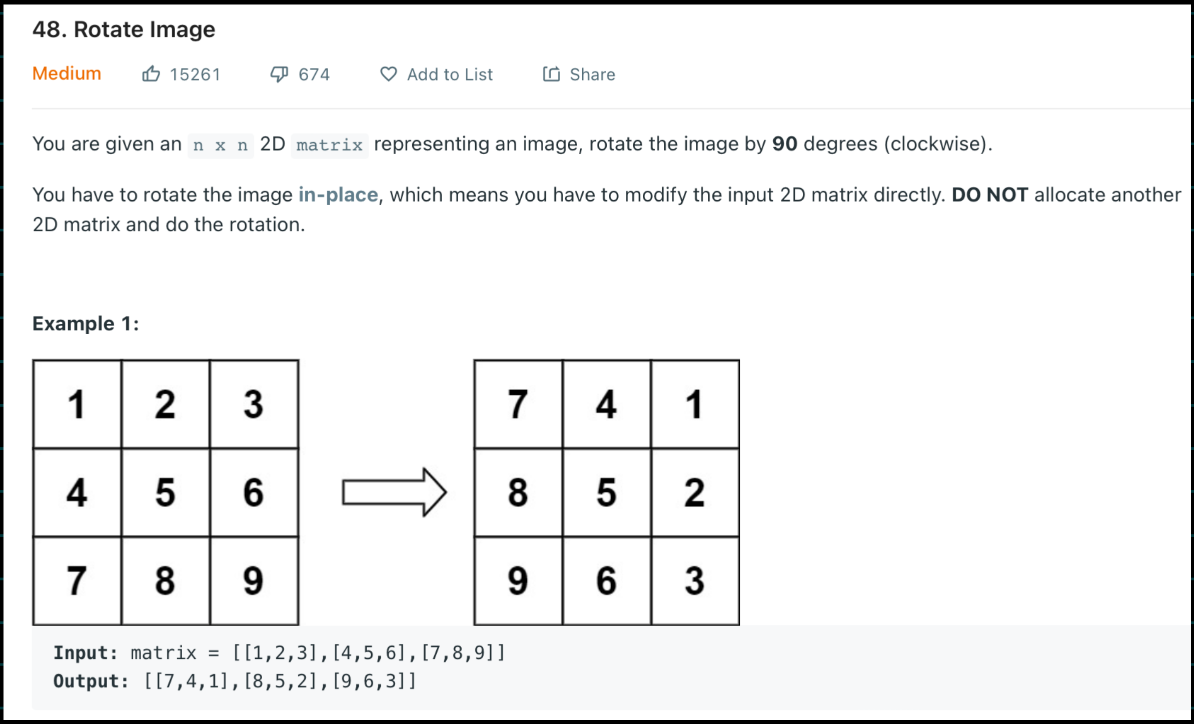
Rote the matrix or rotate image



48. Rotate Image

Medium 15261 674 Add to List Share

You are given an `n x n` 2D `matrix` representing an image, rotate the image by **90** degrees (clockwise).

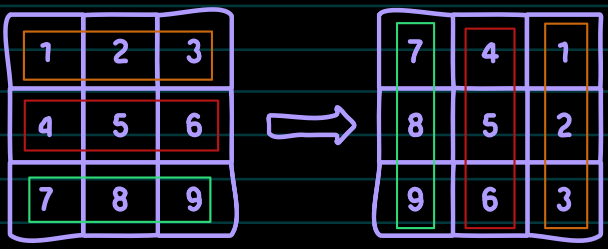
You have to rotate the image **in-place**, which means you have to modify the input 2D matrix directly. **DO NOT** allocate another 2D matrix and do the rotation.

Example 1:

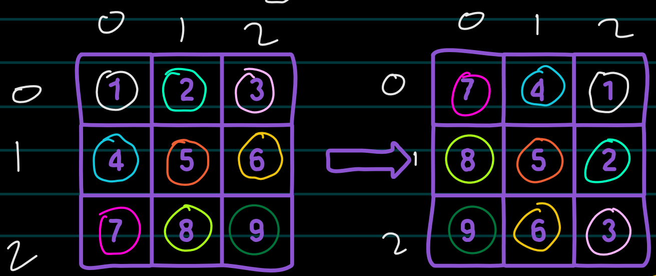
1	2	3
4	5	6
7	8	9

7	4	1
8	5	2
9	6	3

```
Input: matrix = [[1,2,3],[4,5,6],[7,8,9]]
Output: [[7,4,1],[8,5,2],[9,6,3]]
```



Brute-force



same

[0][0] → [0][2]

[0][1] → [1][2]

[0][2] → [2][2]

Constant   Constant

same

[1][0] → [0][1]

[1][1] → [1][1]

[1][2] → [2][1]

at i=2

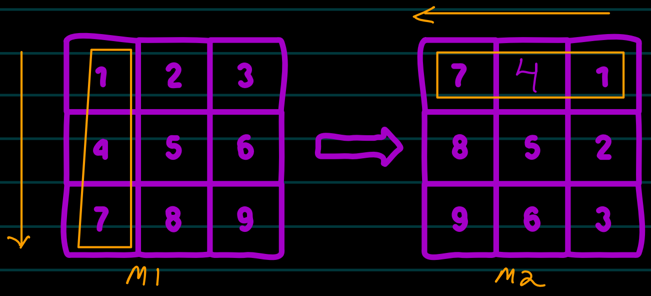
```
                                        3-2-1 => 0
  i      j                    j        (n-1)-i
                                                  → same
[2] [0]       →             [0]  [0]
[2] [1]       →             [1]  [0]
[2] [2]       →             [2]  [0]
```

```
vector<vector<int,int>> ans;
loop(i: 0 -> n)
        loop(j: 0 -> n)
        ans[j][n-i-1] = matrix[i][j]
```

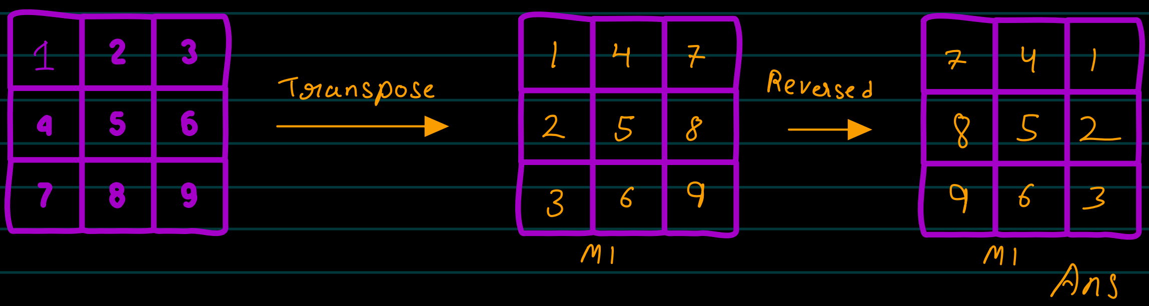
T.C - $O(n^2)$

S.C - $O(n)$

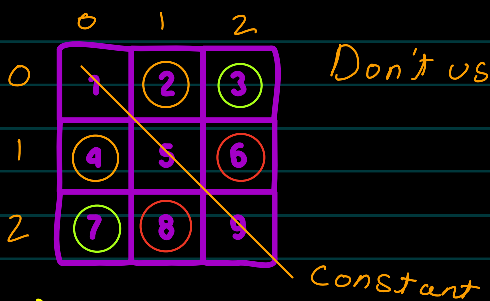
Optimal solution



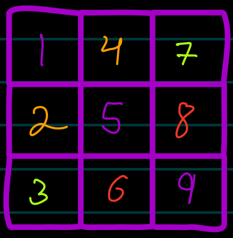
M1 row transposed is reversed in M2 column



How to transpose?



Don't use extra space means swap



swap $\overset{i}{[1]}\overset{j}{[0]} \rightarrow \overset{j}{[0]}\overset{i}{[1]}$

$[2][0] \rightarrow [0][2]$

$[1][2] \rightarrow [2][1]$

so j will always start from $1 \rightarrow n$

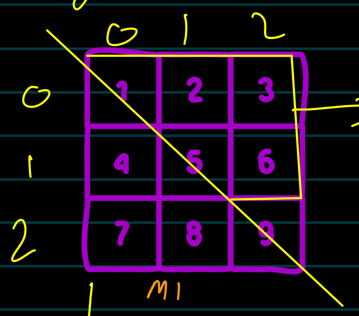
i will from $0 \rightarrow \underbrace{n-2}$

why?

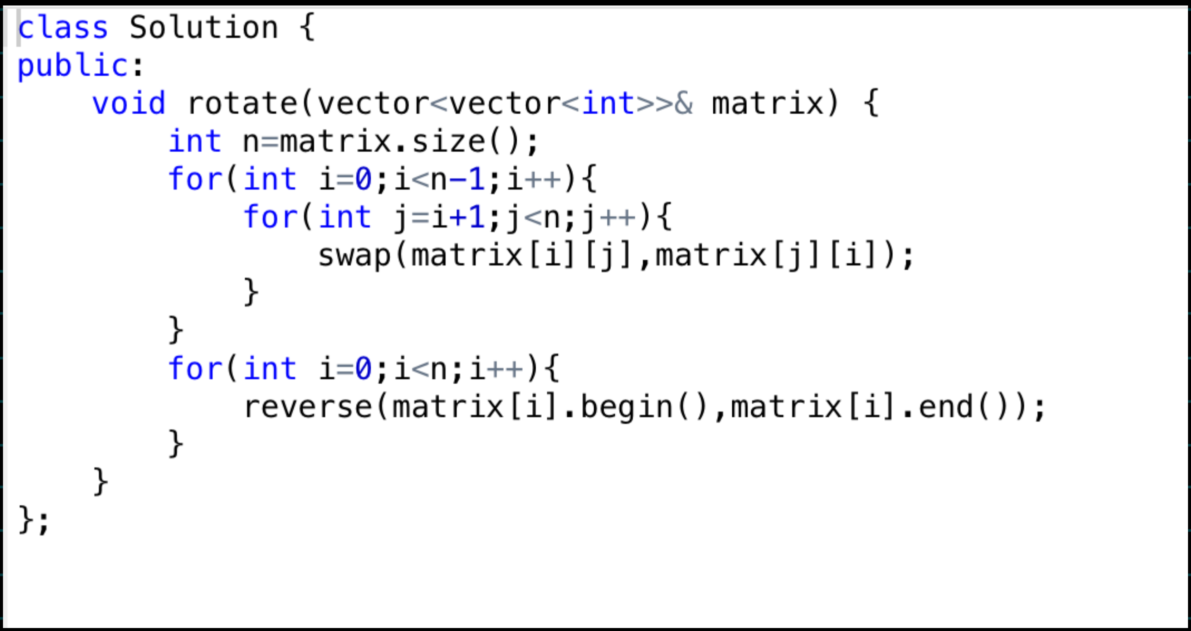
i=0, j=1 swap [j][i]

//2 [1][0]

//4



If we traverse till here we will swap it again



```
class Solution {
public:
    void rotate(vector<vector<int>>& matrix) {
        int n=matrix.size();
        for(int i=0;i<n-1;i++){
            for(int j=i+1;j<n;j++){
                swap(matrix[i][j],matrix[j][i]);
            }
        }
        for(int i=0;i<n;i++){
            reverse(matrix[i].begin(),matrix[i].end());
        }
    }
};
```