The K Weakest Rows in a Matrix - LC : 1337

Code :

import java.util.Arrays;

public class KWeakestRows {

    static int[] kWeakest(int [][]mat, int k) {

        int m = mat.length;

        int n = mat[0].length;

        int pairs[][] = new int[m][2];

        for(int i = 0; i < m; i++) {

            int strength = 0;

            for(int j = 0; j < n; j++) {

                if(mat[i][j] == 0) {

                    break;

                }

                strength++;

            }

            pairs[i][0] = strength;

            pairs[i][1] = i;

        }

        Arrays.sort(pairs, (a,b) -> {

            if(a[0] == b[0]) return a[1] - b[1];

            else return a[0] - b[0];

        });

        int []indexes = new int[k];

        for(int i = 0; i < k; i++) {

            indexes[i] = pairs[i][1];

        }

        return indexes;

    }

    public static void main(String[] args) {

        int 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}};

        int k = 3;

        int output[] = kWeakest(mat, k);

        for(int i = 0; i < output.length; i++) {

            System.out.println(output[i]);

        }

    }}

**Sum of upper and lower triangles**

class Solution  
{  
    //Function to return sum of upper and lower triangles of a matrix.  
    static ArrayList<Integer> sumTriangles(int matrix[][], int n)  
    {  
        // code herem  
        int sum=0;  
        int res=0;  
        ArrayList<Integer> al=new ArrayList<Integer>();  
        for(int i=0;i<n;i++){  
            for(int j=i;j<n;j++){  
                sum+=matrix[i][j];  
                res+=matrix[j][i];  
            }  
        }  
        al.add(sum);  
        al.add(res);  
        return al;  
    }  
}