1658. 将 x 减到 0 的最小操作数

ITCharge 大约 2 分钟

- 标签：数组、哈希表、二分查找、前缀和、滑动窗口
- 难度：中等

题目链接

- 1658. 将 x 减到 0 的最小操作数 - 力扣

题目大意

描述：给定一个整数数组 $nums$ 和一个整数 x 。每一次操作时，你应当移除数组 $nums$ 最左边或最右边的元素，然后从 x 中减去该元素的值。请注意，需要修改数组以供接下来的操作使用。

要求：如果可以将 x 恰好减到 0，返回最小操作数；否则，返回 -1。

说明：

- $1 \leq nums.length \leq 10^5$。
- $1 \leq nums[i] \leq 10^4$。
- $1 \leq x \leq 10^9$。

示例：

- 示例 1：

```py
输入：nums = [1,1,4,2,3], x = 5
输出：2
解释：最佳解决方案是移除后两个元素，将 x 减到 0。
```

- 示例 2：

```py
输入：nums = [3,2,20,1,1,3], x = 10
输出：5
解释：最佳解决方案是移除后三个元素和前两个元素（总共 5 次操作），将 x 减到 0。
```

解题思路

思路 1：滑动窗口

将 x 减到 0 的最小操作数可以转换为求和等于 $sum(nums) - x$ 的最长连续子数组长度。我们可以维护一个区间和为 $sum(nums) - x$ 的滑动窗口，求出最长的窗口长度。具体做法如下：

令 `target = sum(nums) - x`，使用 max_len 维护和等于 $target$ 的最长连续子数组长度。然后用滑动窗口 $window\_sum$ 来记录连续子数组的和，设定两个指针：$left$、$right$，分别指向滑动窗口的左右边界，保证窗口中的和刚好等于 $target$。

- 一开始，$left$、$right$ 都指向 0。
- 向右移动 $right$，将最右侧元素加入当前窗口和 $window\_sum$ 中。
- 如果 $window\_sum > target$，则不断右移 $left$，缩小滑动窗口长度，并更新窗口和的最小值，直到 $window\_sum \leq target$。
- 如果 $window\_sum == target$，则更新最长连续子数组长度。
- 然后继续右移 $right$，直到 $right$ 遍历完 $nums$ 结束。
- 输出 $len(nums) - max\_len$ 作为最小操作数。
- 注意判断题目中的特殊情况。

思路 1：代码

py

```
class Solution:
    def minOperations(self, nums: List[int], x: int) -> int:
        target = sum(nums) - x
        size = len(nums)
        if target < 0:
            return -1
        if target == 0:
            return size
        left, right = 0, 0
        window_sum = 0
        max_len = float('-inf')

        while right < size:
            window_sum += nums[right]
```

```
        while window_sum > target:
            window_sum -= nums[left]
            left += 1
        if window_sum == target:
            max_len = max(max_len, right - left + 1)
        right += 1
    return len(nums) - max_len if max_len != float('-inf') else -1
```

思路 1：复杂度分析

- **时间复杂度**：$O(n)$，其中 n 为数组 $nums$ 的长度。
- **空间复杂度**：$O(1)$。