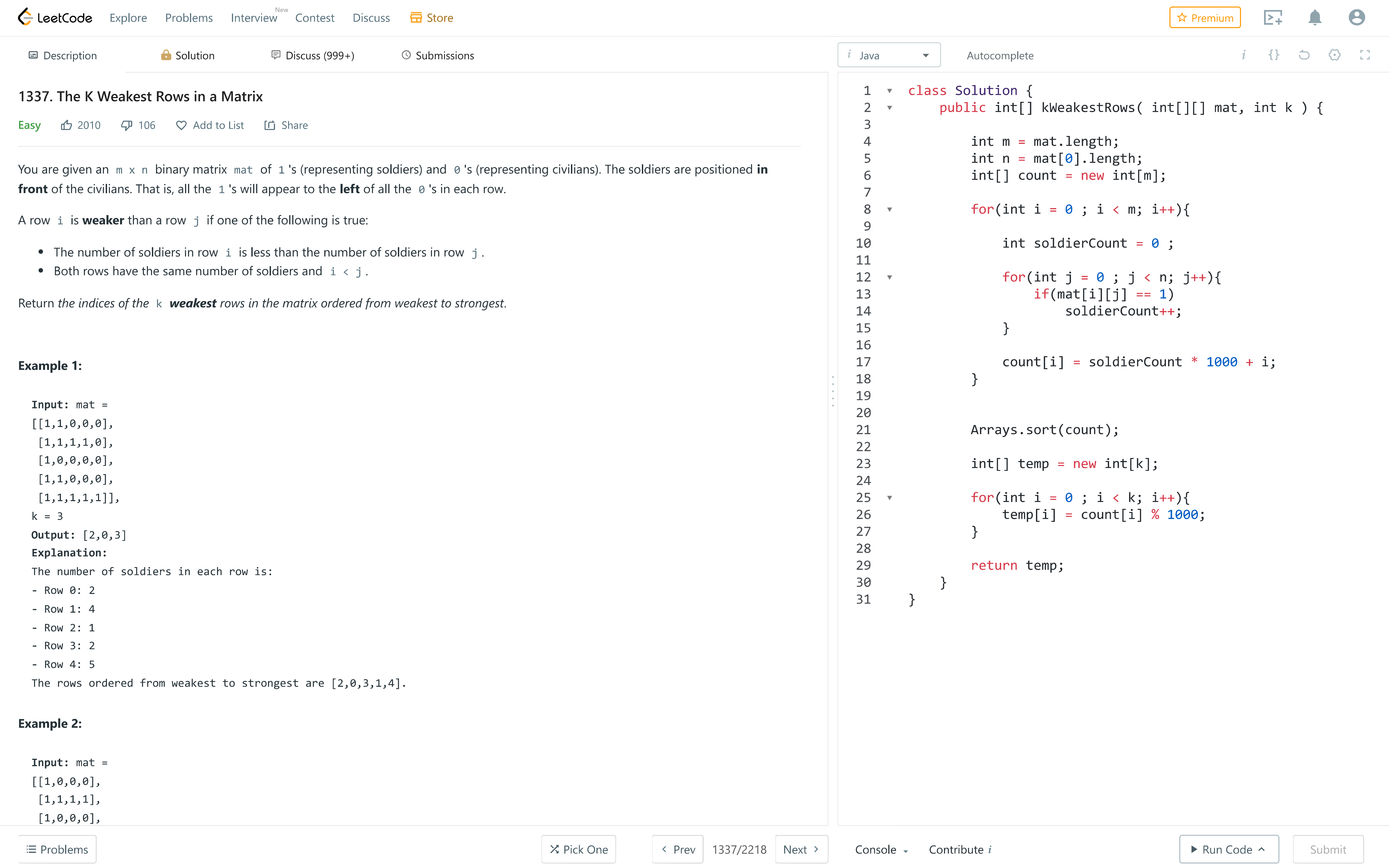

1337. The K Weakest Rows in a Matrix

Easy 2010 106 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].
```

Example 2:

```
Input: mat =
[[1,0,0,0],
 [1,1,1,1],
 [1,0,0,0],
```

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

        int m = mat.length;
        int n = mat[0].length;
        int[] count = new int[m];

        for(int i = 0 ; i < m; i++){

            int soldierCount = 0 ;

            for(int j = 0 ; j < n; j++){
                if(mat[i][j] == 1)
                    soldierCount++;
            }

            count[i] = soldierCount * 1000 + i;
        }


        Arrays.sort(count);

        int[] temp = new int[k];

        for(int i = 0 ; i < k; i++){
            temp[i] = count[i] % 1000;
        }

        return temp;
    }
}
```