Problem # 498 : Diagonal Traverse (Medium)

-----------------------------------------------------------

<https://leetcode.com/problems/diagonal-traverse/>

Solution:

Runtime beats 89.90%

1. If matrix is an empty list, then return it.

2. Otherwise, set m to the number of rows and n to the number of columns.

3. Initialize an empty dictionary adict.

4. Traverse through the matrix by rows and columns using i and j. If the sum of the row index and the column index of the matrix is not in adict, then create a list with the matrix entry. If the sum of the row and column indexes are already there in adict, then append the matrix entry to the list.

5. Sort the keys in the list in ascending order and store them in a list called keys.

6. Initialize the result array res.

7. Iterate through keys and if the key is even (i.e. divisible by 2), reverse the corresponding value in the dictionary and add the value to res. However if the key is odd, then just add the value to res.

8. Return res.

class Solution:

def findDiagonalOrder(self, matrix: List[List[int]]) -> List[int]:

if not matrix:

return matrix

m = len(matrix) # number of rows

n = len(matrix[0]) # number of columns

adict = {}

for i in range(m):

for j in range(n):

if i + j not in adict.keys():

adict[i+j] = [matrix[i][j]]

else:

adict[i+j].append(matrix[i][j])

keys = sorted(adict.keys())

res = []

for key in keys:

if key % 2 == 0:

res += adict[key][::-1]

else:

res += adict[key]

return res