0011. 盛最多水的容器

ITCharge 大约 2 分钟

- 标签：贪心、数组、双指针
- 难度：中等

题目链接

- 0011. 盛最多水的容器 - 力扣

题目大意

描述：给定 n 个非负整数 $a_1, a_2, ..., a_n$，每个数代表坐标中的一个点 (i, a_i)。在坐标内画 n 条垂直线，垂直线 i 的两个端点分别为 (i, a_i) 和 $(i, 0)$。

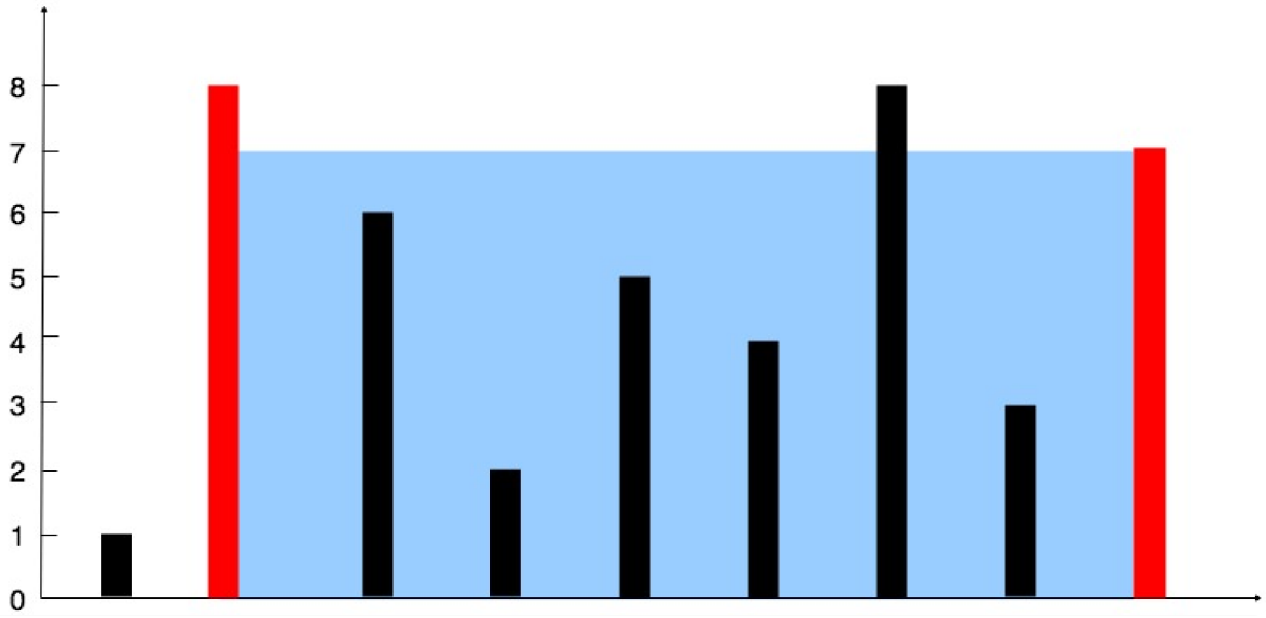
要求：找出其中的两条线，使得它们与 x 轴共同构成的容器可以容纳最多的水。

说明：

- $n == height.length$。
- $2 \le n \le 10^5$。
- $0 \le height[i] \le 10^4$。

示例：

- 示例 1：



py

输入：[1,8,6,2,5,4,8,3,7]
输出：49
解释：图中垂直线代表输入数组 [1,8,6,2,5,4,8,3,7]。在此情况下，容器能够容纳水（表示为蓝色部分）的最大值为 49。

解题思路

思路 1：对撞指针

从示例中可以看出，如果确定好左右两端的直线，容纳的水量是由 左右两端直线中较低直线的高度 * 两端直线之间的距离 所决定的。所以我们应该使得 **较低直线的高度尽可能的高**，这样才能使盛水面积尽可能的大。

可以使用对撞指针求解。移动较低直线所在的指针位置，从而得到不同的高度和面积，最终获取其中最大的面积。具体做法如下：

1. 使用两个指针 `left` ， `right` 。 `left` 指向数组开始位置， `right` 指向数组结束位置。
2. 计算 `left` 和 `right` 所构成的面积值，同时维护更新最大面积值。
3. 判断 `left` 和 `right` 的高度值大小。
    1. 如果 `left` 指向的直线高度比较低，则将 `left` 指针右移。
    2. 如果 `right` 指向的直线高度比较低，则将 `right` 指针左移。
4. 如果遇到 `left == right` ，跳出循环，最后返回最大的面积。

思路 1：代码

```py
class Solution:
    def maxArea(self, height: List[int]) -> int:
        left = 0
        right = len(height) - 1
        ans = 0
        while left < right:
            area = min(height[left], height[right]) * (right-left)
            ans = max(ans, area)
            if height[left] < height[right]:
                left += 1
            else:
                right -= 1
        return ans
```

思路 1：复杂度分析

- **时间复杂度**：$O(n)$。
- **空间复杂度**：$O(1)$。