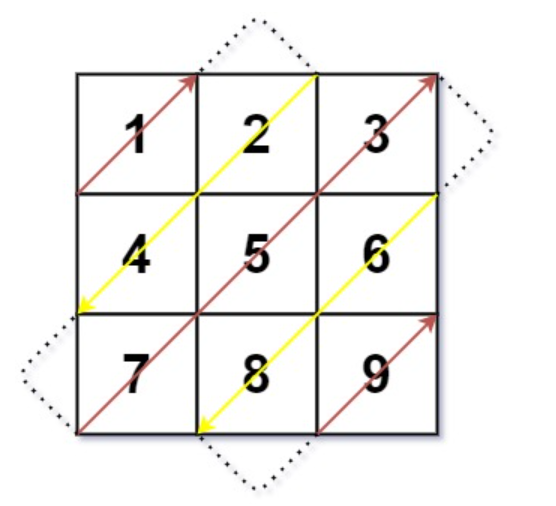
# 题目

给你一个大小为m x n的矩阵mat，请以对角线遍历的顺序，用一个数组返回这个矩阵中的所有元素。

示例 1：



输入：mat = [[1,2,3],[4,5,6],[7,8,9]]

输出：[1,2,4,7,5,3,6,8,9]

示例 2：

输入：mat = [[1,2],[3,4]]

输出：[1,2,3,4]

提示：

m == mat.length

n == mat[i].length

1 <= m, n <= 10^4

1 <= m \* n <= 10^4

-10^5 <= mat[i][j] <= 10^5

# 分析

## 方法一：模拟

思路：

可以按照对角线遍历的顺序来遍历矩阵。对于每条对角线，其元素的行索引和列索引之和是固定的。根据这个特点，可以分两种情况讨论：

当对角线的和为偶数时，按照从左上到右下的顺序遍历。

当对角线的和为奇数时，按照从右上到左下的顺序遍历。

代码：

class Solution {

public:

vector<int> findDiagonalOrder(vector<vector<int>>& mat) {

int m = mat.size();

int n = mat[0].size();

vector<int> result(m \* n);

int row = 0, col = 0;

for (int i = 0; i < m \* n; ++i) {

result[i] = mat[row][col];

if ((row + col) % 2 == 0) { // 对角线和为偶数，向下走

if (col == n - 1) { // 达到右边界，向下移动一格

++row;

} else if (row == 0) { // 达到上边界，向右移动一格

++col;

} else { // 向上移动一格，向右移动一格

--row;

++col;

}

} else { // 对角线和为奇数，向上走

if (row == m - 1) { // 达到下边界，向右移动一格

++col;

} else if (col == 0) { // 达到左边界，向下移动一格

++row;

} else { // 向下移动一格，向左移动一格

++row;

--col;

}

}

}

return result;

}

};