# 题目

矩形以列表 [x1, y1, x2, y2] 的形式表示，其中 (x1, y1) 为左下角的坐标，(x2, y2) 是右上角的坐标。矩形的上下边平行于 x 轴，左右边平行于 y 轴。

如果相交的面积为正，则称两矩形重叠。需要明确的是，只在角或边接触的两个矩形不构成重叠。

给出两个矩形 rec1 和 rec2 。如果它们重叠，返回 true；否则，返回 false 。

示例 1：

输入：rec1 = [0,0,2,2], rec2 = [1,1,3,3]

输出：true

示例 2：

输入：rec1 = [0,0,1,1], rec2 = [1,0,2,1]

输出：false

示例 3：

输入：rec1 = [0,0,1,1], rec2 = [2,2,3,3]

输出：false

提示：

rect1.length == 4

rect2.length == 4

-109 <= rec1[i], rec2[i] <= 109

rec1 和 rec2 表示一个面积不为零的有效矩形

# 方法

思路：

两个矩形不重叠的情况更容易判断，我们可以先找出所有不重叠的情形，然后取反即可得到重叠的判断结果。

对于两个矩形 rec1（[x1, y1, x2, y2]）和 rec2（[x3, y3, x4, y4]），它们不重叠的情况有以下几种：

1、rec1在rec2的左侧（rec1的右边界 <= rec2的左边界）

2、rec1在rec2的右侧（rec1的左边界 >= rec2的右边界）

3、rec1在rec2的下方（rec1的上边界 <= rec2的下边界）

4、rec1在rec2的上方（rec1的下边界 >= rec2的上边界）

如果以上四种情况都不满足，那么两个矩形一定存在面积为正的重叠部分。

代码：

class Solution {

public:

bool isRectangleOverlap(vector<int>& rec1, vector<int>& rec2) {

// 提取矩形的边界坐标

int x1 = rec1[0], y1 = rec1[1], x2 = rec1[2], y2 = rec1[3];

int x3 = rec2[0], y3 = rec2[1], x4 = rec2[2], y4 = rec2[3];

// 判断两个矩形是否不重叠

bool noOverlap = (x2 <= x3) || (x1 >= x4) || (y2 <= y3) || (y1 >= y4);

// 取反即为重叠的情况

return !noOverlap;

}

};

解释

首先提取了两个矩形的边界坐标，分别是 rec1 的左下（x1, y1）、右上（x2, y2）和 rec2 的左下（x3, y3）、右上（x4, y4）。

然后通过判断 rec1 是否在 rec2 的左、右、上、下四个方向上完全分离，来确定它们是否不重叠。

最后，将不重叠的判断结果取反，即可得到两个矩形是否存在面积为正的重叠部分。

这种方法高效且直观，时间复杂度为 O(1)，空间复杂度为 O(1)，只需进行几次简单的比较运算即可得出结果。