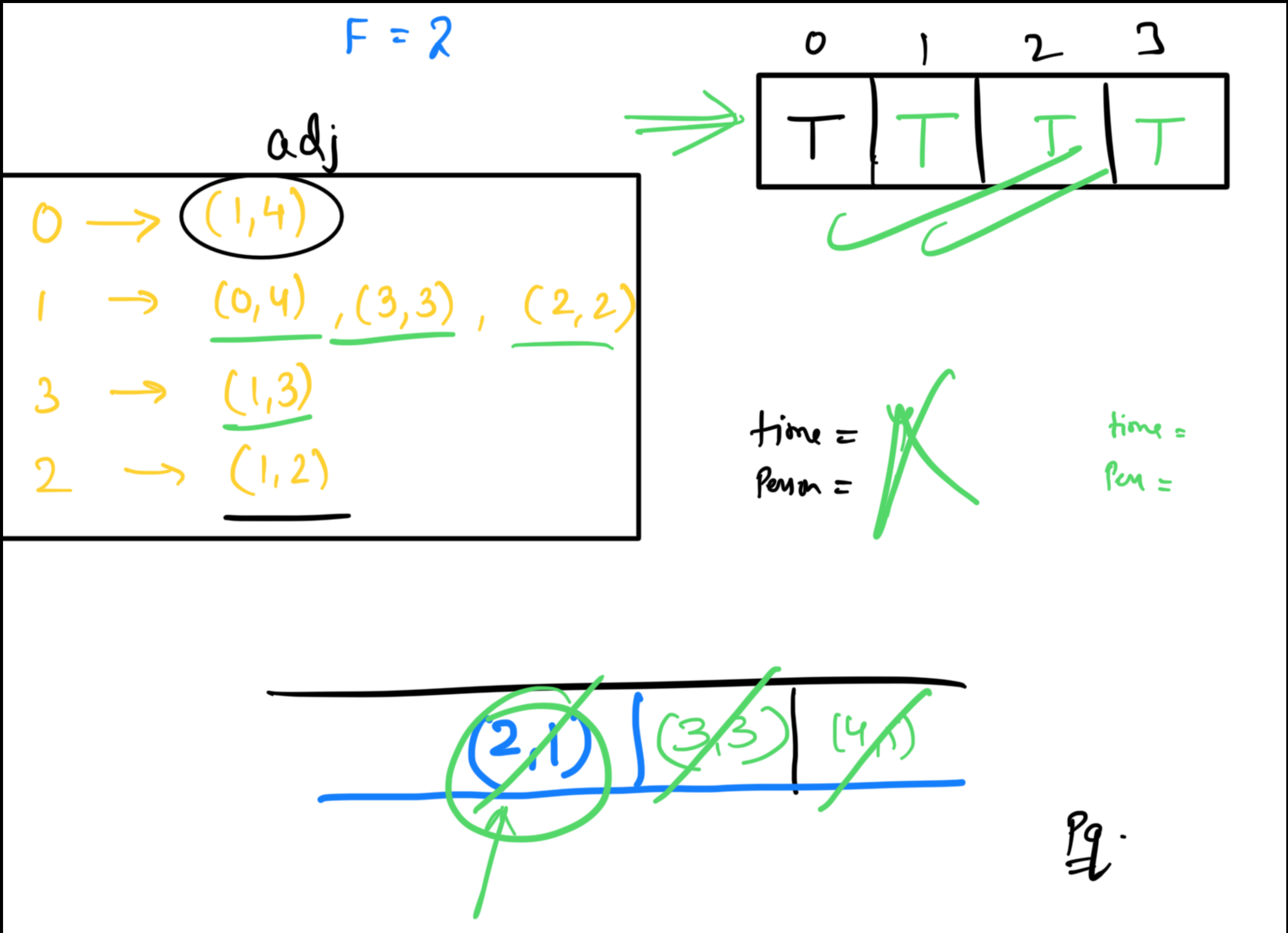
(·) meetings से ही adj बनालो।

Approach-4 :-

" min-heap"

n=4

meetings = [{0, 1, 4}, {1, 3, 3}, {2, 1, 2}]



↳ pq < time, person> (min-heap)