# 题目

给你一幅由N×N矩阵表示的图像，其中每个像素的大小为4字节。请你设计一种算法，将图像旋转90度。

不占用额外内存空间能否做到？

示例 1:

给定 matrix =

[

[1,2,3],

[4,5,6],

[7,8,9]

],

原地旋转输入矩阵，使其变为:

[

[7,4,1],

[8,5,2],

[9,6,3]

]

示例 2:

给定 matrix =

[

[ 5, 1, 9,11],

[ 2, 4, 8,10],

[13, 3, 6, 7],

[15,14,12,16]

],

原地旋转输入矩阵，使其变为:

[

[15,13, 2, 5],

[14, 3, 4, 1],

[12, 6, 8, 9],

[16, 7,10,11]

]

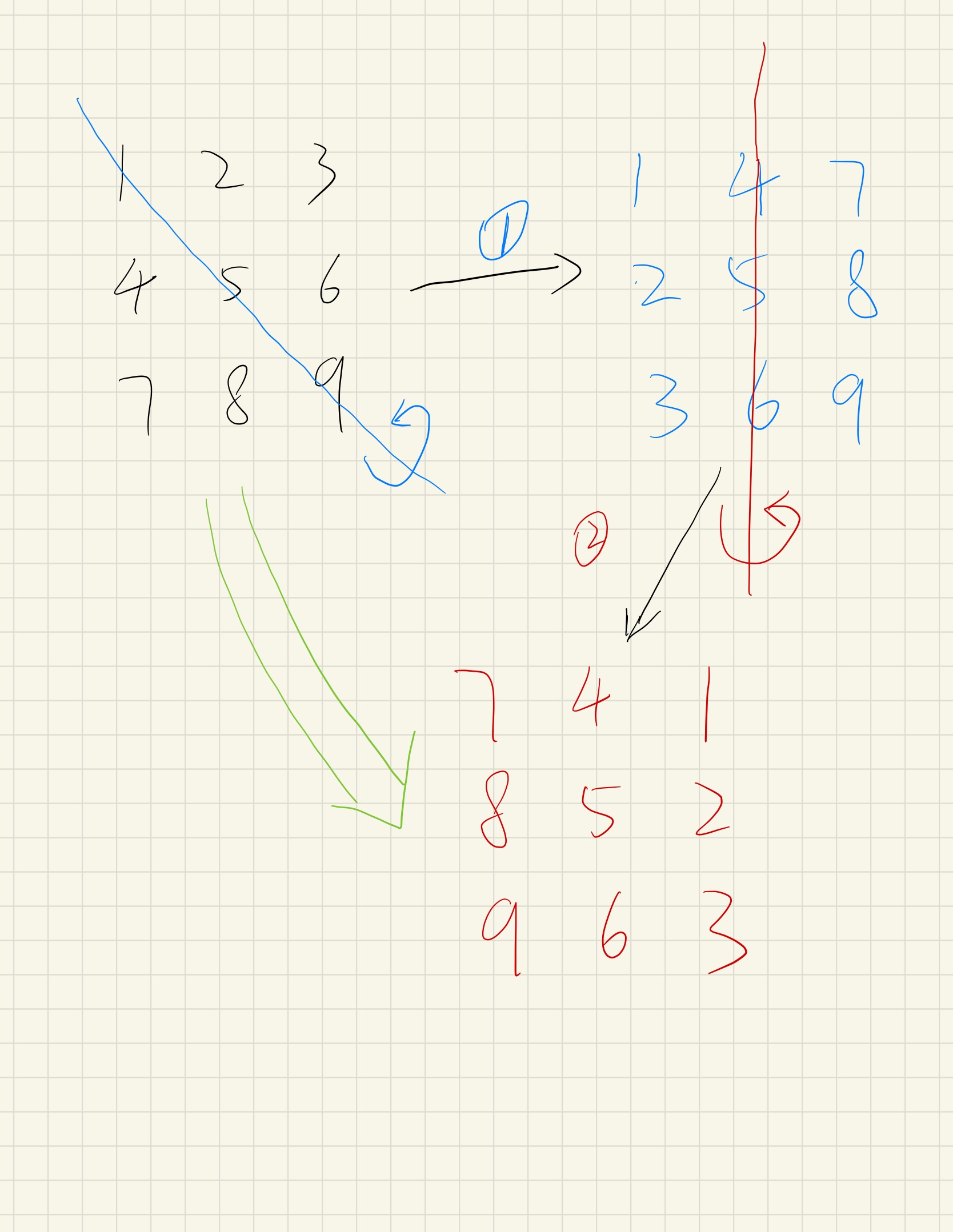
注意：本题与主站 48 题相同：https://leetcode-cn.com/problems/rotate-image/

# 分析

## 方法一：对角线+倒排

**思路：**

先对矩阵做沿主对角线的翻转（其实就是矩阵的转置），原地完成；接着再对操作后的矩阵进行根据列序号倒排。



**代码：**

class Solution {

public:

    //先按照主对角线翻转，再按列号倒排

    void rotate(vector<vector<int>>& matrix) {

        int n = matrix.size();

        for(int i=0 ; i<n ; ++i){

            for(int j = 0 ; j<i ; ++j){ //对角线

                swap(matrix[i][j], matrix[j][i]);

            }

        }

        for(int i=0 ; i<n ; ++i){

            for(int j=0 ; j<n/2 ; ++j){

                swap(matrix[i][j], matrix[i][n-1-j]);

//n-j表示后面的数字，-1表示减掉中间位置的数字

            }

        }

    }

};

## 方法二：原地旋转

class Solution {

public:

void rotate(vector<vector<int>>& matrix) {

int n = matrix.size();

for (int i = 0; i < n / 2; ++i) {

for (int j = 0; j < (n + 1) / 2; ++j) {

int temp = matrix[i][j];

matrix[i][j] = matrix[n - j - 1][i];

matrix[n - j - 1][i] = matrix[n - i - 1][n - j - 1];

matrix[n - i - 1][n - j - 1] = matrix[j][n - i - 1];

matrix[j][n - i - 1] = temp;

}

}

}

};

**复杂度分析：**

时间复杂度：O(N^2)，其中N是matrix的边长。我们需要枚举的子矩阵大小为O(N^2)×⌊(n+1)/2⌋)=O(N^2)。

空间复杂度：O(1)。为原地旋转。