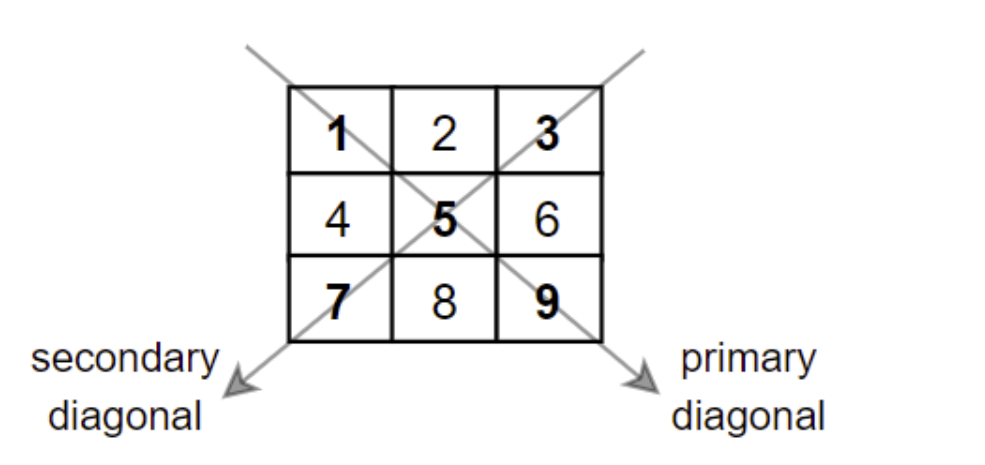
# 题目

给你一个正方形矩阵 mat，请你返回矩阵对角线元素的和。

请你返回在矩阵主对角线上的元素和副对角线上且不在主对角线上元素的和。

示例 1：



输入：mat = [[1,2,3],

[4,5,6],

[7,8,9]]

输出：25

解释：对角线的和为：1 + 5 + 9 + 3 + 7 = 25

请注意，元素 mat[1][1] = 5 只会被计算一次。

示例 2：

输入：mat = [[1,1,1,1],

[1,1,1,1],

[1,1,1,1],

[1,1,1,1]]

输出：8

示例 3：

输入：mat = [[5]]

输出：5

提示：

n == mat.length == mat[i].length

1 <= n <= 100

1 <= mat[i][j] <= 100

# 分析

## 方法一：遍历矩阵

思路：

我们知道矩阵中某个位置(i,j)处于对角线上，则一定满足下列条件之一：

i=j；

i+j=n−1；

根据上述结论，我们可以遍历整个矩阵，如果当前坐标 (i,j)满足i=j或者i+j=n−1则表示该位置一定在对角线上，则把当前的数字加入到答案之中。

代码：

class Solution {

public:

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

int n = mat.size(), sum = 0;

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

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

if (i == j || i + j == n - 1) {

sum += mat[i][j];

}

}

}

return sum;

}

};

复杂度分析

时间复杂度：O(n^2)，其中n是矩阵mat的行数。

空间复杂度：O(1)。