# 题目

给你一个整数数组 nums 。nums 中，子数组的 范围 是子数组中最大元素和最小元素的差值。

返回 nums 中 所有 子数组范围的 和 。

子数组是数组中一个连续 非空 的元素序列。

示例 1：

输入：nums = [1,2,3]

输出：4

解释：nums 的 6 个子数组如下所示：

[1]，范围 = 最大 - 最小 = 1 - 1 = 0

[2]，范围 = 2 - 2 = 0

[3]，范围 = 3 - 3 = 0

[1,2]，范围 = 2 - 1 = 1

[2,3]，范围 = 3 - 2 = 1

[1,2,3]，范围 = 3 - 1 = 2

所有范围的和是 0 + 0 + 0 + 1 + 1 + 2 = 4

示例 2：

输入：nums = [1,3,3]

输出：4

解释：nums 的 6 个子数组如下所示：

[1]，范围 = 最大 - 最小 = 1 - 1 = 0

[3]，范围 = 3 - 3 = 0

[3]，范围 = 3 - 3 = 0

[1,3]，范围 = 3 - 1 = 2

[3,3]，范围 = 3 - 3 = 0

[1,3,3]，范围 = 3 - 1 = 2

所有范围的和是 0 + 0 + 0 + 2 + 0 + 2 = 4

示例 3：

输入：nums = [4,-2,-3,4,1]

输出：59

解释：nums 中所有子数组范围的和是 59

提示：

1 <= nums.length <= 1000

-10^9<= nums[i] <= 10^9

进阶：你可以设计一种时间复杂度为 O(n) 的解决方案吗？

# 分析

## 方法一：模拟

思路：

体步骤如下：

1、初始化结果变量 result 为 0，当前最小值 minVal 和当前最大值 maxVal 分别为数组第一个元素 nums[0]。

2、从数组的第二个元素开始遍历数组：

如果当前元素 nums[i] 大于当前最大值 maxVal，则更新 maxVal 为 nums[i]。

如果当前元素 nums[i] 小于当前最小值 minVal，则更新 minVal 为 nums[i]。

计算当前子数组范围的值为 maxVal - minVal，并将其累加到 result 中。

3、返回 result。

代码：

class Solution {

public:

long long subArrayRanges(vector<int>& nums) {

long long ret = 0;

for (int i = 0; i < nums.size(); i++) {

int minVal = INT\_MAX, maxVal = INT\_MIN;

for (int j = i; j < nums.size(); j++) {

minVal = min(minVal, nums[j]);

maxVal = max(maxVal, nums[j]);

ret += maxVal - minVal;

}

}

return ret;

}

};