1219. Path with Maximum Gold

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.
```

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

        maxgold = 0
        for i in range(len(grid)):
            for j in range(len(grid[0])):
                if grid[i][j]!=0:
                    visited = [[False for j in range(len(grid[0]))] for i
in range(len(grid))]
                    temp = self.collectgold(grid,i,j,visited)
                    if temp>maxgold:
                            maxgold = temp
        return maxgold

    def collectgold(self,grid,i,j,visited):
        if i<0 or j<0 or i>=len(grid) or j>=len(grid[0]) or  visited[i][j]
== True or grid[i][j]==0:
            return 0
        visited[i][j] = True
        up = self.collectgold(grid,i-1,j,visited)
        east=self.collectgold(grid,i,j+1,visited)
        west = self.collectgold(grid,i,j-1,visited)
        south =   self.collectgold(grid,i+1,j,visited)
        visited[i][j] = False
        return (grid[i][j] + max(up,max(east,max(west,south))))
```