**can i make a rotation without transpose.**

Yes, it is possible to rotate a square matrix 90 degrees without explicitly using the transpose operation. However, typical 90-degree rotation patterns often rely on transpose combined with row or column reversals because they follow clear positional symmetries.

If you avoid transpose, you can think about rotating the matrix layer by layer (or ring by ring), moving elements in four-way cycles. This is sometimes called rotating "in place with cyclic swaps" and involves swapping 4 elements at a time around each square "layer" of the matrix.

Here is the key insight:

* The rotation rearranges elements in groups of four, each moving to the position of the next in the cycle.
* You do this for each layer from outside to inside.
* This method does not require a separate transpose step and no direct position equivalence like transpose has.

**Example: Rotate matrix 90 degrees clockwise without transpose**

def rotate\_90\_clockwise(matrix):  
 n = len(matrix)  
 for layer in range(n // 2):  
 first = layer  
 last = n - 1 - layer  
 for i in range(first, last):  
 offset = i - first  
 # Save the top element  
 top = matrix[first][i]  
  
 # left -> top  
 matrix[first][i] = matrix[last - offset][first]  
  
 # bottom -> left  
 matrix[last - offset][first] = matrix[last][last - offset]  
  
 # right -> bottom  
 matrix[last][last - offset] = matrix[i][last]  
  
 # top -> right  
 matrix[i][last] = top  
  
# Sample data and test  
mat = [  
 [1, 2, 3],  
 [4, 5, 6],  
 [7, 8, 9]  
]  
rotate\_90\_clockwise(mat)  
print(mat)

Expected Output:

[  
 [7, 4, 1],  
 [8, 5, 2],  
 [9, 6, 3]  
]

**Remarks**

* This "layer by layer, 4-element cyclical swap" approach is a classic in-place solution for rotation without explicitly transposing and reversing rows or columns.
* It applies **only to square matrices**, because rotation changes a MxN matrix to NxM otherwise.
* There is no direct simple position pattern like transpose between original and final elements, but the algorithm methodically rotates elements in place.

This approach is common in LeetCode problem #48 "Rotate Image" but usually explained as an alternative to transpose-based rotation methods.

If needed, I can provide a similar method for 90 degrees counterclockwise rotation or for other angles as well. Let me know!

⁂

1. <https://stackoverflow.com/questions/3488691/how-to-rotate-a-matrix-90-degrees-without-using-any-extra-space>

1. <https://leetcode.com/problems/rotate-image/>

1. <https://www.geeksforgeeks.org/dsa/inplace-rotate-square-matrix-by-90-degrees/>

1. <https://www.youtube.com/watch?v=Z0R2u6gd3GU>

1. <https://takeuforward.org/data-structure/rotate-image-by-90-degree/>

1. <https://www.enjoyalgorithms.com/blog/rotate-a-matrix-by-90-degrees-in-an-anticlockwise-direction/>

1. <https://stackoverflow.com/questions/42519/how-do-you-rotate-a-two-dimensional-array>

1. <https://takeuforward.org/data-structure/rotate-matrix-anti-clockwise-by-90-degree/>

1. <https://en.wikipedia.org/wiki/Rotation_matrix>

1. <https://www.geeksforgeeks.org/dsa/rotate-matrix-elements/>