给定一个 *n* × *n* 的二维矩阵表示一个图像。

将图像顺时针旋转 90 度。

说明：

你必须在**原地**旋转图像，这意味着你需要直接修改输入的二维矩阵。**请不要**使用另一个矩阵来旋转图像。

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]
]
```

思路，个人对这题不怎么会，就用了暴力解法，直接用一个二位向量存储旋转的矩阵，然后用copy函数赋值给传入的数组，该矩阵旋转有这么一个特点，列的逆序变为行，那么从行访问到列的逆序访问，动态添加即可得到旋转数组。

代码

```
class Solution {
public:
    void rotate(vector<vector<int>>& matrix) {
```

```
        vector<vector<int>> store;
        for(int i=0;i<matrix[0].size();i++){
            vector<int> line;
            for(int j=matrix.size()-1;j>=0;j--){
                line.push_back(matrix[j][i]);
            }
            store.push_back(line);
        }
        matrix=store;
    }
};
```