Problem 2732: Find a Good Subset of the Matrix

Problem Information

Difficulty: Hard
Acceptance Rate: 46.54%
Paid Only: No
Tags: Array, Hash Table, Bit Manipulation, Matrix

Problem Description

You are given a **0-indexed** `m x n` binary matrix `grid`.

Let us call a **non-empty** subset of rows **good** if the sum of each column of the subset is at most half of the length of the subset.

More formally, if the length of the chosen subset of rows is `k`, then the sum of each column should be at most `floor(k / 2)`.

Return _an integer array that contains row indices of a good subset sorted in**ascending** order._

If there are multiple good subsets, you can return any of them. If there are no good subsets, return an empty array.

A **subset** of rows of the matrix `grid` is any matrix that can be obtained by deleting some (possibly none or all) rows from `grid`.

Example 1:

Input: grid = [[0,1,1,0],[0,0,0,1],[1,1,1,1]] **Output:** [0,1] **Explanation:** We can choose the 0th and 1st rows to create a good subset of rows. The length of the chosen subset is 2. - The sum of the 0th column is 0 + 0 = 0, which is at most half of the length of the subset. - The sum of the 1st column is 1 + 0 = 1, which is at most half of the length of the subset. - The sum of the 2nd column is 1 + 0 = 1, which is at most half of the length of the subset. - The sum of the 3rd column is 0 + 1 = 1, which is at most half of the length of the subset.

Example 2:

Input: grid = [[0]] **Output:** [0] **Explanation:** We can choose the 0th row to create a good subset of rows. The length of the chosen subset is 1. - The sum of the 0th column is 0, which is at most half of the length of the subset.

Example 3:

Input: grid = [[1,1,1],[1,1,1]] **Output:** [] **Explanation:** It is impossible to choose any subset of rows to create a good subset.

Constraints:

* `m == grid.length` * `n == grid[i].length` * `1 <= m <= 104` * `1 <= n <= 5` * `grid[i][j]` is either `0` or `1`.

Code Snippets

C++:

```
class Solution {
public:
vector<int> goodSubsetofBinaryMatrix(vector<vector<int>>& grid) {

}
};
```

Java:

```
class Solution {
public List<Integer> goodSubsetofBinaryMatrix(int[][] grid) {

}
}
```

Python3:

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