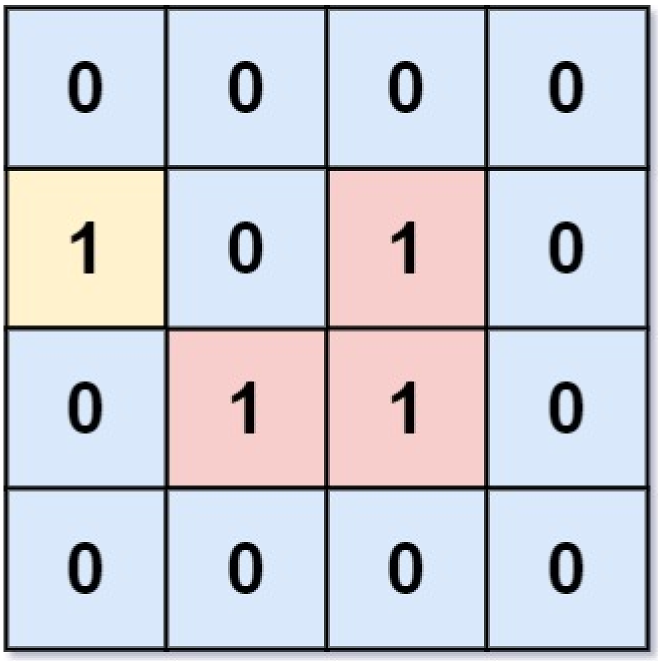
1020. Number of Enclaves

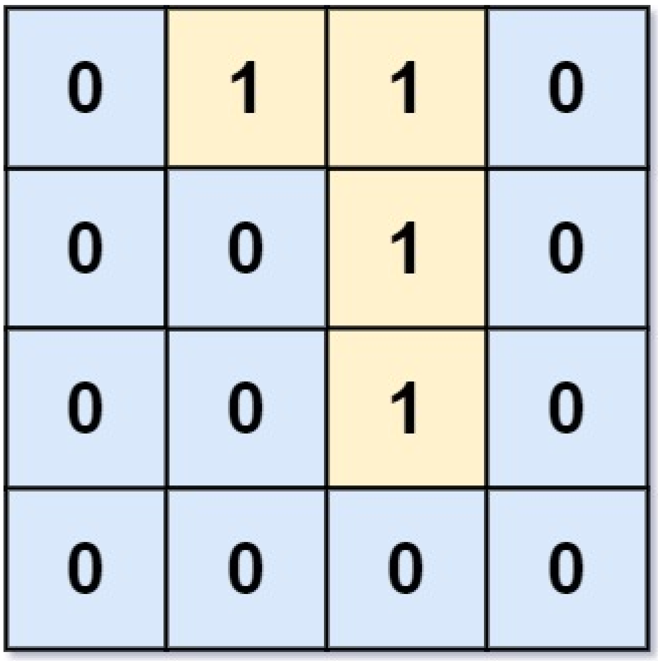
You are given an `m x n` binary matrix `grid`, where `0` represents a sea cell and `1` represents a land cell.

A move consists of walking from one land cell to another adjacent (4-directionally) land cell or walking off the boundary of the `grid`.

Return *the number of land cells in* `grid` *for which we cannot walk off the boundary of the grid in any number of moves*.



```
Input: grid = [[0,0,0,0],[1,0,1,0],[0,1,1,0],[0,0,0,0]]
Output: 3
Explanation: There are three 1s that are enclosed by 0s, and one 1 that is
not enclosed because its on the boundary.
```



~~~
Input: grid = [[0,1,1,0],[0,0,1,0],[0,0,1,0],[0,0,0,0]]
Output: 0
Explanation: All 1s are either on the boundary or can reach the boundary.
``````Python
class Solution:
    def numEnclaves(self, grid: List[List[int]]) -> int:
        for i in range(len(grid)):
            for j in range(len(grid[0])):
                if i==0 or j==0 or i==len(grid)-1 or j==len(grid[0])-1:
                    if grid[i][j]==1:
                        self.dfs(grid,i,j)

        count = 0
        for i in range(len(grid)):
            for j in range(len(grid[0])):
                if grid[i][j]==1:
                    count =count+1
        return count



    def dfs(self,grid,r,c):
        if r<0 or c<0 or r>=len(grid) or c>=len(grid[0]) or grid[r][c]==0:
            return

        grid[r][c]=0
        self.dfs(grid,r-1,c)
        self.dfs(grid,r,c+1)
~~~

```
        self.dfs(grid,r+1,c)
        self.dfs(grid,r,c-1)
```