1、两数之和

给定一个整数数组和一个目标值，找出数组中和为目标值的两个数。

你可以假设每个输入只对应一种答案，且同样的元素不能被重复利用。

示例:

给定 nums = [2, 7, 11, 15], target = 9

因为 nums[0] + nums[1] = 2 + 7 = 9

所以返回 [0, 1]7、反转整数

给定一个 32 位有符号整数，将整数中的数字进行反转。

示例1:

输入: 123

输出: 321

示例 2:

输入: -123

输出: -321

示例 3:

输入: 120

输出: 21

注意:

假设我们的环境只能存储 32 位有符号整数，其数值范围是 [-231, 231- 1]。根据这个假设，如果反转后的整数溢出，则返回 0。9、回文数

判断一个整数是否是回文数。回文数是指正序（从左向右）和倒序（从右向左）读都是一样的整数。

示例 1:

输入: 121

输出: true

示例2:

输入: -121

输出: false

解释: 从左向右读, 为 -121 。 从右向左读, 为 121- 。因此它不是一个回文数。

示例 3:

输入: 10

输出: false

解释: 从右向左读, 为 01 。因此它不是一个回文数。

进阶:

你能不将整数转为字符串来解决这个问题吗？13、罗马数字转整数

罗马数字包含以下七种字符：I，V，X，L，C，D和M。

字符 数值

I 1

V 5

X 10

L 50

C 100

D 500

M 1000

例如， 罗马数字 2 写做II，即为两个并列的 1。12 写做XII，即为X+II。 27 写做XXVII, 即为XX+V+II。

通常情况下，罗马数字中小的数字在大的数字的右边。但也存在特例，例如 4 不写做IIII，而是IV。数字 1 在数字 5 的左边，所表示的数等于大数 5 减小数 1 得到的数值 4 。同样地，数字 9 表示为IX。这个特殊的规则只适用于以下六种情况：

I可以放在V(5) 和X(10) 的左边，来表示 4 和 9。

X可以放在L(50) 和C(100) 的左边，来表示 40 和90。

C可以放在D(500) 和M(1000) 的左边，来表示400 和900。

给定一个罗马数字，将其转换成整数。输入确保在 1到 3999 的范围内。

示例1:

输入:"III"

输出: 3

示例2:

输入:"IV"

输出: 4

示例3:

输入:"IX"

输出: 9

示例4:

输入:"LVIII"

输出: 58

解释: C = 100, L = 50, XXX = 30, III = 3.

示例5:

输入:"MCMXCIV"

输出: 1994

解释: M = 1000, CM = 900, XC = 90, IV = 4.14、最长公共前缀

编写一个函数来查找字符串数组中的最长公共前缀。

如果不存在公共前缀，返回空字符串""。

示例1:

输入: ["flower","flow","flight"]

输出: "fl"

示例2:

输入: ["dog","racecar","car"]

输出: ""

解释: 输入不存在公共前缀。

说明:

所有输入只包含小写字母a-z。20、有效的括号

给定一个只包括 '('，')'，'{'，'}'，'['，']'的字符串，判断字符串是否有效。

有效字符串需满足：

左括号必须用相同类型的右括号闭合。

左括号必须以正确的顺序闭合。

注意空字符串可被认为是有效字符串。

示例 1:

输入: "()"

输出: true

示例2:

输入: "()[]{}"

输出: true

示例3:

输入: "(]"

输出: false

示例4:

输入: "([)]"

输出: false

示例5:

输入: "{[]}"

输出: true21、合并两个有序链表

将两个有序链表合并为一个新的有序链表并返回。新链表是通过拼接给定的两个链表的所有节点组成的。

示例：

输入：1->2->4, 1->3->4

输出：1->1->2->3->4->426、删除排序数组中的重复项

给定一个排序数组，你需要在原地删除重复出现的元素，使得每个元素只出现一次，返回移除后数组的新长度。

不要使用额外的数组空间，你必须在原地修改输入数组并在使用 O(1) 额外空间的条件下完成。

示例1:

给定数组 nums = [1,1,2],

函数应该返回新的长度 2, 并且原数组 nums 的前两个元素被修改为 1, 2。

你不需要考虑数组中超出新长度后面的元素。

示例2:

给定 nums = [0,0,1,1,1,2,2,3,3,4],

函数应该返回新的长度 5, 并且原数组 nums 的前五个元素被修改为 0, 1, 2, 3, 4。

你不需要考虑数组中超出新长度后面的元素。

说明:

为什么返回数值是整数，但输出的答案是数组呢?

请注意，输入数组是以'引用'方式传递的，这意味着在函数里修改输入数组对于调用者是可见的。

你可以想象内部操作如下:

// nums 是以'引用'方式传递的。也就是说，不对实参做任何拷贝

int len = removeDuplicates(nums);

// 在函数里修改输入数组对于调用者是可见的。

// 根据你的函数返回的长度, 它会打印出数组中该长度范围内的所有元素。

for (int i = 0; i < len; i++) {

print(nums[i]);

}27、移除元素

给定一个数组 nums和一个值 val，你需要原地移除所有数值等于val的元素，返回移除后数组的新长度。

不要使用额外的数组空间，你必须在原地修改输入数组并在使用 O(1) 额外空间的条件下完成。

元素的顺序可以改变。你不需要考虑数组中超出新长度后面的元素。

示例 1:

给定 nums = [3,2,2,3], val = 3,

函数应该返回新的长度 2, 并且 nums 中的前两个元素均为 2。

你不需要考虑数组中超出新长度后面的元素。

示例2:

给定 nums = [0,1,2,2,3,0,4,2], val = 2,

函数应该返回新的长度 5, 并且 nums 中的前五个元素为 0, 1, 3, 0, 4。

注意这五个元素可为任意顺序。

你不需要考虑数组中超出新长度后面的元素。

说明:

为什么返回数值是整数，但输出的答案是数组呢?

请注意，输入数组是以'引用'方式传递的，这意味着在函数里修改输入数组对于调用者是可见的。

你可以想象内部操作如下:

// nums 是以'引用'方式传递的。也就是说，不对实参作任何拷贝

int len = removeElement(nums, val);

// 在函数里修改输入数组对于调用者是可见的。

// 根据你的函数返回的长度, 它会打印出数组中该长度范围内的所有元素。

for (int i = 0; i < len; i++) {

print(nums[i]);

}28、实现strStr()

实现strStr()函数。

给定一个haystack 字符串和一个 needle 字符串，在 haystack 字符串中找出 needle 字符串出现的第一个位置 (从0开始)。如果不存在，则返回 -1。

示例 1:

输入: haystack = "hello", needle = "ll"

输出: 2

示例 2:

输入: haystack = "aaaaa", needle = "bba"

输出: -1

说明:

当needle是空字符串时，我们应当返回什么值呢？这是一个在面试中很好的问题。

对于本题而言，当needle是空字符串时我们应当返回 0 。这与C语言的strstr()以及 Java的indexOf()定义相符。35、搜索插入位置

给定一个排序数组和一个目标值，在数组中找到目标值，并返回其索引。如果目标值不存在于数组中，返回它将会被按顺序插入的位置。

你可以假设数组中无重复元素。

示例 1:

输入: [1,3,5,6], 5

输出: 2

示例2:

输入: [1,3,5,6], 2

输出: 1

示例 3:

输入: [1,3,5,6], 7

输出: 4

示例 4:

输入: [1,3,5,6], 0

输出: 038、报数

报数序列是指一个整数序列，按照其中的整数的顺序进行报数，得到下一个数。其前五项如下：

1. 1

2. 11

3. 21

4. 1211

5. 111221

1被读作"one 1"("一个一") , 即11。

11 被读作"two 1s"("两个一"）, 即21。

21 被读作"one 2", "one 1"（"一个二","一个一"), 即1211。

给定一个正整数 n ，输出报数序列的第 n 项。

注意：整数顺序将表示为一个字符串。

示例1:

输入: 1

输出: "1"

示例 2:

输入: 4

输出: "1211"53、最大子序和

给定一个整数数组 nums，找到一个具有最大和的连续子数组（子数组最少包含一个元素），返回其最大和。

示例:

输入: [-2,1,-3,4,-1,2,1,-5,4],

输出: 6

解释:连续子数组[4,-1,2,1] 的和最大，为6。

进阶:

如果你已经实现复杂度为 O(n) 的解法，尝试使用更为精妙的分治法求解。58、最后一个单词的长度

给定一个仅包含大小写字母和空格' '的字符串，返回其最后一个单词的长度。

如果不存在最后一个单词，请返回 0。

说明：一个单词是指由字母组成，但不包含任何空格的字符串。

示例:

输入: "Hello World"

输出: 566、加一

给定一个由整数组成的非空数组所表示的非负整数，在该数的基础上加一。

最高位数字存放在数组的首位， 数组中每个元素只存储一个数字。

你可以假设除了整数 0 之外，这个整数不会以零开头。

示例1:

输入: [1,2,3]

输出: [1,2,4]

解释: 输入数组表示数字 123。

示例2:

输入: [4,3,2,1]

输出: [4,3,2,2]

解释: 输入数组表示数字 4321。67、二进制求和

给定两个二进制字符串，返回他们的和（用二进制表示）。

输入为非空字符串且只包含数字1和0。

示例1:

输入: a = "11", b = "1"

输出: "100"

示例2:

输入: a = "1010", b = "1011"

输出: "10101"69、x 的平方根

实现int sqrt(int x)函数。

计算并返回x的平方根，其中x 是非负整数。

由于返回类型是整数，结果只保留整数的部分，小数部分将被舍去。

示例 1:

输入: 4

输出: 2

示例 2:

输入: 8

输出: 2

说明: 8 的平方根是 2.82842...,

由于返回类型是整数，小数部分将被舍去。70、爬楼梯

假设你正在爬楼梯。需要 n阶你才能到达楼顶。

每次你可以爬 1 或 2 个台阶。你有多少种不同的方法可以爬到楼顶呢？

注意：给定 n 是一个正整数。

示例 1：

输入： 2

输出： 2

解释： 有两种方法可以爬到楼顶。

1. 1 阶 + 1 阶

2. 2 阶

示例 2：

输入： 3

输出： 3

解释： 有三种方法可以爬到楼顶。

1. 1 阶 + 1 阶 + 1 阶

2. 1 阶 + 2 阶

3. 2 阶 + 1 阶83、删除排序链表中的重复元素

给定一个排序链表，删除所有重复的元素，使得每个元素只出现一次。

示例1:

输入: 1->1->2

输出: 1->2

示例2:

输入: 1->1->2->3->3

输出: 1->2->388、合并两个有序数组

给定两个有序整数数组nums1 和 nums2，将 nums2 合并到nums1中，使得num1 成为一个有序数组。

说明:

初始化nums1 和 nums2 的元素数量分别为m 和 n。

你可以假设nums1有足够的空间（空间大小大于或等于m + n）来保存 nums2 中的元素。

示例:

输入:

nums1 = [1,2,3,0,0,0], m = 3

nums2 = [2,5,6], n = 3

输出:[1,2,2,3,5,6]100、相同的树

给定两个二叉树，编写一个函数来检验它们是否相同。

如果两个树在结构上相同，并且节点具有相同的值，则认为它们是相同的。

示例1:

输入: 1 1

/ \ / \

2 3 2 3

[1,2,3], [1,2,3]

输出: true

示例 2:

输入: 1 1

/ \

2 2

[1,2], [1,null,2]

输出: false

示例3:

输入: 1 1

/ \ / \

2 1 1 2

[1,2,1], [1,1,2]

输出: false101、对称二叉树

给定一个二叉树，检查它是否是镜像对称的。

例如，二叉树[1,2,2,3,4,4,3] 是对称的。

1

/ \

2 2

/ \ / \

3 4 4 3

但是下面这个[1,2,2,null,3,null,3] 则不是镜像对称的:

1

/ \

2 2

\ \

3 3

说明:

如果你可以运用递归和迭代两种方法解决这个问题，会很加分。104、二叉树的最大深度

给定一个二叉树，找出其最大深度。

二叉树的深度为根节点到最远叶子节点的最长路径上的节点数。

说明:叶子节点是指没有子节点的节点。

示例：

给定二叉树 [3,9,20,null,null,15,7]，

3

/ \

9 20

/ \

15 7

返回它的最大深度3 。107、二叉树的层次遍历 II

给定一个二叉树，返回其节点值自底向上的层次遍历。 （即按从叶子节点所在层到根节点所在的层，逐层从左向右遍历）

例如：

给定二叉树 [3,9,20,null,null,15,7],

3

/ \

9 20

/ \

15 7

返回其自底向上的层次遍历为：

[

[15,7],

[9,20],

[3]

]108、将有序数组转换为二叉搜索树

将一个按照升序排列的有序数组，转换为一棵高度平衡二叉搜索树。

本题中，一个高度平衡二叉树是指一个二叉树每个节点的左右两个子树的高度差的绝对值不超过 1。

示例:

给定有序数组: [-10,-3,0,5,9],

一个可能的答案是：[0,-3,9,-10,null,5]，它可以表示下面这个高度平衡二叉搜索树：

0

/ \

-3 9

/ /

-10 5110、平衡二叉树

给定一个二叉树，判断它是否是高度平衡的二叉树。

本题中，一棵高度平衡二叉树定义为：

一个二叉树每个节点的左右两个子树的高度差的绝对值不超过1。

示例 1:

给定二叉树 [3,9,20,null,null,15,7]

3

/ \

9 20

/ \

15 7

返回 true 。

示例 2:

给定二叉树 [1,2,2,3,3,null,null,4,4]

1

/ \

2 2

/ \

3 3

/ \

4 4

返回false 。111、二叉树的最小深度

给定一个二叉树，找出其最小深度。

最小深度是从根节点到最近叶子节点的最短路径上的节点数量。

说明:叶子节点是指没有子节点的节点。

示例:

给定二叉树[3,9,20,null,null,15,7],

3

/ \

9 20

/ \

15 7

返回它的最小深度 2.112、路径总和

给定一个二叉树和一个目标和，判断该树中是否存在根节点到叶子节点的路径，这条路径上所有节点值相加等于目标和。

说明:叶子节点是指没有子节点的节点。

示例:

给定如下二叉树，以及目标和 sum = 22，

5

/ \

4 8

/ / \

11 13 4

/ \ \

7 2 1

返回 true, 因为存在目标和为 22 的根节点到叶子节点的路径 5->4->11->2。118、杨辉三角

给定一个非负整数numRows，生成杨辉三角的前numRows行。

在杨辉三角中，每个数是它左上方和右上方的数的和。

示例:

输入: 5

输出:

[

[1],

[1,1],

[1,2,1],

[1,3,3,1],

[1,4,6,4,1]

]119、杨辉三角 II

给定一个非负索引k，其中 k<=33，返回杨辉三角的第 k 行。

在杨辉三角中，每个数是它左上方和右上方的数的和。

示例:

输入: 3

输出: [1,3,3,1]

进阶：

你可以优化你的算法到 O(k) 空间复杂度吗？121、买卖股票的最佳时机

给定一个数组，它的第i 个元素是一支给定股票第 i 天的价格。

如果你最多只允许完成一笔交易（即买入和卖出一支股票），设计一个算法来计算你所能获取的最大利润。

注意你不能在买入股票前卖出股票。

示例 1:

输入: [7,1,5,3,6,4]

输出: 5

解释: 在第 2 天（股票价格 = 1）的时候买入，在第 5 天（股票价格 = 6）的时候卖出，最大利润 = 6-1 = 5 。

注意利润不能是 7-1 = 6, 因为卖出价格需要大于买入价格。

示例 2:

输入: [7,6,4,3,1]

输出: 0

解释: 在这种情况下, 没有交易完成, 所以最大利润为 0。122、买卖股票的最佳时机 II

给定一个数组，它的第i 个元素是一支给定股票第 i 天的价格。

设计一个算法来计算你所能获取的最大利润。你可以尽可能地完成更多的交易（多次买卖一支股票）。

注意：你不能同时参与多笔交易（你必须在再次购买前出售掉之前的股票）。

示例 1:

输入: [7,1,5,3,6,4]

输出: 7

解释: 在第 2 天（股票价格 = 1）的时候买入，在第 3 天（股票价格 = 5）的时候卖出, 这笔交易所能获得利润 = 5-1 = 4 。

随后，在第 4 天（股票价格 = 3）的时候买入，在第 5 天（股票价格 = 6）的时候卖出, 这笔交易所能获得利润 = 6-3 = 3 。

示例 2:

输入: [1,2,3,4,5]

输出: 4

解释: 在第 1 天（股票价格 = 1）的时候买入，在第 5 天 （股票价格 = 5）的时候卖出, 这笔交易所能获得利润 = 5-1 = 4 。

注意你不能在第 1 天和第 2 天接连购买股票，之后再将它们卖出。

因为这样属于同时参与了多笔交易，你必须在再次购买前出售掉之前的股票。

示例3:

输入: [7,6,4,3,1]

输出: 0

解释: 在这种情况下, 没有交易完成, 所以最大利润为 0。125、验证回文串

给定一个字符串，验证它是否是回文串，只考虑字母和数字字符，可以忽略字母的大小写。

说明：本题中，我们将空字符串定义为有效的回文串。

示例 1:

输入: "A man, a plan, a canal: Panama"

输出: true

示例 2:

输入: "race a car"

输出: false136、只出现一次的数字

给定一个非空整数数组，除了某个元素只出现一次以外，其余每个元素均出现两次。找出那个只出现了一次的元素。

说明：

你的算法应该具有线性时间复杂度。 你可以不使用额外空间来实现吗？

示例 1:

输入: [2,2,1]

输出: 1

示例2:

输入: [4,1,2,1,2]

输出: 4141、环形链表

给定一个链表，判断链表中是否有环。

进阶：

你能否不使用额外空间解决此题？155、最小栈

设计一个支持 push，pop，top 操作，并能在常数时间内检索到最小元素的栈。

push(x)-- 将元素 x 推入栈中。

pop()-- 删除栈顶的元素。

top()-- 获取栈顶元素。

getMin() -- 检索栈中的最小元素。

示例:

MinStack minStack = new MinStack();

minStack.push(-2);

minStack.push(0);

minStack.push(-3);

minStack.getMin(); --> 返回 -3.

minStack.pop();

minStack.top(); --> 返回 0.

minStack.getMin(); --> 返回 -2.160、相交链表

编写一个程序，找到两个单链表相交的起始节点。

例如，下面的两个链表：

A: a1 → a2

↘

c1 → c2 → c3

↗

B: b1 → b2 → b3

在节点 c1 开始相交。

注意：

如果两个链表没有交点，返回 null.

在返回结果后，两个链表仍须保持原有的结构。

可假定整个链表结构中没有循环。

程序尽量满足 O(n) 时间复杂度，且仅用 O(1) 内存。

致谢:

特别感谢@stellari 添加此问题并创建所有测试用例。167、两数之和 II - 输入有序数组

给定一个已按照升序排列的有序数组，找到两个数使得它们相加之和等于目标数。

函数应该返回这两个下标值 index1 和 index2，其中 index1必须小于index2。

说明:

返回的下标值（index1 和 index2）不是从零开始的。

你可以假设每个输入只对应唯一的答案，而且你不可以重复使用相同的元素。

示例:

输入: numbers = [2, 7, 11, 15], target = 9

输出: [1,2]

解释: 2 与 7 之和等于目标数 9 。因此 index1 = 1, index2 = 2 。168、Excel表列名称

给定一个正整数，返回它在 Excel 表中相对应的列名称。

例如，

1 -> A

2 -> B

3 -> C

...

26 -> Z

27 -> AA

28 -> AB

...

示例 1:

输入: 1

输出: "A"

示例2:

输入: 28

输出: "AB"

示例3:

输入: 701

输出: "ZY"169、求众数

给定一个大小为 n 的数组，找到其中的众数。众数是指在数组中出现次数大于 n/2 的元素。

你可以假设数组是非空的，并且给定的数组总是存在众数。

示例1:

输入: [3,2,3]

输出: 3

示例2:

输入: [2,2,1,1,1,2,2]

输出: 2171、Excel表列序号

给定一个Excel表格中的列名称，返回其相应的列序号。

例如，

A -> 1

B -> 2

C -> 3

...

Z -> 26

AA -> 27

AB -> 28

...

示例 1:

输入: "A"

输出: 1

示例2:

输入: "AB"

输出: 28

示例3:

输入: "ZY"

输出: 701

致谢：

特别感谢@ts添加此问题并创建所有测试用例。172、阶乘后的零

给定一个整数 n，返回 n! 结果尾数中零的数量。

示例 1:

输入: 3

输出: 0

解释:3! = 6, 尾数中没有零。

示例2:

输入: 5

输出: 1

解释:5! = 120, 尾数中有 1 个零.

说明: 你算法的时间复杂度应为O(logn)。175、组合两个表

表1: Person

+-------------+---------+

| 列名 | 类型 |

+-------------+---------+

| PersonId | int |

| FirstName | varchar |

| LastName | varchar |

+-------------+---------+

PersonId 是上表主键

表2: Address

+-------------+---------+

| 列名 | 类型 |

+-------------+---------+

| AddressId | int |

| PersonId | int |

| City | varchar |

| State | varchar |

+-------------+---------+

AddressId 是上表主键

编写一个 SQL 查询，满足条件：无论 person 是否有地址信息，都需要基于上述两表提供person 的以下信息：

FirstName, LastName, City, State176、第二高的薪水

编写一个 SQL 查询，获取 Employee表中第二高的薪水（Salary）。

+----+--------+

| Id | Salary |

+----+--------+

| 1 | 100 |

| 2 | 200 |

| 3 | 300 |

+----+--------+

例如上述Employee表，SQL查询应该返回200 作为第二高的薪水。如果不存在第二高的薪水，那么查询应返回 null。

+---------------------+

| SecondHighestSalary |

+---------------------+

| 200 |

+---------------------+181、超过经理收入的员工

Employee表包含所有员工，他们的经理也属于员工。每个员工都有一个 Id，此外还有一列对应员工的经理的 Id。

+----+-------+--------+-----------+

| Id | Name | Salary | ManagerId |

+----+-------+--------+-----------+

| 1 | Joe | 70000 | 3 |

| 2 | Henry | 80000 | 4 |

| 3 | Sam | 60000 | NULL |

| 4 | Max | 90000 | NULL |

+----+-------+--------+-----------+

给定Employee表，编写一个 SQL 查询，该查询可以获取收入超过他们经理的员工的姓名。在上面的表格中，Joe 是唯一一个收入超过他的经理的员工。

+----------+

| Employee |

+----------+

| Joe |

+----------+182、查找重复的电子邮箱

编写一个 SQL 查询，查找Person 表中所有重复的电子邮箱。

示例：

+----+---------+

| Id | Email |

+----+---------+

| 1 | a@b.com |

| 2 | c@d.com |

| 3 | a@b.com |

+----+---------+

根据以上输入，你的查询应返回以下结果：

+---------+

| Email |

+---------+

| a@b.com |

+---------+

说明：所有电子邮箱都是小写字母。183、从不订购的客户

某网站包含两个表，Customers 表和 Orders 表。编写一个 SQL 查询，找出所有从不订购任何东西的客户。

Customers 表：

+----+-------+

| Id | Name |

+----+-------+

| 1 | Joe |

| 2 | Henry |

| 3 | Sam |

| 4 | Max |

+----+-------+

Orders 表：

+----+------------+

| Id | CustomerId |

+----+------------+

| 1 | 3 |

| 2 | 1 |

+----+------------+

例如给定上述表格，你的查询应返回：

+-----------+

| Customers |

+-----------+

| Henry |

| Max |

+-----------+189、旋转数组

给定一个数组，将数组中的元素向右移动k个位置，其中k是非负数。

示例 1:

输入: [1,2,3,4,5,6,7] 和 k = 3

输出: [5,6,7,1,2,3,4]

解释:

向右旋转 1 步: [7,1,2,3,4,5,6]

向右旋转 2 步: [6,7,1,2,3,4,5]

向右旋转 3 步: [5,6,7,1,2,3,4]

示例2:

输入: [-1,-100,3,99] 和 k = 2

输出: [3,99,-1,-100]

解释:

向右旋转 1 步: [99,-1,-100,3]

向右旋转 2 步: [3,99,-1,-100]

说明:

尽可能想出更多的解决方案，至少有三种不同的方法可以解决这个问题。

要求使用空间复杂度为O(1) 的原地算法。190、颠倒二进制位

颠倒给定的 32 位无符号整数的二进制位。

示例:

输入: 43261596

输出: 964176192

解释: 43261596 的二进制表示形式为 00000010100101000001111010011100 ，

返回 964176192，其二进制表示形式为 00111001011110000010100101000000 。

进阶:

如果多次调用这个函数，你将如何优化你的算法？191、位1的个数

编写一个函数，输入是一个无符号整数，返回其二进制表达式中数字位数为 '1'的个数（也被称为汉明重量）。

示例:

输入: 11

输出: 3

解释: 整数 11 的二进制表示为 00000000000000000000000000001011

示例2:

输入: 128

输出: 1

解释: 整数 128 的二进制表示为 00000000000000000000000010000000195、第十行

给定一个文本文件 file.txt，请只打印这个文件中的第十行。

示例:

假设 file.txt 有如下内容：

Line 1

Line 2

Line 3

Line 4

Line 5

Line 6

Line 7

Line 8

Line 9

Line 10

你的脚本应当显示第十行：

Line 10

说明:

1. 如果文件少于十行，你应当输出什么？

2. 至少有三种不同的解法，请尝试尽可能多的方法来解题。196、删除重复的电子邮箱

编写一个 SQL 查询，来删除 Person 表中所有重复的电子邮箱，重复的邮箱里只保留 Id 最小 的那个。

+----+------------------+

| Id | Email |

+----+------------------+

| 1 | john@example.com |

| 2 | bob@example.com |

| 3 | john@example.com |

+----+------------------+

Id 是这个表的主键。

例如，在运行你的查询语句之后，上面的 Person 表应返回以下几行:

+----+------------------+

| Id | Email |

+----+------------------+

| 1 | john@example.com |

| 2 | bob@example.com |

+----+------------------+197、上升的温度

给定一个 Weather 表，编写一个 SQL 查询，来查找与之前（昨天的）日期相比温度更高的所有日期的 Id。

+---------+------------------+------------------+

| Id(INT) | RecordDate(DATE) | Temperature(INT) |

+---------+------------------+------------------+

| 1 | 2015-01-01 | 10 |

| 2 | 2015-01-02 | 25 |

| 3 | 2015-01-03 | 20 |

| 4 | 2015-01-04 | 30 |

+---------+------------------+------------------+

例如，根据上述给定的 Weather 表格，返回如下 Id:

+----+

| Id |

+----+

| 2 |

| 4 |

+----+198、打家劫舍

你是一个专业的小偷，计划偷窃沿街的房屋。每间房内都藏有一定的现金，影响你偷窃的唯一制约因素就是相邻的房屋装有相互连通的防盗系统，如果两间相邻的房屋在同一晚上被小偷闯入，系统会自动报警。

给定一个代表每个房屋存放金额的非负整数数组，计算你在不触动警报装置的情况下，能够偷窃到的最高金额。

示例 1:

输入: [1,2,3,1]

输出: 4

解释: 偷窃 1 号房屋 (金额 = 1) ，然后偷窃 3 号房屋 (金额 = 3)。

偷窃到的最高金额 = 1 + 3 = 4 。

示例 2:

输入: [2,7,9,3,1]

输出: 12

解释: 偷窃 1 号房屋 (金额 = 2), 偷窃 3 号房屋 (金额 = 9)，接着偷窃 5 号房屋 (金额 = 1)。

偷窃到的最高金额 = 2 + 9 + 1 = 12 。202、快乐数

编写一个算法来判断一个数是不是&ldquo;快乐数&rdquo;。

一个&ldquo;快乐数&rdquo;定义为：对于一个正整数，每一次将该数替换为它每个位置上的数字的平方和，然后重复这个过程直到这个数变为 1，也可能是无限循环但始终变不到 1。如果可以变为 1，那么这个数就是快乐数。

示例:

输入: 19

输出: true

解释:

12 + 92 = 82

82 + 22 = 68

62 + 82 = 100

12 + 02 + 02 = 1203、删除链表中的节点

删除链表中等于给定值 val 的所有节点。

示例:

输入: 1->2->6->3->4->5->6, val = 6

输出: 1->2->3->4->5204、计数质数

统计所有小于非负整数 n 的质数的数量。

示例:

输入: 10

输出: 4

解释: 小于 10 的质数一共有 4 个, 它们是 2, 3, 5, 7 。205、同构字符串

给定两个字符串 s 和 t，判断它们是否是同构的。

如果 s 中的字符可以被替换得到 t ，那么这两个字符串是同构的。

所有出现的字符都必须用另一个字符替换，同时保留字符的顺序。两个字符不能映射到同一个字符上，但字符可以映射自己本身。

示例 1:

输入: s = "egg", t = "add"

输出: true

示例 2:

输入: s = "foo", t = "bar"

输出: false

示例 3:

输入: s = "paper", t = "title"

输出: true

说明:

你可以假设 s 和 t 具有相同的长度。206、反转链表

反转一个单链表。

示例:

输入: 1->2->3->4->5->NULL

输出: 5->4->3->2->1->NULL

进阶:

你可以迭代或递归地反转链表。你能否用两种方法解决这道题？217、存在重复元素

给定一个整数数组，判断是否存在重复元素。

如果任何值在数组中出现至少两次，函数返回 true。如果数组中每个元素都不相同，则返回 false。

示例 1:

输入: [1,2,3,1]

输出: true

示例 2:

输入: [1,2,3,4]

输出: false

示例 3:

输入: [1,1,1,3,3,4,3,2,4,2]

输出: true219、存在重复元素 II

给定一个整数数组和一个整数 k，判断数组中是否存在两个不同的索引 i 和 j，使得 nums [i] = nums [j]，并且 i 和 j 的差的绝对值最大为 k。

示例 1:

输入: nums = [1,2,3,1], k = 3

输出: true

示例 2:

输入: nums = [1,0,1,1], k = 1

输出: true

示例 3:

输入: nums = [1,2,3,1,2,3], k = 2

输出: false225、用队列实现栈

使用队列实现栈的下列操作：

push(x) -- 元素 x 入栈

pop() -- 移除栈顶元素

top() -- 获取栈顶元素

empty() -- 返回栈是否为空

注意:

你只能使用队列的基本操作-- 也就是 push to back, peek/pop from front, size, 和 is empty 这些操作是合法的。

你所使用的语言也许不支持队列。 你可以使用 list 或者 deque（双端队列）来模拟一个队列 , 只要是标准的队列操作即可。

你可以假设所有操作都是有效的（例如, 对一个空的栈不会调用 pop 或者 top 操作）。226、翻转二叉树

翻转一棵二叉树。

示例：

输入：

4

/ \

2 7

/ \ / \

1 3 6 9

输出：

4

/ \

7 2

/ \ / \

9 6 3 1

备注:

这个问题是受到 Max Howell 的 原问题 启发的 ：

谷歌：我们90％的工程师使用您编写的软件(Homebrew)，但是您却无法在面试时在白板上写出翻转二叉树这道题，这太糟糕了。231、2的幂

给定一个整数，编写一个函数来判断它是否是 2 的幂次方。

示例 1:

输入: 1

输出: true

解释: 20 = 1

示例 2:

输入: 16

输出: true

解释: 24 = 16

示例 3:

输入: 218

输出: false232、用栈实现队列

使用栈实现队列的下列操作：

push(x) -- 将一个元素放入队列的尾部。

pop() -- 从队列首部移除元素。

peek() -- 返回队列首部的元素。

empty() -- 返回队列是否为空。

示例:

MyQueue queue = new MyQueue();

queue.push(1);

queue.push(2);

queue.peek(); // 返回 1

queue.pop(); // 返回 1

queue.empty(); // 返回 false

说明:

你只能使用标准的栈操作 -- 也就是只有 push to top, peek/pop from top, size, 和 is empty 操作是合法的。

你所使用的语言也许不支持栈。你可以使用 list 或者 deque（双端队列）来模拟一个栈，只要是标准的栈操作即可。

假设所有操作都是有效的 （例如，一个空的队列不会调用 pop 或者 peek 操作）。234、回文链表

请判断一个链表是否为回文链表。

示例 1:

输入: 1->2

输出: false

示例 2:

输入: 1->2->2->1

输出: true

进阶：

你能否用 O(n) 时间复杂度和 O(1) 空间复杂度解决此题？235、二叉搜索树的最近公共祖先

给定一个二叉搜索树, 找到该树中两个指定节点的最近公共祖先。

百度百科中最近公共祖先的定义为：&ldquo;对于有根树 T 的两个结点 p、q，最近公共祖先表示为一个结点 x，满足 x 是 p、q 的祖先且 x 的深度尽可能大（一个节点也可以是它自己的祖先）。&rdquo;

例如，给定如下二叉搜索树: root = [6,2,8,0,4,7,9,null,null,3,5]

\_\_\_\_\_\_\_6\_\_\_\_\_\_

/ \

\_\_\_2\_\_ \_\_\_8\_\_

/ \ / \

0 \_4 7 9

/ \

3 5

示例 1:

输入: root = [6,2,8,0,4,7,9,null,null,3,5], p = 2, q = 8

输出: 6

解释: 节点 2 和节点 8 的最近公共祖先是 6。

示例 2:

输入: root = [6,2,8,0,4,7,9,null,null,3,5], p = 2, q = 4

输出: 2

解释: 节点 2 和节点 4 的最近公共祖先是 2, 因为根据定义最近公共祖先节点可以为节点本身。

说明:

所有节点的值都是唯一的。

p、q 为不同节点且均存在于给定的二叉搜索树中。237、删除链表中的节点

请编写一个函数，使其可以删除某个链表中给定的（非末尾）节点，你将只被给定要求被删除的节点。

现有一个链表 -- head = [4,5,1,9]，它可以表示为:

4 -> 5 -> 1 -> 9

示例 1:

输入: head = [4,5,1,9], node = 5

输出: [4,1,9]

解释: 给定你链表中值为 5 的第二个节点，那么在调用了你的函数之后，该链表应变为 4 -> 1 -> 9.

示例 2:

输入: head = [4,5,1,9], node = 1

输出: [4,5,9]

解释: 给定你链表中值为 1 的第三个节点，那么在调用了你的函数之后，该链表应变为 4 -> 5 -> 9.

说明:

链表至少包含两个节点。

链表中所有节点的值都是唯一的。

给定的节点为非末尾节点并且一定是链表中的一个有效节点。

不要从你的函数中返回任何结果。242、有效的字母异位词

给定两个字符串 s 和 t ，编写一个函数来判断 t 是否是 s 的一个字母异位词。

示例 1:

输入: s = "anagram", t = "nagaram"

输出: true

示例 2:

输入: s = "rat", t = "car"

输出: false

说明:

你可以假设字符串只包含小写字母。

进阶:

如果输入字符串包含 unicode 字符怎么办？你能否调整你的解法来应对这种情况？257、二叉树的所有路径

给定一个二叉树，返回所有从根节点到叶子节点的路径。

说明: 叶子节点是指没有子节点的节点。

示例:

输入:

1

/ \

2 3

\

5

输出: ["1->2->5", "1->3"]

解释: 所有根节点到叶子节点的路径为: 1->2->5, 1->3258、各位相加

给定一个非负整数 num，反复将各个位上的数字相加，直到结果为一位数。

示例:

输入: 38

输出: 2

解释: 各位相加的过程为：3 + 8 = 11, 1 + 1 = 2。 由于 2 是一位数，所以返回 2。

进阶:

你可以不使用循环或者递归，且在 O(1) 时间复杂度内解决这个问题吗？263、丑数

编写一个程序判断给定的数是否为丑数。

丑数就是只包含质因数 2, 3, 5 的正整数。

示例 1:

输入: 6

输出: true

解释: 6 = 2 &times; 3

示例 2:

输入: 8

输出: true

解释: 8 = 2 &times; 2 &times; 2

示例 3:

输入: 14

输出: false

解释: 14 不是丑数，因为它包含了另外一个质因数 7。

说明：

1 是丑数。

输入不会超过 32 位有符号整数的范围: [-231, 231 - 1]。268、缺失数字

给定一个包含 0, 1, 2, ..., n 中 n 个数的序列，找出 0 .. n 中没有出现在序列中的那个数。

示例 1:

输入: [3,0,1]

输出: 2

示例 2:

输入: [9,6,4,2,3,5,7,0,1]

输出: 8

说明:

你的算法应具有线性时间复杂度。你能否仅使用额外常数空间来实现?278、第一个错误的版本

你是产品经理，目前正在带领一个团队开发新的产品。不幸的是，你的产品的最新版本没有通过质量检测。由于每个版本都是基于之前的版本开发的，所以错误的版本之后的所有版本都是错的。

假设你有 n 个版本 [1, 2, ..., n]，你想找出导致之后所有版本出错的第一个错误的版本。

你可以通过调用 bool isBadVersion(version) 接口来判断版本号 version 是否在单元测试中出错。实现一个函数来查找第一个错误的版本。你应该尽量减少对调用 API 的次数。

示例:

给定 n = 5，并且 version = 4 是第一个错误的版本。

调用 isBadVersion(3) -> false

调用 isBadVersion(5) -> true

调用 isBadVersion(4) -> true

所以，4 是第一个错误的版本。 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。

示例:

输入: [0,1,0,3,12]

输出: [1,3,12,0,0]

说明:

必须在原数组上操作，不能拷贝额外的数组。

尽量减少操作次数。290、单词模式

给定一种 pattern(模式) 和一个字符串 str ，判断 str 是否遵循相同的模式。

这里的遵循指完全匹配，例如， pattern 里的每个字母和字符串 str 中的每个非空单词之间存在着双向连接的对应模式。

示例1:

输入: pattern = "abba", str = "dog cat cat dog"

输出: true

示例 2:

输入:pattern = "abba", str = "dog cat cat fish"

输出: false

示例 3:

输入: pattern = "aaaa", str = "dog cat cat dog"

输出: false

示例 4:

输入: pattern = "abba", str = "dog dog dog dog"

输出: false

说明:

你可以假设 pattern 只包含小写字母， str 包含了由单个空格分隔的小写字母。 292、Nim游戏

你和你的朋友，两个人一起玩 Nim游戏：桌子上有一堆石头，每次你们轮流拿掉 1 - 3 块石头。 拿掉最后一块石头的人就是获胜者。你作为先手。

你们是聪明人，每一步都是最优解。 编写一个函数，来判断你是否可以在给定石头数量的情况下赢得游戏。

示例:

输入: 4

输出: false

解释: 如果堆中有 4 块石头，那么你永远不会赢得比赛；

因为无论你拿走 1 块、2 块 还是 3 块石头，最后一块石头总是会被你的朋友拿走。303、区域和检索 - 数组不可变

给定一个整数数组 nums，求出数组从索引 i 到 j (i <= j) 范围内元素的总和，包含 i, j 两点。

示例：

给定 nums = [-2, 0, 3, -5, 2, -1]，求和函数为 sumRange()

sumRange(0, 2) -> 1

sumRange(2, 5) -> -1

sumRange(0, 5) -> -3

说明:

你可以假设数组不可变。

会多次调用 sumRange 方法。326、3的幂

给定一个整数，写一个函数来判断它是否是 3 的幂次方。

示例 1:

输入: 27

输出: true

示例 2:

输入: 0

输出: false

示例 3:

输入: 9

输出: true

示例 4:

输入: 45

输出: false

进阶：

你能不使用循环或者递归来完成本题吗？342、4的幂

给定一个整数 (32 位有符号整数)，请编写一个函数来判断它是否是 4 的幂次方。

示例 1:

输入: 16

输出: true

示例 2:

输入: 5

输出: false

进阶：

你能不使用循环或者递归来完成本题吗？344、反转字符串

编写一个函数，其作用是将输入的字符串反转过来。

示例 1:

输入: "hello"

输出: "olleh"

示例 2:

输入: "A man, a plan, a canal: Panama"

输出: "amanaP :lanac a ,nalp a ,nam A"345、反转字符串中的元音字母

编写一个函数，以字符串作为输入，反转该字符串中的元音字母。

示例 1:

输入: "hello"

输出: "holle"

示例 2:

输入: "leetcode"

输出: "leotcede"

说明:

元音字母不包含字母"y"。349、两个数组的交集

给定两个数组，编写一个函数来计算它们的交集。

示例 1:

输入: nums1 = [1,2,2,1], nums2 = [2,2]

输出: [2]

示例 2:

输入: nums1 = [4,9,5], nums2 = [9,4,9,8,4]

输出: [9,4]

说明:

输出结果中的每个元素一定是唯一的。

我们可以不考虑输出结果的顺序。350、两个数组的交集 II

给定两个数组，编写一个函数来计算它们的交集。

示例 1:

输入: nums1 = [1,2,2,1], nums2 = [2,2]

输出: [2,2]

示例 2:

输入: nums1 = [4,9,5], nums2 = [9,4,9,8,4]

输出: [4,9]

说明：

输出结果中每个元素出现的次数，应与元素在两个数组中出现的次数一致。

我们可以不考虑输出结果的顺序。

进阶:

如果给定的数组已经排好序呢？你将如何优化你的算法？

如果 nums1 的大小比 nums2 小很多，哪种方法更优？

如果 nums2 的元素存储在磁盘上，磁盘内存是有限的，并且你不能一次加载所有的元素到内存中，你该怎么办？367、有效的完全平方数

给定一个正整数 num，编写一个函数，如果 num 是一个完全平方数，则返回 True，否则返回 False。

说明：不要使用任何内置的库函数，如 sqrt。

示例 1：

输入：16

输出：True

示例 2：

输入：14

输出：False371、两整数之和

不使用运算符 + 和 - ​​​​​​​，计算两整数 ​​​​​​​a 、b ​​​​​​​之和。

示例 1:

输入: a = 1, b = 2

输出: 3

示例 2:

输入: a = -2, b = 3

输出: 1374、猜数字大小

我们正在玩一个猜数字游戏。 游戏规则如下：

我从 1 到 n 选择一个数字。 你需要猜我选择了哪个数字。

每次你猜错了，我会告诉你这个数字是大了还是小了。

你调用一个预先定义好的接口 guess(int num)，它会返回 3 个可能的结果（-1，1 或 0）：

-1 : 我的数字比较小

1 : 我的数字比较大

0 : 恭喜！你猜对了！

示例 :

输入: n = 10, pick = 6

输出: 6383、赎金信

给定一个赎金信 (ransom) 字符串和一个杂志(magazine)字符串，判断第一个字符串ransom能不能由第二个字符串magazines里面的字符构成。如果可以构成，返回 true ；否则返回 false。

(题目说明：为了不暴露赎金信字迹，要从杂志上搜索各个需要的字母，组成单词来表达意思。)

注意：

你可以假设两个字符串均只含有小写字母。

canConstruct("a", "b") -> false

canConstruct("aa", "ab") -> false

canConstruct("aa", "aab") -> true387、字符串中的第一个唯一字符

给定一个字符串，找到它的第一个不重复的字符，并返回它的索引。如果不存在，则返回 -1。

案例:

s = "leetcode"

返回 0.

s = "loveleetcode",

返回 2.

注意事项：您可以假定该字符串只包含小写字母。389、找不同

给定两个字符串 s 和 t，它们只包含小写字母。

字符串 t 由字符串 s 随机重排，然后在随机位置添加一个字母。

请找出在 t 中被添加的字母。

示例:

输入：

s = "abcd"

t = "abcde"

输出：

e

解释：

'e' 是那个被添加的字母。400、第N个数字

在无限的整数序列 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, ...中找到第 n 个数字。

注意:

n 是正数且在32为整形范围内 ( n < 231)。

示例 1:

输入:

3

输出:

3

示例 2:

输入:

11

输出:

0

说明:

第11个数字在序列 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, ... 里是0，它是10的一部分。401、二进制手表

二进制手表顶部有 4 个 LED 代表小时（0-11），底部的 6 个 LED 代表分钟（0-59）。

每个 LED 代表一个 0 或 1，最低位在右侧。

例如，上面的二进制手表读取 &ldquo;3:25&rdquo;。

给定一个非负整数 n 代表当前 LED 亮着的数量，返回所有可能的时间。

案例:

输入: n = 1

返回: ["1:00", "2:00", "4:00", "8:00", "0:01", "0:02", "0:04", "0:08", "0:16", "0:32"]

注意事项:

输出的顺序没有要求。

小时不会以零开头，比如 &ldquo;01:00&rdquo; 是不允许的，应为 &ldquo;1:00&rdquo;。

分钟必须由两位数组成，可能会以零开头，比如 &ldquo;10:2&rdquo; 是无效的，应为 &ldquo;10:02&rdquo;。404、左叶子之和

计算给定二叉树的所有左叶子之和。

示例：

3

/ \

9 20

/ \

15 7

在这个二叉树中，有两个左叶子，分别是 9 和 15，所以返回 24

405、数字转换为十六进制数

给定一个整数，编写一个算法将这个数转换为十六进制数。对于负整数，我们通常使用 补码运算 方法。

注意:

十六进制中所有字母(a-f)都必须是小写。

十六进制字符串中不能包含多余的前导零。如果要转化的数为0，那么以单个字符'0'来表示；对于其他情况，十六进制字符串中的第一个字符将不会是0字符。

给定的数确保在32位有符号整数范围内。

不能使用任何由库提供的将数字直接转换或格式化为十六进制的方法。

示例 1：

输入:

26

输出:

"1a"

示例 2：

输入:

-1

输出:

"ffffffff"409、最长回文串

给定一个包含大写字母和小写字母的字符串，找到通过这些字母构造成的最长的回文串。

在构造过程中，请注意区分大小写。比如 "Aa" 不能当做一个回文字符串。

注意:

假设字符串的长度不会超过 1010。

示例 1:

输入:

"abccccdd"

输出:

7

解释:

我们可以构造的最长的回文串是"dccaccd", 它的长度是 7。412、Fizz Buzz

写一个程序，输出从 1 到 n 数字的字符串表示。

1. 如果 n 是3的倍数，输出&ldquo;Fizz&rdquo;；

2. 如果 n 是5的倍数，输出&ldquo;Buzz&rdquo;；

3.如果 n 同时是3和5的倍数，输出 &ldquo;FizzBuzz&rdquo;。

示例：

n = 15,

返回:

[

"1",

"2",

"Fizz",

"4",

"Buzz",

"Fizz",

"7",

"8",

"Fizz",

"Buzz",

"11",

"Fizz",

"13",

"14",

"FizzBuzz"

]414、第三大的数

给定一个非空数组，返回此数组中第三大的数。如果不存在，则返回数组中最大的数。要求算法时间复杂度必须是O(n)。

示例 1:

输入: [3, 2, 1]

输出: 1

解释: 第三大的数是 1.

示例 2:

输入: [1, 2]

输出: 2

解释: 第三大的数不存在, 所以返回最大的数 2 .

示例 3:

输入: [2, 2, 3, 1]

输出: 1

解释: 注意，要求返回第三大的数，是指第三大且唯一出现的数。

存在两个值为2的数，它们都排第二。415、字符串相加

给定两个字符串形式的非负整数 num1 和num2 ，计算它们的和。

注意：

num1 和num2 的长度都小于 5100.

num1 和num2 都只包含数字 0-9.

num1 和num2 都不包含任何前导零。

你不能使用任何內建 BigInteger 库， 也不能直接将输入的字符串转换为整数形式。427、建立四叉树

我们想要使用一棵四叉树来储存一个 N x N 的布尔值网络。网络中每一格的值只会是真或假。树的根结点代表整个网络。对于每个结点, 它将被分等成四个孩子结点直到这个区域内的值都是相同的.

每个结点还有另外两个布尔变量: isLeaf 和 val。isLeaf 当这个节点是一个叶子结点时为真。val 变量储存叶子结点所代表的区域的值。

你的任务是使用一个四叉树表示给定的网络。下面的例子将有助于你理解这个问题：

给定下面这个8 x 8 网络，我们将这样建立一个对应的四叉树：

由上文的定义，它能被这样分割：

对应的四叉树应该像下面这样，每个结点由一对 (isLeaf, val) 所代表.

对于非叶子结点，val 可以是任意的，所以使用 \* 代替。

提示：

N 将小于 1000 且确保是 2 的整次幂。

如果你想了解更多关于四叉树的知识，你可以参考这个 wiki 页面。429、N叉树的层序遍历

给定一个N叉树，返回其节点值的层序遍历。 (即从左到右，逐层遍历)。

例如，给定一个 3叉树 :

返回其层序遍历:

[

[1],

[3,2,4],

[5,6]

]

说明:

树的深度不会超过 1000。

树的节点总数不会超过 5000。434、字符串中的单词数

统计字符串中的单词个数，这里的单词指的是连续的不是空格的字符。

请注意，你可以假定字符串里不包括任何不可打印的字符。

示例:

输入: "Hello, my name is John"

输出: 5437、路径总和 III

给定一个二叉树，它的每个结点都存放着一个整数值。

找出路径和等于给定数值的路径总数。

路径不需要从根节点开始，也不需要在叶子节点结束，但是路径方向必须是向下的（只能从父节点到子节点）。

二叉树不超过1000个节点，且节点数值范围是 [-1000000,1000000] 的整数。

示例：

root = [10,5,-3,3,2,null,11,3,-2,null,1], sum = 8

10

/ \

5 -3

/ \ \

3 2 11

/ \ \

3 -2 1

返回 3。和等于 8 的路径有:

1. 5 -> 3

2. 5 -> 2 -> 1

3. -3 -> 11438、找到字符串中所有字母异位词

给定一个字符串 s 和一个非空字符串 p，找到 s 中所有是 p 的字母异位词的子串，返回这些子串的起始索引。

字符串只包含小写英文字母，并且字符串 s 和 p 的长度都不超过 20100。

说明：

字母异位词指字母相同，但排列不同的字符串。

不考虑答案输出的顺序。

示例 1:

输入:

s: "cbaebabacd" p: "abc"

输出:

[0, 6]

解释:

起始索引等于 0 的子串是 "cba", 它是 "abc" 的字母异位词。

起始索引等于 6 的子串是 "bac", 它是 "abc" 的字母异位词。

示例 2:

输入:

s: "abab" p: "ab"

输出:

[0, 1, 2]

解释:

起始索引等于 0 的子串是 "ab", 它是 "ab" 的字母异位词。

起始索引等于 1 的子串是 "ba", 它是 "ab" 的字母异位词。

起始索引等于 2 的子串是 "ab", 它是 "ab" 的字母异位词。441、排列硬币

你总共有 n 枚硬币，你需要将它们摆成一个阶梯形状，第 k 行就必须正好有 k 枚硬币。

给定一个数字 n，找出可形成完整阶梯行的总行数。

n 是一个非负整数，并且在32位有符号整型的范围内。

示例 1:

n = 5

硬币可排列成以下几行:

&curren;

&curren; &curren;

&curren; &curren;

因为第三行不完整，所以返回2.

示例 2:

n = 8

硬币可排列成以下几行:

&curren;

&curren; &curren;

&curren; &curren; &curren;

&curren; &curren;

因为第四行不完整，所以返回3.443、压缩字符串

给定一组字符，使用原地算法将其压缩。

压缩后的长度必须始终小于或等于原数组长度。

数组的每个元素应该是长度为1 的字符（不是 int 整数类型）。

在完成原地修改输入数组后，返回数组的新长度。

进阶：

你能否仅使用O(1) 空间解决问题？

示例 1：

输入：

["a","a","b","b","c","c","c"]

输出：

返回6，输入数组的前6个字符应该是：["a","2","b","2","c","3"]

说明：

"aa"被"a2"替代。"bb"被"b2"替代。"ccc"被"c3"替代。

示例 2：

输入：

["a"]

输出：

返回1，输入数组的前1个字符应该是：["a"]

说明：

没有任何字符串被替代。

示例 3：

输入：

["a","b","b","b","b","b","b","b","b","b","b","b","b"]

输出：

返回4，输入数组的前4个字符应该是：["a","b","1","2"]。

说明：

由于字符"a"不重复，所以不会被压缩。"bbbbbbbbbbbb"被&ldquo;b12&rdquo;替代。

注意每个数字在数组中都有它自己的位置。

注意：

所有字符都有一个ASCII值在[35, 126]区间内。

1 <= len(chars) <= 1000。447、回旋镖的数量

给定平面上 n 对不同的点，&ldquo;回旋镖&rdquo; 是由点表示的元组 (i, j, k) ，其中 i 和 j 之间的距离和 i 和 k 之间的距离相等（需要考虑元组的顺序）。

找到所有回旋镖的数量。你可以假设 n 最大为 500，所有点的坐标在闭区间 [-10000, 10000] 中。

示例:

输入:

[[0,0],[1,0],[2,0]]

输出:

2

解释:

两个回旋镖为 [[1,0],[0,0],[2,0]] 和 [[1,0],[2,0],[0,0]]448、找到所有数组中消失的数字

给定一个范围在 1 <= a[i] <= n ( n = 数组大小 ) 的 整型数组，数组中的元素一些出现了两次，另一些只出现一次。

找到所有在 [1, n] 范围之间没有出现在数组中的数字。

您能在不使用额外空间且时间复杂度为O(n)的情况下完成这个任务吗? 你可以假定返回的数组不算在额外空间内。

示例:

输入:

[4,3,2,7,8,2,3,1]

输出:

[5,6]453、最小移动次数使数组元素相等

给定一个长度为 n 的非空整数数组，找到让数组所有元素相等的最小移动次数。每次移动可以使 n - 1 个元素增加 1。

示例:

输入:

[1,2,3]

输出:

3

解释:

只需要3次移动（注意每次移动会增加两个元素的值）：

[1,2,3] => [2,3,3] => [3,4,3] => [4,4,4]455、分发饼干

假设你是一位很棒的家长，想要给你的孩子们一些小饼干。但是，每个孩子最多只能给一块饼干。对每个孩子 i ，都有一个胃口值 gi ，这是能让孩子们满足胃口的饼干的最小尺寸；并且每块饼干 j ，都有一个尺寸 sj 。如果 sj >= gi ，我们可以将这个饼干 j 分配给孩子 i ，这个孩子会得到满足。你的目标是尽可能满足越多数量的孩子，并输出这个最大数值。

注意：

你可以假设胃口值为正。

一个小朋友最多只能拥有一块饼干。

示例 1:

输入: [1,2,3], [1,1]

输出: 1

解释:

你有三个孩子和两块小饼干，3个孩子的胃口值分别是：1,2,3。

虽然你有两块小饼干，由于他们的尺寸都是1，你只能让胃口值是1的孩子满足。

所以你应该输出1。

示例 2:

输入: [1,2], [1,2,3]

输出: 2

解释:

你有两个孩子和三块小饼干，2个孩子的胃口值分别是1,2。

你拥有的饼干数量和尺寸都足以让所有孩子满足。

所以你应该输出2.458、可怜的小猪

有1000只水桶，其中有且只有一桶装的含有毒药，其余装的都是水。它们从外观看起来都一样。如果小猪喝了毒药，它会在15分钟内死去。

问题来了，如果需要你在一小时内，弄清楚哪只水桶含有毒药，你最少需要多少只猪？

回答这个问题，并为下列的进阶问题编写一个通用算法。

进阶:

假设有 n 只水桶，猪饮水中毒后会在 m 分钟内死亡，你需要多少猪（x）就能在 p 分钟内找出&ldquo;有毒&rdquo;水桶？n只水桶里有且仅有一只有毒的桶。459、重复的子字符串

给定一个非空的字符串，判断它是否可以由它的一个子串重复多次构成。给定的字符串只含有小写英文字母，并且长度不超过10000。

示例 1:

输入: "abab"

输出: True

解释: 可由子字符串 "ab" 重复两次构成。

示例 2:

输入: "aba"

输出: False

示例 3:

输入: "abcabcabcabc"

输出: True

解释: 可由子字符串 "abc" 重复四次构成。 (或者子字符串 "abcabc" 重复两次构成。)461、汉明距离

两个整数之间的汉明距离指的是这两个数字对应二进制位不同的位置的数目。

给出两个整数 x 和 y，计算它们之间的汉明距离。

注意：

0 <= x, y < 231.

示例:

输入: x = 1, y = 4

输出: 2

解释:

1 (0 0 0 1)

4 (0 1 0 0)

&uarr; &uarr;

上面的箭头指出了对应二进制位不同的位置。463、岛屿的周长

给定一个包含 0 和 1 的二维网格地图，其中 1 表示陆地 0 表示水域。网格中的格子水平和垂直方向相连（对角线方向不相连）。整个网格被水完全包围，但其中恰好有一个岛屿（或者说，一个或多个表示陆地的格子相连组成的岛屿）。岛屿中没有&ldquo;湖&rdquo;（&ldquo;湖&rdquo; 指水域在岛屿内部且不和岛屿周围的水相连）。格子是边长为 1 的正方形。网格为长方形，且宽度和高度均不超过 100 。计算这个岛屿的周长。

示例 :

[[0,1,0,0],

[1,1,1,0],

[0,1,0,0],

[1,1,0,0]]

答案: 16

解释: 它的周长是下面图片中的 16 个黄色的边：475、供暖器

冬季已经来临。 你的任务是设计一个有固定加热半径的供暖器向所有房屋供暖。

现在，给出位于一条水平线上的房屋和供暖器的位置，找到可以覆盖所有房屋的最小加热半径。

所以，你的输入将会是房屋和供暖器的位置。你将输出供暖器的最小加热半径。

说明:

给出的房屋和供暖器的数目是非负数且不会超过 25000。

给出的房屋和供暖器的位置均是非负数且不会超过10^9。

只要房屋位于供暖器的半径内(包括在边缘上)，它就可以得到供暖。

所有供暖器都遵循你的半径标准，加热的半径也一样。

示例 1:

输入: [1,2,3],[2]

输出: 1

解释: 仅在位置2上有一个供暖器。如果我们将加热半径设为1，那么所有房屋就都能得到供暖。

示例 2:

输入: [1,2,3,4],[1,4]

输出: 1

解释: 在位置1, 4上有两个供暖器。我们需要将加热半径设为1，这样所有房屋就都能得到供暖。476、数字的补数

给定一个正整数，输出它的补数。补数是对该数的二进制表示取反。

注意:

给定的整数保证在32位带符号整数的范围内。

你可以假定二进制数不包含前导零位。

示例 1:

输入: 5

输出: 2

解释: 5的二进制表示为101（没有前导零位），其补数为010。所以你需要输出2。

示例 2:

输入: 1

输出: 0

解释: 1的二进制表示为1（没有前导零位），其补数为0。所以你需要输出0。479、最大回文数乘积

你需要找到由两个 n 位数的乘积组成的最大回文数。

由于结果会很大，你只需返回最大回文数 mod 1337得到的结果。

示例:

输入: 2

输出: 987

解释: 99 x 91 = 9009, 9009 % 1337 = 987

说明:

n 的取值范围为 [1,8]。482、密钥格式化

给定一个密钥字符串S，只包含字母，数字以及 '-'（破折号）。N 个 '-' 将字符串分成了 N+1 组。给定一个数字 K，重新格式化字符串，除了第一个分组以外，每个分组要包含 K 个字符，第一个分组至少要包含 1 个字符。两个分组之间用 '-'（破折号）隔开，并且将所有的小写字母转换为大写字母。

给定非空字符串 S 和数字 K，按照上面描述的规则进行格式化。

示例 1：

输入：S = "5F3Z-2e-9-w", K = 4

输出："5F3Z-2E9W"

解释：字符串 S 被分成了两个部分，每部分 4 个字符；

注意，两个额外的破折号需要删掉。

示例 2：

输入：S = "2-5g-3-J", K = 2

输出："2-5G-3J"

解释：字符串 S 被分成了 3 个部分，按照前面的规则描述，第一部分的字符可以少于给定的数量，其余部分皆为 2 个字符。

提示:

S 的长度不超过 12,000，K 为正整数

S 只包含字母数字（a-z，A-Z，0-9）以及破折号'-'

S 非空

485、最大连续1的个数

给定一个二进制数组， 计算其中最大连续1的个数。

示例 1:

输入: [1,1,0,1,1,1]

输出: 3

解释: 开头的两位和最后的三位都是连续1，所以最大连续1的个数是 3.

注意：

输入的数组只包含 0 和1。

输入数组的长度是正整数，且不超过 10,000。492、构造矩形

作为一位web开发者， 懂得怎样去规划一个页面的尺寸是很重要的。 现给定一个具体的矩形页面面积，你的任务是设计一个长度为 L 和宽度为 W 且满足以下要求的矩形的页面。要求：

1. 你设计的矩形页面必须等于给定的目标面积。

2. 宽度 W 不应大于长度 L，换言之，要求 L >= W 。

3. 长度 L 和宽度 W 之间的差距应当尽可能小。

你需要按顺序输出你设计的页面的长度 L 和宽度 W。

示例：

输入: 4

输出: [2, 2]

解释: 目标面积是 4， 所有可能的构造方案有 [1,4], [2,2], [4,1]。

但是根据要求2，[1,4] 不符合要求; 根据要求3，[2,2] 比 [4,1] 更能符合要求. 所以输出长度 L 为 2， 宽度 W 为 2。

说明:

给定的面积不大于 10,000,000 且为正整数。

你设计的页面的长度和宽度必须都是正整数。496、下一个更大元素 I

给定两个没有重复元素的数组 nums1 和 nums2 ，其中nums1 是 nums2 的子集。找到 nums1 中每个元素在 nums2 中的下一个比其大的值。

nums1 中数字 x 的下一个更大元素是指 x 在 nums2 中对应位置的右边的第一个比 x 大的元素。如果不存在，对应位置输出-1。

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。

注意:

nums1和nums2中所有元素是唯一的。

nums1和nums2 的数组大小都不超过1000。500、键盘行

给定一个单词列表，只返回可以使用在键盘同一行的字母打印出来的单词。键盘如下图所示。示例1:

输入: ["Hello", "Alaska", "Dad", "Peace"]

输出: ["Alaska", "Dad"]注意: 你可以重复使用键盘上同一字符。

你可以假设输入的字符串将只包含字母。501、二叉搜索树中的众数

给定一个有相同值的二叉搜索树（BST），找出 BST 中的所有众数（出现频率最高的元素）。

假定 BST 有如下定义：

结点左子树中所含结点的值小于等于当前结点的值

结点右子树中所含结点的值大于等于当前结点的值

左子树和右子树都是二叉搜索树

例如：

给定 BST [1,null,2,2],

1

\

2

/

2

返回[2].

提示：如果众数超过1个，不需考虑输出顺序

进阶：你可以不使用额外的空间吗？（假设由递归产生的隐式调用栈的开销不被计算在内）504、七进制数

给定一个整数，将其转化为7进制，并以字符串形式输出。

示例 1:

输入: 100

输出: "202"

示例 2:

输入: -7

输出: "-10"

注意: 输入范围是 [-1e7, 1e7] 。506、相对名次

给出 N 名运动员的成绩，找出他们的相对名次并授予前三名对应的奖牌。前三名运动员将会被分别授予 &ldquo;金牌&rdquo;，&ldquo;银牌&rdquo; 和&ldquo; 铜牌&rdquo;（"Gold Medal", "Silver Medal", "Bronze Medal"）。

(注：分数越高的选手，排名越靠前。)

示例 1:

输入: [5, 4, 3, 2, 1]

输出: ["Gold Medal", "Silver Medal", "Bronze Medal", "4", "5"]

解释: 前三名运动员的成绩为前三高的，因此将会分别被授予 &ldquo;金牌&rdquo;，&ldquo;银牌&rdquo;和&ldquo;铜牌&rdquo; ("Gold Medal", "Silver Medal" and "Bronze Medal").

余下的两名运动员，我们只需要通过他们的成绩计算将其相对名次即可。

提示:

N 是一个正整数并且不会超过 10000。

所有运动员的成绩都不相同。507、完美数

对于一个 正整数，如果它和除了它自身以外的所有正因子之和相等，我们称它为&ldquo;完美数&rdquo;。

给定一个 正整数 n， 如果他是完美数，返回 True，否则返回 False

示例：

输入: 28

输出: True

解释: 28 = 1 + 2 + 4 + 7 + 14

注意:

输入的数字 n 不会超过 100,000,000. (1e8)520、检测大写字母

给定一个单词，你需要判断单词的大写使用是否正确。

我们定义，在以下情况时，单词的大写用法是正确的：

全部字母都是大写，比如"USA"。

单词中所有字母都不是大写，比如"leetcode"。

如果单词不只含有一个字母，只有首字母大写， 比如 "Google"。

否则，我们定义这个单词没有正确使用大写字母。

示例 1:

输入: "USA"

输出: True

示例 2:

输入: "FlaG"

输出: False

注意: 输入是由大写和小写拉丁字母组成的非空单词。521、最长特殊序列 Ⅰ

给定两个字符串，你需要从这两个字符串中找出最长的特殊序列。最长特殊序列定义如下：该序列为某字符串独有的最长子序列（即不能是其他字符串的子序列）。

子序列可以通过删去字符串中的某些字符实现，但不能改变剩余字符的相对顺序。空序列为所有字符串的子序列，任何字符串为其自身的子序列。

输入为两个字符串，输出最长特殊序列的长度。如果不存在，则返回 -1。

示例 :

输入: "aba", "cdc"

输出: 3

解析: 最长特殊序列可为 "aba" (或 "cdc")

说明:

两个字符串长度均小于100。

字符串中的字符仅含有 'a'~'z'。530、二叉搜索树的最小绝对差

给定一个所有节点为非负值的二叉搜索树，求树中任意两节点的差的绝对值的最小值。

示例 :

输入:

1

\

3

/

2

输出:

1

解释:

最小绝对差为1，其中 2 和 1 的差的绝对值为 1（或者 2 和 3）。

注意: 树中至少有2个节点。532、数组中的K-diff数对

给定一个整数数组和一个整数 k, 你需要在数组里找到不同的 k-diff 数对。这里将 k-diff 数对定义为一个整数对 (i, j), 其中 i 和 j 都是数组中的数字，且两数之差的绝对值是 k.

示例 1:

输入: [3, 1, 4, 1, 5], k = 2

输出: 2

解释: 数组中有两个 2-diff 数对, (1, 3) 和 (3, 5)。

尽管数组中有两个1，但我们只应返回不同的数对的数量。

示例 2:

输入:[1, 2, 3, 4, 5], k = 1

输出: 4

解释: 数组中有四个 1-diff 数对, (1, 2), (2, 3), (3, 4) 和 (4, 5)。

示例 3:

输入: [1, 3, 1, 5, 4], k = 0

输出: 1

解释: 数组中只有一个 0-diff 数对，(1, 1)。

注意:

数对 (i, j) 和数对 (j, i) 被算作同一数对。

数组的长度不超过10,000。

所有输入的整数的范围在 [-1e7, 1e7]。538、把二叉搜索树转换为累加树

给定一个二叉搜索树（Binary Search Tree），把它转换成为累加树（Greater Tree)，使得每个节点的值是原来的节点值加上所有大于它的节点值之和。

例如：

输入: 二叉搜索树:

5

/ \

2 13

输出: 转换为累加树:

18

/ \

20 13541、反转字符串 II

给定一个字符串和一个整数 k，你需要对从字符串开头算起的每个 2k 个字符的前k个字符进行反转。如果剩余少于 k 个字符，则将剩余的所有全部反转。如果有小于 2k 但大于或等于 k 个字符，则反转前 k 个字符，并将剩余的字符保持原样。

示例:

输入: s = "abcdefg", k = 2

输出: "bacdfeg"

要求:

该字符串只包含小写的英文字母。

给定字符串的长度和 k 在[1, 10000]范围内。543、二叉树的直径

给定一棵二叉树，你需要计算它的直径长度。一棵二叉树的直径长度是任意两个结点路径长度中的最大值。这条路径可能穿过根结点。

示例 :

给定二叉树

1

/ \

2 3

/ \

4 5

返回 3, 它的长度是路径 [4,2,1,3] 或者 [5,2,1,3]。

注意：两结点之间的路径长度是以它们之间边的数目表示。551、学生出勤纪录 I

给定一个字符串来代表一个学生的出勤纪录，这个纪录仅包含以下三个字符：

'A' : Absent，缺勤

'L' : Late，迟到

'P' : Present，到场

如果一个学生的出勤纪录中不超过一个'A'(缺勤)并且不超过两个连续的'L'(迟到),那么这个学生会被奖赏。

你需要根据这个学生的出勤纪录判断他是否会被奖赏。

示例 1:

输入: "PPALLP"

输出: True

示例 2:

输入: "PPALLL"

输出: False557、反转字符串中的单词 III

给定一个字符串，你需要反转字符串中每个单词的字符顺序，同时仍保留空格和单词的初始顺序。

示例 1:

输入: "Let's take LeetCode contest"

输出: "s'teL ekat edoCteeL tsetnoc"

注意：在字符串中，每个单词由单个空格分隔，并且字符串中不会有任何额外的空格。558、Quad Tree Intersection

A quadtree is a tree data in which each internal node has exactly four children: topLeft, topRight, bottomLeft and bottomRight. Quad trees are often used to partition a two-dimensional space by recursively subdividing it into four quadrants or regions.

We want to store True/False information in our quad tree. The quad tree is used to represent a N \* N boolean grid. For each node, it will be subdivided into four children nodes until the values in the region it represents are all the same. Each node has another two boolean attributes : isLeaf and val. isLeaf is true if and only if the node is a leaf node. The val attribute for a leaf node contains the value of the region it represents.

For example, below are two quad trees A and B:A:

+-------+-------+ T: true

| | | F: false

| T | T |

| | |

+-------+-------+

| | |

| F | F |

| | |

+-------+-------+

topLeft: T

topRight: T

bottomLeft: F

bottomRight: F

B:

+-------+---+---+

| | F | F |

| T +---+---+

| | T | T |

+-------+---+---+

| | |

| T | F |

| | |

+-------+-------+

topLeft: T

topRight:

topLeft: F

topRight: F

bottomLeft: T

bottomRight: T

bottomLeft: T

bottomRight: F

Your task is to implement a function that will take two quadtrees and return a quadtree that represents the logical OR (or union) of the two trees.A: B: C (A or B):

+-------+-------+ +-------+---+---+ +-------+-------+

| | | | | F | F | | | |

| T | T | | T +---+---+ | T | T |

| | | | | T | T | | | |

+-------+-------+ +-------+---+---+ +-------+-------+

| | | | | | | | |

| F | F | | T | F | | T | F |

| | | | | | | | |

+-------+-------+ +-------+-------+ +-------+-------+Note: Both A and B represent grids of size N \* N.

N is guaranteed to be a power of 2.

If you want to know more about the quad tree, you can refer to its wiki.

The logic OR operation is defined as this: "A or B" is true if A is true, or if B is true, or if both A and B are true.559、N叉树的最大深度

给定一个N叉树，找到其最大深度。

最大深度是指从根节点到最远叶子节点的最长路径上的节点总数。

例如，给定一个 3叉树 :

我们应返回其最大深度，3。

说明:

树的深度不会超过 1000。

树的节点总不会超过 5000。561、数组拆分 I

给定长度为 2n 的数组, 你的任务是将这些数分成 n 对, 例如 (a1, b1), (a2, b2), ..., (an, bn) ，使得从1 到 n 的 min(ai, bi) 总和最大。

示例 1:

输入: [1,4,3,2]

输出: 4

解释: n 等于 2, 最大总和为 4 = min(1, 2) + min(3, 4).

提示:

n 是正整数,范围在 [1, 10000].

数组中的元素范围在 [-10000, 10000].563、二叉树的坡度

给定一个二叉树，计算整个树的坡度。

一个树的节点的坡度定义即为，该节点左子树的结点之和和右子树结点之和的差的绝对值。空结点的的坡度是0。

整个树的坡度就是其所有节点的坡度之和。

示例:

输入:

1

/ \

2 3

输出: 1

解释:

结点的坡度 2 : 0

结点的坡度 3 : 0

结点的坡度 1 : |2-3| = 1

树的坡度 : 0 + 0 + 1 = 1

注意:

任何子树的结点的和不会超过32位整数的范围。

坡度的值不会超过32位整数的范围。566、重塑矩阵

在MATLAB中，有一个非常有用的函数 reshape，它可以将一个矩阵重塑为另一个大小不同的新矩阵，但保留其原始数据。

给出一个由二维数组表示的矩阵，以及两个正整数r和c，分别表示想要的重构的矩阵的行数和列数。

重构后的矩阵需要将原始矩阵的所有元素以相同的行遍历顺序填充。

如果具有给定参数的reshape操作是可行且合理的，则输出新的重塑矩阵；否则，输出原始矩阵。

示例 1:

输入:

nums =

[[1,2],

[3,4]]

r = 1, c = 4

输出:

[[1,2,3,4]]

解释:

行遍历nums的结果是 [1,2,3,4]。新的矩阵是 1 \* 4 矩阵, 用之前的元素值一行一行填充新矩阵。

示例 2:

输入:

nums =

[[1,2],

[3,4]]

r = 2, c = 4

输出:

[[1,2],

[3,4]]

解释:

没有办法将 2 \* 2 矩阵转化为 2 \* 4 矩阵。 所以输出原矩阵。

注意：

给定矩阵的宽和高范围在 [1, 100]。

给定的 r 和 c 都是正数。572、另一个树的子树

给定两个非空二叉树 s 和 t，检验 s 中是否包含和 t 具有相同结构和节点值的子树。s 的一个子树包括 s 的一个节点和这个节点的所有子孙。s 也可以看做它自身的一棵子树。

示例 1:

给定的树 s:

3

/ \

4 5

/ \

1 2

给定的树 t：

4

/ \

1 2

返回 true，因为 t 与 s 的一个子树拥有相同的结构和节点值。

示例 2:

给定的树 s：

3

/ \

4 5

/ \

1 2

/

0

给定的树 t：

4

/ \

1 2

返回 false。575、分糖果

给定一个偶数长度的数组，其中不同的数字代表着不同种类的糖果，每一个数字代表一个糖果。你需要把这些糖果平均分给一个弟弟和一个妹妹。返回妹妹可以获得的最大糖果的种类数。

示例 1:

输入: candies = [1,1,2,2,3,3]

输出: 3

解析: 一共有三种种类的糖果，每一种都有两个。

最优分配方案：妹妹获得[1,2,3],弟弟也获得[1,2,3]。这样使妹妹获得糖果的种类数最多。

示例 2 :

输入: candies = [1,1,2,3]

输出: 2

解析: 妹妹获得糖果[2,3],弟弟获得糖果[1,1]，妹妹有两种不同的糖果，弟弟只有一种。这样使得妹妹可以获得的糖果种类数最多。

注意:

数组的长度为[2, 10,000]，并且确定为偶数。

数组中数字的大小在范围[-100,000, 100,000]内。

581、最短无序连续子数组

给定一个整数数组，你需要寻找一个连续的子数组，如果对这个子数组进行升序排序，那么整个数组都会变为升序排序。

你找到的子数组应是最短的，请输出它的长度。

示例 1:

输入: [2, 6, 4, 8, 10, 9, 15]

输出: 5

解释: 你只需要对 [6, 4, 8, 10, 9] 进行升序排序，那么整个表都会变为升序排序。

说明 :

输入的数组长度范围在 [1, 10,000]。

输入的数组可能包含重复元素 ，所以升序的意思是<=。589、N叉树的前序遍历

给定一个N叉树，返回其节点值的前序遍历。

例如，给定一个 3叉树 :

返回其前序遍历: [1,3,5,6,2,4]。

说明: 递归法很简单，你可以使用迭代法完成此题吗?590、N叉树的后序遍历

给定一个N叉树，返回其节点值的后序遍历。

例如，给定一个 3叉树 :

返回其后序遍历: [5,6,3,2,4,1].

说明: 递归法很简单，你可以使用迭代法完成此题吗?594、最长和谐子序列

和谐数组是指一个数组里元素的最大值和最小值之间的差别正好是1。

现在，给定一个整数数组，你需要在所有可能的子序列中找到最长的和谐子序列的长度。

示例 1:

输入: [1,3,2,2,5,2,3,7]

输出: 5

原因: 最长的和谐数组是：[3,2,2,2,3].

说明: 输入的数组长度最大不超过20,000.595、大的国家

这里有张 World 表

+-----------------+------------+------------+--------------+---------------+

| name | continent | area | population | gdp |

+-----------------+------------+------------+--------------+---------------+

| Afghanistan | Asia | 652230 | 25500100 | 20343000 |

| Albania | Europe | 28748 | 2831741 | 12960000 |

| Algeria | Africa | 2381741 | 37100000 | 188681000 |

| Andorra | Europe | 468 | 78115 | 3712000 |

| Angola | Africa | 1246700 | 20609294 | 100990000 |

+-----------------+------------+------------+--------------+---------------+

如果一个国家的面积超过300万平方公里，或者人口超过2500万，那么这个国家就是大国家。

编写一个SQL查询，输出表中所有大国家的名称、人口和面积。

例如，根据上表，我们应该输出:

+--------------+-------------+--------------+

| name | population | area |

+--------------+-------------+--------------+

| Afghanistan | 25500100 | 652230 |

| Algeria | 37100000 | 2381741 |

+--------------+-------------+--------------+596、超过5名学生的课

有一个courses 表 ，有: student (学生) 和 class (课程)。

请列出所有超过或等于5名学生的课。

例如,表:

+---------+------------+

| student | class |

+---------+------------+

| A | Math |

| B | English |

| C | Math |

| D | Biology |

| E | Math |

| F | Computer |

| G | Math |

| H | Math |

| I | Math |

+---------+------------+

应该输出:

+---------+

| class |

+---------+

| Math |

+---------+

Note:

学生在每个课中不应被重复计算。598、范围求和 II

给定一个初始元素全部为 0，大小为 m\*n 的矩阵 M 以及在 M 上的一系列更新操作。

操作用二维数组表示，其中的每个操作用一个含有两个正整数 a 和 b 的数组表示，含义是将所有符合 0 <= i < a 以及 0 <= j < b 的元素 M[i][j] 的值都增加 1。

在执行给定的一系列操作后，你需要返回矩阵中含有最大整数的元素个数。

示例 1:

输入:

m = 3, n = 3

operations = [[2,2],[3,3]]

输出: 4

解释:

初始状态, M =

[[0, 0, 0],

[0, 0, 0],

[0, 0, 0]]

执行完操作 [2,2] 后, M =

[[1, 1, 0],

[1, 1, 0],

[0, 0, 0]]

执行完操作 [3,3] 后, M =

[[2, 2, 1],

[2, 2, 1],

[1, 1, 1]]

M 中最大的整数是 2, 而且 M 中有4个值为2的元素。因此返回 4。

注意:

m 和 n 的范围是 [1,40000]。

a 的范围是 [1,m]，b 的范围是 [1,n]。

操作数目不超过 10000。599、两个列表的最小索引总和

假设Andy和Doris想在晚餐时选择一家餐厅，并且他们都有一个表示最喜爱餐厅的列表，每个餐厅的名字用字符串表示。

你需要帮助他们用最少的索引和找出他们共同喜爱的餐厅。 如果答案不止一个，则输出所有答案并且不考虑顺序。 你可以假设总是存在一个答案。

示例 1:

输入:

["Shogun", "Tapioca Express", "Burger King", "KFC"]

["Piatti", "The Grill at Torrey Pines", "Hungry Hunter Steakhouse", "Shogun"]

输出: ["Shogun"]

解释: 他们唯一共同喜爱的餐厅是&ldquo;Shogun&rdquo;。

示例 2:

输入:

["Shogun", "Tapioca Express", "Burger King", "KFC"]

["KFC", "Shogun", "Burger King"]

输出: ["Shogun"]

解释: 他们共同喜爱且具有最小索引和的餐厅是&ldquo;Shogun&rdquo;，它有最小的索引和1(0+1)。

提示:

两个列表的长度范围都在 [1, 1000]内。

两个列表中的字符串的长度将在[1，30]的范围内。

下标从0开始，到列表的长度减1。

两个列表都没有重复的元素。605、种花问题

假设你有一个很长的花坛，一部分地块种植了花，另一部分却没有。可是，花卉不能种植在相邻的地块上，它们会争夺水源，两者都会死去。

给定一个花坛（表示为一个数组包含0和1，其中0表示没种植花，1表示种植了花），和一个数 n 。能否在不打破种植规则的情况下种入 n 朵花？能则返回True，不能则返回False。

示例 1:

输入: flowerbed = [1,0,0,0,1], n = 1

输出: True

示例 2:

输入: flowerbed = [1,0,0,0,1], n = 2

输出: False

注意:

数组内已种好的花不会违反种植规则。

输入的数组长度范围为 [1, 20000]。

n 是非负整数，且不会超过输入数组的大小。606、根据二叉树创建字符串

你需要采用前序遍历的方式，将一个二叉树转换成一个由括号和整数组成的字符串。

空节点则用一对空括号 "()" 表示。而且你需要省略所有不影响字符串与原始二叉树之间的一对一映射关系的空括号对。

示例 1:

输入: 二叉树: [1,2,3,4]

1

/ \

2 3

/

4

输出: "1(2(4))(3)"

解释: 原本将是&ldquo;1(2(4)())(3())&rdquo;，

在你省略所有不必要的空括号对之后，

它将是&ldquo;1(2(4))(3)&rdquo;。

示例 2:

输入: 二叉树: [1,2,3,null,4]

1

/ \

2 3

\

4

输出: "1(2()(4))(3)"

解释: 和第一个示例相似，

除了我们不能省略第一个对括号来中断输入和输出之间的一对一映射关系。617、合并二叉树

给定两个二叉树，想象当你将它们中的一个覆盖到另一个上时，两个二叉树的一些节点便会重叠。

你需要将他们合并为一个新的二叉树。合并的规则是如果两个节点重叠，那么将他们的值相加作为节点合并后的新值，否则不为 NULL 的节点将直接作为新二叉树的节点。

示例 1:

输入:

Tree 1 Tree 2

1 2

/ \ / \

3 2 1 3

/ \ \

5 4 7

输出:

合并后的树:

3

/ \

4 5

/ \ \

5 4 7

注意: 合并必须从两个树的根节点开始。620、有趣的电影

某城市开了一家新的电影院，吸引了很多人过来看电影。该电影院特别注意用户体验，专门有个 LED显示板做电影推荐，上面公布着影评和相关电影描述。

作为该电影院的信息部主管，您需要编写一个 SQL查询，找出所有影片描述为非 boring (不无聊) 的并且 id 为奇数 的影片，结果请按等级 rating 排列。

例如，下表 cinema:

+---------+-----------+--------------+-----------+

| id | movie | description | rating |

+---------+-----------+--------------+-----------+

| 1 | War | great 3D | 8.9 |

| 2 | Science | fiction | 8.5 |

| 3 | irish | boring | 6.2 |

| 4 | Ice song | Fantacy | 8.6 |

| 5 | House card| Interesting| 9.1 |

+---------+-----------+--------------+-----------+

对于上面的例子，则正确的输出是为：

+---------+-----------+--------------+-----------+

| id | movie | description | rating |

+---------+-----------+--------------+-----------+

| 5 | House card| Interesting| 9.1 |

| 1 | War | great 3D | 8.9 |

+---------+-----------+--------------+-----------+

622、设计循环队列

设计你的循环队列实现。 循环队列是一种线性数据结构，其操作表现基于 FIFO（先进先出）原则并且队尾被连接在队首之后以形成一个循环。它也被称为&ldquo;环形缓冲器&rdquo;。

循环队列的一个好处是我们可以利用这个队列之前用过的空间。在一个普通队列里，一旦一个队列满了，我们就不能插入下一个元素，即使在队列前面仍有空间。但是使用循环队列，我们能使用这些空间去存储新的值。

你的实现应该支持如下操作：

MyCircularQueue(k): 构造器，设置队列长度为 k 。

Front: 从队首获取元素。如果队列为空，返回 -1 。

Rear: 获取队尾元素。如果队列为空，返回 -1 。

enQueue(value): 向循环队列插入一个元素。如果成功插入则返回真。

deQueue(): 从循环队列中删除一个元素。如果成功删除则返回真。

isEmpty(): 检查循环队列是否为空。

isFull(): 检查循环队列是否已满。

示例：

MyCircularQueue circularQueue = new MycircularQueue(3); // 设置长度为3

circularQueue.enQueue(1); // 返回true

circularQueue.enQueue(2); // 返回true

circularQueue.enQueue(3); // 返回true

circularQueue.enQueue(4); // 返回false,队列已满

circularQueue.Rear(); // 返回3

circularQueue.isFull(); // 返回true

circularQueue.deQueue(); // 返回true

circularQueue.enQueue(4); // 返回true

circularQueue.Rear(); // 返回4

提示：

所有的值都在 1 至 1000 的范围内；

操作数将在 1 至 1000 的范围内；

请不要使用内置的队列库。627、交换工资

给定一个 salary表，如下所示，有m=男性 和 f=女性的值 。交换所有的 f 和 m 值(例如，将所有 f 值更改为 m，反之亦然)。要求使用一个更新查询，并且没有中间临时表。

例如:

| id | name | sex | salary |

|----|------|-----|--------|

| 1 | A | m | 2500 |

| 2 | B | f | 1500 |

| 3 | C | m | 5500 |

| 4 | D | f | 500 |

运行你所编写的查询语句之后，将会得到以下表:

| id | name | sex | salary |

|----|------|-----|--------|

| 1 | A | f | 2500 |

| 2 | B | m | 1500 |

| 3 | C | f | 5500 |

| 4 | D | m | 500 |628、三个数的最大乘积

给定一个整型数组，在数组中找出由三个数组成的最大乘积，并输出这个乘积。

示例 1:

输入: [1,2,3]

输出: 6

示例 2:

输入: [1,2,3,4]

输出: 24

注意:

给定的整型数组长度范围是[3,104]，数组中所有的元素范围是[-1000, 1000]。

输入的数组中任意三个数的乘积不会超出32位有符号整数的范围。633、平方数之和

给定一个非负整数 c ，你要判断是否存在两个整数 a 和 b，使得 a2 + b2 = c。

示例1:

输入: 5

输出: True

解释: 1 \* 1 + 2 \* 2 = 5

示例2:

输入: 3

输出: False637、二叉树的层平均值

给定一个非空二叉树, 返回一个由每层节点平均值组成的数组.

示例 1:

输入:

3

/ \

9 20

/ \

15 7

输出: [3, 14.5, 11]

解释:

第0层的平均值是 3, 第1层是 14.5, 第2层是 11. 因此返回 [3, 14.5, 11].

注意：

节点值的范围在32位有符号整数范围内。641、设计循环双端队列

设计实现双端队列。

你的实现需要支持以下操作：

MyCircularDeque(k)：构造函数,双端队列的大小为k。

insertFront()：将一个元素添加到双端队列头部。 如果操作成功返回 true。

insertLast()：将一个元素添加到双端队列尾部。如果操作成功返回 true。

deleteFront()：从双端队列头部删除一个元素。 如果操作成功返回 true。

deleteLast()：从双端队列尾部删除一个元素。如果操作成功返回 true。

getFront()：从双端队列头部获得一个元素。如果双端队列为空，返回 -1。

getRear()：获得双端队列的最后一个元素。 如果双端队列为空，返回 -1。

isEmpty()：检查双端队列是否为空。

isFull()：检查双端队列是否满了。

示例：

MyCircularDeque circularDeque = new MycircularDeque(3); // 设置容量大小为3

circularDeque.insertLast(1); // 返回 true

circularDeque.insertLast(2); // 返回 true

circularDeque.insertFront(3); // 返回 true

circularDeque.insertFront(4); // 已经满了，返回 false

circularDeque.getRear(); // 返回 32

circularDeque.isFull(); // 返回 true

circularDeque.deleteLast(); // 返回 true

circularDeque.insertFront(4); // 返回 true

circularDeque.getFront(); // 返回 4

提示：

所有值的范围为 [1, 1000]

操作次数的范围为 [1, 1000]

请不要使用内置的双端队列库。643、子数组最大平均数 I

给定 n 个整数，找出平均数最大且长度为 k 的连续子数组，并输出该最大平均数。

示例 1:

输入: [1,12,-5,-6,50,3], k = 4

输出: 12.75

解释: 最大平均数 (12-5-6+50)/4 = 51/4 = 12.75

注意:

1 <= k <= n <= 30,000。

所给数据范围 [-10,000，10,000]。645、错误的集合

集合 S 包含从1到 n 的整数。不幸的是，因为数据错误，导致集合里面某一个元素复制了成了集合里面的另外一个元素的值，导致集合丢失了一个整数并且有一个元素重复。

给定一个数组 nums 代表了集合 S 发生错误后的结果。你的任务是首先寻找到重复出现的整数，再找到丢失的整数，将它们以数组的形式返回。

示例 1:

输入: nums = [1,2,2,4]

输出: [2,3]

注意:

给定数组的长度范围是 [2, 10000]。

给定的数组是无序的。653、两数之和 IV - 输入 BST

给定一个二叉搜索树和一个目标结果，如果 BST 中存在两个元素且它们的和等于给定的目标结果，则返回 true。

案例 1:

输入:

5

/ \

3 6

/ \ \

2 4 7

Target = 9

输出: True

案例 2:

输入:

5

/ \

3 6

/ \ \

2 4 7

Target = 28

输出: False

657、判断路线成圈

初始位置 (0, 0) 处有一个机器人。给出它的一系列动作，判断这个机器人的移动路线是否形成一个圆圈，换言之就是判断它是否会移回到原来的位置。

移动顺序由一个字符串表示。每一个动作都是由一个字符来表示的。机器人有效的动作有 R（右），L（左），U（上）和 D（下）。输出应为 true 或 false，表示机器人移动路线是否成圈。

示例 1:

输入: "UD"

输出: true

示例 2:

输入: "LL"

输出: false661、图片平滑器

包含整数的二维矩阵 M 表示一个图片的灰度。你需要设计一个平滑器来让每一个单元的灰度成为平均灰度 (向下舍入) ，平均灰度的计算是周围的8个单元和它本身的值求平均，如果周围的单元格不足八个，则尽可能多的利用它们。

示例 1:

输入:

[[1,1,1],

[1,0,1],

[1,1,1]]

输出:

[[0, 0, 0],

[0, 0, 0],

[0, 0, 0]]

解释:

对于点 (0,0), (0,2), (2,0), (2,2): 平均(3/4) = 平均(0.75) = 0

对于点 (0,1), (1,0), (1,2), (2,1): 平均(5/6) = 平均(0.83333333) = 0

对于点 (1,1): 平均(8/9) = 平均(0.88888889) = 0

注意:

给定矩阵中的整数范围为 [0, 255]。

矩阵的长和宽的范围均为 [1, 150]。665、非递减数列

给定一个长度为 n 的整数数组，你的任务是判断在最多改变 1 个元素的情况下，该数组能否变成一个非递减数列。

我们是这样定义一个非递减数列的： 对于数组中所有的 i (1 <= i < n)，满足 array[i] <= array[i + 1]。

示例 1:

输入: [4,2,3]

输出: True

解释: 你可以通过把第一个4变成1来使得它成为一个非递减数列。

示例 2:

输入: [4,2,1]

输出: False

解释: 你不能在只改变一个元素的情况下将其变为非递减数列。

说明: n 的范围为 [1, 10,000]。669、修剪二叉搜索树

给定一个二叉搜索树，同时给定最小边界L 和最大边界 R。通过修剪二叉搜索树，使得所有节点的值在[L, R]中 (R>=L) 。你可能需要改变树的根节点，所以结果应当返回修剪好的二叉搜索树的新的根节点。

示例 1:

输入:

1

/ \

0 2

L = 1

R = 2

输出:

1

\

2

示例 2:

输入:

3

/ \

0 4

\

2

/

1

L = 1

R = 3

输出:

3

/

2

/

1671、二叉树中第二小的节点

给定一个非空特殊的二叉树，每个节点都是正数，并且每个节点的子节点数量只能为 2 或 0。如果一个节点有两个子节点的话，那么这个节点的值不大于它的子节点的值。

给出这样的一个二叉树，你需要输出所有节点中的第二小的值。如果第二小的值不存在的话，输出 -1 。

示例 1:

输入:

2

/ \

2 5

/ \

5 7

输出: 5

说明: 最小的值是 2 ，第二小的值是 5 。

示例 2:

输入:

2

/ \

2 2

输出: -1

说明: 最小的值是 2, 但是不存在第二小的值。674、最长连续递增序列

给定一个未经排序的整数数组，找到最长且连续的的递增序列。

示例 1:

输入: [1,3,5,4,7]

输出: 3

解释: 最长连续递增序列是 [1,3,5], 长度为3。

尽管 [1,3,5,7] 也是升序的子序列, 但它不是连续的，因为5和7在原数组里被4隔开。

示例 2:

输入: [2,2,2,2,2]

输出: 1

解释: 最长连续递增序列是 [2], 长度为1。

注意：数组长度不会超过10000。680、验证回文字符串 Ⅱ

给定一个非空字符串 s，最多删除一个字符。判断是否能成为回文字符串。

示例 1:

输入: "aba"

输出: True

示例 2:

输入: "abca"

输出: True

解释: 你可以删除c字符。

注意:

字符串只包含从 a-z 的小写字母。字符串的最大长度是50000。682、棒球比赛

你现在是棒球比赛记录员。

给定一个字符串列表，每个字符串可以是以下四种类型之一：

1.整数（一轮的得分）：直接表示您在本轮中获得的积分数。

2. "+"（一轮的得分）：表示本轮获得的得分是前两轮有效 回合得分的总和。

3. "D"（一轮的得分）：表示本轮获得的得分是前一轮有效 回合得分的两倍。

4. "C"（一个操作，这不是一个回合的分数）：表示您获得的最后一个有效 回合的分数是无效的，应该被移除。

每一轮的操作都是永久性的，可能会对前一轮和后一轮产生影响。

你需要返回你在所有回合中得分的总和。

示例 1:

输入: ["5","2","C","D","+"]

输出: 30

解释:

第1轮：你可以得到5分。总和是：5。

第2轮：你可以得到2分。总和是：7。

操作1：第2轮的数据无效。总和是：5。

第3轮：你可以得到10分（第2轮的数据已被删除）。总数是：15。

第4轮：你可以得到5 + 10 = 15分。总数是：30。

示例 2:

输入: ["5","-2","4","C","D","9","+","+"]

输出: 27

解释:

第1轮：你可以得到5分。总和是：5。

第2轮：你可以得到-2分。总数是：3。

第3轮：你可以得到4分。总和是：7。

操作1：第3轮的数据无效。总数是：3。

第4轮：你可以得到-4分（第三轮的数据已被删除）。总和是：-1。

第5轮：你可以得到9分。总数是：8。

第6轮：你可以得到-4 + 9 = 5分。总数是13。

第7轮：你可以得到9 + 5 = 14分。总数是27。

注意：

输入列表的大小将介于1和1000之间。

列表中的每个整数都将介于-30000和30000之间。686、重复叠加字符串匹配

给定两个字符串 A 和 B, 寻找重复叠加字符串A的最小次数，使得字符串B成为叠加后的字符串A的子串，如果不存在则返回 -1。

举个例子，A = "abcd"，B = "cdabcdab"。

答案为 3， 因为 A 重复叠加三遍后为 &ldquo;abcdabcdabcd&rdquo;，此时 B 是其子串；A 重复叠加两遍后为"abcdabcd"，B 并不是其子串。

注意:

A 与 B 字符串的长度在1和10000区间范围内。687、最长同值路径

给定一个二叉树，找到最长的路径，这个路径中的每个节点具有相同值。 这条路径可以经过也可以不经过根节点。

注意：两个节点之间的路径长度由它们之间的边数表示。

示例 1:

输入:

5

/ \

4 5

/ \ \

1 1 5

输出:

2

示例 2:

输入:

1

/ \

4 5

/ \ \

4 4 5

输出:

2

注意: 给定的二叉树不超过10000个结点。 树的高度不超过1000。690、员工的重要性

给定一个保存员工信息的数据结构，它包含了员工唯一的id，重要度 和 直系下属的id。

比如，员工1是员工2的领导，员工2是员工3的领导。他们相应的重要度为15, 10, 5。那么员工1的数据结构是[1, 15, [2]]，员工2的数据结构是[2, 10, [3]]，员工3的数据结构是[3, 5, []]。注意虽然员工3也是员工1的一个下属，但是由于并不是直系下属，因此没有体现在员工1的数据结构中。

现在输入一个公司的所有员工信息，以及单个员工id，返回这个员工和他所有下属的重要度之和。

示例 1:

输入: [[1, 5, [2, 3]], [2, 3, []], [3, 3, []]], 1

输出: 11

解释:

员工1自身的重要度是5，他有两个直系下属2和3，而且2和3的重要度均为3。因此员工1的总重要度是 5 + 3 + 3 = 11。

注意:

一个员工最多有一个直系领导，但是可以有多个直系下属

员工数量不超过2000。693、交替位二进制数

给定一个正整数，检查他是否为交替位二进制数：换句话说，就是他的二进制数相邻的两个位数永不相等。

示例 1:

输入: 5

输出: True

解释:

5的二进制数是: 101

示例 2:

输入: 7

输出: False

解释:

7的二进制数是: 111

示例 3:

输入: 11

输出: False

解释:

11的二进制数是: 1011

示例 4:

输入: 10

输出: True

解释:

10的二进制数是: 1010695、岛屿的最大面积

给定一个包含了一些 0 和 1的非空二维数组 grid , 一个 岛屿 是由四个方向 (水平或垂直) 的 1 (代表土地) 构成的组合。你可以假设二维矩阵的四个边缘都被水包围着。

找到给定的二维数组中最大的岛屿面积。(如果没有岛屿，则返回面积为0。)

示例 1:

[[0,0,1,0,0,0,0,1,0,0,0,0,0],

[0,0,0,0,0,0,0,1,1,1,0,0,0],

[0,1,1,0,1,0,0,0,0,0,0,0,0],

[0,1,0,0,1,1,0,0,1,0,1,0,0],

[0,1,0,0,1,1,0,0,1,1,1,0,0],

[0,0,0,0,0,0,0,0,0,0,1,0,0],

[0,0,0,0,0,0,0,1,1,1,0,0,0],

[0,0,0,0,0,0,0,1,1,0,0,0,0]]

对于上面这个给定矩阵应返回 6。注意答案不应该是11，因为岛屿只能包含水平或垂直的四个方向的'1'。

示例 2:

[[0,0,0,0,0,0,0,0]]

对于上面这个给定的矩阵, 返回 0。

注意: 给定的矩阵grid 的长度和宽度都不超过 50。696、计数二进制子串

给定一个字符串 s，计算具有相同数量0和1的非空(连续)子字符串的数量，并且这些子字符串中的所有0和所有1都是组合在一起的。

重复出现的子串要计算它们出现的次数。

示例 1 :

输入: "00110011"

输出: 6

解释: 有6个子串具有相同数量的连续1和0：&ldquo;0011&rdquo;，&ldquo;01&rdquo;，&ldquo;1100&rdquo;，&ldquo;10&rdquo;，&ldquo;0011&rdquo; 和 &ldquo;01&rdquo;。

请注意，一些重复出现的子串要计算它们出现的次数。

另外，&ldquo;00110011&rdquo;不是有效的子串，因为所有的0（和1）没有组合在一起。

示例 2 :

输入: "10101"

输出: 4

解释: 有4个子串：&ldquo;10&rdquo;，&ldquo;01&rdquo;，&ldquo;10&rdquo;，&ldquo;01&rdquo;，它们具有相同数量的连续1和0。

注意：

s.length 在1到50,000之间。

s 只包含&ldquo;0&rdquo;或&ldquo;1&rdquo;字符。697、数组的度

给定一个非空且只包含非负数的整数数组 nums, 数组的度的定义是指数组里任一元素出现频数的最大值。

你的任务是找到与 nums 拥有相同大小的度的最短连续子数组，返回其长度。

示例 1:

输入: [1, 2, 2, 3, 1]

输出: 2

解释:

输入数组的度是2，因为元素1和2的出现频数最大，均为2.

连续子数组里面拥有相同度的有如下所示:

[1, 2, 2, 3, 1], [1, 2, 2, 3], [2, 2, 3, 1], [1, 2, 2], [2, 2, 3], [2, 2]

最短连续子数组[2, 2]的长度为2，所以返回2.

示例 2:

输入: [1,2,2,3,1,4,2]

输出: 6

注意:

nums.length 在1到50,000区间范围内。

nums[i] 是一个在0到49,999范围内的整数。700、二叉搜索树中的搜索

给定二叉搜索树（BST）的根节点和一个值。 你需要在BST中找到节点值等于给定值的节点。 返回以该节点为根的子树。 如果节点不存在，则返回 NULL。

例如，

给定二叉搜索树:

4

/ \

2 7

/ \

1 3

和值: 2

你应该返回如下子树:

2

/ \

1 3

在上述示例中，如果要找的值是 5，但因为没有节点值为 5，我们应该返回 NULL。703、数据流中的第K大元素

设计一个找到数据流中第K大元素的类（class）。注意是排序后的第K大元素，不是第K个不同的元素。

你的 KthLargest 类需要一个同时接收整数 k 和整数数组nums 的构造器，它包含数据流中的初始元素。每次调用 KthLargest.add，返回当前数据流中第K大的元素。

示例:

int k = 3;

int[] arr = [4,5,8,2];

KthLargest kthLargest = new KthLargest(3, arr);

kthLargest.add(3); // returns 4

kthLargest.add(5); // returns 5

kthLargest.add(10); // returns 5

kthLargest.add(9); // returns 8

kthLargest.add(4); // returns 8

说明:

你可以假设 nums 的长度&ge; k-1 且k &ge; 1。704、二分查找

给定一个 n 个元素有序的（升序）整型数组 nums 和一个目标值 target ，写一个函数搜索 nums 中的 target，如果目标值存在返回下标，否则返回 -1。

示例 1:

输入: nums = [-1,0,3,5,9,12], target = 9

输出: 4

解释: 9 出现在 nums 中并且下标为 4

示例 2:

输入: nums = [-1,0,3,5,9,12], target = 2

输出: -1

解释: 2 不存在 nums 中因此返回 -1

提示：

你可以假设 nums 中的所有元素是不重复的。

n 将在 [1, 10000]之间。

nums 的每个元素都将在 [-9999, 9999]之间。705、设计哈希集合

不使用任何内建的哈希表库设计一个哈希集合

具体地说，你的设计应该包含以下的功能

add(value)：向哈希集合中插入一个值。

contains(value) ：返回哈希集合中是否存在这个值。

remove(value)：将给定值从哈希集合中删除。如果哈希集合中没有这个值，什么也不做。

示例:

MyHashSet hashSet = new MyHashSet();

hashSet.add(1);

hashSet.add(2);

hashSet.contains(1); // 返回 true

hashSet.contains(3); // 返回 false (未找到)

hashSet.add(2);

hashSet.contains(2); // 返回 true

hashSet.remove(2);

hashSet.contains(2); // 返回 false (已经被删除)

注意：

所有的值都在 [1, 1000000]的范围内。

操作的总数目在[1, 10000]范围内。

不要使用内建的哈希集合库。706、设计哈希映射

不使用任何内建的哈希表库设计一个哈希映射

具体地说，你的设计应该包含以下的功能

put(key, value)：向哈希映射中插入(键,值)的数值对。如果键对应的值已经存在，更新这个值。

get(key)：返回给定的键所对应的值，如果映射中不包含这个键，返回-1。

remove(key)：如果映射中存在这个键，删除这个数值对。

示例：

MyHashMap hashMap = new MyHashMap();

hashMap.put(1, 1);

hashMap.put(2, 2);

hashMap.get(1); // 返回 1

hashMap.get(3); // 返回 -1 (未找到)

hashMap.put(2, 1); // 更新已有的值

hashMap.get(2); // 返回 1

hashMap.remove(2); // 删除键为2的数据

hashMap.get(2); // 返回 -1 (未找到)

注意：

所有的值都在 [1, 1000000]的范围内。

操作的总数目在[1, 10000]范围内。

不要使用内建的哈希库。707、设计链表

设计链表的实现。您可以选择使用单链表或双链表。单链表中的节点应该具有两个属性：val 和 next。val 是当前节点的值，next 是指向下一个节点的指针/引用。如果要使用双向链表，则还需要一个属性 prev 以指示链表中的上一个节点。假设链表中的所有节点都是 0-index 的。

在链表类中实现这些功能：

get(index)：获取链表中第 index 个节点的值。如果索引无效，则返回-1。

addAtHead(val)：在链表的第一个元素之前添加一个值为 val 的节点。插入后，新节点将成为链表的第一个节点。

addAtTail(val)：将值为 val 的节点追加到链表的最后一个元素。

addAtIndex(index,val)：在链表中的第 index 个节点之前添加值为 val 的节点。如果 index 等于链表的长度，则该节点将附加到链表的末尾。如果 index 大于链表长度，则不会插入节点。

deleteAtIndex(index)：如果索引 index 有效，则删除链表中的第 index 个节点。

示例：

MyLinkedList linkedList = new MyLinkedList();

linkedList.addAtHead(1);

linkedList.addAtTail(3);

linkedList.addAtIndex(1,2); //链表变为1-> 2-> 3

linkedList.get(1); //返回2

linkedList.deleteAtIndex(1); //现在链表是1-> 3

linkedList.get(1); //返回3

提示：

所有值都在 [1, 1000] 之内。

操作次数将在 [1, 1000] 之内。

请不要使用内置的 LinkedList 库。709、转换成小写字母

实现函数 ToLowerCase()，该函数接收一个字符串参数 str，并将该字符串中的大写字母转换成小写字母，之后返回新的字符串。

示例 1：

输入: "Hello"

输出: "hello"

示例 2：

输入: "here"

输出: "here"

示例 3：

输入: "LOVELY"

输出: "lovely"717、1比特与2比特字符

有两种特殊字符。第一种字符可以用一比特0来表示。第二种字符可以用两比特(10 或 11)来表示。

现给一个由若干比特组成的字符串。问最后一个字符是否必定为一个一比特字符。给定的字符串总是由0结束。

示例 1:

输入:

bits = [1, 0, 0]

输出: True

解释:

唯一的编码方式是一个两比特字符和一个一比特字符。所以最后一个字符是一比特字符。

示例 2:

输入:

bits = [1, 1, 1, 0]

输出: False

解释:

唯一的编码方式是两比特字符和两比特字符。所以最后一个字符不是一比特字符。

注意:

1 <= len(bits) <= 1000.

bits[i] 总是0 或 1.720、词典中最长的单词

给出一个字符串数组words组成的一本英语词典。从中找出最长的一个单词，该单词是由words词典中其他单词逐步添加一个字母组成。若其中有多个可行的答案，则返回答案中字典序最小的单词。

若无答案，则返回空字符串。

示例 1:

输入:

words = ["w","wo","wor","worl", "world"]

输出: "world"

解释:

单词"world"可由"w", "wo", "wor", 和 "worl"添加一个字母组成。

示例 2:

输入:

words = ["a", "banana", "app", "appl", "ap", "apply", "apple"]

输出: "apple"

解释:

"apply"和"apple"都能由词典中的单词组成。但是"apple"得字典序小于"apply"。

注意:

所有输入的字符串都只包含小写字母。

words数组长度范围为[1,1000]。

words[i]的长度范围为[1,30]。724、寻找数组的中心索引

给定一个整数类型的数组 nums，请编写一个能够返回数组&ldquo;中心索引&rdquo;的方法。

我们是这样定义数组中心索引的：数组中心索引的左侧所有元素相加的和等于右侧所有元素相加的和。

如果数组不存在中心索引，那么我们应该返回 -1。如果数组有多个中心索引，那么我们应该返回最靠近左边的那一个。

示例 1:

输入:

nums = [1, 7, 3, 6, 5, 6]

输出: 3

解释:

索引3 (nums[3] = 6) 的左侧数之和(1 + 7 + 3 = 11)，与右侧数之和(5 + 6 = 11)相等。

同时, 3 也是第一个符合要求的中心索引。

示例 2:

输入:

nums = [1, 2, 3]

输出: -1

解释:

数组中不存在满足此条件的中心索引。

说明:

nums 的长度范围为 [0, 10000]。

任何一个 nums[i] 将会是一个范围在 [-1000, 1000]的整数。728、自除数

自除数 是指可以被它包含的每一位数除尽的数。

例如，128 是一个自除数，因为 128 % 1 == 0，128 % 2 == 0，128 % 8 == 0。

还有，自除数不允许包含 0 。

给定上边界和下边界数字，输出一个列表，列表的元素是边界（含边界）内所有的自除数。

示例 1：

输入：

上边界left = 1, 下边界right = 22

输出： [1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 12, 15, 22]

注意：

每个输入参数的边界满足 1 <= left <= right <= 10000。733、图像渲染

有一幅以二维整数数组表示的图画，每一个整数表示该图画的像素值大小，数值在 0 到 65535 之间。

给你一个坐标 (sr, sc) 表示图像渲染开始的像素值（行 ，列）和一个新的颜色值 newColor，让你重新上色这幅图像。

为了完成上色工作，从初始坐标开始，记录初始坐标的上下左右四个方向上像素值与初始坐标相同的相连像素点，接着再记录这四个方向上符合条件的像素点与他们对应四个方向上像素值与初始坐标相同的相连像素点，&hellip;&hellip;，重复该过程。将所有有记录的像素点的颜色值改为新的颜色值。

最后返回经过上色渲染后的图像。

示例 1:

输入:

image = [[1,1,1],[1,1,0],[1,0,1]]

sr = 1, sc = 1, newColor = 2

输出: [[2,2,2],[2,2,0],[2,0,1]]

解析:

在图像的正中间，(坐标(sr,sc)=(1,1)),

在路径上所有符合条件的像素点的颜色都被更改成2。

注意，右下角的像素没有更改为2，

因为它不是在上下左右四个方向上与初始点相连的像素点。

注意:

image 和 image[0] 的长度在范围 [1, 50] 内。

给出的初始点将满足 0 <= sr < image.length 和 0 <= sc < image[0].length。

image[i][j] 和 newColor 表示的颜色值在范围 [0, 65535]内。744、寻找比目标字母大的最小字母

给定一个只包含小写字母的有序数组letters 和一个目标字母 target，寻找有序数组里面比目标字母大的最小字母。

数组里字母的顺序是循环的。举个例子，如果目标字母target = 'z' 并且有序数组为 letters = ['a', 'b']，则答案返回 'a'。

示例:

输入:

letters = ["c", "f", "j"]

target = "a"

输出: "c"

输入:

letters = ["c", "f", "j"]

target = "c"

输出: "f"

输入:

letters = ["c", "f", "j"]

target = "d"

输出: "f"

输入:

letters = ["c", "f", "j"]

target = "g"

输出: "j"

输入:

letters = ["c", "f", "j"]

target = "j"

输出: "c"

输入:

letters = ["c", "f", "j"]

target = "k"

输出: "c"

注:

letters长度范围在[2, 10000]区间内。

letters 仅由小写字母组成，最少包含两个不同的字母。

目标字母target 是一个小写字母。746、使用最小花费爬楼梯

数组的每个索引做为一个阶梯，第 i个阶梯对应着一个非负数的体力花费值 cost[i](索引从0开始)。

每当你爬上一个阶梯你都要花费对应的体力花费值，然后你可以选择继续爬一个阶梯或者爬两个阶梯。

您需要找到达到楼层顶部的最低花费。在开始时，你可以选择从索引为 0 或 1 的元素作为初始阶梯。

示例 1:

输入: cost = [10, 15, 20]

输出: 15

解释: 最低花费是从cost[1]开始，然后走两步即可到阶梯顶，一共花费15。

示例 2:

输入: cost = [1, 100, 1, 1, 1, 100, 1, 1, 100, 1]

输出: 6

解释: 最低花费方式是从cost[0]开始，逐个经过那些1，跳过cost[3]，一共花费6。

注意：

cost 的长度将会在 [2, 1000]。

每一个 cost[i] 将会是一个Integer类型，范围为 [0, 999]。747、至少是其他数字两倍的最大数

在一个给定的数组nums中，总是存在一个最大元素 。

查找数组中的最大元素是否至少是数组中每个其他数字的两倍。

如果是，则返回最大元素的索引，否则返回-1。

示例 1:

输入: nums = [3, 6, 1, 0]

输出: 1

解释: 6是最大的整数, 对于数组中的其他整数,

6大于数组中其他元素的两倍。6的索引是1, 所以我们返回1.

示例 2:

输入: nums = [1, 2, 3, 4]

输出: -1

解释: 4没有超过3的两倍大, 所以我们返回 -1.

提示:

nums 的长度范围在[1, 50].

每个 nums[i] 的整数范围在 [0, 99].762、二进制表示中质数个计算置位

给定两个整数 L 和 R ，找到闭区间 [L, R] 范围内，计算置位位数为质数的整数个数。

（注意，计算置位代表二进制表示中1的个数。例如 21 的二进制表示 10101 有 3 个计算置位。还有，1 不是质数。）

示例 1:

输入: L = 6, R = 10

输出: 4

解释:

6 -> 110 (2 个计算置位，2 是质数)

7 -> 111 (3 个计算置位，3 是质数)

9 -> 1001 (2 个计算置位，2 是质数)

10-> 1010 (2 个计算置位，2 是质数)

示例 2:

输入: L = 10, R = 15

输出: 5

解释:

10 -> 1010 (2 个计算置位, 2 是质数)

11 -> 1011 (3 个计算置位, 3 是质数)

12 -> 1100 (2 个计算置位, 2 是质数)

13 -> 1101 (3 个计算置位, 3 是质数)

14 -> 1110 (3 个计算置位, 3 是质数)

15 -> 1111 (4 个计算置位, 4 不是质数)

注意:

L, R 是 L <= R 且在 [1, 10^6] 中的整数。

R - L 的最大值为 10000。766、托普利茨矩阵

如果一个矩阵的每一方向由左上到右下的对角线上具有相同元素，那么这个矩阵是托普利茨矩阵。

给定一个 M x N 的矩阵，当且仅当它是托普利茨矩阵时返回 True。

示例 1:

输入:

matrix = [

[1,2,3,4],

[5,1,2,3],

[9,5,1,2]

]

输出: True

解释:

在上述矩阵中, 其对角线为:

"[9]", "[5, 5]", "[1, 1, 1]", "[2, 2, 2]", "[3, 3]", "[4]"。

各条对角线上的所有元素均相同, 因此答案是True。

示例 2:

输入:

matrix = [

[1,2],

[2,2]

]

输出: False

解释:

对角线"[1, 2]"上的元素不同。

说明:

matrix 是一个包含整数的二维数组。

matrix 的行数和列数均在 [1, 20]范围内。

matrix[i][j] 包含的整数在 [0, 99]范围内。

进阶:

如果矩阵存储在磁盘上，并且磁盘内存是有限的，因此一次最多只能将一行矩阵加载到内存中，该怎么办？

如果矩阵太大以至于只能一次将部分行加载到内存中，该怎么办？771、宝石与石头

给定字符串J 代表石头中宝石的类型，和字符串 S代表你拥有的石头。 S 中每个字符代表了一种你拥有的石头的类型，你想知道你拥有的石头中有多少是宝石。

J 中的字母不重复，J 和 S中的所有字符都是字母。字母区分大小写，因此"a"和"A"是不同类型的石头。

示例 1:

输入: J = "aA", S = "aAAbbbb"

输出: 3

示例 2:

输入: J = "z", S = "ZZ"

输出: 0

注意:

S 和 J 最多含有50个字母。

J 中的字符不重复。783、二叉搜索树结点最小距离

给定一个二叉搜索树的根结点 root, 返回树中任意两节点的差的最小值。

示例：

输入: root = [4,2,6,1,3,null,null]

输出: 1

解释:

注意，root是树结点对象(TreeNode object)，而不是数组。

给定的树 [4,2,6,1,3,null,null] 可表示为下图:

4

/ \

2 6

/ \

1 3

最小的差值是 1, 它是节点1和节点2的差值, 也是节点3和节点2的差值。

注意：

二叉树的大小范围在 2 到 100。

二叉树总是有效的，每个节点的值都是整数，且不重复。784、字母大小写全排列

给定一个字符串S，通过将字符串S中的每个字母转变大小写，我们可以获得一个新的字符串。返回所有可能得到的字符串集合。

示例:

输入: S = "a1b2"

输出: ["a1b2", "a1B2", "A1b2", "A1B2"]

输入: S = "3z4"

输出: ["3z4", "3Z4"]

输入: S = "12345"

输出: ["12345"]

注意：

S 的长度不超过12。

S 仅由数字和字母组成。788、旋转数字

我们称一个数 X 为好数, 如果它的每位数字逐个地被旋转 180 度后，我们仍可以得到一个有效的，且和 X 不同的数。要求每位数字都要被旋转。

如果一个数的每位数字被旋转以后仍然还是一个数字， 则这个数是有效的。0, 1, 和 8 被旋转后仍然是它们自己；2 和 5 可以互相旋转成对方；6 和 9 同理，除了这些以外其他的数字旋转以后都不再是有效的数字。

现在我们有一个正整数 N, 计算从 1 到 N 中有多少个数 X 是好数？

示例:

输入: 10

输出: 4

解释:

在[1, 10]中有四个好数： 2, 5, 6, 9。

注意 1 和 10 不是好数, 因为他们在旋转之后不变。

注意:

N 的取值范围是 [1, 10000]。796、旋转字符串

给定两个字符串, A 和 B。

A 的旋转操作就是将 A 最左边的字符移动到最右边。 例如, 若 A = 'abcde'，在移动一次之后结果就是'bcdea' 。如果在若干次旋转操作之后，A 能变成B，那么返回True。

示例 1:

输入: A = 'abcde', B = 'cdeab'

输出: true

示例 2:

输入: A = 'abcde', B = 'abced'

输出: false

注意：

A 和 B 长度不超过 100。804、唯一摩尔斯密码词

国际摩尔斯密码定义一种标准编码方式，将每个字母对应于一个由一系列点和短线组成的字符串， 比如: "a" 对应 ".-", "b" 对应 "-...", "c" 对应 "-.-.", 等等。

为了方便，所有26个英文字母对应摩尔斯密码表如下：

[".-","-...","-.-.","-..",".","..-.","--.","....","..",".---","-.-",".-..","--","-.","---",".--.","--.-",".-.","...","-","..-","...-",".--","-..-","-.--","--.."]

给定一个单词列表，每个单词可以写成每个字母对应摩尔斯密码的组合。例如，"cab" 可以写成 "-.-.-....-"，(即 "-.-." + "-..." + ".-"字符串的结合)。我们将这样一个连接过程称作单词翻译。

返回我们可以获得所有词不同单词翻译的数量。

例如:

输入: words = ["gin", "zen", "gig", "msg"]

输出: 2

解释:

各单词翻译如下:

"gin" -> "--...-."

"zen" -> "--...-."

"gig" -> "--...--."

"msg" -> "--...--."

共有 2 种不同翻译, "--...-." 和 "--...--.".

注意:

单词列表words 的长度不会超过 100。

每个单词 words[i]的长度范围为 [1, 12]。

每个单词 words[i]只包含小写字母。806、写字符串需要的行数

我们要把给定的字符串 S 从左到右写到每一行上，每一行的最大宽度为100个单位，如果我们在写某个字母的时候会使这行超过了100 个单位，那么我们应该把这个字母写到下一行。我们给定了一个数组 widths ，这个数组 widths[0] 代表 'a' 需要的单位， widths[1] 代表 'b' 需要的单位，...， widths[25] 代表 'z' 需要的单位。

现在回答两个问题：至少多少行能放下S，以及最后一行使用的宽度是多少个单位？将你的答案作为长度为2的整数列表返回。

示例 1:

输入:

widths = [10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10]

S = "abcdefghijklmnopqrstuvwxyz"

输出: [3, 60]

解释:

所有的字符拥有相同的占用单位10。所以书写所有的26个字母，

我们需要2个整行和占用60个单位的一行。

示例 2:

输入:

widths = [4,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10,10]

S = "bbbcccdddaaa"

输出: [2, 4]

解释:

除去字母'a'所有的字符都是相同的单位10，并且字符串 "bbbcccdddaa" 将会覆盖 9 \* 10 + 2 \* 4 = 98 个单位.

最后一个字母 'a' 将会被写到第二行，因为第一行只剩下2个单位了。

所以，这个答案是2行，第二行有4个单位宽度。

注:

字符串 S 的长度在 [1, 1000] 的范围。

S 只包含小写字母。

widths 是长度为 26的数组。

widths[i] 值的范围在 [2, 10]。811、子域名访问计数

一个网站域名，如"discuss.leetcode.com"，包含了多个子域名。作为顶级域名，常用的有"com"，下一级则有"leetcode.com"，最低的一级为"discuss.leetcode.com"。当我们访问域名"discuss.leetcode.com"时，也同时访问了其父域名"leetcode.com"以及顶级域名 "com"。

给定一个带访问次数和域名的组合，要求分别计算每个域名被访问的次数。其格式为访问次数+空格+地址，例如："9001 discuss.leetcode.com"。

接下来会给出一组访问次数和域名组合的列表cpdomains 。要求解析出所有域名的访问次数，输出格式和输入格式相同，不限定先后顺序。

示例 1:

输入:

["9001 discuss.leetcode.com"]

输出:

["9001 discuss.leetcode.com", "9001 leetcode.com", "9001 com"]

说明:

例子中仅包含一个网站域名："discuss.leetcode.com"。按照前文假设，子域名"leetcode.com"和"com"都会被访问，所以它们都被访问了9001次。

示例 2

输入:

["900 google.mail.com", "50 yahoo.com", "1 intel.mail.com", "5 wiki.org"]

输出:

["901 mail.com","50 yahoo.com","900 google.mail.com","5 wiki.org","5 org","1 intel.mail.com","951 com"]

说明:

按照假设，会访问"google.mail.com" 900次，"yahoo.com" 50次，"intel.mail.com" 1次，"wiki.org" 5次。

而对于父域名，会访问"mail.com" 900+1 = 901次，"com" 900 + 50 + 1 = 951次，和 "org" 5 次。

注意事项：

cpdomains 的长度小于 100。

每个域名的长度小于100。

每个域名地址包含一个或两个"."符号。

输入中任意一个域名的访问次数都小于10000。812、最大三角形面积

给定包含多个点的集合，从其中取三个点组成三角形，返回能组成的最大三角形的面积。

示例:

输入: points = [[0,0],[0,1],[1,0],[0,2],[2,0]]

输出: 2

解释:

这五个点如下图所示。组成的橙色三角形是最大的，面积为2。

注意:

3 <= points.length <= 50.

不存在重复的点。

-50 <= points[i][j] <= 50.

结果误差值在 10^-6 以内都认为是正确答案。819、最常见的单词

给定一个段落 (paragraph) 和一个禁用单词列表 (banned)。返回出现次数最多，同时不在禁用列表中的单词。题目保证至少有一个词不在禁用列表中，而且答案唯一。

禁用列表中的单词用小写字母表示，不含标点符号。段落中的单词不区分大小写。答案都是小写字母。

示例:

输入:

paragraph = "Bob hit a ball, the hit BALL flew far after it was hit."

banned = ["hit"]

输出: "ball"

解释:

"hit" 出现了3次，但它是一个禁用的单词。

"ball" 出现了2次 (同时没有其他单词出现2次)，所以它是段落里出现次数最多的，且不在禁用列表中的单词。

注意，所有这些单词在段落里不区分大小写，标点符号需要忽略（即使是紧挨着单词也忽略， 比如 "ball,"），

"hit"不是最终的答案，虽然它出现次数更多，但它在禁用单词列表中。

说明:

1 <= 段落长度 <= 1000.

1 <= 禁用单词个数 <= 100.

1 <= 禁用单词长度 <= 10.

答案是唯一的, 且都是小写字母 (即使在 paragraph 里是大写的，即使是一些特定的名词，答案都是小写的。)

paragraph 只包含字母、空格和下列标点符号!?',;.

paragraph 里单词之间都由空格隔开。

不存在没有连字符或者带有连字符的单词。

单词里只包含字母，不会出现省略号或者其他标点符号。821、字符的最短距离

给定一个字符串 S 和一个字符 C。返回一个代表字符串 S 中每个字符到字符串 S 中的字符 C 的最短距离的数组。

示例 1:

输入: S = "loveleetcode", C = 'e'

输出: [3, 2, 1, 0, 1, 0, 0, 1, 2, 2, 1, 0]

说明:

字符串 S 的长度范围为 [1, 10000]。

C 是一个单字符，且保证是字符串 S 里的字符。

S 和 C 中的所有字母均为小写字母。824、山羊拉丁文

给定一个由空格分割单词的句子 S。每个单词只包含大写或小写字母。

我们要将句子转换为 &ldquo;Goat Latin&rdquo;（一种类似于 猪拉丁文 - Pig Latin 的虚构语言）。

山羊拉丁文的规则如下：

如果单词以元音开头（a, e, i, o, u），在单词后添加"ma"。

例如，单词"apple"变为"applema"。

如果单词以辅音字母开头（即非元音字母），移除第一个字符并将它放到末尾，之后再添加"ma"。

例如，单词"goat"变为"oatgma"。

根据单词在句子中的索引，在单词最后添加与索引相同数量的字母'a'，索引从1开始。

例如，在第一个单词后添加"a"，在第二个单词后添加"aa"，以此类推。

返回将 S 转换为山羊拉丁文后的句子。

示例 1:

输入: "I speak Goat Latin"

输出: "Imaa peaksmaaa oatGmaaaa atinLmaaaaa"

示例 2:

输入: "The quick brown fox jumped over the lazy dog"

输出: "heTmaa uickqmaaa rownbmaaaa oxfmaaaaa umpedjmaaaaaa overmaaaaaaa hetmaaaaaaaa azylmaaaaaaaaa ogdmaaaaaaaaaa"

说明:

S 中仅包含大小写字母和空格。单词间有且仅有一个空格。

1 <= S.length <= 150。830、较大分组的位置

在一个由小写字母构成的字符串 S 中，包含由一些连续的相同字符所构成的分组。

例如，在字符串 S = "abbxxxxzyy" 中，就含有 "a", "bb", "xxxx", "z" 和 "yy" 这样的一些分组。

我们称所有包含大于或等于三个连续字符的分组为较大分组。找到每一个较大分组的起始和终止位置。

最终结果按照字典顺序输出。

示例 1:

输入: "abbxxxxzzy"

输出: [[3,6]]

解释: "xxxx" 是一个起始于 3 且终止于 6 的较大分组。

示例 2:

输入: "abc"

输出: []

解释: "a","b" 和 "c" 均不是符合要求的较大分组。

示例 3:

输入: "abcdddeeeeaabbbcd"

输出: [[3,5],[6,9],[12,14]]

说明: 1 <= S.length <= 1000832、翻转图像

给定一个二进制矩阵 A，我们想先水平翻转图像，然后反转图像并返回结果。

水平翻转图片就是将图片的每一行都进行翻转，即逆序。例如，水平翻转 [1, 1, 0] 的结果是 [0, 1, 1]。

反转图片的意思是图片中的 0 全部被 1 替换， 1 全部被 0 替换。例如，反转 [0, 1, 1] 的结果是 [1, 0, 0]。

示例 1:

输入: [[1,1,0],[1,0,1],[0,0,0]]

输出: [[1,0,0],[0,1,0],[1,1,1]]

解释: 首先翻转每一行: [[0,1,1],[1,0,1],[0,0,0]]；

然后反转图片: [[1,0,0],[0,1,0],[1,1,1]]

示例 2:

输入: [[1,1,0,0],[1,0,0,1],[0,1,1,1],[1,0,1,0]]

输出: [[1,1,0,0],[0,1,1,0],[0,0,0,1],[1,0,1,0]]

解释: 首先翻转每一行: [[0,0,1,1],[1,0,0,1],[1,1,1,0],[0,1,0,1]]；

然后反转图片: [[1,1,0,0],[0,1,1,0],[0,0,0,1],[1,0,1,0]]

说明:

1 <= A.length = A[0].length <= 20

0 <= A[i][j] <= 1836、矩形重叠

矩形以列表 [x1, y1, x2, y2] 的形式表示，其中 (x1, y1) 为左下角的坐标，(x2, y2) 是右上角的坐标。

如果相交的面积为正，则称两矩形重叠。需要明确的是，只在角或边接触的两个矩形不构成重叠。

给出两个矩形，判断它们是否重叠并返回结果。

示例 1：

输入：rec1 = [0,0,2,2], rec2 = [1,1,3,3]

输出：true

示例 2：

输入：rec1 = [0,0,1,1], rec2 = [1,0,2,1]

输出：false

说明：

两个矩形 rec1 和 rec2 都以含有四个整数的列表的形式给出。

矩形中的所有坐标都处于 -10^9 和 10^9 之间。840、矩阵中的幻方

3 x 3 的幻方是一个填充有从 1 到 9 的不同数字的 3 x 3 矩阵，其中每行，每列以及两条对角线上的各数之和都相等。

给定一个由整数组成的 N &times; N 矩阵，其中有多少个 3 &times; 3 的 &ldquo;幻方&rdquo; 子矩阵？（每个子矩阵都是连续的）。

示例 1:

输入: [[4,3,8,4],

[9,5,1,9],

[2,7,6,2]]

输出: 1

解释:

下面的子矩阵是一个 3 x 3 的幻方：

438

951

276

而这一个不是：

384

519

762

总的来说，在本示例所给定的矩阵中只有一个 3 x 3 的幻方子矩阵。

提示:

1 <= grid.length = grid[0].length <= 10

0 <= grid[i][j] <= 15844、比较含退格的字符串

给定 S 和 T 两个字符串，当它们分别被输入到空白的文本编辑器后，判断二者是否相等，并返回结果。 # 代表退格字符。

示例 1：

输入：S = "ab#c", T = "ad#c"

输出：true

解释：S 和 T 都会变成 &ldquo;ac&rdquo;。

示例 2：

输入：S = "ab##", T = "c#d#"

输出：true

解释：S 和 T 都会变成 &ldquo;&rdquo;。

示例 3：

输入：S = "a##c", T = "#a#c"

输出：true

解释：S 和 T 都会变成 &ldquo;c&rdquo;。

示例 4：

输入：S = "a#c", T = "b"

输出：false

解释：S 会变成 &ldquo;c&rdquo;，但 T 仍然是 &ldquo;b&rdquo;。

提示：

1 <= S.length <= 200

1 <= T.length <= 200

S 和 T 只含有小写字母以及字符 '#'。

849、到最近的人的最大距离

在一排座位（ seats）中，1 代表有人坐在座位上，0 代表座位上是空的。

至少有一个空座位，且至少有一人坐在座位上。

亚历克斯希望坐在一个能够使他与离他最近的人之间的距离达到最大化的座位上。

返回他到离他最近的人的最大距离。

示例 1：

输入：[1,0,0,0,1,0,1]

输出：2

解释：

如果亚历克斯坐在第二个空位（seats[2]）上，他到离他最近的人的距离为 2 。

如果亚历克斯坐在其它任何一个空位上，他到离他最近的人的距离为 1 。

因此，他到离他最近的人的最大距离是 2 。

示例 2：

输入：[1,0,0,0]

输出：3

解释：

如果亚历克斯坐在最后一个座位上，他离最近的人有 3 个座位远。

这是可能的最大距离，所以答案是 3 。

提示：

1 <= seats.length <= 20000

seats 中只含有 0 和 1，至少有一个 0，且至少有一个 1。852、山脉数组的峰顶索引

我们把符合下列属性的数组 A 称作山脉：

A.length >= 3

存在 0 < i < A.length - 1 使得A[0] < A[1] < ... A[i-1] < A[i] > A[i+1] > ... > A[A.length - 1]

给定一个确定为山脉的数组，返回任何满足 A[0] < A[1] < ... A[i-1] < A[i] > A[i+1] > ... > A[A.length - 1] 的 i 的值。

示例 1：

输入：[0,1,0]

输出：1

示例 2：

输入：[0,2,1,0]

输出：1

提示：

3 <= A.length <= 10000

0 <= A[i] <= 10^6

A 是如上定义的山脉

859、亲密字符串

给定两个由小写字母构成的字符串 A 和 B ，只要我们可以通过交换 A 中的两个字母得到与 B 相等的结果，就返回 true ；否则返回 false 。

示例 1：

输入： A = "ab", B = "ba"

输出： true

示例 2：

输入： A = "ab", B = "ab"

输出： false

示例 3:

输入： A = "aa", B = "aa"

输出： true

示例 4：

输入： A = "aaaaaaabc", B = "aaaaaaacb"

输出： true

示例 5：

输入： A = "", B = "aa"

输出： false

提示：

0 <= A.length <= 20000

0 <= B.length <= 20000

A 和 B 仅由小写字母构成。860、柠檬水找零

在柠檬水摊上，每一杯柠檬水的售价为 5 美元。

顾客排队购买你的产品，（按账单 bills 支付的顺序）一次购买一杯。

每位顾客只买一杯柠檬水，然后向你付 5 美元、10 美元或 20 美元。你必须给每个顾客正确找零，也就是说净交易是每位顾客向你支付 5 美元。

注意，一开始你手头没有任何零钱。

如果你能给每位顾客正确找零，返回 true ，否则返回 false 。

示例 1：

输入：[5,5,5,10,20]

输出：true

解释：

前 3 位顾客那里，我们按顺序收取 3 张 5 美元的钞票。

第 4 位顾客那里，我们收取一张 10 美元的钞票，并返还 5 美元。

第 5 位顾客那里，我们找还一张 10 美元的钞票和一张 5 美元的钞票。

由于所有客户都得到了正确的找零，所以我们输出 true。

示例 2：

输入：[5,5,10]

输出：true

示例 3：

输入：[10,10]

输出：false

示例 4：

输入：[5,5,10,10,20]

输出：false

解释：

前 2 位顾客那里，我们按顺序收取 2 张 5 美元的钞票。

对于接下来的 2 位顾客，我们收取一张 10 美元的钞票，然后返还 5 美元。

对于最后一位顾客，我们无法退回 15 美元，因为我们现在只有两张 10 美元的钞票。

由于不是每位顾客都得到了正确的找零，所以答案是 false。

提示：

0 <= bills.length <= 10000

bills[i] 不是 5 就是 10 或是 20 867、转置矩阵

给定一个矩阵 A， 返回 A 的转置矩阵。

矩阵的转置是指将矩阵的主对角线翻转，交换矩阵的行索引与列索引。

示例 1：

输入：[[1,2,3],[4,5,6],[7,8,9]]

输出：[[1,4,7],[2,5,8],[3,6,9]]

示例 2：

输入：[[1,2,3],[4,5,6]]

输出：[[1,4],[2,5],[3,6]]

提示：

1 <= A.length <= 1000

1 <= A[0].length <= 1000868、二进制间距

给定一个正整数 N，找到并返回 N 的二进制表示中两个连续的 1 之间的最长距离。

如果没有两个连续的 1，返回 0 。

示例 1：

输入：22

输出：2

解释：

22 的二进制是 0b10110 。

在 22 的二进制表示中，有三个 1，组成两对连续的 1 。

第一对连续的 1 中，两个 1 之间的距离为 2 。

第二对连续的 1 中，两个 1 之间的距离为 1 。

答案取两个距离之中最大的，也就是 2 。

示例 2：

输入：5

输出：2

解释：

5 的二进制是 0b101 。

示例 3：

输入：6

输出：1

解释：

6 的二进制是 0b110 。

示例 4：

输入：8

输出：0

解释：

8 的二进制是 0b1000 。

在 8 的二进制表示中没有连续的 1，所以返回 0 。

提示：

1 <= N <= 10^9872、叶子相似的树

请考虑一颗二叉树上所有的叶子，这些叶子的值按从左到右的顺序排列形成一个 叶值序列 。

举个例子，如上图所示，给定一颗叶值序列为 (6, 7, 4, 9, 8) 的树。

如果有两颗二叉树的叶值序列是相同，那么我们就认为它们是 叶相似 的。

如果给定的两个头结点分别为 root1 和 root2 的树是叶相似的，则返回 true；否则返回 false 。

提示：

给定的两颗树可能会有 1 到 100 个结点。874、模拟行走机器人

机器人在一个无限大小的网格上行走，从点 (0, 0) 处开始出发，面向北方。该机器人可以接收以下三种类型的命令：

-2：向左转 90 度

-1：向右转 90 度

1 <= x <= 9：向前移动 x 个单位长度

在网格上有一些格子被视为障碍物。

第 i 个障碍物位于网格点 (obstacles[i][0], obstacles[i][1])

如果机器人试图走到障碍物上方，那么它将停留在障碍物的前一个网格方块上，但仍然可以继续该路线的其余部分。

返回从原点到机器人的最大欧式距离的平方。

示例 1：

输入: commands = [4,-1,3], obstacles = []

输出: 25

解释: 机器人将会到达 (3, 4)

示例 2：

输入: commands = [4,-1,4,-2,4], obstacles = [[2,4]]

输出: 65

解释: 机器人在左转走到 (1, 8) 之前将被困在 (1, 4) 处

提示：

0 <= commands.length <= 10000

0 <= obstacles.length <= 10000

-30000 <= obstacle[i][0] <= 30000

-30000 <= obstacle[i][1] <= 30000

答案保证小于 2 ^ 31876、链表的中间结点

给定一个带有头结点 head 的非空单链表，返回链表的中间结点。

如果有两个中间结点，则返回第二个中间结点。

示例 1：

输入：[1,2,3,4,5]

输出：此列表中的结点 3 (序列化形式：[3,4,5])

返回的结点值为 3 。 (测评系统对该结点序列化表述是 [3,4,5])。

注意，我们返回了一个 ListNode 类型的对象 ans，这样：

ans.val = 3, ans.next.val = 4, ans.next.next.val = 5, 以及 ans.next.next.next = NULL.

示例 2：

输入：[1,2,3,4,5,6]

输出：此列表中的结点 4 (序列化形式：[4,5,6])

由于该列表有两个中间结点，值分别为 3 和 4，我们返回第二个结点。

提示：

给定链表的结点数介于 1 和 100 之间。883、三维形体投影面积

在 N \* N 的网格中，我们放置了一些与 x，y，z 三轴对齐的 1 \* 1 \* 1 立方体。

每个值 v = grid[i][j] 表示 v 个正方体叠放在单元格 (i, j) 上。

现在，我们查看这些立方体在 xy、yz 和 zx 平面上的投影。

投影就像影子，将三维形体映射到一个二维平面上。

在这里，从顶部、前面和侧面看立方体时，我们会看到&ldquo;影子&rdquo;。

返回所有三个投影的总面积。

示例 1：

输入：[[2]]

输出：5

示例 2：

输入：[[1,2],[3,4]]

输出：17

解释：

这里有该形体在三个轴对齐平面上的三个投影(&ldquo;阴影部分&rdquo;)。

示例 3：

输入：[[1,0],[0,2]]

输出：8

示例 4：

输入：[[1,1,1],[1,0,1],[1,1,1]]

输出：14

示例 5：

输入：[[2,2,2],[2,1,2],[2,2,2]]

输出：21

提示：

1 <= grid.length = grid[0].length <= 50

0 <= grid[i][j] <= 50884、两句话中的不常见单词

给定两个句子 A 和 B 。 （句子是一串由空格分隔的单词。每个单词仅由小写字母组成。）

如果一个单词在其中一个句子中只出现一次，在另一个句子中却没有出现，那么这个单词就是不常见的。

返回所有不常用单词的列表。

您可以按任何顺序返回列表。

示例 1：

输入：A = "this apple is sweet", B = "this apple is sour"

输出：["sweet","sour"]

示例 2：

输入：A = "apple apple", B = "banana"

输出：["banana"]

提示：

0 <= A.length <= 200

0 <= B.length <= 200

A 和 B 都只包含空格和小写字母。888、公平的糖果交换

爱丽丝和鲍勃有不同大小的糖果棒：A[i] 是爱丽丝拥有的第 i 块糖的大小，B[j] 是鲍勃拥有的第 j 块糖的大小。

因为他们是朋友，所以他们想交换一个糖果棒，这样交换后，他们都有相同的糖果总量。（一个人拥有的糖果总量是他们拥有的糖果棒大小的总和。）

返回一个整数数组 ans，其中 ans[0] 是爱丽丝必须交换的糖果棒的大小，ans[1] 是 Bob 必须交换的糖果棒的大小。

如果有多个答案，你可以返回其中任何一个。保证答案存在。

示例 1：

输入：A = [1,1], B = [2,2]

输出：[1,2]

示例 2：

输入：A = [1,2], B = [2,3]

输出：[1,2]

示例 3：

输入：A = [2], B = [1,3]

输出：[2,3]

示例 4：

输入：A = [1,2,5], B = [2,4]

输出：[5,4]

提示：

1 <= A.length <= 10000

1 <= B.length <= 10000

1 <= A[i] <= 100000

1 <= B[i] <= 100000

保证爱丽丝与鲍勃的糖果总量不同。

答案肯定存在。892、三维形体的表面积

在 N \* N 的网格上，我们放置一些 1 \* 1 \* 1 的立方体。

每个值 v = grid[i][j] 表示 v 个正方体叠放在单元格 (i, j) 上。

返回结果形体的总表面积。

示例 1：

输入：[[2]]

输出：10

示例 2：

输入：[[1,2],[3,4]]

输出：34

示例 3：

输入：[[1,0],[0,2]]

输出：16

示例 4：

输入：[[1,1,1],[1,0,1],[1,1,1]]

输出：32

示例 5：

输入：[[2,2,2],[2,1,2],[2,2,2]]

输出：46

提示：

1 <= N <= 50

0 <= grid[i][j] <= 50893、特殊等价字符串组

你将得到一个字符串数组 A。

如果经过任意次数的移动，S == T，那么两个字符串 S 和 T 是特殊等价的。

一次移动包括选择两个索引 i 和 j，且 i％2 == j％2，并且交换 S[j] 和 S [i]。

现在规定，A 中的特殊等价字符串组是 A 的非空子集 S，这样不在 S 中的任何字符串与 S 中的任何字符串都不是特殊等价的。

返回 A 中特殊等价字符串组的数量。

示例 1：

输入：["a","b","c","a","c","c"]

输出：3

解释：3 组 ["a","a"]，["b"]，["c","c","c"]

示例 2：

输入：["aa","bb","ab","ba"]

输出：4

解释：4 组 ["aa"]，["bb"]，["ab"]，["ba"]

示例 3：

输入：["abc","acb","bac","bca","cab","cba"]

输出：3

解释：3 组 ["abc","cba"]，["acb","bca"]，["bac","cab"]

示例 4：

输入：["abcd","cdab","adcb","cbad"]

输出：1

解释：1 组 ["abcd","cdab","adcb","cbad"]

提示：

1 <= A.length <= 1000

1 <= A[i].length <= 20

所有 A[i] 都具有相同的长度。

所有 A[i] 都只由小写字母组成。896、单调数列

如果数组是单调递增或单调递减的，那么它是单调的。

如果对于所有 i <= j，A[i] <= A[j]，那么数组 A 是单调递增的。 如果对于所有 i <= j，A[i]> = A[j]，那么数组 A 是单调递减的。

当给定的数组 A 是单调数组时返回 true，否则返回 false。

示例 1：

输入：[1,2,2,3]

输出：true

示例 2：

输入：[6,5,4,4]

输出：true

示例 3：

输入：[1,3,2]

输出：false

示例 4：

输入：[1,2,4,5]

输出：true

示例 5：

输入：[1,1,1]

输出：true

提示：

1 <= A.length <= 50000

-100000 <= A[i] <= 100000897、递增顺序查找树

给定一个树，按顺序重新排列树，使树中最左边的结点现在是树的根，并且每个结点没有左子结点，只有一个右子结点。

示例 ：

输入：[5,3,6,2,4,null,8,1,null,null,null,7,9]

5

/ \

3 6

/ \ \

2 4 8

/ / \

1 7 9

输出：[1,null,2,null,3,null,4,null,5,null,6,null,7,null,8,null,9]

1

\

2

\

3

\

4

\

5

\

6

\

7

\

8

\

9

提示：

给定树中的结点数介于 1 和 100 之间。

每个结点都有一个从 0 到 1000 范围内的唯一整数值。