Problem 1293: Shortest Path in a Grid with Obstacles Elimination

Problem Information

Difficulty: Hard
Acceptance Rate: 45.91%
Paid Only: No
Tags: Array, Breadth-First Search, Matrix

Problem Description

You are given an `m x n` integer matrix `grid` where each cell is either `0` (empty) or `1` (obstacle). You can move up, down, left, or right from and to an empty cell in **one step**.

Return _the minimum number of**steps** to walk from the upper left corner _`(0, 0)`_to the lower right corner_`(m - 1, n - 1)`_given that you can eliminate**at most** _`k` _obstacles_. If it is not possible to find such walk return `-1`.

Example 1:

![](https://assets.leetcode.com/uploads/2021/09/30/short1-grid.jpg)

Input: grid = [[0,0,0],[1,1,0],[0,0,0],[0,1,1],[0,0,0]], k = 1 **Output:** 6 **Explanation:** The shortest path without eliminating any obstacle is 10. The shortest path with one obstacle elimination at position (3,2) is 6. Such path is (0,0) -> (0,1) -> (0,2) -> (1,2) -> (2,2) -> **(3,2)** -> (4,2).

Example 2:

![](https://assets.leetcode.com/uploads/2021/09/30/short2-grid.jpg)

Input: grid = [[0,1,1],[1,1,1],[1,0,0]], k = 1 **Output:** -1 **Explanation:** We need to eliminate at least two obstacles to find such a walk.

Constraints:

* `m == grid.length` * `n == grid[i].length` * `1 <= m, n <= 40` * `1 <= k <= m * n` * `grid[i][j]` is either `0` **or** `1`. * `grid[0][0] == grid[m - 1][n - 1] == 0`

Code Snippets

C++:

```
class Solution {
public:
int shortestPath(vector<vector<int>>& grid, int k) {

}
};
```

Java:

```
class Solution {
public int shortestPath(int[][] grid, int k) {

}
}
```

Python3:

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