

Problem 1337: The K Weakest Rows in a Matrix

Problem Information

Difficulty: Easy

Acceptance Rate: 74.18%

Paid Only: No

Tags: Array, Binary Search, Sorting, Heap (Priority Queue), Matrix

Problem Description

You are given an $m \times n$ 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]`.

Example 2:

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

Constraints:

* `m == mat.length` * `n == mat[i].length` * `2 <= n, m <= 100` * `1 <= k <= m` * `matrix[i][j]` is either 0 or 1.

Code Snippets

C++:

```
class Solution {
public:
    vector<int> kWeakestRows(vector<vector<int>>& mat, int k) {

    }
};
```

Java:

```
class Solution {
    public int[] kWeakestRows(int[][] mat, int k) {

    }
}
```

Python3:

```
class Solution:
    def kWeakestRows(self, mat: List[List[int]], k: int) -> List[int]:
```