1765. Map of Highest Peak

Description

You are given an integer matrix `isWater` of size `m x n` that represents a map of **land** and **water** cells.

- If `isWater[i][j] == 0`, cell `(i, j)` is a **land** cell.
- If `isWater[i][j] == 1`, cell `(i, j)` is a **water** cell.

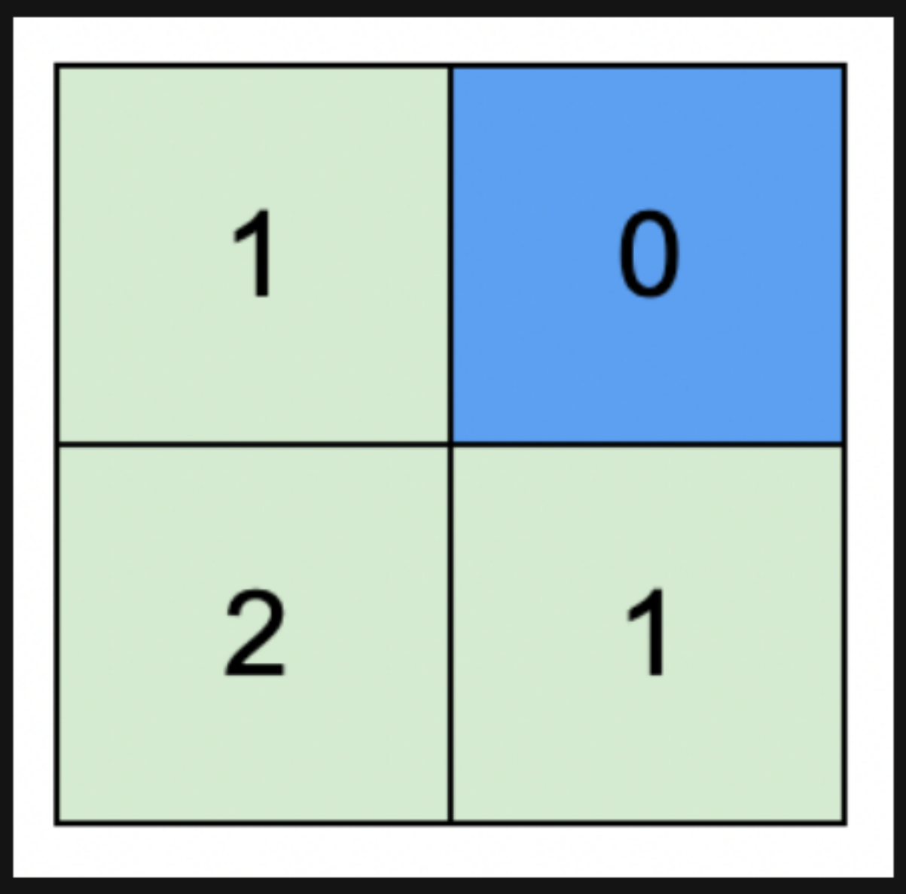
You must assign each cell a height in a way that follows these rules:

- The height of each cell must be non-negative.
- If the cell is a **water** cell, its height must be `0`.
- Any two adjacent cells must have an absolute height difference of **at most** `1`. A cell is adjacent to another cell if the former is directly north, east, south, or west of the latter (i.e., their sides are touching).

Find an assignment of heights such that the maximum height in the matrix is **maximized**.

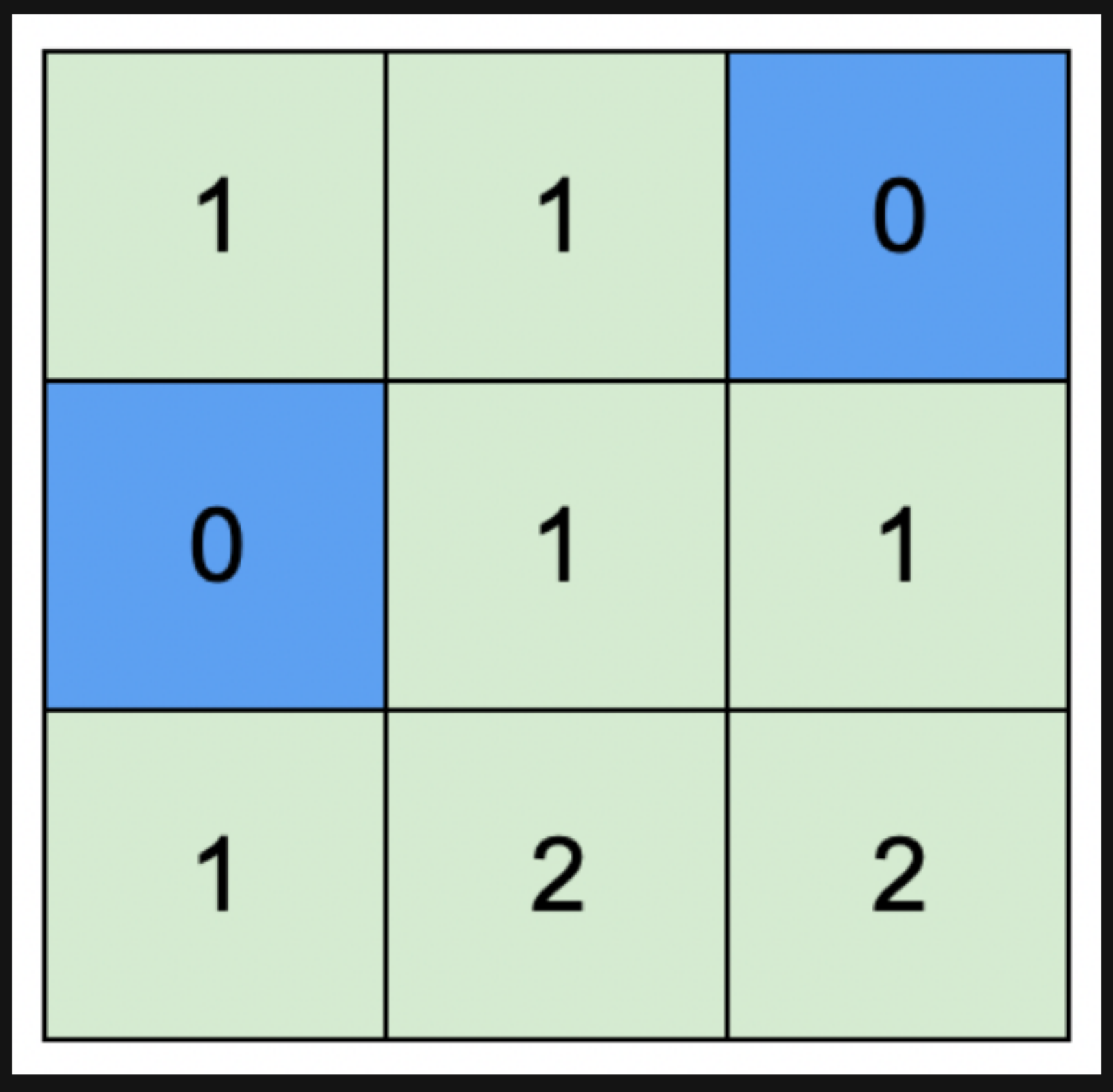
Return *an integer matrix* `height` *of size* `m x n` *where* `height[i][j]` *is cell* `(i, j)` *'s height. If there are multiple solutions, return* ***any*** *of them*.

Example 1:



```
Input: isWater = [[0,1],[0,0]]
Output: [[1,0],[2,1]]
Explanation: The image shows the assigned heights of each cell.
The blue cell is the water cell, and the green cells are the land cells.
```

Example 2:



```
Input: isWater = [[0,0,1],[1,0,0],[0,0,0]]
Output: [[1,1,0],[0,1,1],[1,2,2]]
Explanation: A height of 2 is the maximum possible height of any assignment.
Any height assignment that has a maximum height of 2 while still meeting the rules will also be accepted.
```

Constraints:

- `m == isWater.length`
- `n == isWater[i].length`
- `1 <= m, n <= 1000`
- `isWater[i][j]` is `0` or `1`.
- There is at least **one** water cell.