人工智能学院

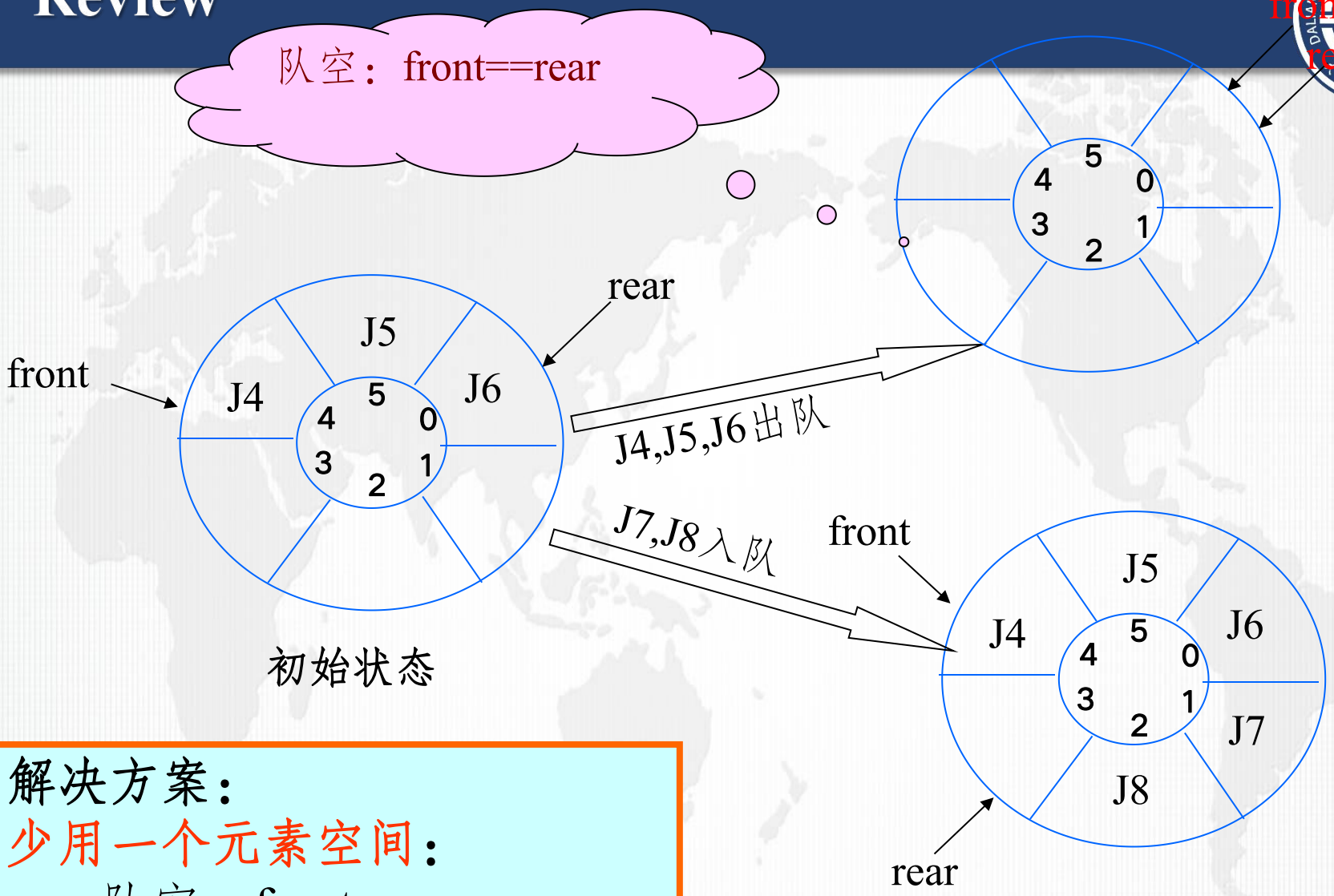
Data Structures and Algorithms

Stacks and Queues (栈和队列) 4

Assessment

1. Review
2. Practice

Review



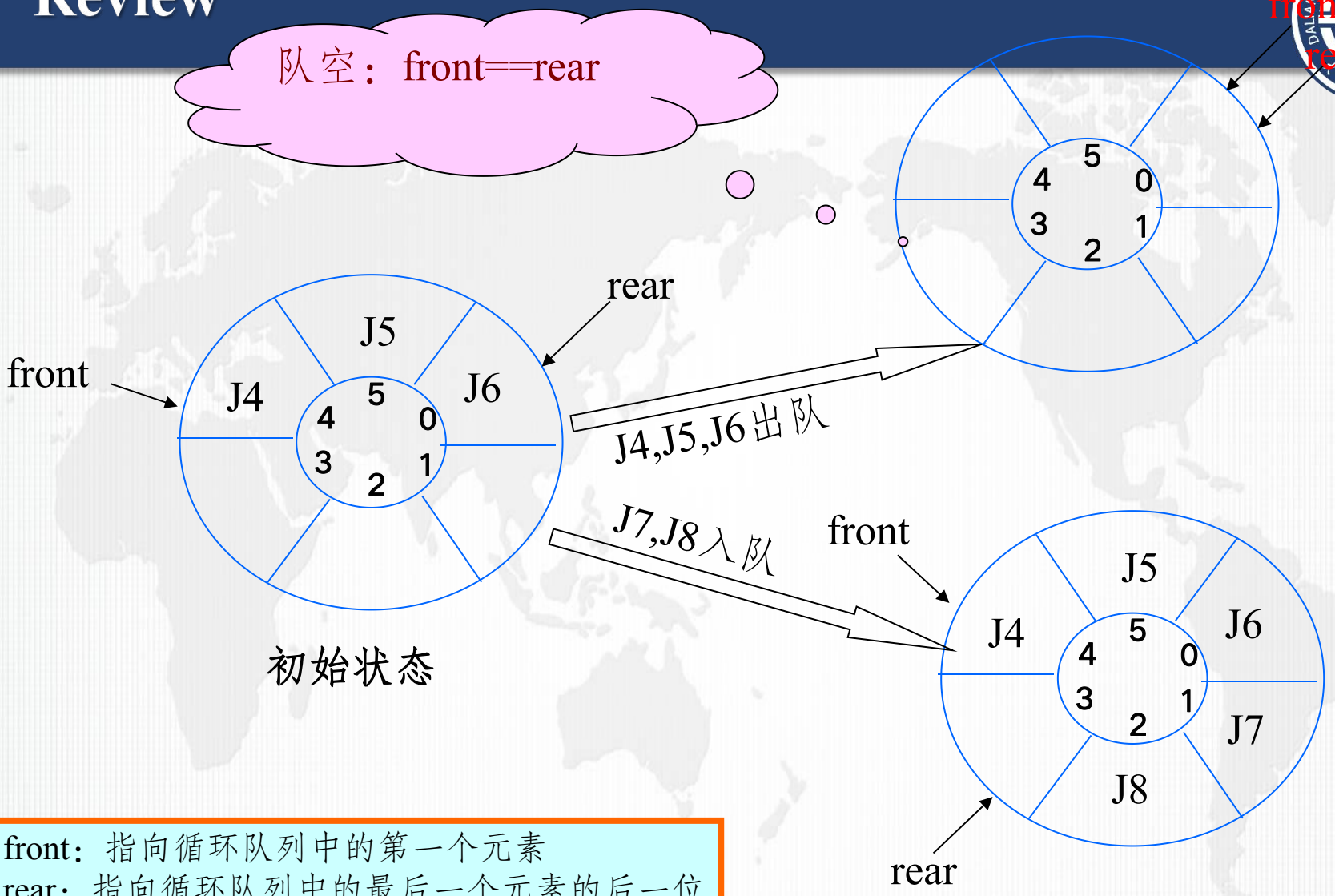
解决方案：

少用一个元素空间：

队空：front==rear

队满：(rear+1)%M==front

Review



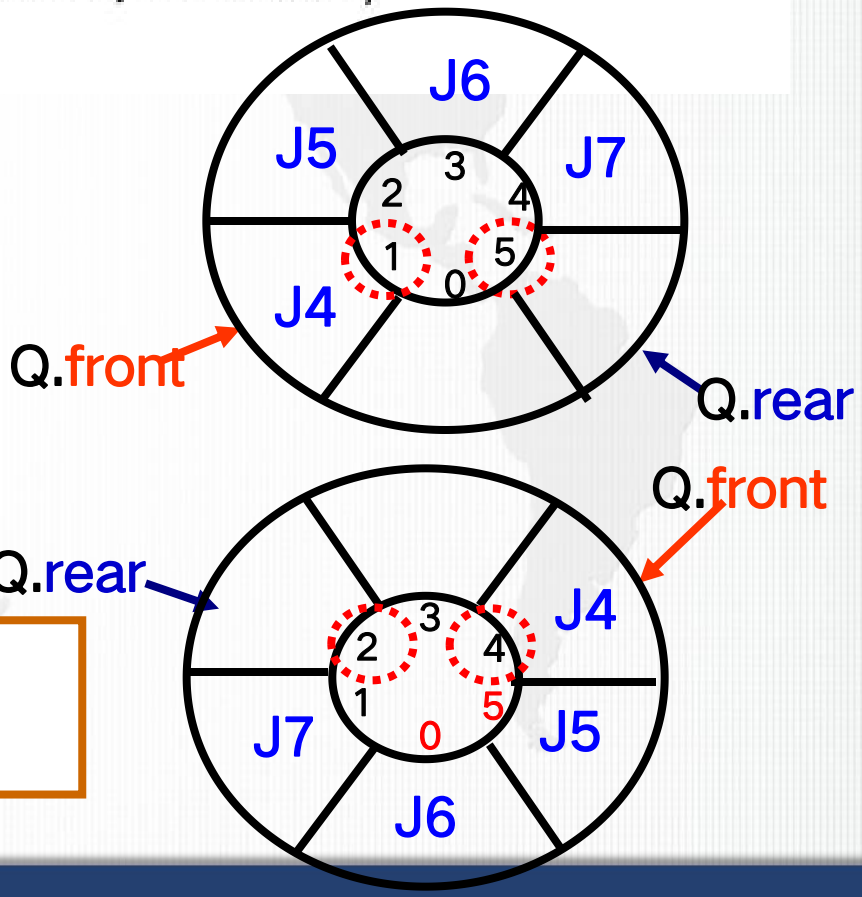
front：指向循环队列中的第一个元素
rear：指向循环队列中的最后一个元素的后一位

Review

◆ 求循环队列中元素个数：

```
31  int queue_length(SqQueue *Q) {
32      return (Q->rear - Q->front + MAXSIZE) % MAXSIZE;
33  }
```

T(n)=O(1)



队中结点的个数：

(rear - front + maxSize) % maxSize

Practice

◆ 练习1

给定 n 个非负整数表示每个宽度为 1 的柱子的高度图，计算按此排列的柱子，下雨之后能接多少雨水。

示例 1:

输入：height = [0,1,0,2,1,0,1,3,2,1,2,1]

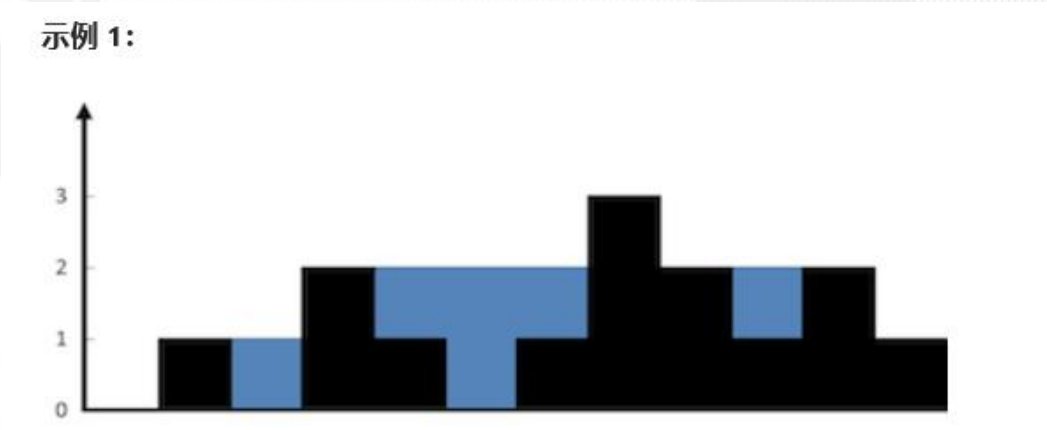
输出：6

解释：上面是由数组 [0,1,0,2,1,0,1,3,2,1,2,1] 表示的高度图，在这种情况下，可以接 6 个单位的雨水（蓝色部分表示雨水）。

示例 2:

输入：height = [4,2,0,3,2,5]

输出：9



Practice

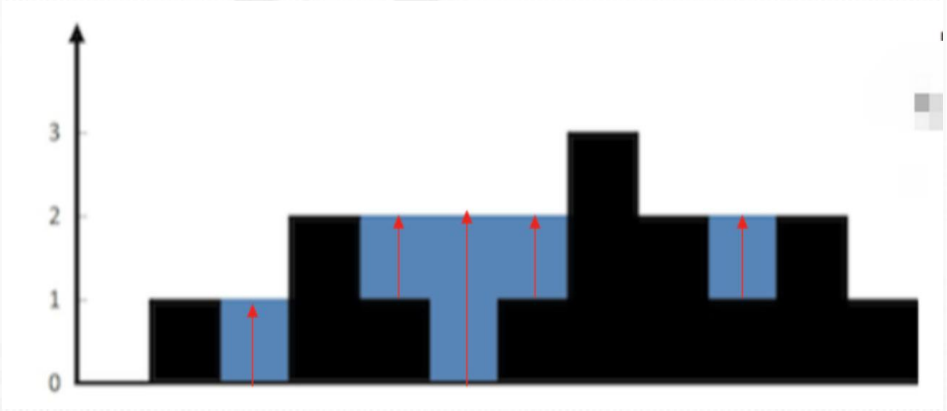
◆ 练习1

给定 n 个非负整数表示每个宽度为 1 的柱子的高度图，计算按此排列的柱子，下雨之后能接多少雨水。

暴力解法：

按照列的来计算的话，宽度一定是1，那么需要确定的是雨水的高度

每一列雨水的高度取决于该列左侧最高的柱子和右侧最高柱子中最矮的那个柱子高度。



Practice

◆ 练习1

给定 n 个非负整数表示每个宽度为 1 的柱子的高度图，计算按此排列的柱子，下雨之后能接多少雨水。

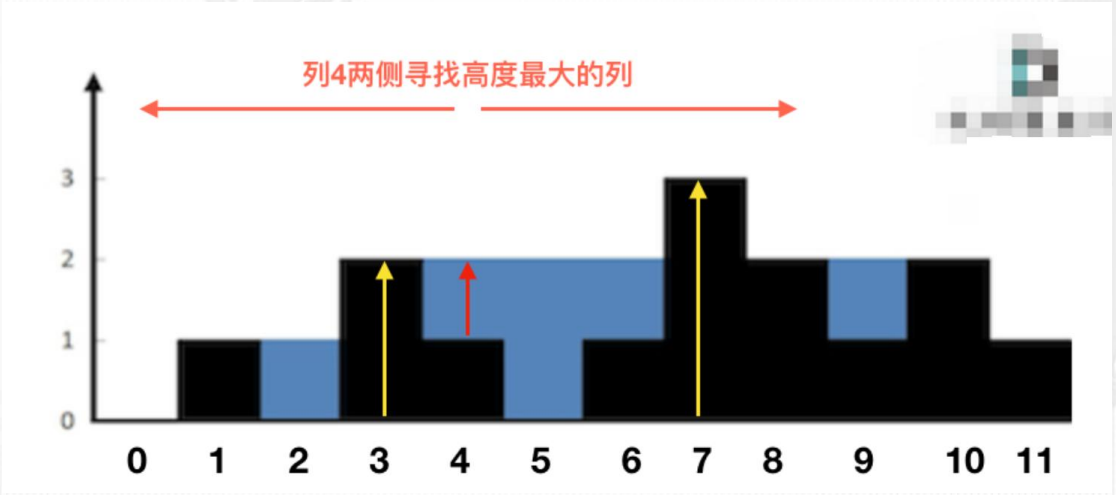
暴力解法：

列4左侧最高的柱子的列是3，高度为2

列4右侧最高的柱子的列是7，高度为3

列4柱子的高度为1

那么列4的雨水的高度为：min(2, 3) - 1 = 2 - 1 = 1



Practice

◆ 练习1

给定 n 个非负整数表示每个宽度为 1 的柱子的高度图，计算按此排列的柱子，下雨之后能接多少雨水。

方法二：

直观想法

在暴力方法中，我们仅仅为了找到最大值每次都要向左和向右扫描一次。但是我们可以先从左到右遍历一遍找到每个元素的左侧的最大高度；然后从右到左遍历一遍找到每个元素的

算法：

1. 找到数组中从下标 i 到最左端最高的条形块高度 left_max
2. 找到数组中从下标 i 到最右端最高的条形块高度 right_max。
3. 扫描数组 height 并更新答案：
4. 累加 min(max_left[i],max_right[i])−height[i]到 ans上

复杂度为O（n）

Practice

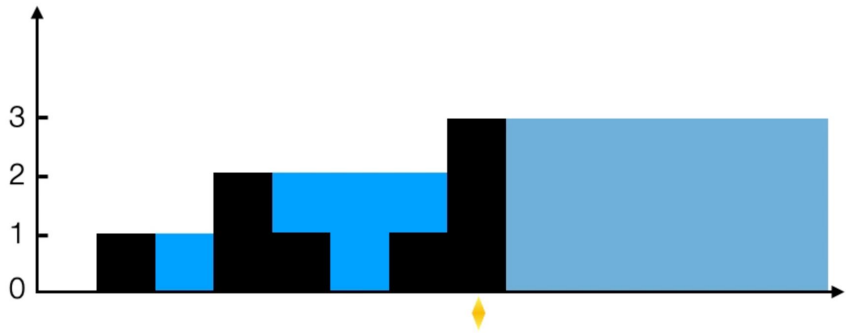
◆ 练习1

给定 n 个非负整数表示每个宽度为 1 的柱子的高度图，计算按此排列的柱子，下雨之后能接多少雨水。

方法三：

直观想法

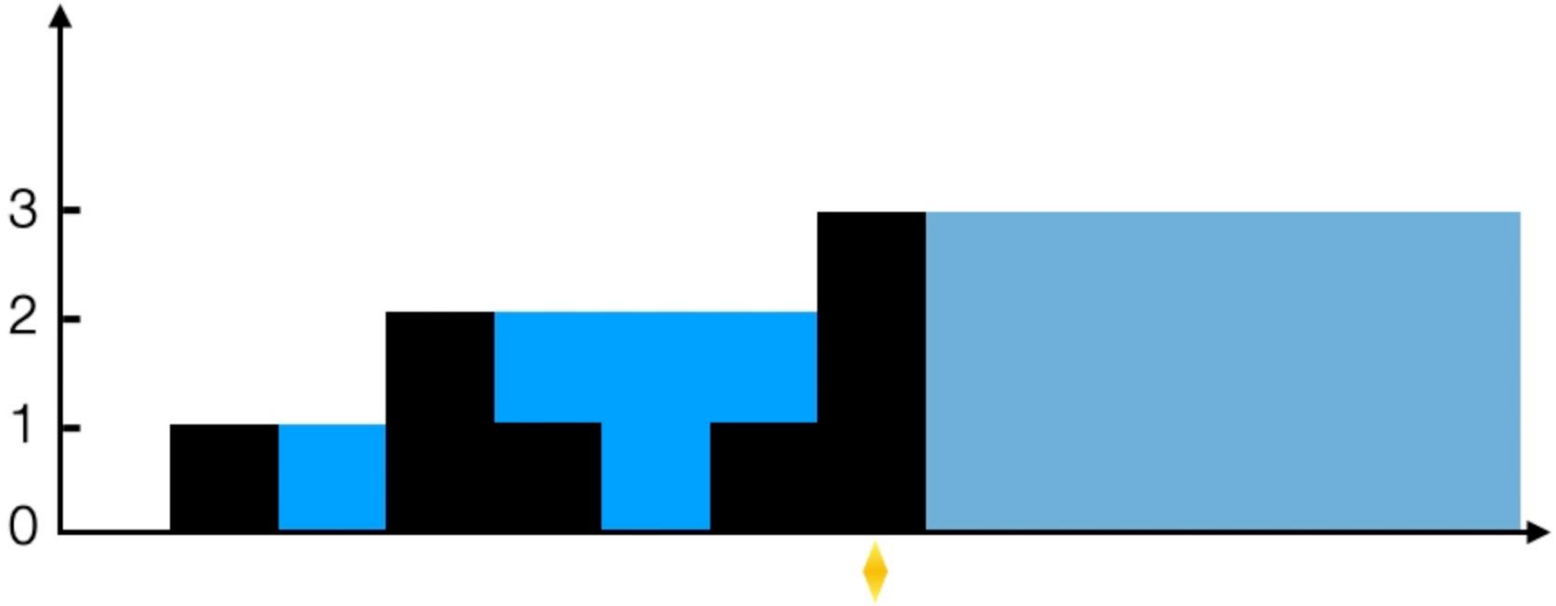
积水只能在低洼的地方形成，当后面的柱子高度比前面的低的时候，是没有办法接雨水的。所以使用单调递减栈存储可能储水的柱子，当找到一根比前面高的柱子，就可以计算接到的雨水。



height = [0,1,0,2,1,0,1,3,2,1,2,1]

Practice

- 练习1



height = [0,1,0,2,1,0,1,3,2,1,2,1]

height = [0,1,0,2,1,0,1,3,2,1,2,1]

Practice

◆ 练习2（括号匹配进阶）

给你一个只包含 '(' 和 ')' 的字符串，找出最长有效（格式正确且连续）括号子串的长度

始终保持栈底元素为当前已经遍历过的元素中「最后一个没有被匹配的右括号的下标」，这样的做法主要是考虑了边界条件的处理，栈里其他元素维护左括号的下标：

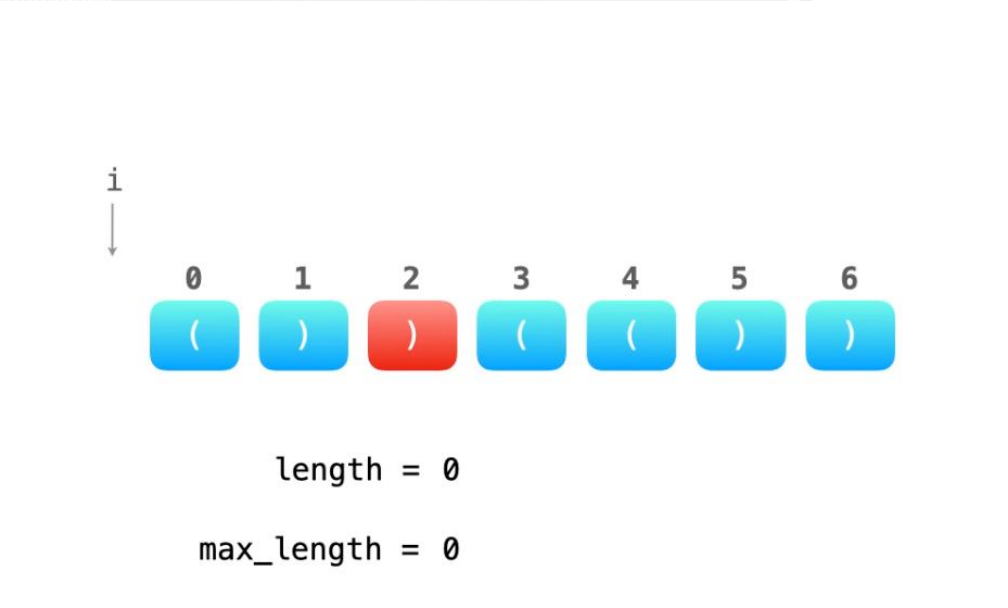
1. 对于遇到的每个‘(’ ，我们将它的下标放入栈中
2. 对于遇到的每个 ‘)’ ，我们先弹出栈顶元素表示匹配了当前右括号：
   ①如果栈为空，说明当前的右括号为没有被匹配的右括号，我们将其下标放入栈中来更新我们之前提到的「最后一个没有被匹配的右括号的下标」
   ②如果栈不为空，当前右括号的下标减去栈顶元素即为「以该右括号为结尾的最长有效括号的长度」

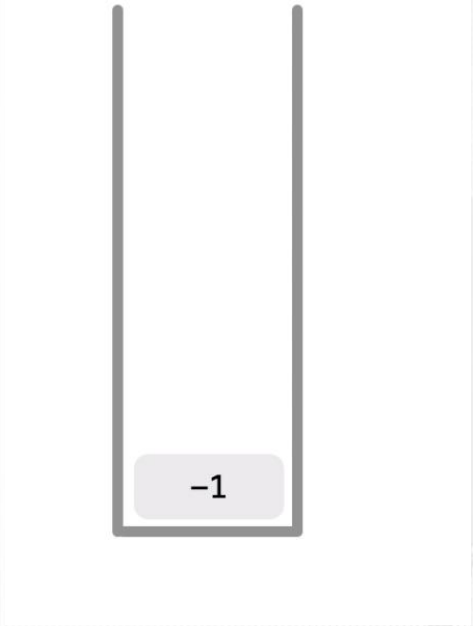
我们从前往后遍历字符串并更新答案即可。

Practice

◆ 练习2（括号匹配进阶）

给你一个只包含 '(' 和 ')' 的字符串，找出最长有效（格式正确且连续）括号子串的长度

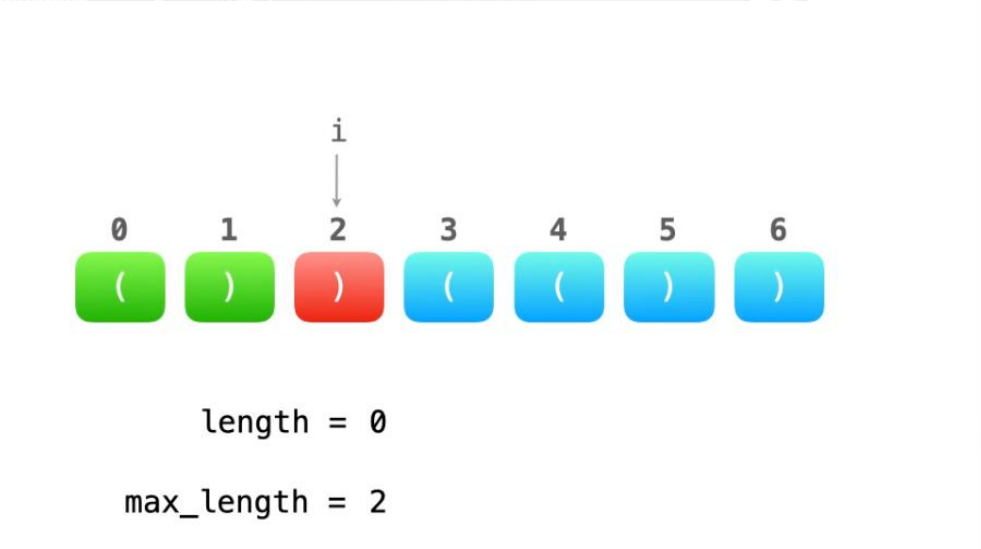


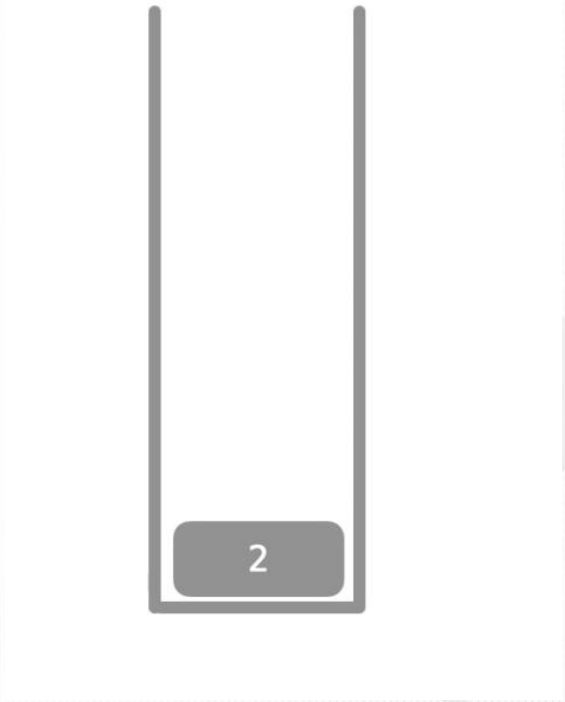


Practice

◆ 练习2（括号匹配进阶）

给你一个只包含 '(' 和 ')' 的字符串，找出最长有效（格式正确且连续）括号子串的长度





Practice

◆ 练习3（用栈和队列实现）

给你一个链表，两两交换其中相邻的节点，并返回交换后链表的头节点。你必须在不修改节点内部的值的情况下完成本题（即，只能进行节点交换）。

直观思想：

实现两两交换，后面的一个和前面的一个交换，可以联想到“栈”后入先出的特点

两两交换之后整体链表的生成与存储可以利用“队列”存储

利用队列实现，遍历完原链表后还需要一次遍历生成新链表

算法：

1. 遍历给定链表，拆解出当前遍历的元素（使 next 为空）
2. 将拆解出的元素入栈，入栈前判断栈中是否有两个元素
3. 如果栈中有两个元素，先将元素依次出栈，然后入队
4. 经过一轮链表的遍历，就得到了满足结果的“链表节点”队列，每个节点 next 均为空
5. 遍历队列组成新链表即可，此时链表元素依旧是原链表的元素，内存地址没有改变

Practice

◆ 练习4

给定一个整数数组 temperatures ，表示每天的温度，返回一个数组 answer ，其中 answer[i] 是指对于第 i 天，下一个更高温度出现在几天后。如果气温在这之后都不会升高，请在该位置用 0 来代替。

例如，给定一个列表 temperatures = [73, 74, 75, 71, 69, 72, 76, 73]，你的输出应该是 [1, 1, 4, 2, 1, 1, 0, 0]。

解法一：

暴力解法，两层for循环，把至少需要等待的天数就搜出来了。时间复杂度是O(n^2)

Practice

◆ 练习4

给定一个整数数组 temperatures，表示每天的温度，返回一个数组 answer，其中 answer[i] 是指对于第 i 天，下一个更高温度出现在几天后。如果气温在这之后都不会升高，请在该位置用 0 来代替。

解法二：单调栈

那么单调栈的原理是什么呢？为什么时间复杂度是O(n)就可以找到每一个元素的右边第一个比它大的元素位置呢？

- 单调栈的本质是空间换时间，因为在遍历的过程中需要用一个栈来记录右边第一个比当前元素高的元素，优点是整个数组只需要遍历一次。
- 更直白来说，就是用一个栈来记录我们遍历过的元素，因为我们遍历数组的时候，我们不知道之前都遍历了哪些元素，以至于遍历一个元素不确定是否比之前的遍历过的元素大，所以我们需要用一个容器（这里用单调栈）来记录我们遍历过的元素。

Practice

◆ 练习4

给定一个整数数组 temperatures ，表示每天的温度，返回一个数组 answer ，其中 answer[i] 是指对于第 i 天，下一个更高温度出现在几天后。如果气温在这之后都不会升高，请在该位置用 0 来代替。

解法二：单调栈

1. 单调栈里存放的元素是什么？
单调栈里只需要存放元素的下标i就可以了
如果需要使用对应的元素，直接T[i]就可以获取。

2. 单调栈里元素是递增呢？ 还是递减呢？

Practice

◆ 练习5

给你两个没有重复元素的数组 nums1 和 nums2 ，其中nums1 是 nums2 的子集。
请你找出 nums1 中每个元素在 nums2 中的下一个比其大的值。。

示例 1:

输入: nums1 = [4,1,2], nums2 = [1,3,4,2].

输出: [-1,3,-1]

解释:

对于 num1 中的数字 4 ，你无法在第二个数组中找到下一个更大的数字，因此输出 -1 。

对于 num1 中的数字 1 ，第二个数组中数字1右边的下一个较大数字是 3 。

对于 num1 中的数字 2 ，第二个数组中没有下一个更大的数字，因此输出 -1 。

示例 2:

输入: nums1 = [2,4], nums2 = [1,2,3,4].

输出: [3,-1]

解释:

对于 num1 中的数字 2 ，第二个数组中的下一个较大数字是 3 。

对于 num1 中的数字 4 ，第二个数组中没有下一个更大的数字，因此输出-1 。

Practice

◆ 练习5

- 使用单调栈，首先要想单调栈是从大到小还是从小到大（栈头到栈底的顺序）？
- 使用单调栈，需要遍历的是哪个数组？

Practice

◆ 练习5

- 使用单调栈，首先要想单调栈是从大到小还是从小到大（栈头到栈底的顺序）？
  栈里的元素为递增顺序（栈头到栈底的顺序）
  （递减栈：求右边第一个比自己小的元素了）
- 使用单调栈，需要遍历的是哪个数组？
  遍历num2，找到current_element右边比它大的第一个元素
  同时判断current_element是否在num1中

Practice

◆ 练习5

- 当前遍历的元素T[i]小于栈顶元素T[st.top()]的情况
  此时满足递增栈（栈头到栈底的顺序），所以直接入栈
- 当前遍历的元素T[i]等于栈顶元素T[st.top()]的情况
  如果相等的话，依然直接入栈，因为我们要求的是右边第一个比自己大的元素，而不是大于等于
- 当前遍历的元素T[i]大于栈顶元素T[st.top()]的情况
  此时如果入栈就不满足递增栈了，这也是找到右边第一个比自己大的元素的时候。

判断栈顶元素是否在nums1里出现过，（注意栈里的元素是nums2的元素），如果出现过，开始记录结果。

Homework

- 选择题全部
- leetcode 题目全部AC

Thanks!

See you in the next session!