54. Sort the Matrix Diagonally

A matrix diagonal is a diagonal line of cells starting from some cell in either the topmost row or leftmost column and going in the bottom-right direction until reaching the matrix's end. For example, the matrix diagonal starting from mat[2][0], where mat is a 6 x 3 matrix, includes cells mat[2][0], mat[3][1], and mat[4][2]. Given an m x n matrix mat of integers, sort each matrix diagonal in ascending order and return the resulting matrix.

Example 1: Input: mat = [[3,3,1,1],[2,2,1,2],[1,1,1,2]] Output: [[1,1,1,1],[1,2,2,2],[1,2,3,3]]

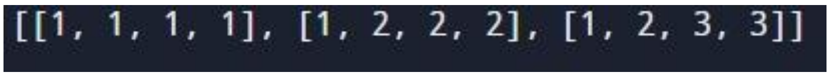
Example 2: Input: mat = [[11,25,66,1,69,7],[23,55,17,45,15,52],[75,31,36,44,58,8],[22,27,33,25,68,4],[84,28,14,11,5,50]] Output: [[5,17,4,1,52,7],[11,11,25,45,8,69],[14,23,25,44,58,15],[22,27,31,36,50,66],[84,28,75,33,55,68]]

AIM: To sort the Matrix Diagonally

PROGRAM:

```
def diagonalSort(mat):
    m, n = len(mat), len(mat[0])
    def collect_diagonal(i, j):
        diagonal = []
        while i < m and j < n:
            diagonal.append(mat[i][j])
            i += 1
            j += 1
        return diagonal
    def put_diagonal(diagonal, i, j):
        idx = 0
        while i < m and j < n:
            mat[i][j] = diagonal[idx]
            i += 1
            j += 1
            idx += 1
    for i in range(m):
        diagonal = collect_diagonal(i, 0)
        diagonal.sort()
        put_diagonal(diagonal, i, 0)
    for j in range(1, n):
        diagonal = collect_diagonal(0, j)
        diagonal.sort()
        put_diagonal(diagonal, 0, j)
    return mat
mat1 = [[3, 3, 1, 1], [2, 2, 1, 2], [1, 1, 1, 2]]
mat2 = [[11, 25, 66, 1, 69, 7], [23, 55, 17, 45, 15, 52], [75, 31, 36, 44, 58, 8],
        [22, 27, 33, 25, 68, 4], [84, 28, 14, 11, 5, 50]]


print(diagonalSort(mat1))
print(diagonalSort(mat2))
```

OUTPUT: 

```
[[1, 1, 1, 1], [1, 2, 2, 2], [1, 2, 3, 3]]
```

TIME COMPLEXITY: O(m * n * log(min(m, n)))