# 题目

给你一个二维整数数组 ranges 和两个整数 left 和 right 。每个 ranges[i] = [starti, endi] 表示一个从 starti 到 endi 的 闭区间 。

如果闭区间 [left, right] 内每个整数都被 ranges 中 至少一个 区间覆盖，那么请你返回 true ，否则返回 false 。

已知区间 ranges[i] = [starti, endi] ，如果整数 x 满足 starti <= x <= endi ，那么我们称整数x 被覆盖了。

示例 1：

输入：ranges = [[1,2],[3,4],[5,6]], left = 2, right = 5

输出：true

解释：2 到 5 的每个整数都被覆盖了：

- 2 被第一个区间覆盖。

- 3 和 4 被第二个区间覆盖。

- 5 被第三个区间覆盖。

示例 2：

输入：ranges = [[1,10],[10,20]], left = 21, right = 21

输出：false

解释：21 没有被任何一个区间覆盖。

提示：

1 <= ranges.length <= 50

1 <= starti <= endi <= 50

1 <= left <= right <= 50

# 分析

## 方法一：模拟

思路：

可以使用一个数组 covered 来记录每个整数是否被覆盖。然后遍历 ranges，对于每个区间 [start, end]，将 covered[start] 到 covered[end] 都置为 true。最后，检查从 left 到 right 的所有整数是否都被覆盖了。

代码：

class Solution {

public:

bool isCovered(vector<vector<int>>& ranges, int left, int right) {

vector<bool> covered(51, false); // 初始化为 false，范围在 1 到 50

for (auto& range : ranges) {

for (int i = range[0]; i <= range[1]; ++i) {

covered[i] = true;

}

}

for (int i = left; i <= right; ++i) {

if (!covered[i]) {

return false;

}

}

return true;

}

};

这段代码遍历了 ranges，对于每个区间 [start, end]，将 covered[start] 到 covered[end] 都置为 true。然后，它遍历从 left 到 right 的所有整数，如果发现有任何一个整数没有被覆盖，则返回 false。如果所有整数都被覆盖，则返回 true。