Array : Video - 57

E.A.S.Y

Leetcode -1337

The Weakest Rows in a Matrix

Facebook
Instagram → codestorywithMIK

(Twitter) → CSwithMIK

Company :- amazon

(Different Problem Statement)

1337. The K Weakest Rows in a Matrix

Easy 3457 208 Add to List Share

You are given an `m x n` binary matrix `mat` of `1`'s (representing soldiers) and `0`'s (representing civilians). The soldiers are positioned **in front** of the civilians. That is, all the `1`'s will appear to the **left** of all the `0`'s in each row.

A row `i` is **weaker** than a row `j` if one of the following is true:

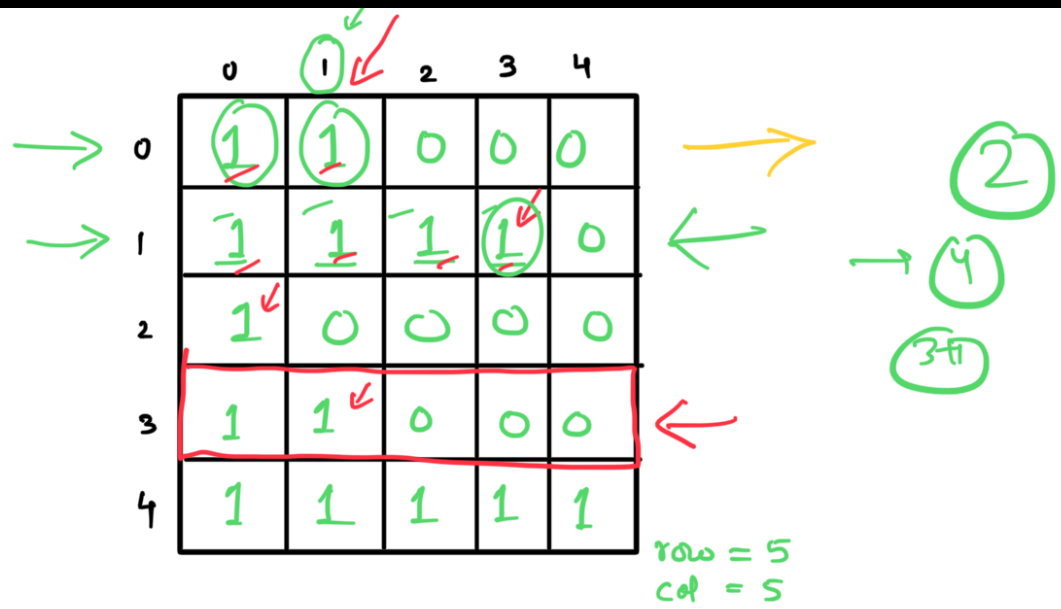
- The number of soldiers in row `i` is less than the number of soldiers in row `j`.
- Both rows have the same number of soldiers and `i < j`.

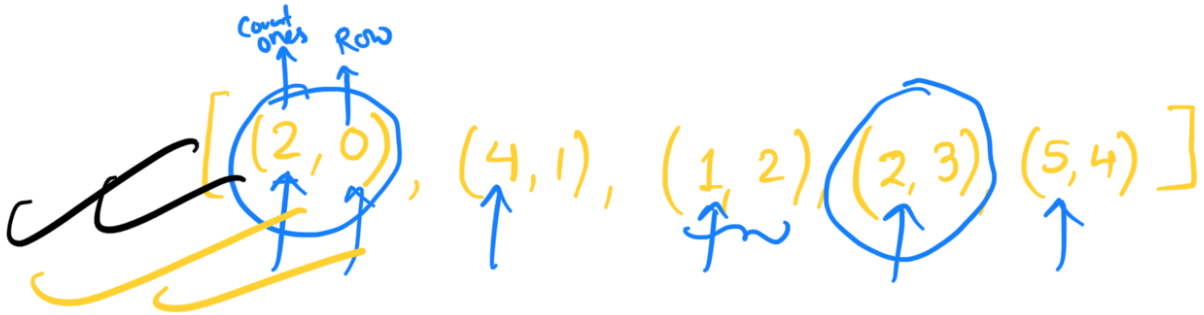
Return *the indices of the* `k` ***weakest*** *rows in the matrix ordered from weakest to strongest.*

Example 1:

```
Input: mat =
[[1,1,0,0,0],
 [1,1,1,1,0],
 [1,0,0,0,0],
 [1,1,0,0,0],
 [1,1,1,1,1]],
k = 3
Output: [2,0,3]
Explanation:
The number of soldiers in each row is:
- Row 0: 2
- Row 1: 4
- Row 2: 1
- Row 3: 2
- Row 4: 5
The rows ordered from weakest to strongest are [2,0,3,1,4].
```

Intuition :-





$$cat = [(1,2), (2,0), (2,3), (4,1), (5,4)]$$

Sorting

Counting 1s :-

1. Linearly count 1s.
2. Binary search in each row

Approach-1

$O(m * \log(n))$

```
for(int row=0; row<m; row++) {
    int count_of_one = binarySearch(grid[row]);
    countOnes.push_back(count_one, row);
}
```

```
//O(mlogm)
Sort(begin(countOnes), end(c-a)); //ascend.
// O(K)
→ Choose first K elements.
}
```

T.C:- $O(m \log n) + O(m \log m) + O(K)$

S.C:- $O(m)$

m = row, n = col

1, 1, 1, 1, 0

result = 3

r

l

Count = (result + 1) = 4

Approach 2 ...

Intuition समझो।

K weakest row indices

(weakest to str)

(Heap).

min-heap

max-heap

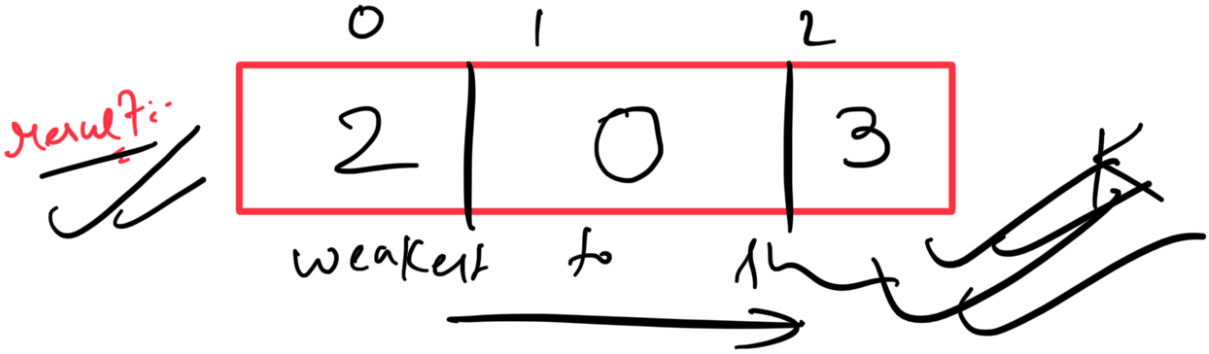
K = 3

0 1 2 3 4

0	1	1	0	0	0	← (2, 0)
1	1	1	1	1	0	← (4, 1)
2	1	0	0	0	0	← (1, 2)
3	1	1	0	0	0	← (2, 3)
4	1	1	1	1	1	← (5, 4)

Push/Pop = log(K)

(1, 2)



① row → O(m)

↳ Binary search → log(n) → O(m log(n))

↳ log(K) → + O(m log(K))

② O(K)

T.C: O(m log(n)) + O(m log(K)) + O(K)

Space: ∴ O(K).