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.

Program:

```
def diagonal_sort(mat):
    from collections import defaultdict
    d = defaultdict(list)
    for i in range(len(mat)):
        for j in range(len(mat[0])):
            d[i - j].append(mat[i][j])
    for k in d:
        d[k].sort()
    for i in range(len(mat)):
        for j in range(len(mat[0])):
            mat[i][j] = d[i - j].pop(0)
    return mat

# Example usage:
mat = [[3, 3, 1, 1], [2, 2, 1, 2], [1, 1, 1, 2]]
print(diagonal_sort(mat))  # Output: [[1, 1, 1, 1], [1, 2, 2, 2], [1, 2, 3, 3]]def
diagonalSort(mat):
    from collections import defaultdict
    import heapq

    # Dictionary to hold all diagonals
    diagonals = defaultdict(list)

    # Populate the dictionary with the diagonals
    m, n = len(mat), len(mat[0])
    for i in range(m):
        for j in range(n):
            heapq.heappush(diagonals[i - j], mat[i][j])

    # Take sorted elements from the heap and put them back in the matrix
    for i in range(m):
        for j in range(n):
            mat[i][j] = heapq.heappop(diagonals[i - j])

    return mat

# Example usage
mat = [
    [3, 3, 1, 1],
    [2, 2, 1, 2],
```

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

sorted_mat = diagonalSort(mat)
print(sorted_mat)
```

Output:

```
"C:\Program Files\Python312\python.exe" "C:\Work Space\DAA COADS.PYTHON\program 54.py"
[[1, 1, 1, 1], [1, 2, 2, 2], [1, 2, 3, 3]]

Process finished with exit code 0
```

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