**Problem #2946: Matrix Similarity After Cyclic Shifts (Easy)**

<https://leetcode.com/problems/matrix-similarity-after-cyclic-shifts/description/>

**My Solution:**

**Intuition**

If k is equal number of columns then it is the same as the original matrix. So return True.

After n rotations, since matrix return to its original position, we can divide k by n and just take the remainder using modulo (i.e. k = k % n)

Construct a new matrix called newmat with the shifts and compare newmat with mat

**Approach**

Iterate through each row of the matrix mat and do the shifts to the left or right according to whether the row is even or odd.

So we go through the matrix just once to construct the new matrix after the shifts. Finally we compare the original matrix with the new matrix.

**Complexity**

Time complexity:

O(n) -- linear time since we visit each element once

Space complexity:

O(n) -- we construct a new matrix of the same size as the original matrix.

**Code**

class Solution:

def areSimilar(self, mat: List[List[int]], k: int) -> bool:

if k == len(mat[0]):

return True

m = len(mat) # number of rows

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

k = k % n # after n rotations, matrix will come to its original position

newmat = []

for i in range(m):

if i % 2 == 0: # odd index shift left

newmat.append(mat[i][k : n] + mat[i][0 : k])

else: # even index shift right

newmat.append(mat[i][n - k : n] + mat[i][0 : n - k])

return mat == newmat