

Problem 3462: Maximum Sum With at Most K Elements

Problem Information

Difficulty: Medium

Acceptance Rate: 60.27%

Paid Only: No

Tags: Array, Greedy, Sorting, Heap (Priority Queue), Matrix

Problem Description

You are given a 2D integer matrix `grid` of size `n x m`, an integer array `limits` of length `n`, and an integer `k`. The task is to find the **maximum sum** of **at most** `k` elements from the matrix `grid` such that:

* The number of elements taken from the `ith` row of `grid` does not exceed `limits[i]`.

Return the **maximum sum**.

Example 1:

Input: grid = [[1,2],[3,4]], limits = [1,2], k = 2

Output: 7

Explanation:

* From the second row, we can take at most 2 elements. The elements taken are 4 and 3. *
The maximum possible sum of at most 2 selected elements is `4 + 3 = 7`.

Example 2:

Input: grid = [[5,3,7],[8,2,6]], limits = [2,2], k = 3

Output: 21

****Explanation:****

* From the first row, we can take at most 2 elements. The element taken is 7. * From the second row, we can take at most 2 elements. The elements taken are 8 and 6. * The maximum possible sum of at most 3 selected elements is `7 + 8 + 6 = 21`.

****Constraints:****

```
* `n == grid.length == limits.length` * `m == grid[i].length` * `1 <= n, m <= 500` * `0 <= grid[i][j] <= 105` * `0 <= limits[i] <= m` * `0 <= k <= min(n * m, sum(limits))`
```

Code Snippets

C++:

```
class Solution {
public:
    long long maxSum(vector<vector<int>>& grid, vector<int>& limits, int k) {
        }
};
```

Java:

```
class Solution {
    public long maxSum(int[][][] grid, int[] limits, int k) {
        }
}
```

Python3:

```
class Solution:
    def maxSum(self, grid: List[List[int]], limits: List[int], k: int) -> int:
```