Problem 1219: Path with Maximum Gold

Problem Information

Difficulty: Medium
Acceptance Rate: 68.23%
Paid Only: No
Tags: Array, Backtracking, Matrix

Problem Description

In a gold mine `grid` of size `m x n`, each cell in this mine has an integer representing the amount of gold in that cell, `0` if it is empty.

Return the maximum amount of gold you can collect under the conditions:

* Every time you are located in a cell you will collect all the gold in that cell. * From your position, you can walk one step to the left, right, up, or down. * You can't visit the same cell more than once. * Never visit a cell with `0` gold. * You can start and stop collecting gold from **any** position in the grid that has some gold.

Example 1:

Input: grid = [[0,6,0],[5,8,7],[0,9,0]] **Output:** 24 **Explanation:** [[0,6,0], [5,8,7], [0,9,0]] Path to get the maximum gold, 9 -> 8 -> 7.

Example 2:

Input: grid = [[1,0,7],[2,0,6],[3,4,5],[0,3,0],[9,0,20]] **Output:** 28 **Explanation:** [[1,0,7], [2,0,6], [3,4,5], [0,3,0], [9,0,20]] Path to get the maximum gold, 1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7.

Constraints:

* `m == grid.length` * `n == grid[i].length` * `1 <= m, n <= 15` * `0 <= grid[i][j] <= 100` * There are at most **25** cells containing gold.

Code Snippets

C++:

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

}
};
```

Java:

```
class Solution {
public int getMaximumGold(int[][] grid) {

}
}
```

Python3:

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