

高频题

283. 移动零

给定一个数组 `nums`，编写一个函数将所有 `0` 移动到数组的末尾，同时保持非零元素的相对顺序。

输入: `[0,1,0,3,12]`

输出: `[1,3,12,0,0]`

1. 必须在原数组上操作，不能拷贝额外的数组。
2. 尽量减少操作次数。

cur



0	1	2	3	4
0	1	0	3	12



i

1. 两数之和

给定一个整数数组 `nums` 和一个目标值 `target`，请你在该数组中找出和为目标值的那 **两个** 整数，并返回他们的数组下标。

你可以假设每种输入只会对应一个答案。但是，你不能重复利用这个数组中同样的元素。

给定 `nums = [2, 7, 11, 15]`, `target = 9`

因为 `nums[0] + nums[1] = 2 + 7 = 9`

所以返回 `[0, 1]`

■ 暴力法

□ 枚举每一对整数

□ 时间复杂度 $O(n^2)$

□ 空间复杂度 $O(1)$

优化

目标: 14					
0	1	2	3	4	5
2	4	5	7	9	11

↑
i

哈希表	
key	value
2	0
4	1
5	2
7	3

■ 时间复杂度 $O(n)$

■ 空间复杂度 $O(n)$

15. 三数之和

给你一个包含 n 个整数的数组 `nums`，判断 `nums` 中是否存在三个元素 a ， b ， c ，使得 $a + b + c = 0$ ？请你找出所有满足条件且不重复的三元组。

注意：答案中不可以包含重复的三元组。

给定数组 `nums = [-1, 0, 1, 2, -1, -4]`，

满足要求的三元组集合为：

```
[  
  [-1, 0, 1],  
  [-1, -1, 2]  
]
```

■ 暴力法

□ 枚举每一个三元组

□ 时间复杂度 $O(n^3)$

□ 空间复杂度 $O(1)$

优化

0	1	2	3	4	5
-1	0	1	2	-1	-4

排序

0	1	2	3	4	5
-4	-1	-1	0	1	2

↑

i

↑

l

↑

r

■ 时间复杂度 $O(n^2)$

■ 空间复杂度 $O(1)$

50. Pow(x, n)

实现 `pow(x, n)`，即计算 x 的 n 次幂函数。

输入：2.00000, 10
输出：1024.00000

输入：2.00000, -2
输出：0.25000
解释： $2^{-2} = 1/2^2 = 1/4 = 0.25$

说明：

- $-100.0 < x < 100.0$
- n 是 32 位有符号整数，其数值范围是 $[-2^{31}, 2^{31} - 1]$ 。

■ 最简单的做法

□ 将 n 个 x 进行相乘

□ 时间复杂度： $O(n)$

□ 空间复杂度： $O(1)$

■ 快速幂（分治）

□ 时间复杂度： $O(\log n)$

□ 非递归空间复杂度： $O(1)$

□ 递归空间复杂度： $O(\log n)$

递归

$$3^{20} = 3^{10} * 3^{10}$$

$$3^{21} = 3^{10} * 3^{10} * 3$$

$$3^{-20} = 3^{-10} * 3^{-10}$$

$$3^{-21} = 3^{-10} * 3^{-10} * 3^{-1}$$

$$3^{-21} = 3^{-11} * 3^{-11} * 3$$

非递归

$$3^{21}$$

$$21 = (10101)_2$$

$$21 = (1 * 2^4) + (0 * 2^3) + (1 * 2^2) + (0 * 2^1) + (1 * 2^0)$$

$$3^{21} = 3^{(1 * 2^4) + (0 * 2^3) + (1 * 2^2) + (0 * 2^1) + (1 * 2^0)}$$

$$3^{21} = 3^{1 * 2^4} * 3^{0 * 2^3} * 3^{1 * 2^2} * 3^{0 * 2^1} * 3^{1 * 2^0}$$

非递归

$$21 = (10101)_2$$

$$3^{2^1} = 3^{1*2^4} * 3^{0*2^3} * 3^{1*2^2} * 3^{0*2^1} * 3^{1*2^0}$$

$$3^{2^1} = 3^{2^0} * 3^{2^0} = 3^1 * 3^1 = 3^2$$

$$3^{2^2} = 3^{2^1} * 3^{2^1} = 3^2 * 3^2 = 3^4$$

$$3^{2^3} = 3^{2^2} * 3^{2^2} = 3^4 * 3^4 = 3^8$$

$$3^{2^4} = 3^{2^3} * 3^{2^3} = 3^8 * 3^8 = 3^{16}$$

快速幂补充

■ 请设计一个算法求x的y次幂模z的结果: $x^y \% z$

□ 假设x、y都可能是很大的整数

□ $y \geq 0, z \neq 0$

■ 公式须知

□ $(a * b) \% p == ((a \% p) * (b \% p)) \% p$

面试题62. 圆圈中最后剩下的数字

0,1,...,n-1这n个数字排成一个圆圈，从数字0开始，每次从这个圆圈里删除第m个数字。求出这个圆圈里剩下的最后一个数字。

例如，0、1、2、3、4这5个数字组成一个圆圈，从数字0开始每次删除第3个数字，则删除的前4个数字依次是2、0、4、1，因此最后剩下的数字是3。

输入：n = 5, m = 3

输出：3

输入：n = 10, m = 17

输出：2

■ 计算公式

$$\square f(n, m) = (f(n - 1, m) + m) \% n$$

■ 这其实就是著名的约瑟夫环问题

□ 有n个人，编号分别为0, 1, ..., n - 1，每当报数到第m个人时，就杀掉他，求最后胜利者编号

公式推导

■ $f(11, 3) == 6$

□ 从A开始报数，最后能活下来的是G

0	1	2	3	4	5	6	7	8	9	10
A	B	C	D	E	F	G	H	I	J	K

■ 从A开始报数，杀掉C之后，剩下10个人，接下来从D开始报数，最后能活下来的依然是G

8	9	0	1	2	3	4	5	6	7
A	B	D	E	F	G	H	I	J	K

■ 从11人变为10人，胜利者的编号由6变为3，所以 $f(10, 3) = f(11, 3) - 3$

■ 也就是说 $f(11, 3) = f(10, 3) + 3$

■ 通用结论: $f(n, m) = (f(n - 1, m) + m) \% n$

■ 最后的%n是为了防止索引越界

如果编号从1开始

```
public int lastRemaining(int n, int m) {  
    int res = 0;  
    for (int i = 2; i <= n; i++) {  
        res = (res + m) % i;  
    }  
    return res + 1;  
}
```

```
public int lastRemaining(int n, int m) {  
    return f(n, m) + 1;  
}  
  
public int f(int n, int m) {  
    return (n == 1) ? 0 : (f(n - 1, m) + m) % n;  
}
```

54. 螺旋矩阵

给定一个包含 $m \times n$ 个元素的矩阵 (m 行, n 列), 请按照顺时针螺旋顺序, 返回矩阵中的所有元素。

输入:

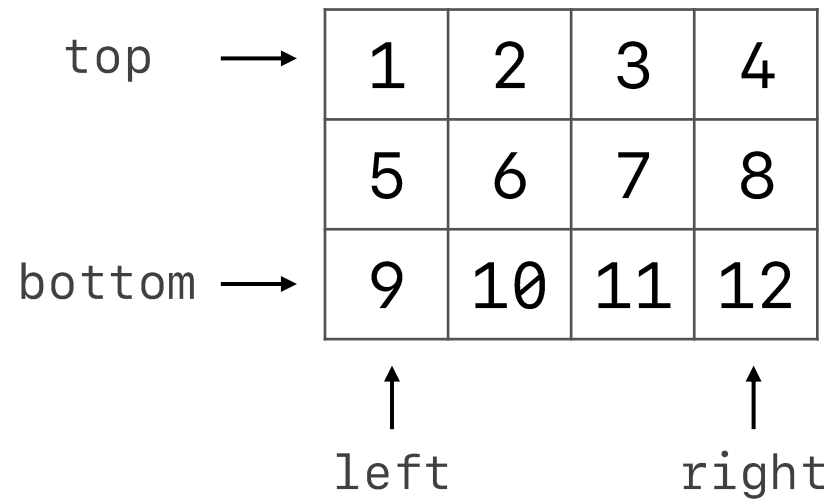
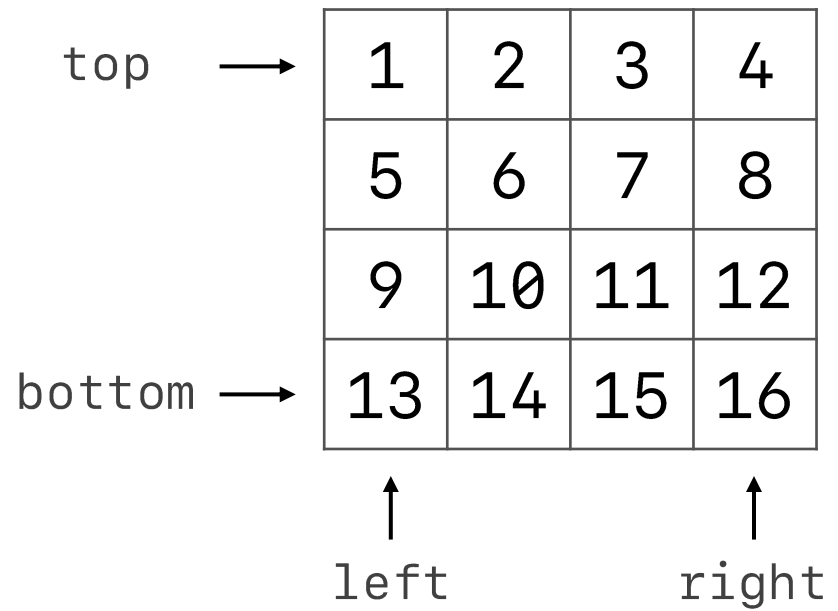
```
[  
  [ 1, 2, 3 ],  
  [ 4, 5, 6 ],  
  [ 7, 8, 9 ]  
]
```

输出: [1,2,3,6,9,8,7,4,5]

输入:

```
[  
  [1, 2, 3, 4],  
  [5, 6, 7, 8],  
  [9,10,11,12]  
]
```

输出: [1,2,3,4,8,12,11,10,9,5,6,7]



146. LRU缓存机制

- LRU (Least Recently Used) : 最近最少使用、最近最久未使用
- 是操作系统常用的一种页面置换算法, 选择最近最久未使用的页面予以淘汰

运用你所掌握的数据结构, 设计和实现一个 LRU (最近最少使用) 缓存机制。它应该支持以下操作: 获取数据 `get` 和 写入数据 `put` 。

获取数据 `get(key)` - 如果密钥 (key) 存在于缓存中, 则获取密钥的值 (总是正数), 否则返回 -1。

写入数据 `put(key, value)` - 如果密钥不存在, 则写入其数据值。当缓存容量达到上限时, 它应该在写入新数据之前删除最久未使用的数据值, 从而为新的数据值留出空间。

进阶:

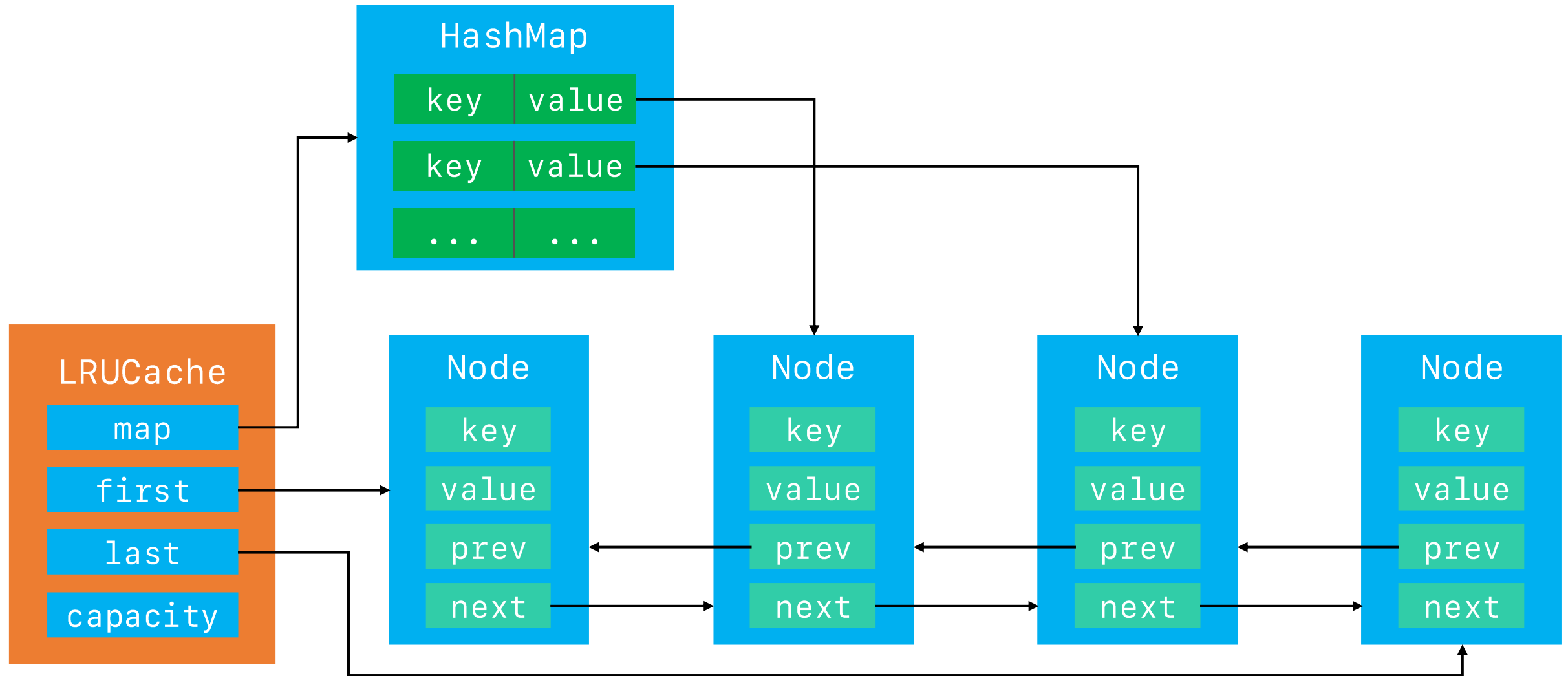
你是否可以在 **O(1)** 时间复杂度内完成这两种操作?

```
LRUCache cache = new LRUCache( 2 /* 缓存容量 */ );

cache.put(1, 1);
cache.put(2, 2);
cache.get(1);           // 返回 1
cache.put(3, 3);         // 该操作会使得密钥 2 作废
cache.get(2);           // 返回 -1 (未找到)
cache.put(4, 4);         // 该操作会使得密钥 1 作废
cache.get(1);           // 返回 -1 (未找到)
cache.get(3);           // 返回 3
cache.get(4);           // 返回 4
```

- LRUCache的常见实现方式是: 哈希表+双向链表

LRUCache的设计



7. 整数反转

给出一个 32 位的有符号整数，你需要将这个整数中每位上的数字进行反转。

输入：123

输出：321

输入：-123

输出：-321

输入：120

输出：21

假设我们的环境只能存储得下 32 位的有符号整数，则其数值范围为 $[-2^{31}, 2^{31} - 1]$ 。请根据这个假设，如果反转后整数溢出那么就返回 0。

252. 会议室

给定一个会议时间安排的数组，每个会议时间都会包括开始和结束的时间 $[[s_1, e_1], [s_2, e_2], \dots]$ ($s_i < e_i$)，请你判断一个人是否能够参加这里面的全部会议。

输入: `[[0,30],[5,10],[15,20]]`

输出: `false`

输入: `[[7,10],[2,4]]`

输出: `true`

253. 会议室 II

给定一个会议时间安排的数组，每个会议时间都会包括开始和结束的时间 $[[s_1, e_1], [s_2, e_2], \dots]$ ($s_i < e_i$)，为避免会议冲突，同时要考虑充分利用会议室资源，请你计算至少需要多少间会议室，才能满足这些会议安排。

输入： $[[0, 30], [5, 10], [15, 20]]$

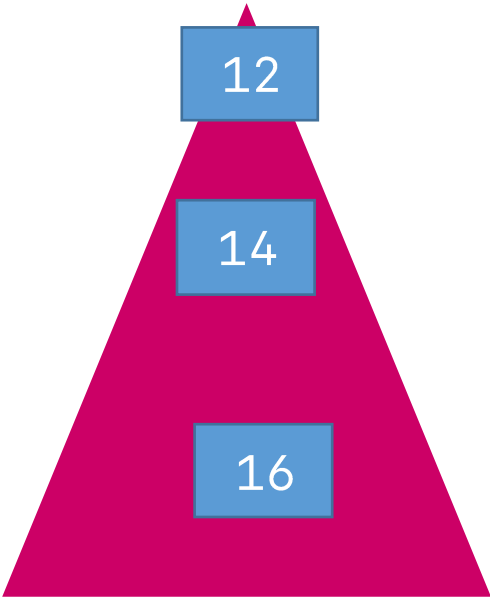
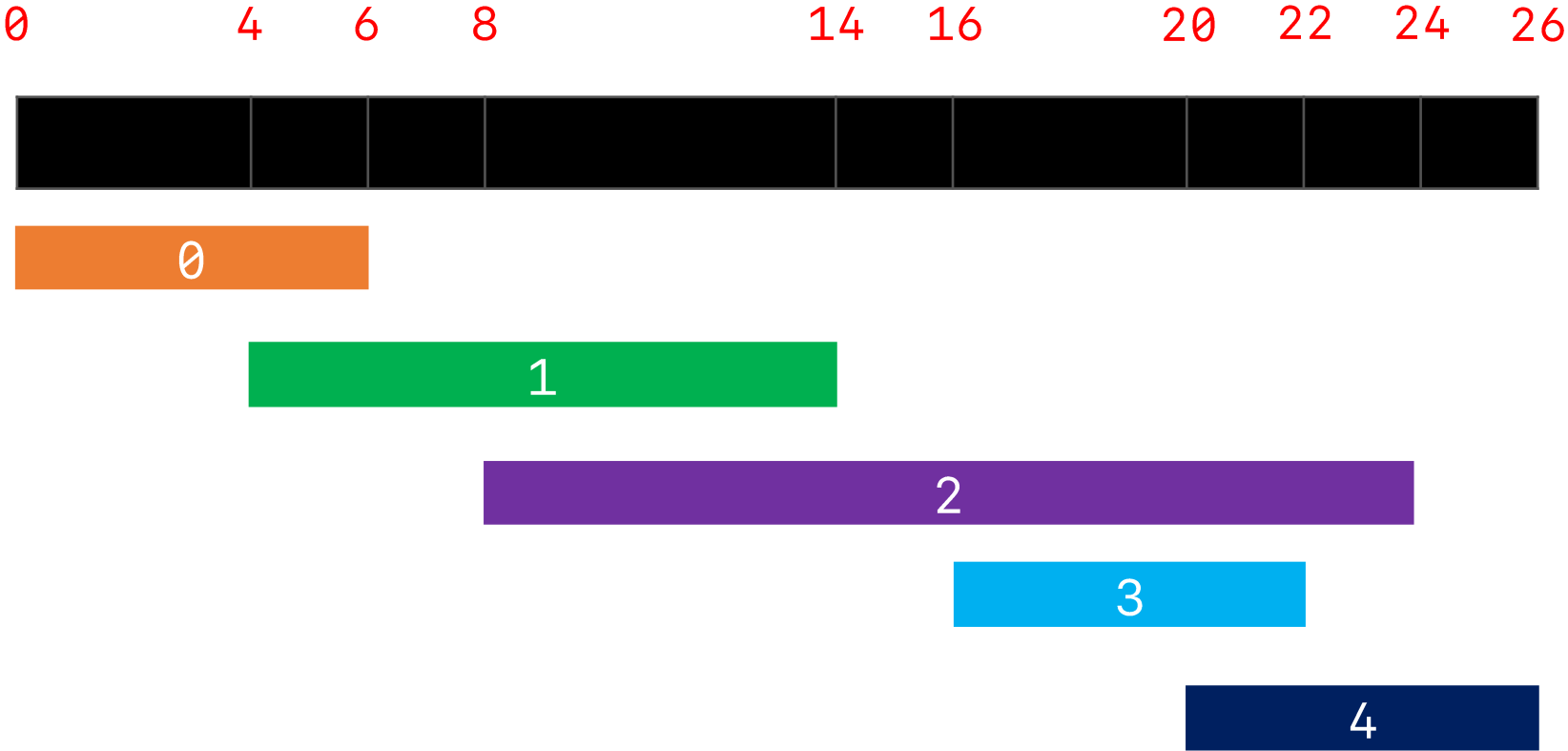
输出：2

输入： $[[7, 10], [2, 4]]$

输出：1

最小堆

[0, 6], [4, 14], [8, 24], [16, 22], [20, 26]



分开排序

[0, 6], [4, 14], [8, 24], [16, 22], [20, 26]

开始时间				
0	1	2	3	4
0	4	8	16	20

↑
beginIdx

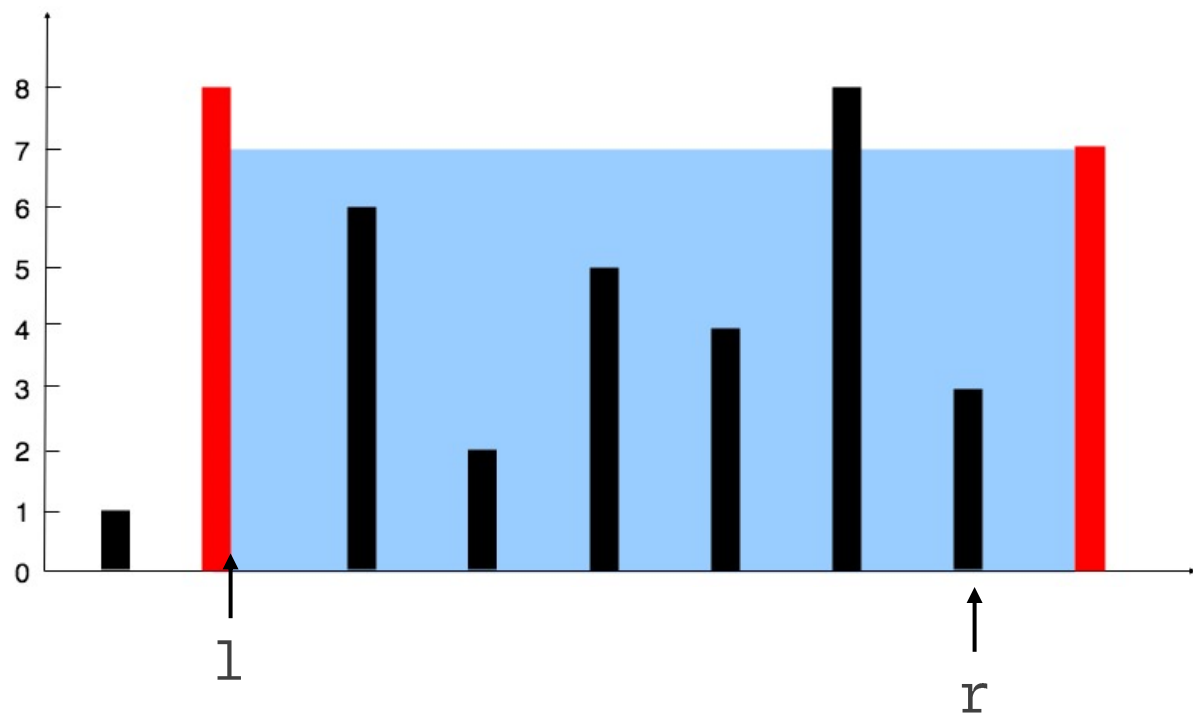
结束时间				
0	1	2	3	4
6	14	22	24	26

↑
endIdx

11. 盛最多水的容器

给你 n 个非负整数 a_1, a_2, \dots, 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 $(i, 0)$ 。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：你不能倾斜容器，且 n 的值至少为 2。

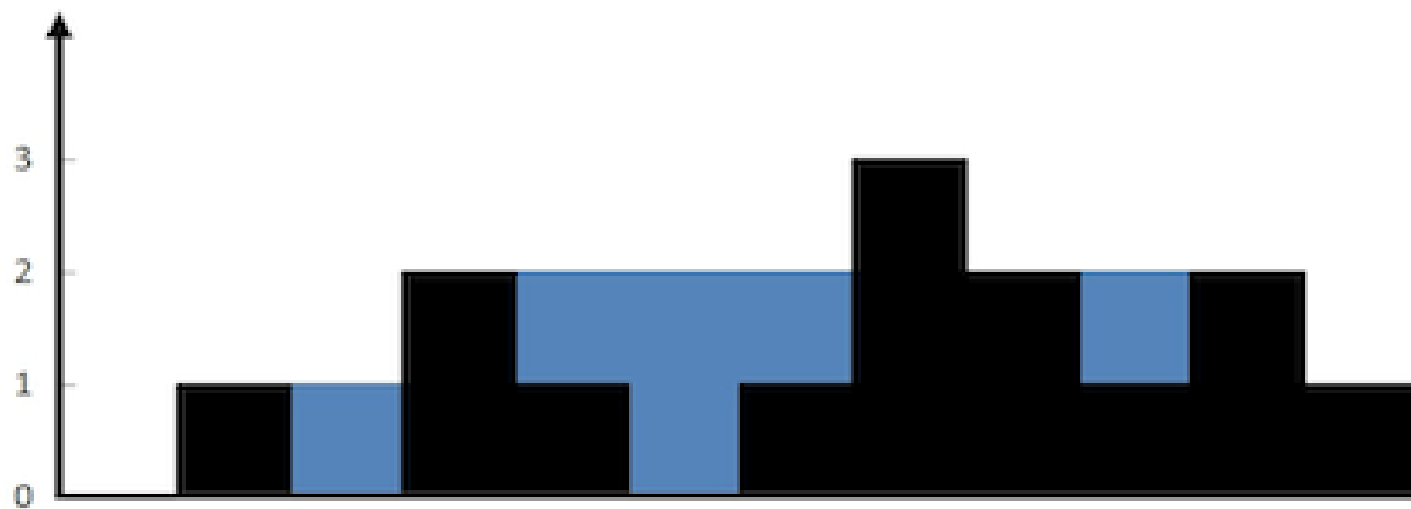


输入: $[1, 8, 6, 2, 5, 4, 8, 3, 7]$

输出: 49

42. 接雨水

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。



输入: `[0,1,0,2,1,0,1,3,2,1,2,1]`

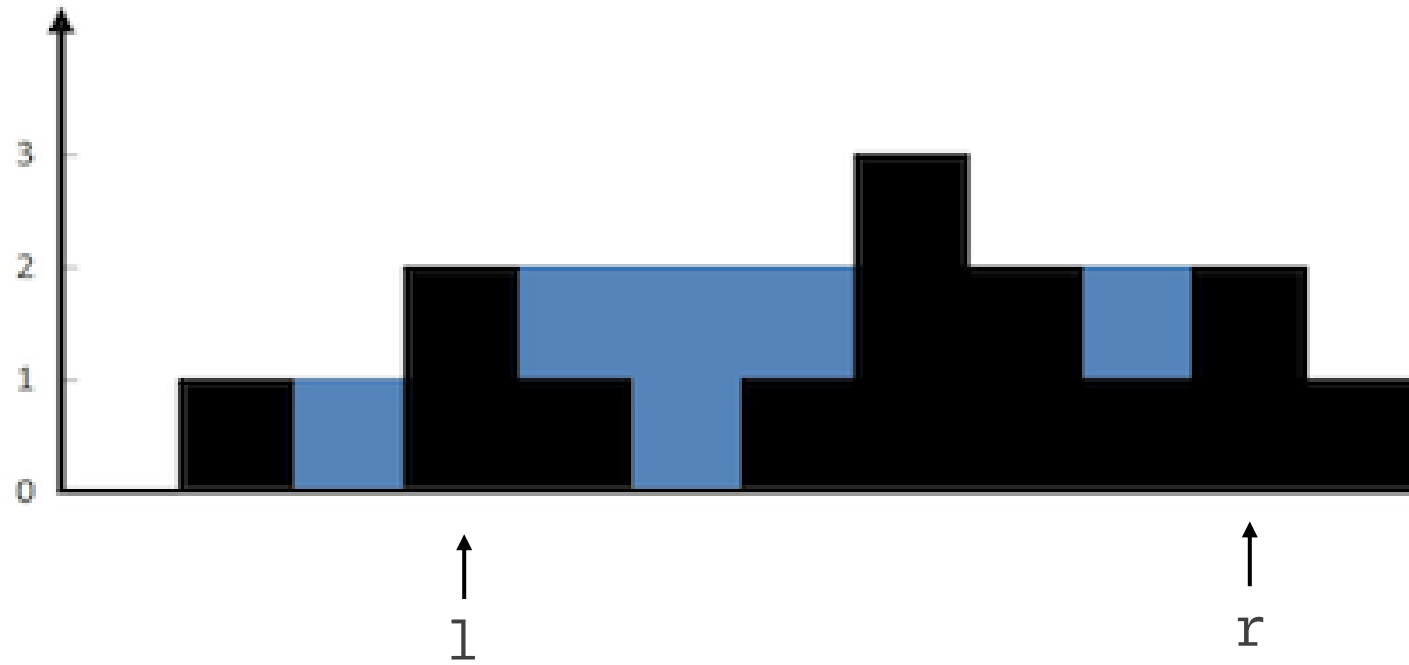
输出: 6

water = 0

lowerMax = 1

lower = 1

这个柱子能放多少水: $\text{lowerMax} - \text{lower}$



思考题

- [215. 数组中的第K个最大元素](#)
- [315. 计算右侧小于当前元素的个数](#)
- [4. 寻找两个有序数组的中位数](#)
- [149. 直线上最多的点数](#)
- [200. 岛屿数量](#)

第三季总共讲解的题目 (21)

■ 数组 (3)

- [88. 合并两个有序数组](#)
- [75. 颜色分类](#)
- [面试题 16.16. 部分排序](#)

■ 链表 (5)

- [203. 移除链表元素](#)
- [2. 两数相加](#)
- [160. 相交链表](#)
- [86. 分隔链表](#)
- [234. 回文链表](#)

■ 栈_队列 (4)

- [155. 最小栈](#)
- [239. 滑动窗口最大值](#)
- [654. 最大二叉树](#)
- [739. 每日温度](#)

■ 字符串 (5)

- [面试题 01.09. 字符串轮转](#)
- [572. 另一个树的子树](#)
- [242. 有效的字母异位词](#)
- [151. 翻转字符串里的单词](#)
- [3. 无重复字符的最长子串](#)

■ 动态规划 (4)

- [面试题47. 礼物的最大价值](#)
- [121. 买卖股票的最佳时机](#)
- [72. 编辑距离](#)
- [5. 最长回文子串](#)

第三季总共讲解的题目 (19)

■ 二叉树 (3)

- [236. 二叉树的最近公共祖先](#)
- [99. 恢复二叉搜索树](#)
- [333. 最大BST子树](#)

■ DFS (4)

- [17. 电话号码的字母组合](#)
- [46. 全排列](#)
- [47. 全排列 II](#)
- [22. 括号生成](#)

■ 高频题 (12)

- [283. 移动零](#)
- [1. 两数之和](#)
- [15. 三数之和](#)
- [50. Pow\(x, n\)](#)
- [面试题62. 圆圈中最后剩下的数字](#)
- [54. 螺旋矩阵](#)
- [146. LRU缓存机制](#)
- [7. 整数反转](#)
- [252. 会议室](#)
- [253. 会议室 II](#)
- [11. 盛最多水的容器](#)
- [42. 接雨水](#)

刷题建议

- 读懂题意后，如果思考10~15分钟没有思路，直接看答案
 - 读懂答案后，照着抄一遍
 - 然后默写一遍
-
- 每道题的所有解法都看一遍（不要只看最优解）
-
- 建议观摩一下leetcode.com的Discuss阅读量靠前的代码
-
- 建议观摩一下leetcode-cn.com、leetcode.com打败时间（接近）100%的提交代码

面试建议

■ 与面试官沟通题目细节

- 有无空间\时间复杂度要求?
- 数组\链表是否有序?
- 数组\链表是否有重复元素?
-

■ 面试

- 把你能想到的专业术语关键词、解法都可以说出来

■ 笔试\机试

- 写思路比写代码更重要（简单写一下主要思路即可）
- 写完代码后，最好顺便写出空间、时间复杂度
- 优先写出最优解，若时间允许，也可以写出其他解法

联想关键词

■ 数组

□ 排序、双指针、三指针、扫描方向（左→右、右→左）

■ 链表

□ 虚拟头结点、双指针、快慢指针、翻转、中间节点

■ 排列组合

□ DFS

■ 最值

□ 贪心、排序、动态规划

■ 对称\顺序

□ 栈\队列

■ 二叉树

□ 递归、遍历

■ 搜索数据要求 $O(1)$ 时间

□ 哈希表