

53.Maximum Subarray

Find the contiguous subarray within an array (containing at least one number) which has the largest sum.

For example, given the array [-2,1,-3,4,-1,2,1,-5,4],

the contiguous subarray [4,-1,2,1] has the largest sum = 6.

**More practice:**

If you have figured out the O(*n*) solution, try coding another solution using the divide and conquer approach, which is more subtle.

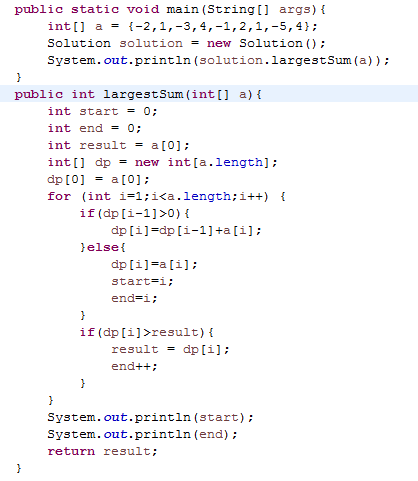
2 solutions：

(1)最大值的子array，要不就是当前index值加上之前的sum，要不就是之前的sum，要不就是当前index的值本身；

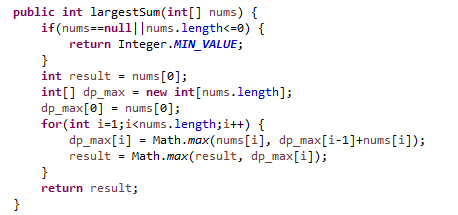
algorithm that operates on arrays: it starts at the left end (element A[1]) and scans through to the right end (element A[n]), keeping track of the maximum sum subvector seen so far. The maximum is initially A[0]. Suppose we’ve solved the problem for A[1 … i - 1]; how can we extend that to A[1 … i]? The maximum sum in the first I elements is either the maximum sum in the first i - 1 elements (which we’ll call MaxSoFar), or it is that of a subvector that ends in position i (which we’ll call MaxEndingHere)

(2)DP算法（Dynamic Programming）

用一个值result记录截至当前的最大subarray的sum值，用一个数组dp[i]记录截至当前的元素中前i-1个的sum值或当前值nums[i],根据前i-1是正还是负，是正则nums[i]加上之前的sum，否则就等于当前nums[i];



Beats:89.53%



58. Length of Last Word

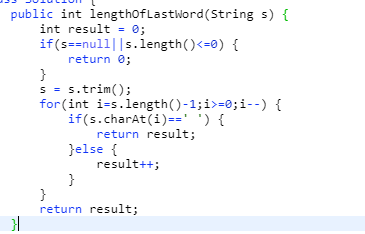
Given a string *s* consists of upper/lower-case alphabets and empty space characters ' ', return the length of last word in the string.

If the last word does not exist, return 0.

**Note:** A word is defined as a character sequence consists of non-space characters only.

倒着往前数

100%：



66. Plus One

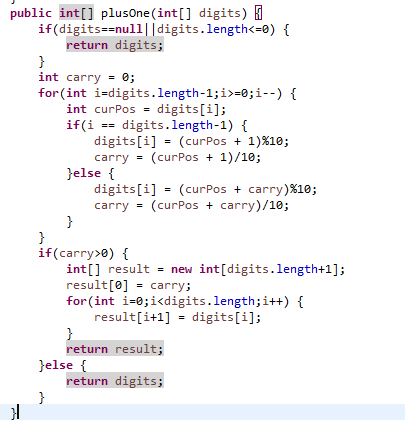
Given a non-negative integer represented as a **non-empty** array of digits, plus one to the integer.

You may assume the integer do not contain any leading zero, except the number 0 itself.

The digits are stored such that the most significant digit is at the head of the list.

设置一个flag来标记一下当前的进位，倒着往前遍历并相加；

100%：



67. Add Binary

Given two binary strings, return their sum (also a binary string).

For example,

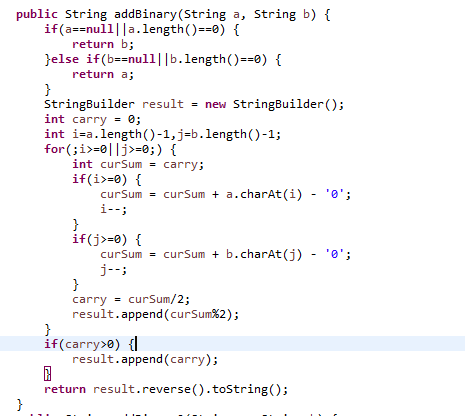
a = "11"

b = "1"

Return "100"

类似上面的66题，倒着往前加，用StringBuilder，直接用char加不要换算成数字（直接当前char相加后减去'0'即可），最后用StringBuilder的reverse翻转一下；

100%：



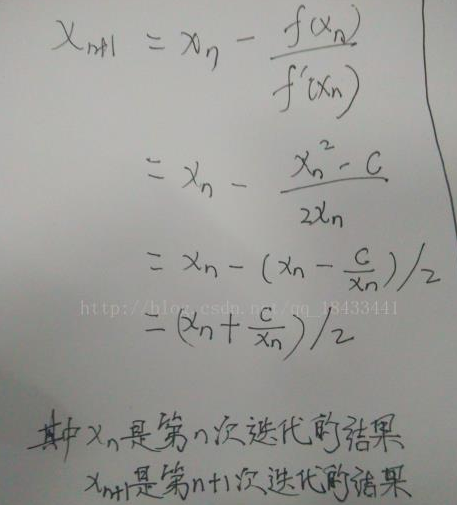
69. Sqrt(x)

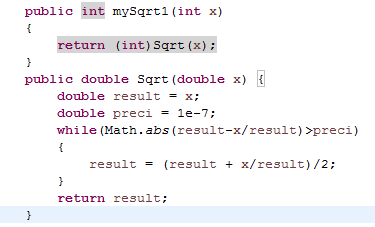
Implement int sqrt(int x).

Compute and return the square root of *x*.

**x** is guaranteed to be a non-negative integer.

牛顿迭代法：



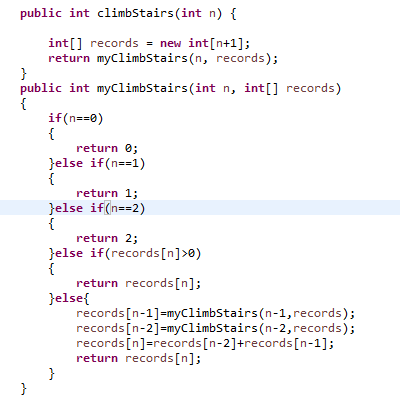


70. Climbing Stairs

You are climbing a stair case. It takes *n* steps to reach to the top.

Each time you can either climb 1 or 2 steps. In how many distinct ways can you climb to the top?

递归，但是要缓存每一次的结果，否则时间复杂度就是2^n就超时了，缓存结果的话就是O(n)



Beats:100%



83. Remove Duplicates from Sorted List

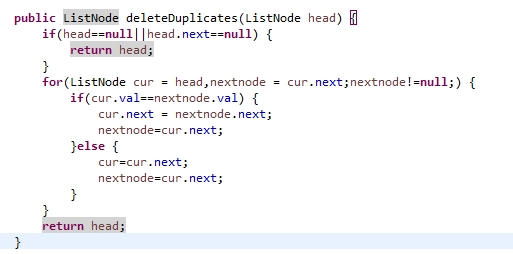
Given a sorted linked list, delete all duplicates such that each element appear only *once*.

For example,

Given 1->1->2, return 1->2.

Given 1->1->2->3->3, return 1->2->3.

Beats:100%



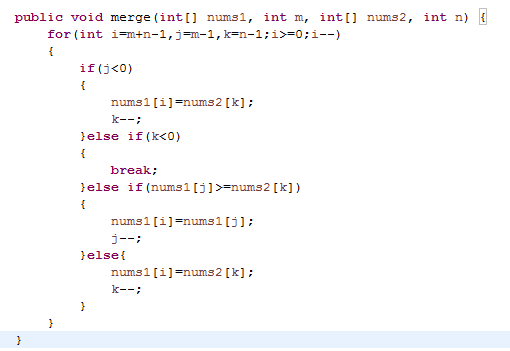
88. Merge Sorted Array

Given two sorted integer arrays *nums1* and *nums2*, merge *nums2* into *nums1* as one sorted array.

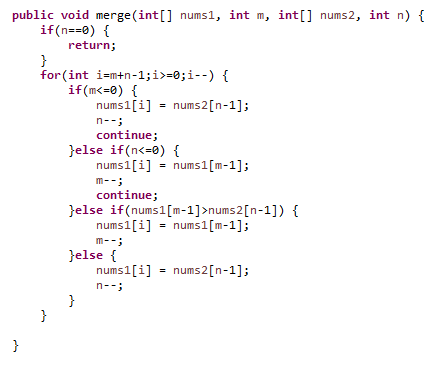
**Note:**

You may assume that *nums1* has enough space (size that is greater or equal to *m* + *n*) to hold additional elements from *nums2*. The number of elements initialized in *nums1* and *nums2* are *m* and *n* respectively

从nums1的尾部（第m+n个位置）往前依次插入,两个指针分别指向nums1和nums2的队胃往前遍历并将大的数值插入到当前位置；复杂度O（n）



Beats:100%



100. Same Tree

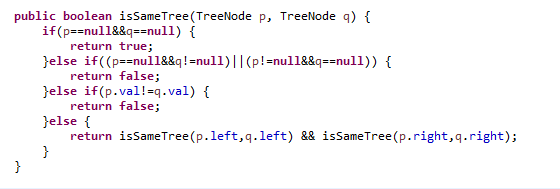
Given two binary trees, write a function to check if they are the same or not.

Two binary trees are considered the same if they are structurally identical and the nodes have the same value.

用递归（recursively）：



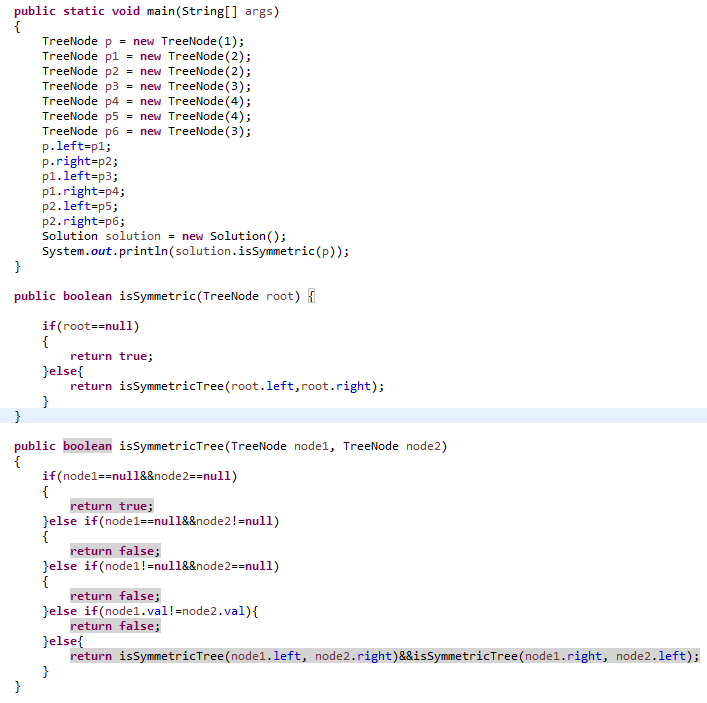
Beats:100%



101. Symmetric Tree

Given a binary tree, check whether it is a mirror of itself (ie, symmetric around its center).

用递归（recursively）：



Beats:100%



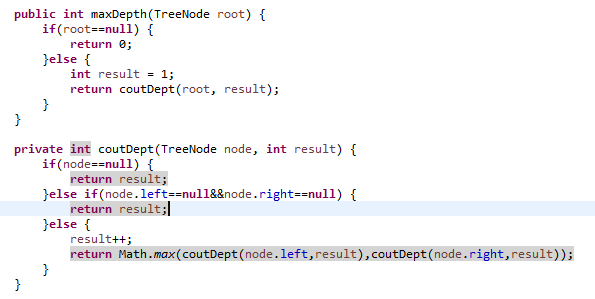
104. Maximum Depth of Binary Tree

Given a binary tree, find its maximum depth.

The maximum depth is the number of nodes along the longest path from the root node down to the farthest leaf node.

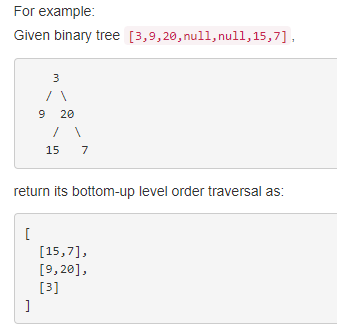


Beats:100%



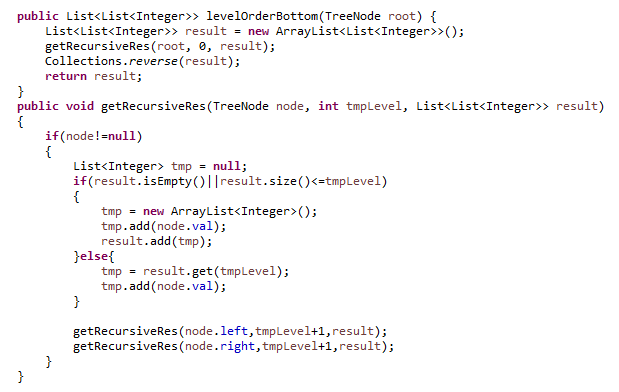
107. Binary Tree Level Order Traversal II

Given a binary tree, return the *bottom-up level order* traversal of its nodes' values. (ie, from left to right, level by level from leaf to root).

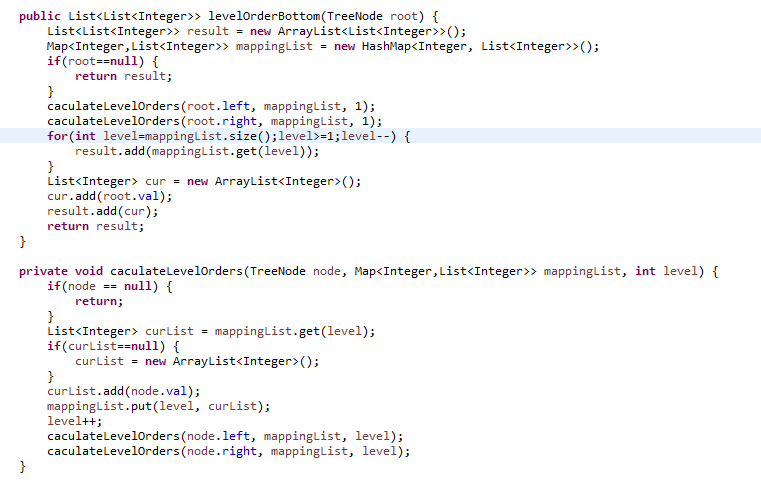


按先根序算法（先遍历根再左子树再右子树）递归遍历，同时用一个二维list记录当前node值（二维list可以通过get拿到第几行的list），这样最终结果是个从上到下的顺序的，最后反转一下外层list即可：

beats 100%



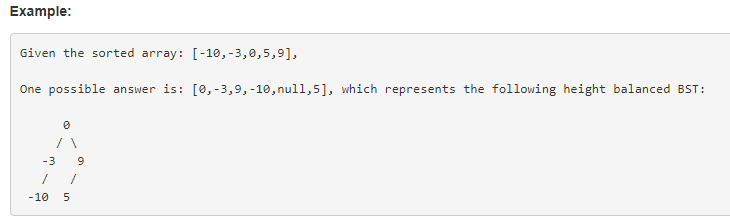
Beats:96.35%



108. Convert Sorted Array to Binary Search Tree

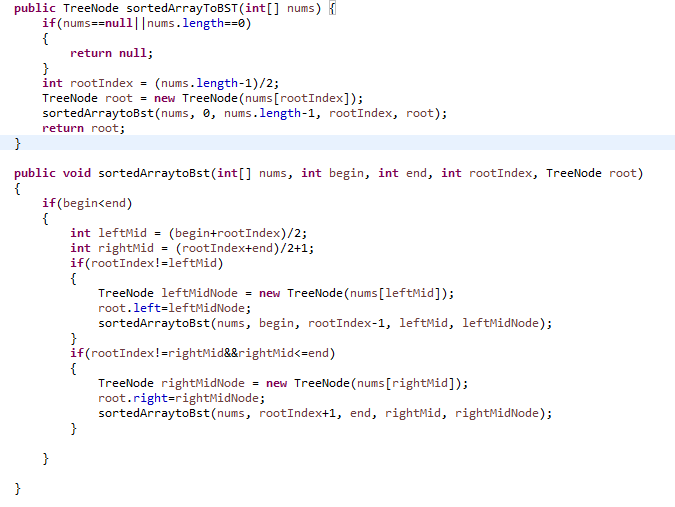
Given an array where elements are sorted in ascending order, convert it to a height balanced BST.

For this problem, a height-balanced binary tree is defined as a binary tree in which the depth of the two subtrees of *every* node never differ by more than 1.



利用Binary Search Tree的一个特性：一个根节点的左边的数肯定都是小于右边的数据的；再加上本身nums是排序好的，所以可先找到数组中间的数作为根节点，然后依次从左边字数组（字数组中间位置）和右边字数组（字数组中间位置）中找到这个根节点的左右子节点； 然后一次递归左右子节点和左右子树（左节点和左子数组；右子节点和右子数组）：

beats 100%



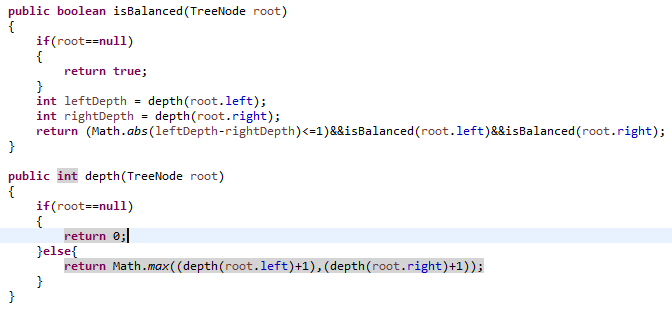
110. Balanced Binary Tree

Given a binary tree, determine if it is height-balanced.

For this problem, a height-balanced binary tree is defined as

a binary tree in which the depth of the two subtrees of *every* node never differ by more than 1.

利用递归获取高度，递归判断每个子树；



Beats:94.19%



111. Minimum Depth of Binary Tree

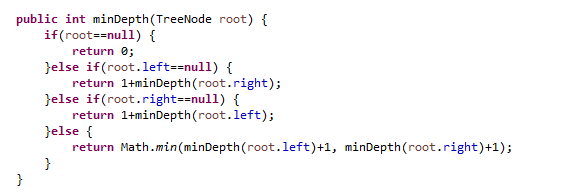
Given a binary tree, find its minimum depth.

The minimum depth is the number of nodes along the shortest path from the root node down to the nearest leaf node.

递归：beats 99%



Beats:100%

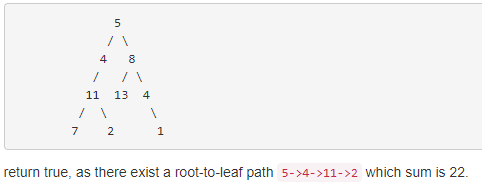


112. Path Sum

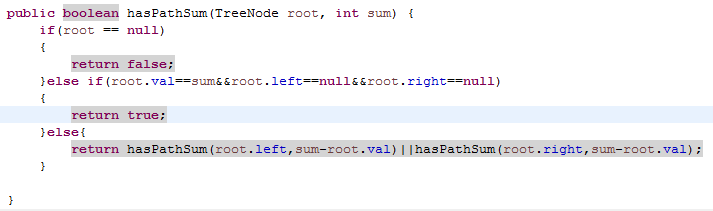
Given a binary tree and a sum, determine if the tree has a root-to-leaf path such that adding up all the values along the path equals the given sum.

For example:

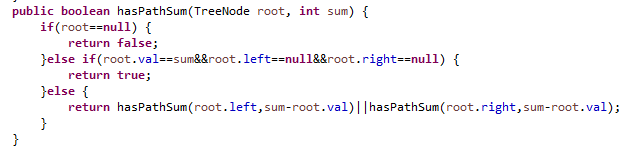
Given the below binary tree and sum = 22



递归：beats 98%



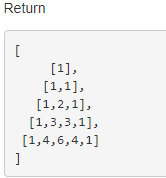
Beats:100%



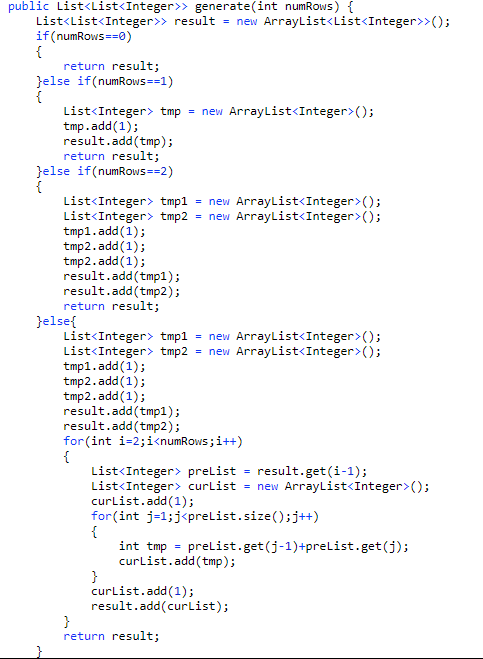
118. Pascal's Triangle

Given *numRows*, generate the first *numRows* of Pascal's triangle.

For example, given *numRows* = 5,



先初始化前两行，用前一行计算本行的方式计算后面的。beats 99.62%



Beats:100%



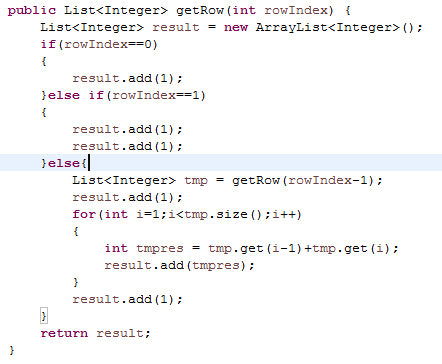
119. Pascal's Triangle II

Given an index *k*, return the *k*th row of the Pascal's triangle.

For example, given *k* = 3,

Return [1,3,3,1].

由第1行和2第二行递归计算出前一行的：beats 50%



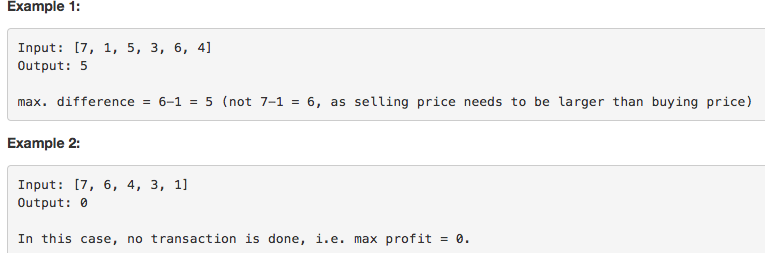
Beats:89.06%



121. Best Time to Buy and Sell Stock

Say you have an array for which the *i*th element is the price of a given stock on day *i*.

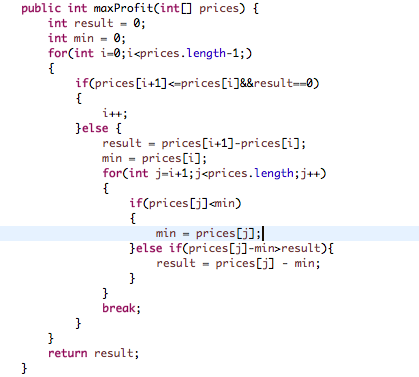
If you were only permitted to complete at most one transaction (ie, buy one and sell one share of the stock), design an algorithm to find the maximum profit.



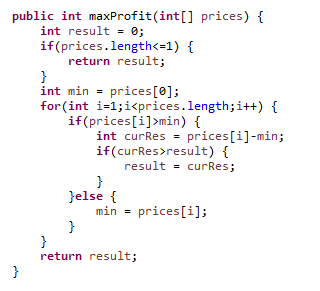
beats:85%

先找到曲线的上升点，记录这个点的值为min，并更新result为这点后的那个数减掉这个点的值，然后从这个点开始依次循环到结束，遍历中的每个值都与min相减，值比result大就替换result，并且途中遇到比这个点更低的就替换min；

O（n）



Beats:88.4%



122. Best Time to Buy and Sell Stock II

Say you have an array for which the *i*th element is the price of a given stock on day *i*.

Design an algorithm to find the maximum profit. You may complete as many transactions as you like (ie, buy one and sell one share of the stock multiple times). However, you may not engage in multiple transactions at the same time (ie, you must sell the stock before you buy again).

beats 60.43%

凡是上升的区间差值都加到result上去；



Beats:95.11%



125. Valid Palindrome

Given a string, determine if it is a palindrome, considering only alphanumeric characters and ignoring cases.

For example,

"A man, a plan, a canal: Panama" is a palindrome.

"race a car" is *not* a palindrome.

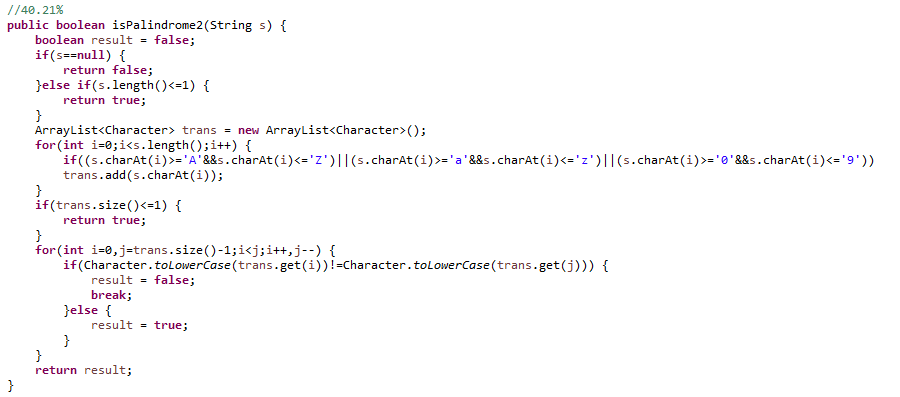
**Note:**

Have you consider that the string might be empty? This is a good question to ask during an interview.

For the purpose of this problem, we define empty string as valid palindrome.

beats 49.71%





Beats:83.08%



136. Single Number

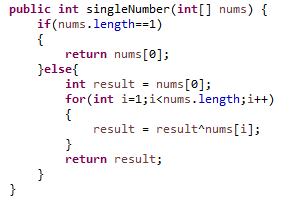
Given an array of integers, every element appears *twice* except for one. Find that single one.

**Note:**

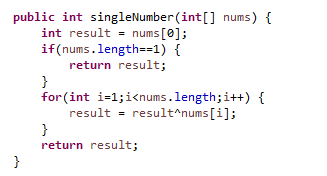
Your algorithm should have a linear runtime complexity. Could you implement it without using extra memory?

利用异或运算（相同为0不同为1（位计算）），beats 100%

异或^ ： 快速比较两个值，比判断等于高效； 不用额外空间交换两个值：a=a^b; b=a^b; a=a^b;



Beats:100%



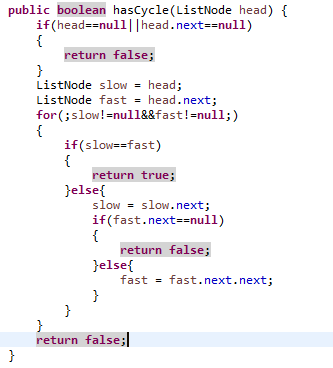
141. Linked List Cycle

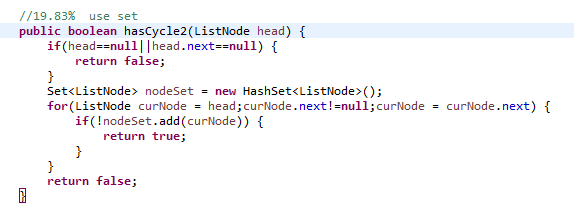
Given a linked list, determine if it has a cycle in it.

Follow up:

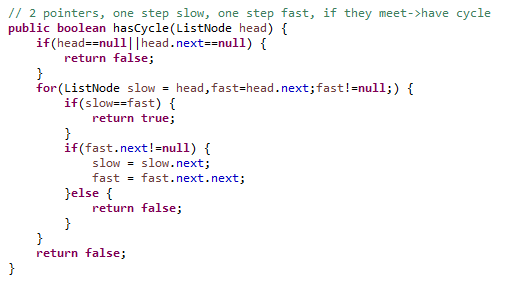
Can you solve it without using extra space?

设置两个指针，一个慢一个快，慢的每次往前走一个node,快的走两个node,如果两个相等的时候就说明有环： beats 97.82%

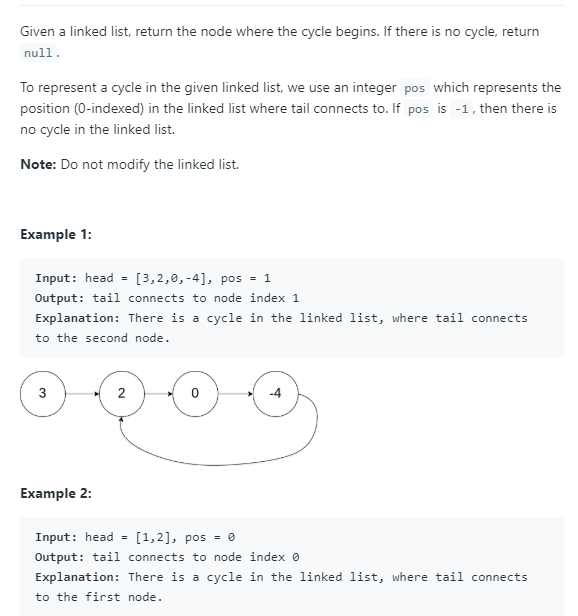




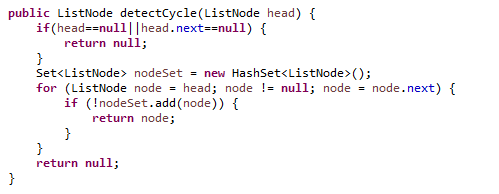
Beats:100%:



**142. Linked List Cycle II**



Use Set:beats 21.16%

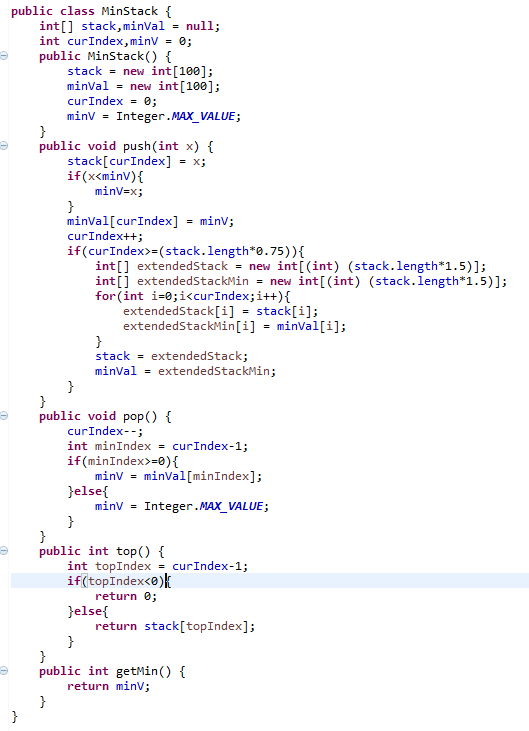


155. Min Stack

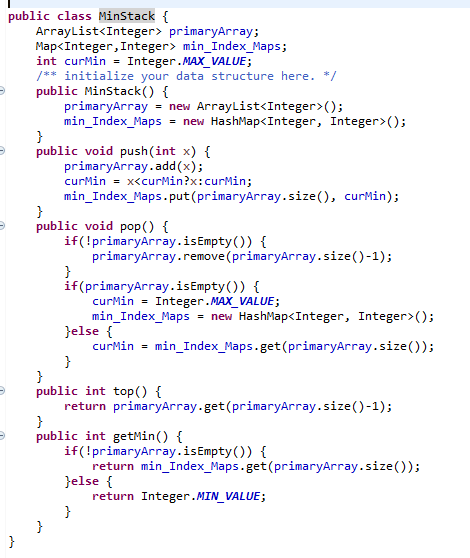
Design a stack that supports push, pop, top, and retrieving the minimum element in constant time.

* push(x) -- Push element x onto stack.
* pop() -- Removes the element on top of the stack.
* top() -- Get the top element.
* getMin() -- Retrieve the minimum element in the stack.

这种用数组实现的方式，用两个数组分别记录stack的值和stack每一个index处对应的minValue, beats了66.38%：

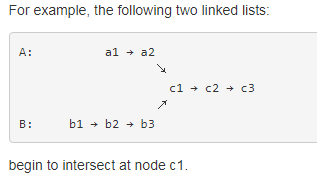


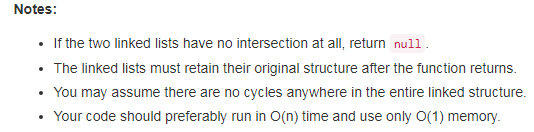
Beats:89.60%



160. Intersection of Two Linked Lists

Write a program to find the node at which the intersection of two singly linked lists begins.







Beats:98.08%



167. Two Sum II - Input array is sorted

Given an array of integers that is already ***sorted in ascending order***, find two numbers such that they add up to a specific target number.

The function twoSum should return indices of the two numbers such that they add up to the target, where index1 must be less than index2. Please note that your returned answers (both index1 and index2) are not zero-based.

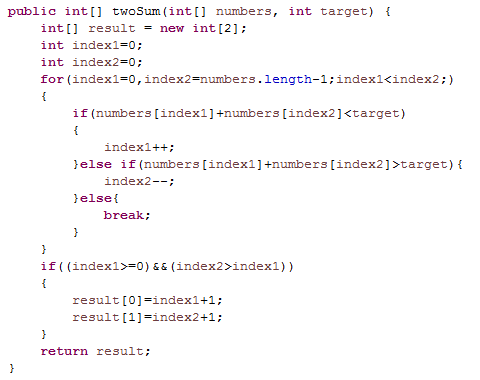
You may assume that each input would have *exactly* one solution and you may not use the *same* element twice.

**Input:** numbers={2, 7, 11, 15}, target=9

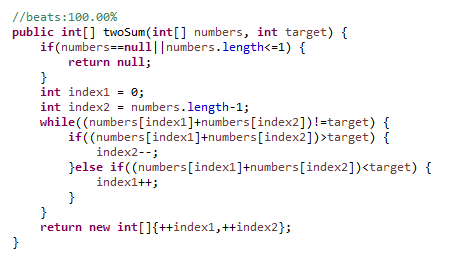
**Output:** index1=1, index2=2

beats 99.26%

用两个指针从两头往中间遍历，加起来的和比target大就把尾部指针前移一个，如果加起来的和比target小就把头部指针往后移一个；

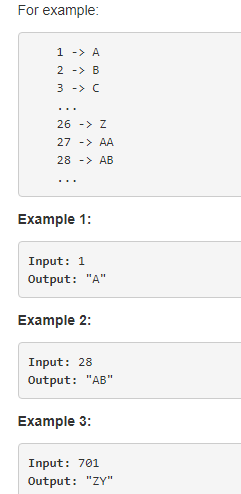


Beats:100%



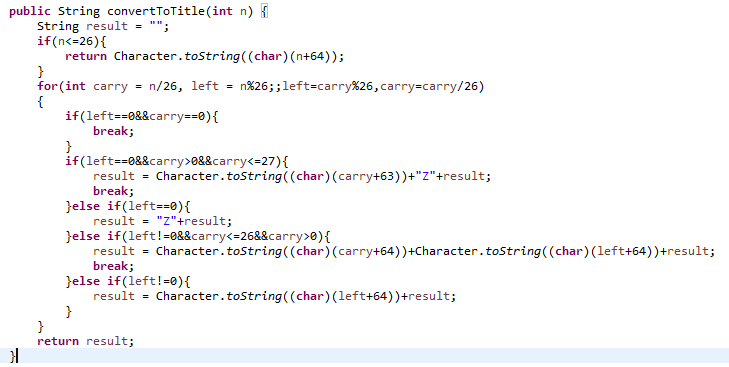
168. Excel Sheet Column Title

Given a positive integer, return its corresponding column title as appear in an Excel sheet

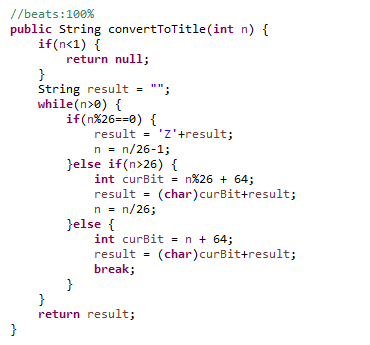


beats 100%

26进制，逢26进一，注意临界值（刚好整除26时和不能整除时两种情况）



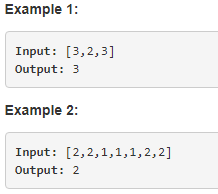
Beats:100%



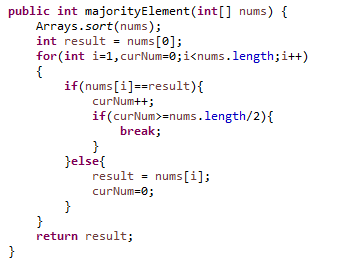
169. Majority Element

Given an array of size *n*, find the majority element. The majority element is the element that appears **more than** ⌊ n/2 ⌋ times.

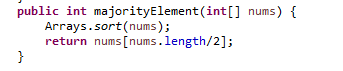
You may assume that the array is non-empty and the majority element always exist in the array.



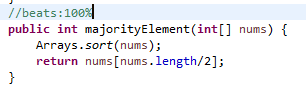
beats:37.59% 先排序，再逐个依次计数：



beats:79.93%

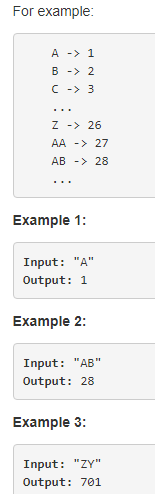


Beats:100%

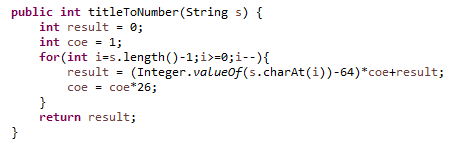


171. Excel Sheet Column Number

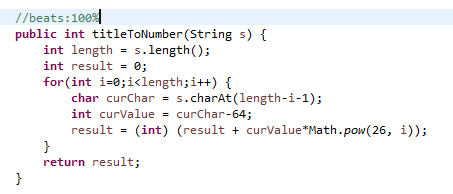
Given a column title as appear in an Excel sheet, return its corresponding column number



beats:61.52%

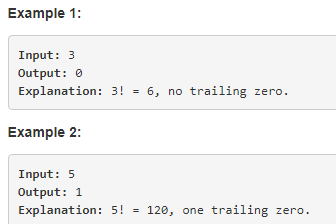


Beats:100%

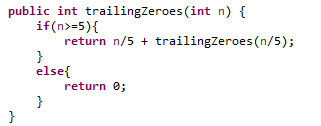


172. Factorial Trailing Zeroes

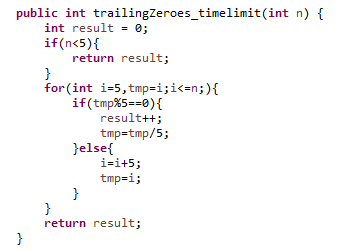
Given an integer *n*, return the number of trailing zeroes in *n*!.



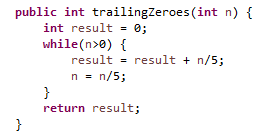
beats:77.85%(other)



mine:



Beats:100%



175. Combine Two Tables



Beats 100%

Mysql:

select FirstName,LastName,City, State from Person left join Address on Person.PersonId=Address.PersonId

left join: (以左边为主，左边的表全列出来，右边没有匹配的就显示为null)

select a1,a2,b1,b2 from A left join B on A.a1=B.b1;

right join:(以右边的为主，右边的全列出来，左边的没有匹配的就显示为null)

select \* from A right join B on A.id=B.id;

inner join:(不以谁为主，只列出符合条件的)

select \* from A inner join B on A.id=B.id

select \* from A,B where A.id=B.id;

Cross join:

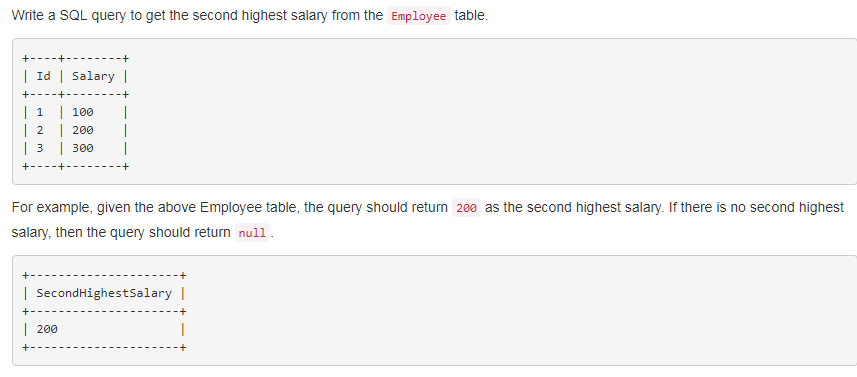
select \* from A cross join B;

select \* from A join B;

select \* from A,B;

(where, on, using)

176. Second Highest Salary



(select Salary as SecondHighestSalary from Employee order by Salary desc limit 1,1)

select IFNULL(Salary,null) as SecondHighestSalary from (select Salary from Employee order by Salary desc limit 1,1) as t

select IFNULL(Salary,null) as SecondHighestSalary from Employee order by Salary desc limit 1,1

**IFNULL(expr1,expr2)**

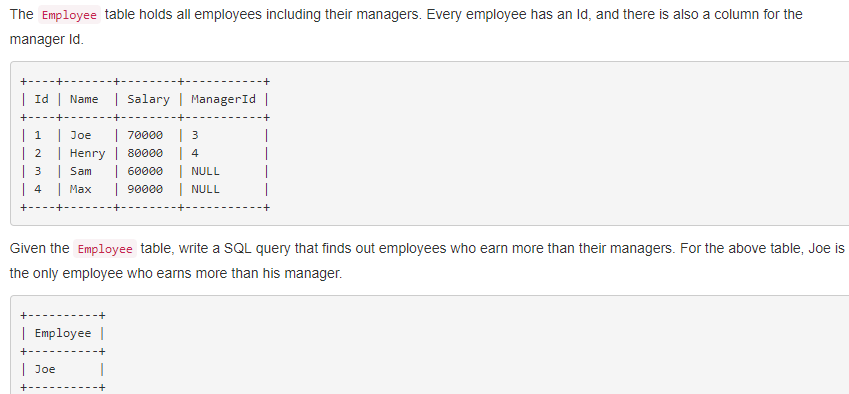
假如expr1 不为 NULL，则 IFNULL() 的返回值为 expr1; 否则其返回值为 expr2。IFNULL()的返回值是数字或是字符串，具体情况取决于其所使用的语境。

answer:

beats:100%

select IFNULL((select distinct Salary from Employee order by Salary desc limit 1,1),null) as SecondHighestSalary

181. Employees Earning More Than Their Managers



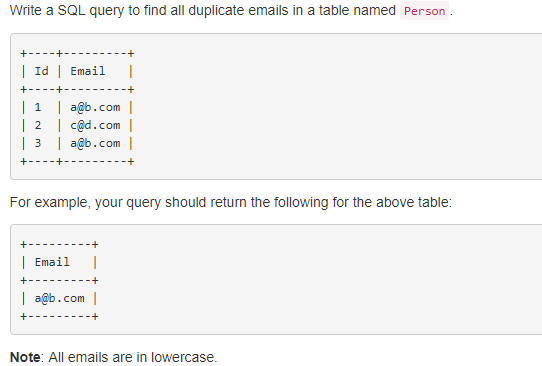
beats:100%

select t1.Name as Employee from Employee t1,(select Id,Salary,ManagerId from Employee) as t2 where t1.Salary>t2.Salary and t1.ManagerId=t2.Id

beats:71.87%

select a.Name as Employee from Employee a, Employee b where a.Salary>b.Salary and a.ManagerId=b.Id;

182. Duplicate Emails



beats: 100%

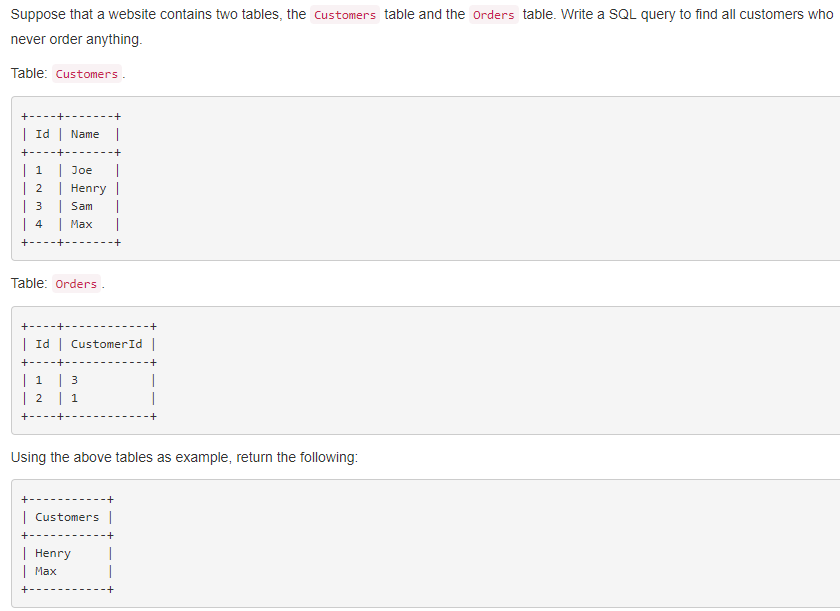
select distinct t1.Email from Person t1,Person t2 where t1.Email=t2.Email and t1.Id<>t2.Id

select distinct t1.Email from Person t1,Person t2 where t1.Email=t2.Email and t1.Id<>t2.Id

Or

select distinct(a.Email) from Person a join Person b where a.Id<>b.Id and a.Email=b.Email;

183. Customers Who Never Order



（查相等的时候这么查：

select Customers.Name as Customers from Customers,Orders where Customers.Id=Orders.CustomerId）

beats 40%:

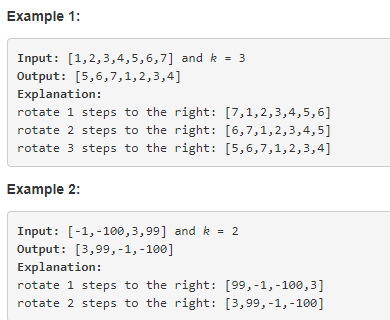
select Customers.Name as Customers from Customers where Customers.Id not in (select CustomerId from Orders)

beats:97%:

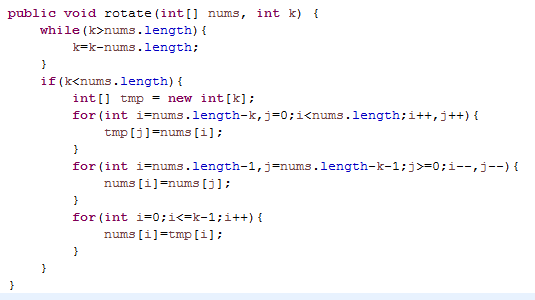
select Customers.Name as Customers from Customers left join Orders on Customers.Id=Orders.CustomerId where Orders.CustomerId is null;

189. Rotate Array

Given an array, rotate the array to the right by *k* steps, where *k* is non-negative.

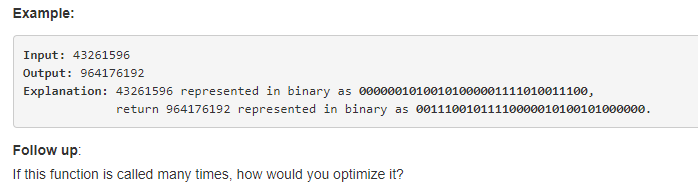


beats:98.09%

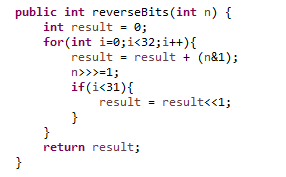


190. Reverse Bits

Reverse bits of a given 32 bits unsigned integer.

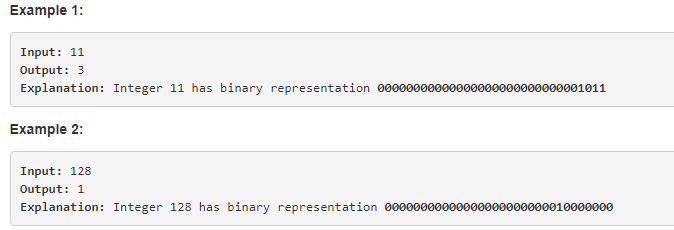


referred to disscussion: beats 100%

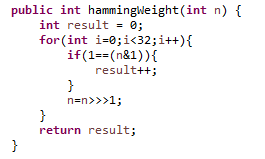


191. Number of 1 Bits

Write a function that takes an unsigned integer and returns the number of '1' bits it has (also known as the [Hamming weight](http://en.wikipedia.org/wiki/Hamming_weight)).



beats:63%

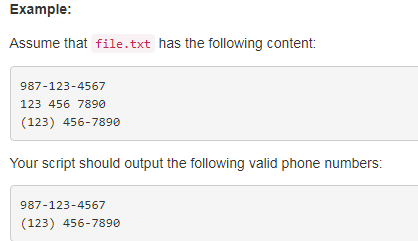


193. Valid Phone Numbers

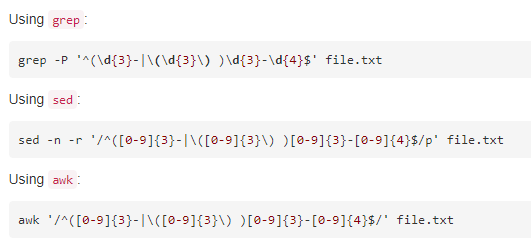
Given a text file file.txt that contains list of phone numbers (one per line), write a one liner bash script to print all valid phone numbers.

You may assume that a valid phone number must appear in one of the following two formats: (xxx) xxx-xxxx or xxx-xxx-xxxx. (x means a digit)

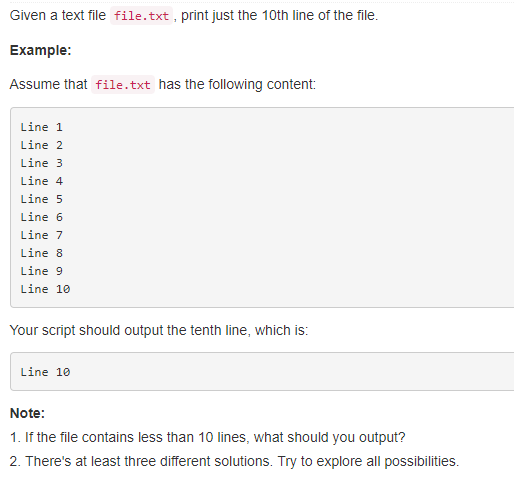
You may also assume each line in the text file must not contain leading or trailing white spaces.



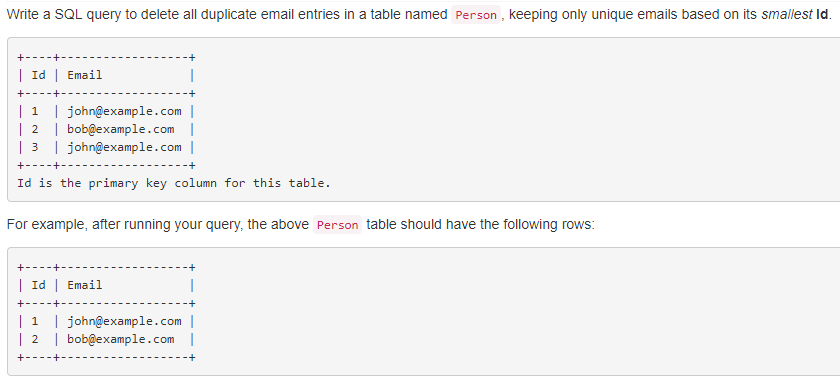
bash(refered)



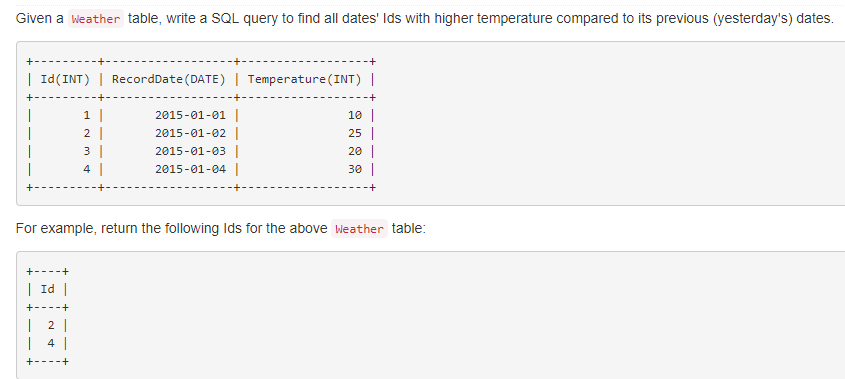
195. Tenth Line



196. Delete Duplicate Emails



197. Rising Temperature



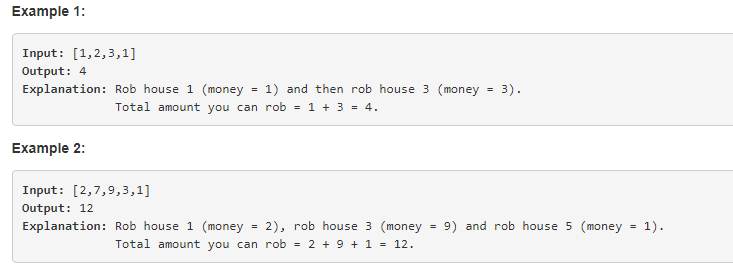
Beats:100%

select t0.Id from Weather t0 join (select Id,RecordDate,Temperature from Weather) as t on t0.Id<>t.Id and t0.Temperature>t.Temperature and t0.RecordDate=date\_add(t.RecordDate,interval 1 day);

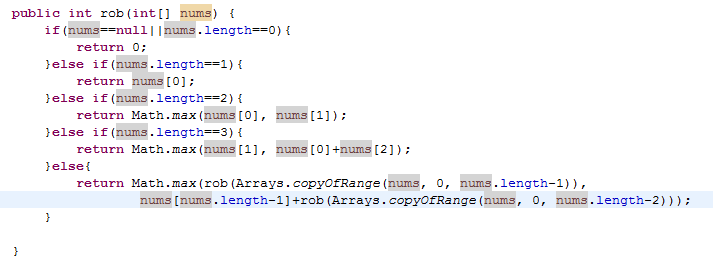
198. House Robber

You are a professional robber planning to rob houses along a street. Each house has a certain amount of money stashed, the only constraint stopping you from robbing each of them is that adjacent houses have security system connected and **it will automatically contact the police if two adjacent houses were broken into on the same night**.

Given a list of non-negative integers representing the amount of money of each house, determine the maximum amount of money you can rob tonight **without alerting the police**.



time limit Exceeded:

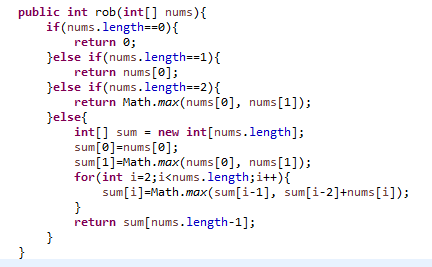


加个缓存：

beats：1.9%



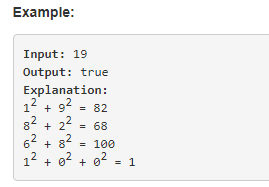
beats:1.28%



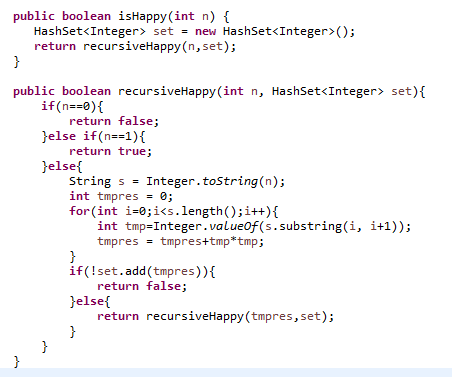
202. Happy Number

Write an algorithm to determine if a number is "happy".

A happy number is a number defined by the following process: Starting with any positive integer, replace the number by the sum of the squares of its digits, and repeat the process until the number equals 1 (where it will stay), or it loops endlessly in a cycle which does not include 1. Those numbers for which this process ends in 1 are happy numbers.

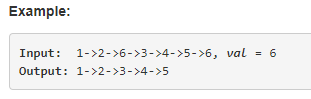


用递归法，beats： 8.6%



203. Remove Linked List Elements

Remove all elements from a linked list of integers that have value ***val***.



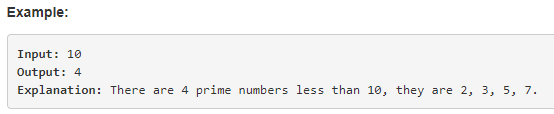
两个指针，一前一后，遍历list（注意处理临界值）

beats:65.44%



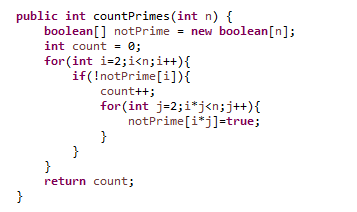
204. Count Primes

Count the number of prime numbers less than a non-negative number, ***n***.



合数都可以分解成若干质数，所以，只要2-sqrt(i)间的质数不能整除i即可

beats:67%：(referenced)

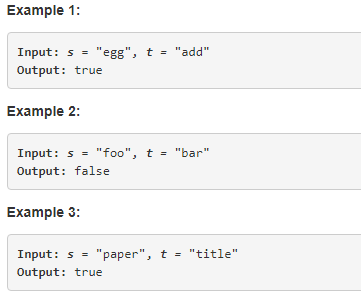


205. Isomorphic Strings

Given two strings ***s*** and ***t***, determine if they are isomorphic.

Two strings are isomorphic if the characters in ***s*** can be replaced to get ***t***.

All occurrences of a character must be replaced with another character while preserving the order of characters. No two characters may map to the same character but a character may map to itself.



beats:5.65%

依次给每个字母编号（从1开始），用map存储字母与编号之间的映射，前面出现过就用第一次出现的那个编号，否则用他的序号（第一个就是1，第二个就是2...），两个编号一样就是同构的。

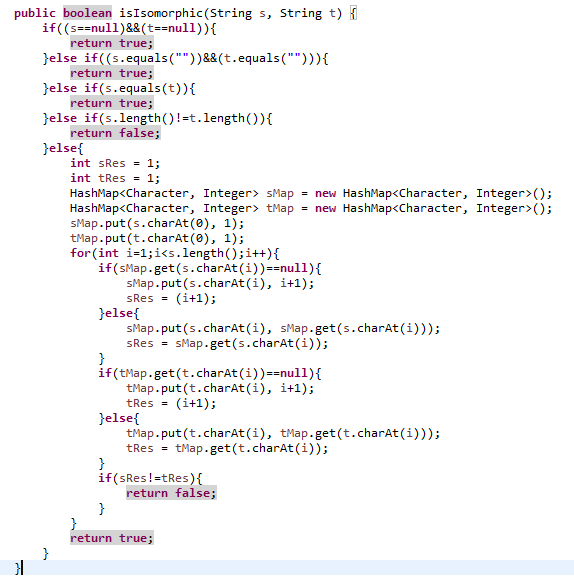
比如：

paper

12145

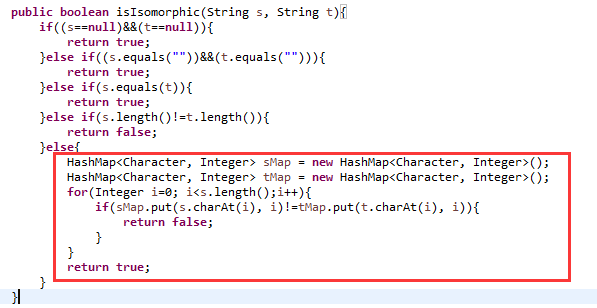
title

12145



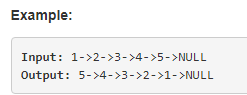
beats:24.47%(referenced)

充分利用map的特性：



206. Reverse Linked List

Reverse a singly linked list



beats:100%

三个指针：head, i, j 一起移动

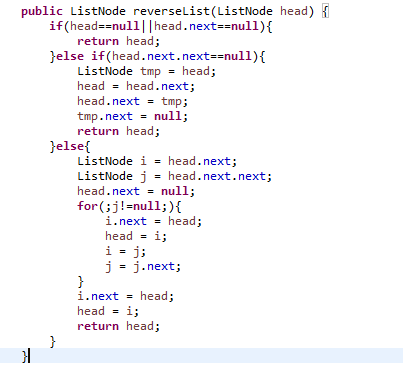
1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> null

head i j

先把head的next设为null

把第一个指针转向： i.next = head;

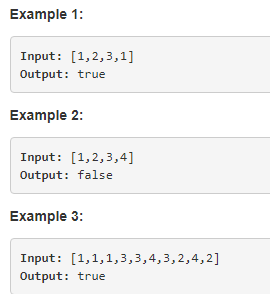
head i j全都往前移动一步



217. Contains Duplicate

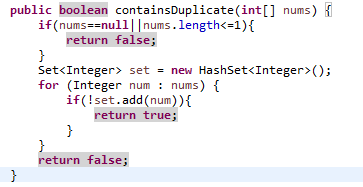
Given an array of integers, find if the array contains any duplicates.

Your function should return true if any value appears at least twice in the array, and it should return false if every element is distinct.



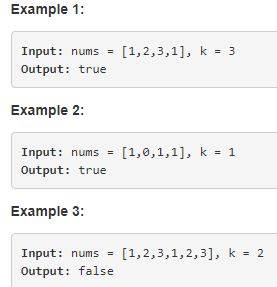
beats:68.5%

利用set：



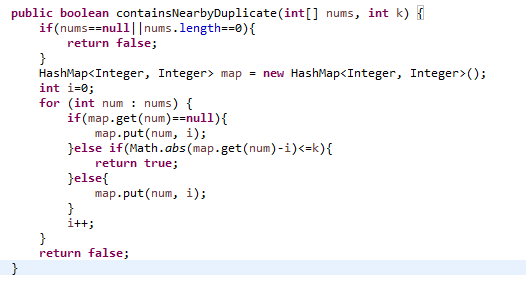
219. Contains Duplicate II

Given an array of integers and an integer *k*, find out whether there are two distinct indices *i* and *j* in the array such that **nums[i] = nums[j]** and the **absolute** difference between *i* and *j* is at most *k*.



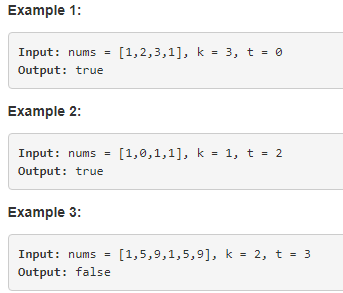
beats:9.34%

利用map，key是元素，value是对应的index值，每次添加时判断map中是否已经存在了，存在的话计算一下这个元素的index与当前i的距离，小于等于k就直接返回否则就更新这个元素的index；



220. Contains Duplicate III

Given an array of integers, find out whether there are two distinct indices *i* and *j* in the array such that the **absolute** difference between **nums[i]** and **nums[j]** is at most *t* and the **absolute** difference between *i* and *j* is at most *k*.

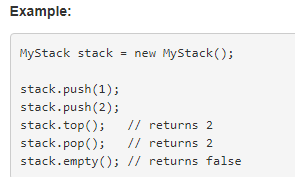


？？？

225. Implement Stack using Queues

Implement the following operations of a stack using queues.

* push(x) -- Push element x onto stack.
* pop() -- Removes the element on top of the stack.
* top() -- Get the top element.
* empty() -- Return whether the stack is empty.

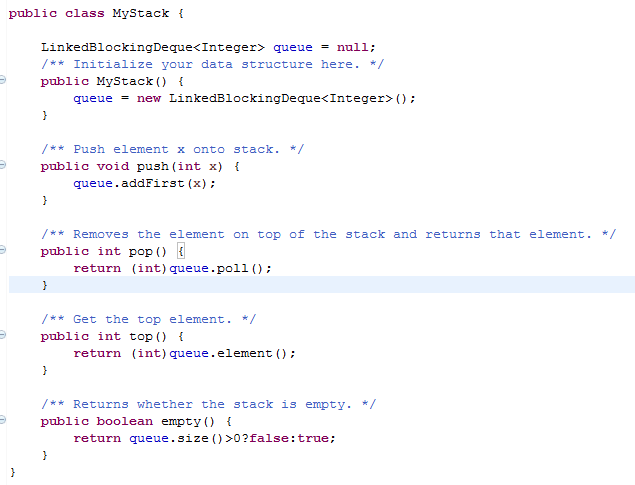


**Notes:**

* You must use *only* standard operations of a queue -- which means only push to back, peek/pop from front, size, and is empty operations are valid.
* Depending on your language, queue may not be supported natively. You may simulate a queue by using a list or deque (double-ended queue), as long as you use only standard operations of a queue.
* You may assume that all operations are valid (for example, no pop or top operations will be called on an empty stack).

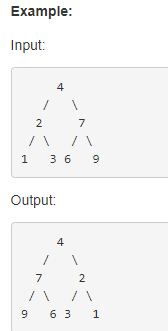
beats:99.8%

利用concurrent包里的LinkedBlockingDeque：



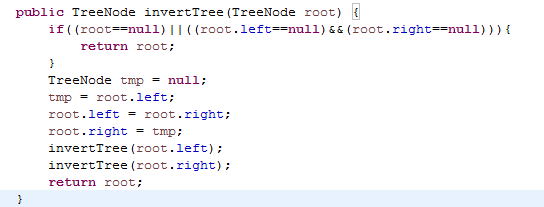
226. Invert Binary Tree

Invert a binary tree.



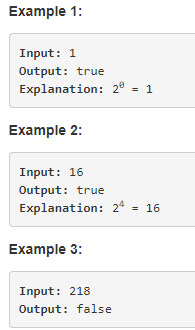
beats:100%

用递归：

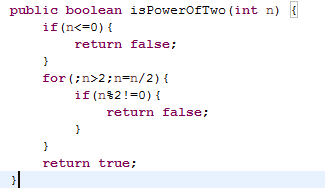


231. Power of Two

Given an integer, write a function to determine if it is a power of two.



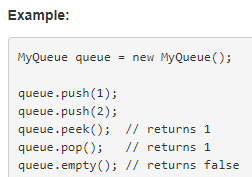
beats:100%



232. Implement Queue using Stacks

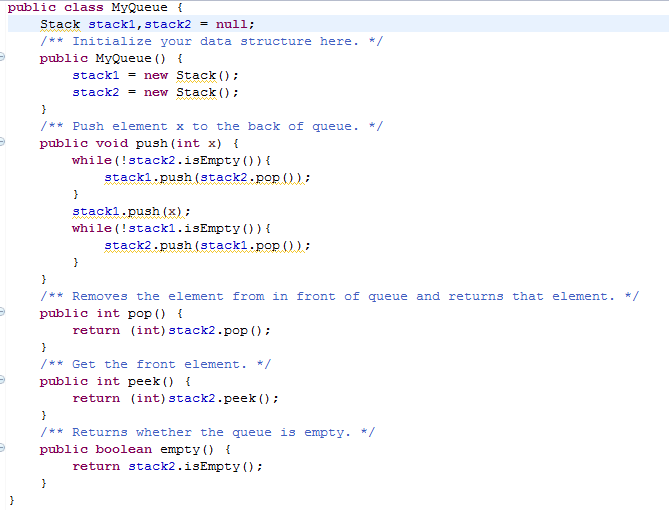
Implement the following operations of a queue using stacks.

* push(x) -- Push element x to the back of queue.
* pop() -- Removes the element from in front of queue.
* peek() -- Get the front element.
* empty() -- Return whether the queue is empty.



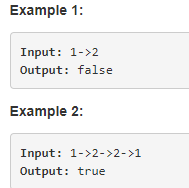
beats:98.7%

用两个stack来回倒数据，stack1 和 stack2,stack2存储实时数据，当push新数据时先将stack2的依次pop出来并push到stack1中，然后将新数据push到stack1，然后依次再将stack1中数据pop出来push到stack2中：



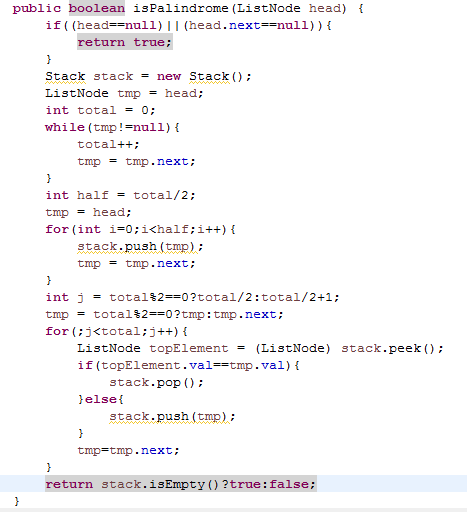
234. Palindrome Linked List

Given a singly linked list, determine if it is a palindrome.



beats:39.83

利用好对称性，用stack：

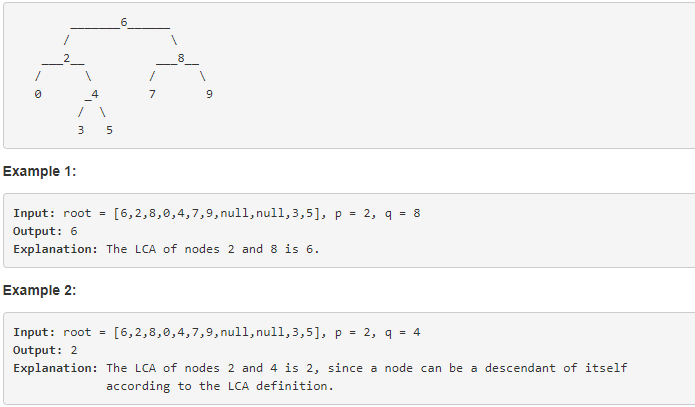


235. Lowest Common Ancestor of a Binary Search Tree

Given a binary search tree (BST), find the lowest common ancestor (LCA) of two given nodes in the BST.

According to the [definition of LCA on Wikipedia](https://en.wikipedia.org/wiki/Lowest_common_ancestor): “The lowest common ancestor is defined between two nodes p and q as the lowest node in T that has both p and q as descendants (where we allow **a node to be a descendant of itself**).”

Given binary search tree:  root = [6,2,8,0,4,7,9,null,null,3,5]

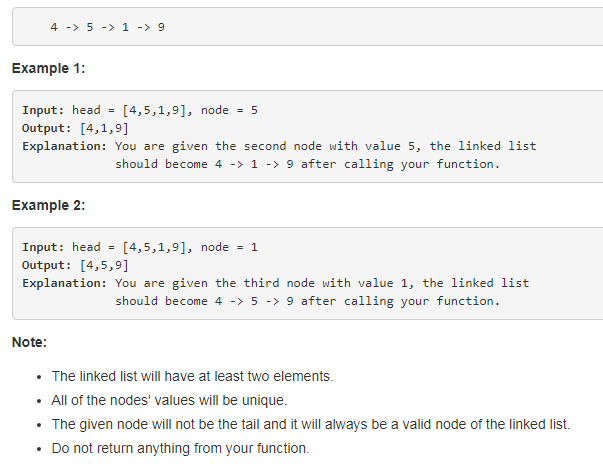


???

237. Delete Node in a Linked List

Write a function to delete a node (except the tail) in a singly linked list, given only access to that node.

Given linked list -- head = [4,5,1,9], which looks like following:

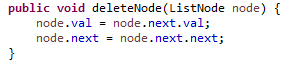


beats:100%

referenced:

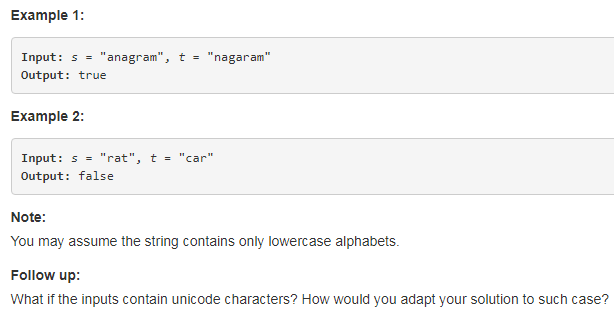
直接操作要被删掉的那个节点，把它后面一个节点的值赋给它，然后把它的next指针指向next的next：

时间复杂读和空间复杂度均为：O（1）



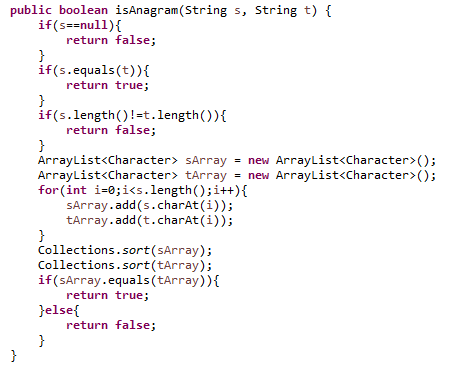
242. Valid Anagram

Given two strings *s* and *t*, write a function to determine if *t* is an anagram of *s*.

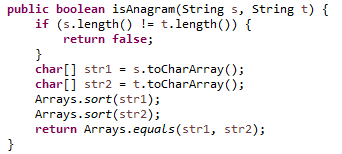


beats:13.4%

先排序，再比较：



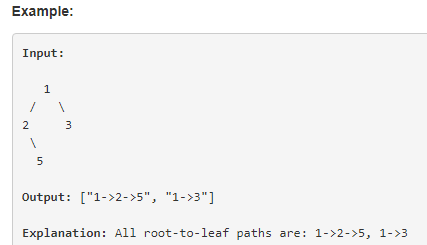
referenced:



257. Binary Tree Paths

Given a binary tree, return all root-to-leaf paths.

**Note:** A leaf is a node with no children.



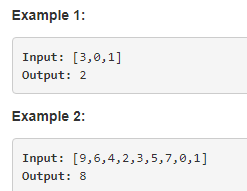
beats:99.41%

递归：维持一个存储结果的list，和一个存储遍历到当前位置为止的字符串curString，每遍历到一个叶子节点就把这个curString加到List中去，若不是叶子就把list，curString作为参数继续递归该节点的左子树和右子树：



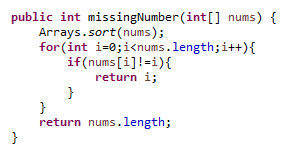
268. Missing Number

Given an array containing *n* distinct numbers taken from 0, 1, 2, ..., n, find the one that is missing from the array.



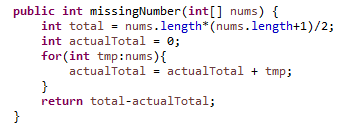
beats:20.23%

先排序，再依次遍历，不等于遍历序号i的即为missing的数：



referenced：beats:98.56%

计算1...n累加和，计算数组实际的和，这两个的差就是missing的数：

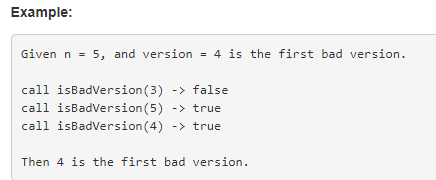


278. First Bad Version

You are a product manager and currently leading a team to develop a new product. Unfortunately, the latest version of your product fails the quality check. Since each version is developed based on the previous version, all the versions after a bad version are also bad.

Suppose you have n versions [1, 2, ..., n] and you want to find out the first bad one, which causes all the following ones to be bad.

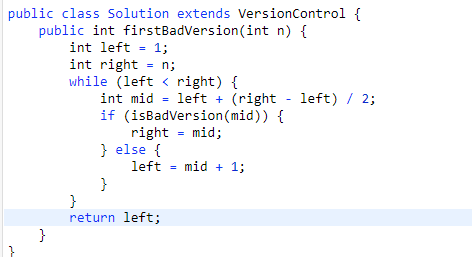
You are given an API bool isBadVersion(version) which will return whether version is bad. Implement a function to find the first bad version. You should minimize the number of calls to the API.



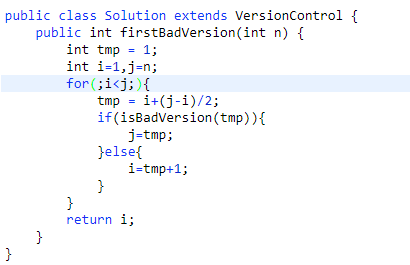
beats:99%

referenced:

二分法：

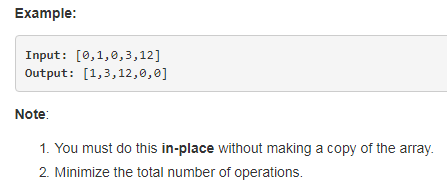


my:



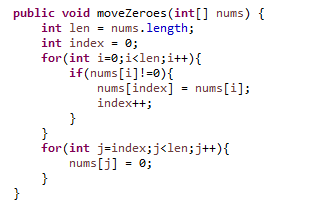
283. Move Zeroes

Given an array nums, write a function to move all 0's to the end of it while maintaining the relative order of the non-zero elements.



beats:100%

依次遍历，把非0的依次覆盖，并用一个index来记录非零的个数，最后把从index到最后的全部置零。



290. Word Pattern

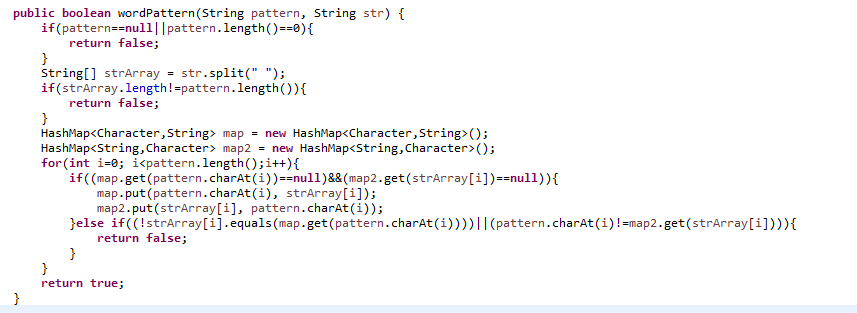
Given a pattern and a string str, find if str follows the same pattern.

Here **follow** means a full match, such that there is a bijection between a letter in pattern and a **non-empty** word in str.



beats:99.95%

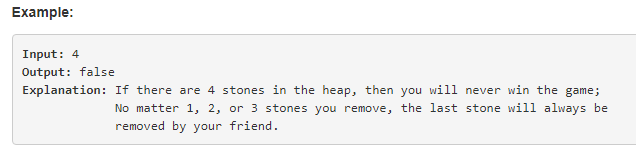
用两个map分别存储pattern到str的映射和str到pattern的映射，没有就加进去，有的话就比较下是否相同。



292. Nim Game

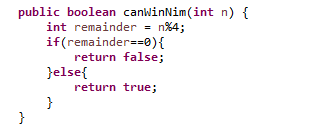
You are playing the following Nim Game with your friend: There is a heap of stones on the table, each time one of you take turns to remove 1 to 3 stones. The one who removes the last stone will be the winner. You will take the first turn to remove the stones.

Both of you are very clever and have optimal strategies for the game. Write a function to determine whether you can win the game given the number of stones in the heap.



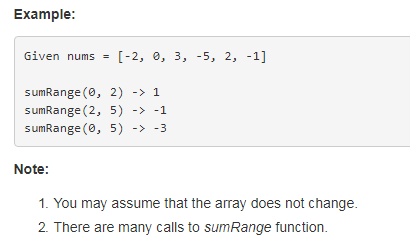
beats:100%

如果是4的整数倍，谁先选就会失去控制权，如果n%4有余数mod，谁先选谁就得到控制权（可以全选了，然后让对方先在那些整数倍4的里面先选）：



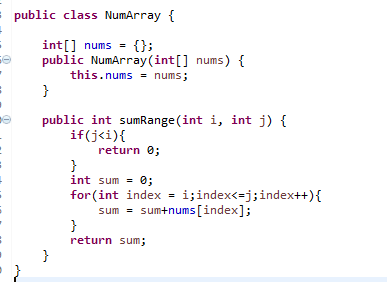
303. Range Sum Query - Immutable

Given an integer array *nums*, find the sum of the elements between indices *i* and *j* (*i* ≤ *j*), inclusive.



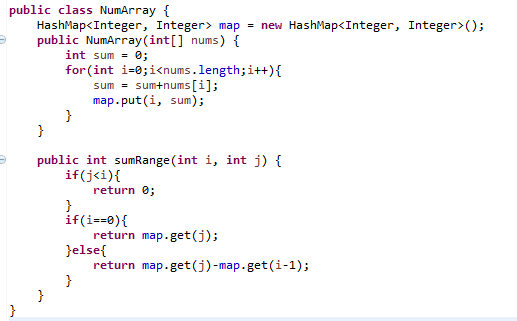
beats:5.91%

从i到j依次循环相加：



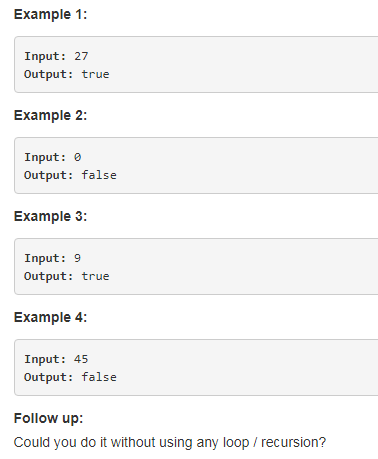
beats:92.4%

利用map，初始化时就存储index从0到当前i的和，计算时只需要从map取出第j个减掉第i-1个的值就可以了：



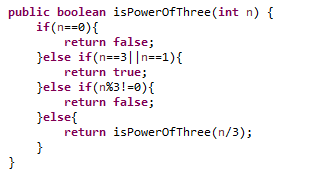
326. Power of Three

Given an integer, write a function to determine if it is a power of three.



beats:99.25%

递归



beats:95.58%

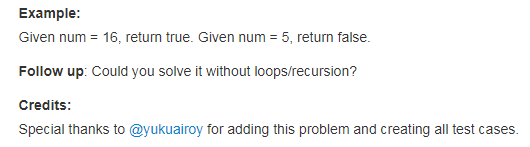
referenced:

转成3进制的数，然后判断是否是100...000的格式即可：

clipboard.png

342. Power of Four

Given an integer (signed 32 bits), write a function to check whether it is a power of 4.



beats:3.86%

同上面326题Power of Three:

clipboard.png

beats:100%

referenced:

利用二进制的位运算：

clipboard.png

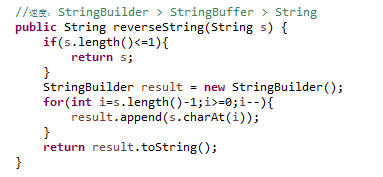
344. Reverse String

Write a function that takes a string as input and returns the string reversed.

clipboard.png

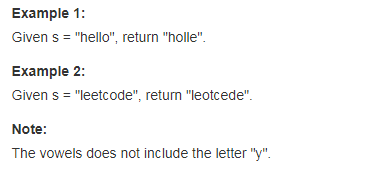
beats:97.17%

用StringBuilder倒着依次添加每个字符（速度比较： StringBuilder > StringBuffer > String, StringBuilder不是线程安全的，StringBuffer是线程安全的）



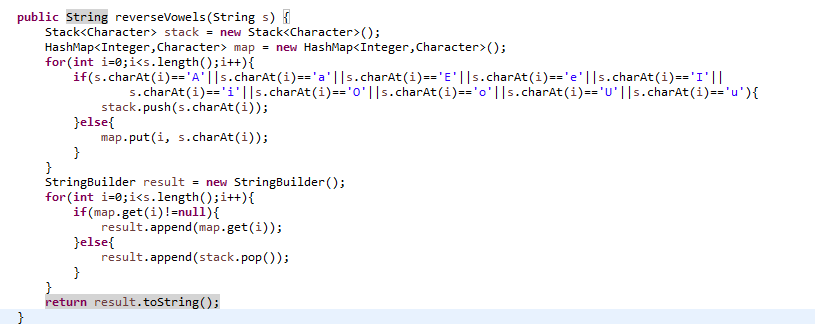
345. Reverse Vowels of a String

Write a function that takes a string as input and reverse only the vowels of a string.



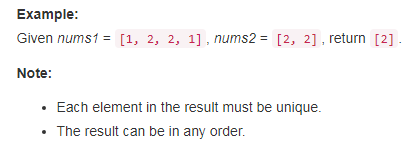
beats:6.08%

用一个map，一个stack分别按序存储非元音字符和元音字符，然后用stringBuilder一次拼起来：



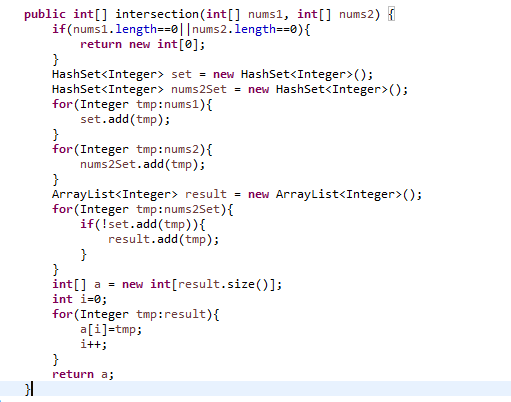
349. Intersection of Two Arrays

Given two arrays, write a function to compute their intersection.



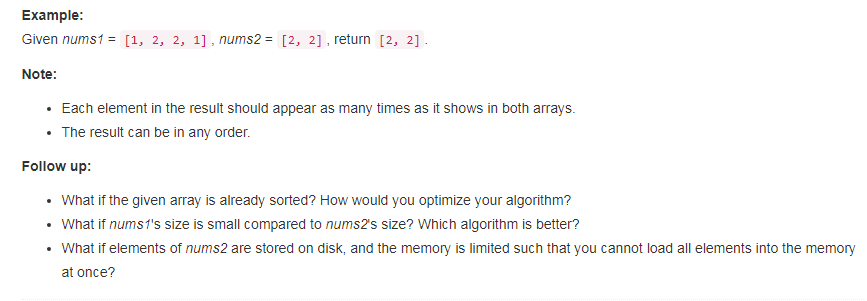
beats:98.67%

把两个数组分别放到两个set里，然后把第二个set的数据往第一个里加，如果加失败就把这个数放到数组里，最后返回这个数据就可以了：



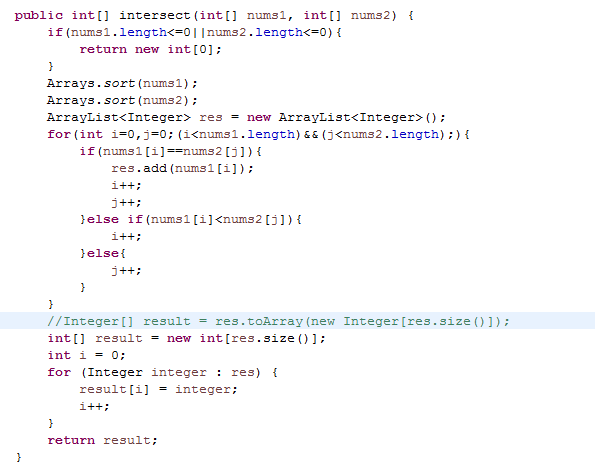
350. Intersection of Two Arrays II

Given two arrays, write a function to compute their intersection.



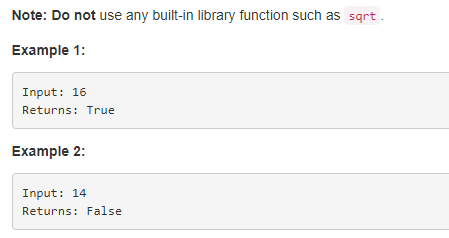
beats:99.8%

先将两个数组排序，然后两个指针i,j依次遍历比较对应的值：



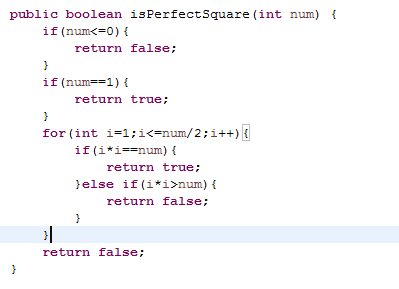
367. Valid Perfect Square

Given a positive integer *num*, write a function which returns True if *num* is a perfect square else False.



beats:7.28%

从1到n/2,依次计算i\*i是否等于n:



371. Sum of Two Integers

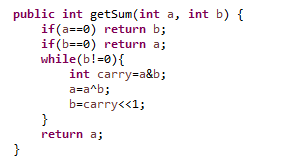
Calculate the sum of two integers *a* and *b*, but you are **not allowed** to use the operator + and -.

clipboard.png

beats:100%

referenced:

用按位与计算进位，用按位异或计算不带进位的加：



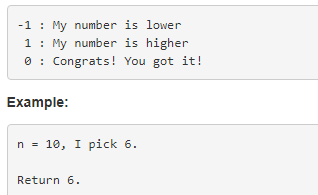
374. Guess Number Higher or Lower

We are playing the Guess Game. The game is as follows:

I pick a number from **1** to ***n***. You have to guess which number I picked.

Every time you guess wrong, I'll tell you whether the number is higher or lower.

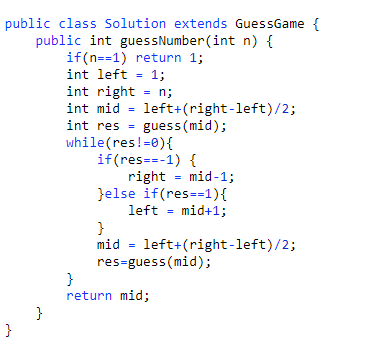
You call a pre-defined API guess(int num) which returns 3 possible results (-1, 1, or 0):



二分法：time limit Exceeded:

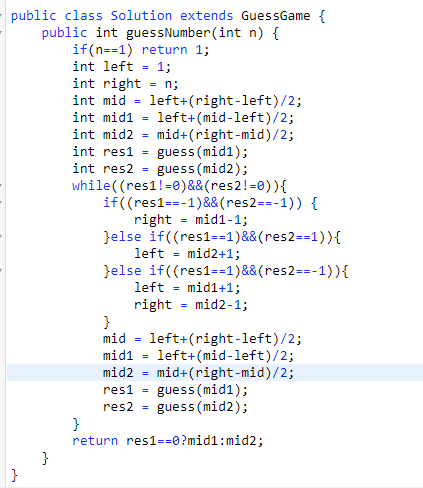
beats:100%

注意：取中间数要用mid=left+(right-left)/2, 不能直接用(left+right)/2



beats:100%

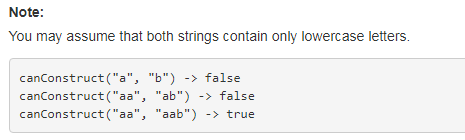
三分法：



383. Ransom Note

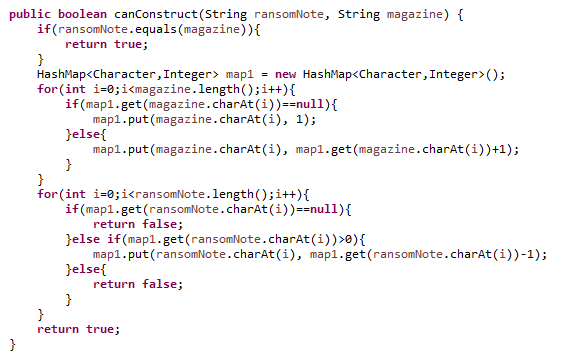
Given an arbitrary ransom note string and another string containing letters from all the magazines, write a function that will return true if the ransom note can be constructed from the magazines ; otherwise, it will return false.

Each letter in the magazine string can only be used once in your ransom note.



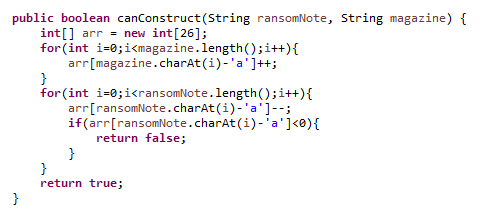
beats:31.52%

用map存储magazine中每个字母出现的次数，然后遍历ransom，出现一个就在map中对应的字符减掉一次：



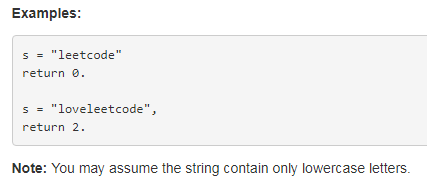
beats:97.07%

referenced:



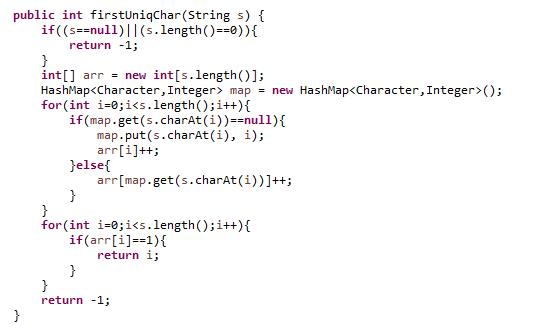
387. First Unique Character in a String

Given a string, find the first non-repeating character in it and return it's index. If it doesn't exist, return -1.



beats:43.46%

用一个map记录位置（如果map中有对应的字符就把arr对应的index加一），用一个数据记录个数，最后遍历数组，返回第一个是1的：

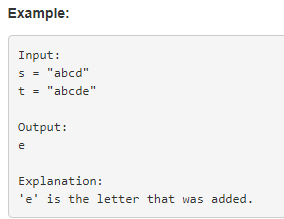


389. Find the Difference

Given two strings ***s*** and ***t*** which consist of only lowercase letters.

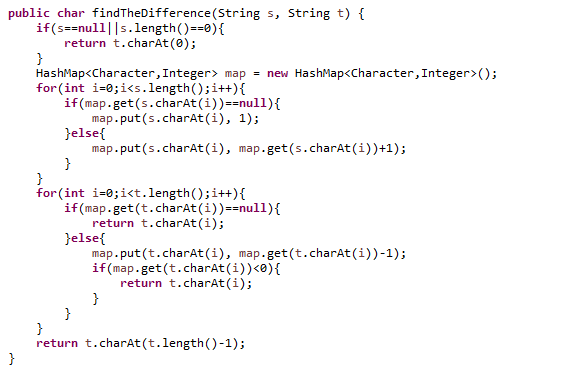
String ***t*** is generated by random shuffling string ***s*** and then add one more letter at a random position.

Find the letter that was added in ***t***.



beats:18.25%

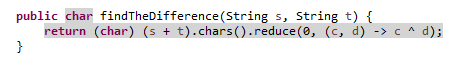
用一个map记录s中每个字符的个数，遍历t，map中没有就是要找的字符,map中有就把它的数量减掉1



beats:2.5%

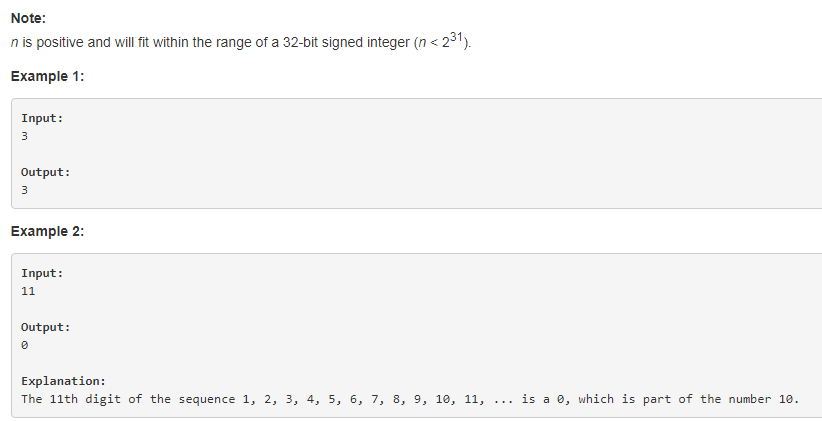
referenced:

转成intStream, 然后排个序，再异或一下



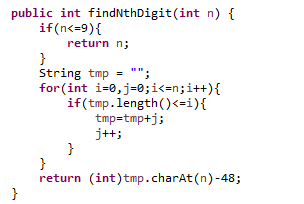
400. Nth Digit

Find the *n*th digit of the infinite integer sequence 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, ...

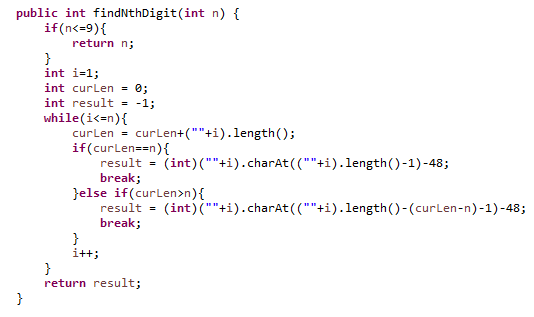


拼字符串，找第n个，转成int

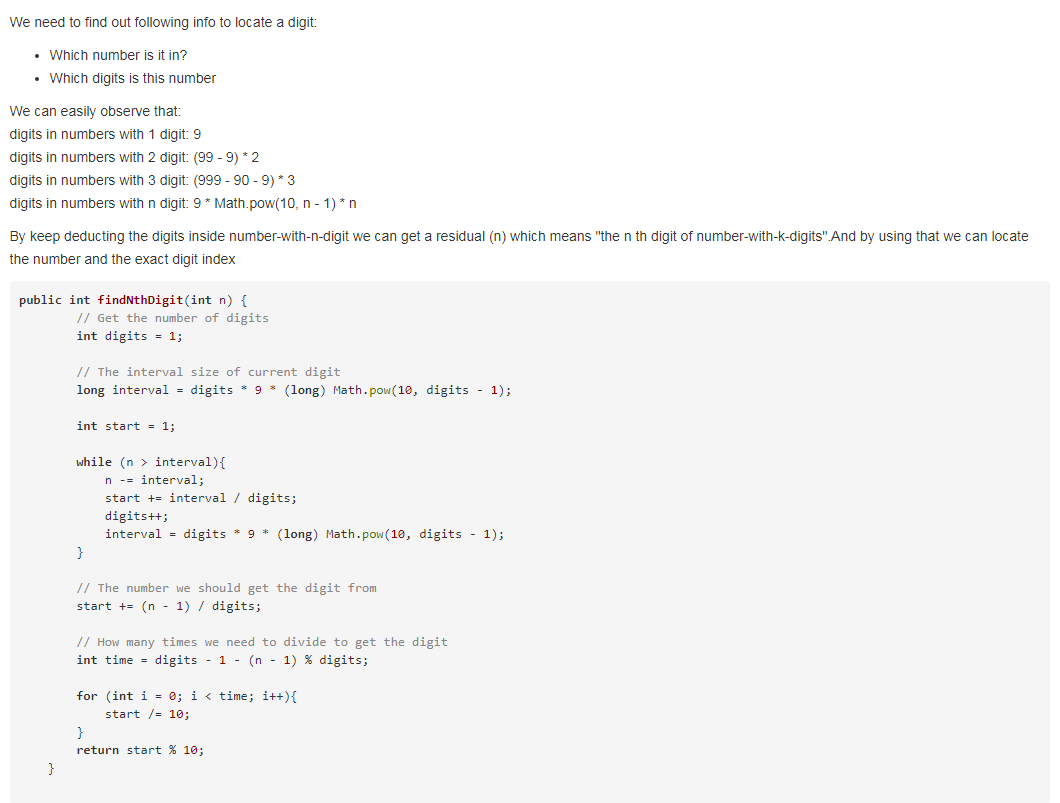
Time Limit Exceeded:



Time Limit Exceeded:



referenced：？？？？？



401. Binary Watch

A binary watch has 4 LEDs on the top which represent the **hours** (**0-11**), and the 6 LEDs on the bottom represent the **minutes** (**0-59**).

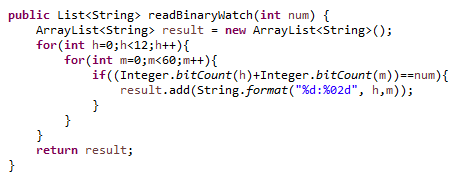
Each LED represents a zero or one, with the least significant bit on the right.



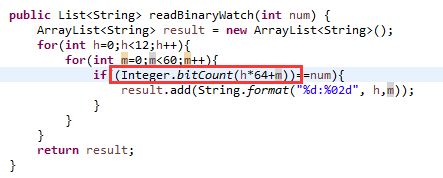
beats:38.5%

referenced:

只要把所有时间遍历一下，转成二进制后看看哪些的1加起来是num就可以了：

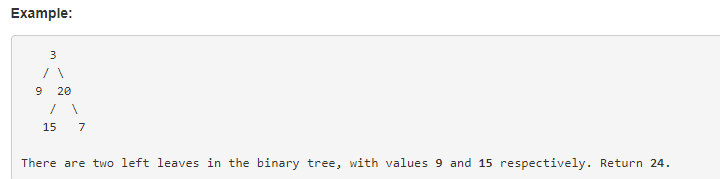


或：



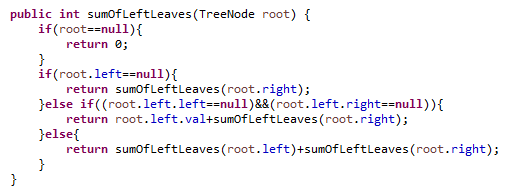
404. Sum of Left Leaves

Find the sum of all left leaves in a given binary tree.



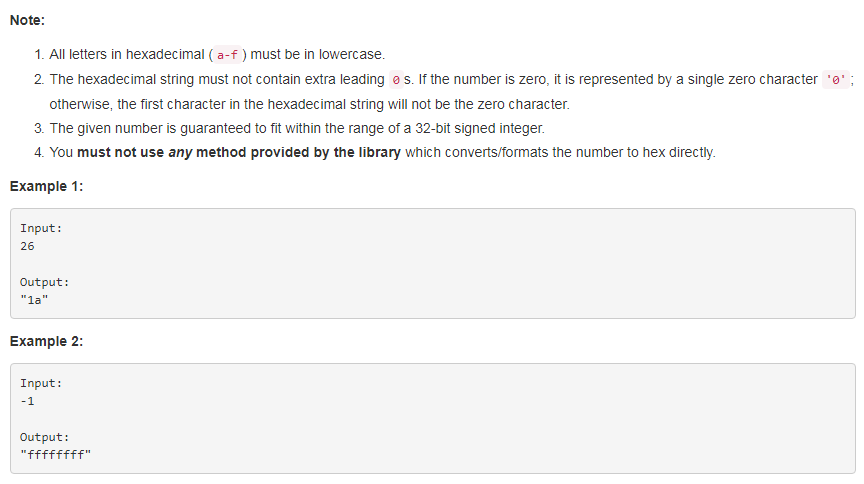
beats:100%

递归：



405. Convert a Number to Hexadecimal

Given an integer, write an algorithm to convert it to hexadecimal. For negative integer, [two’s complement](https://en.wikipedia.org/wiki/Two%27s_complement) method is used.



two's complement:

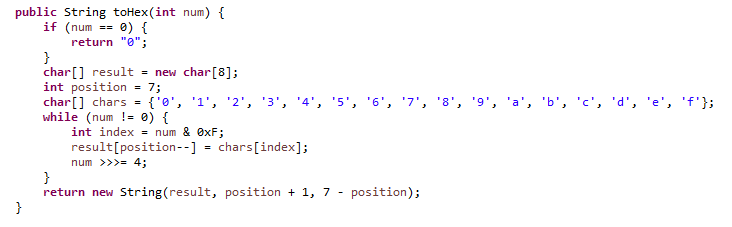
负数：求绝对值的二进制，然后每一位都反转，然后加1：

比如-1，

1是0001， 反转后得1110，加一是1111

所以-1的two's complement是1111

Referenced: 100%

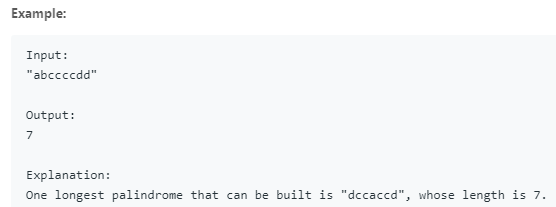


**409. Longest Palindrome**

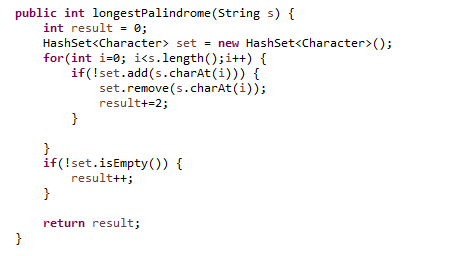
Given a string which consists of lowercase or uppercase letters, find the length of the longest palindromes that can be built with those letters.

This is case sensitive, for example "Aa" is not considered a palindrome here.

**Note:**  
Assume the length of given string will not exceed 1,010.



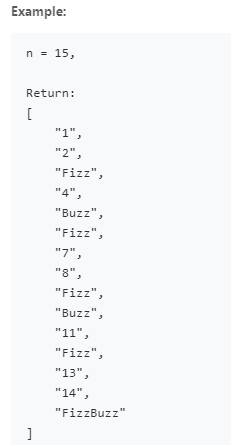
Beats: 69.83%



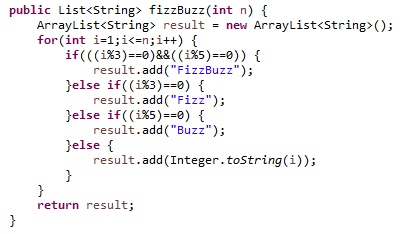
**412. Fizz Buzz**

Write a program that outputs the string representation of numbers from 1 to *n*.

But for multiples of three it should output “Fizz” instead of the number and for the multiples of five output “Buzz”. For numbers which are multiples of both three and five output “FizzBuzz”.

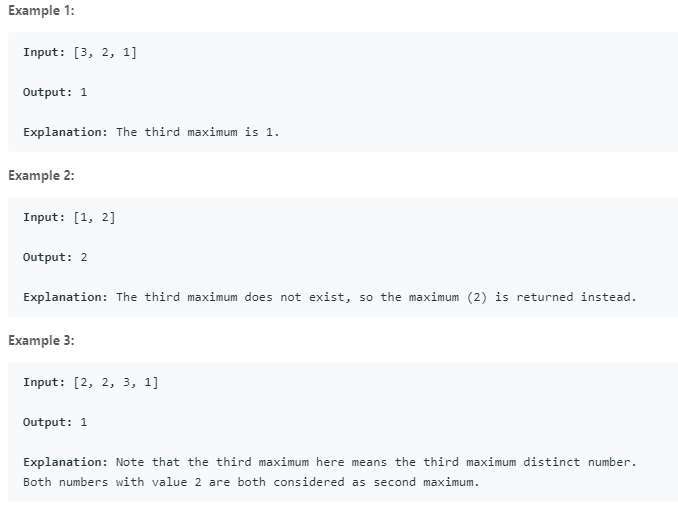


Beats: 100.00%



**414. Third Maximum Number**

Given a **non-empty** array of integers, return the **third** maximum number in this array. If it does not exist, return the maximum number. The time complexity must be in O(n).

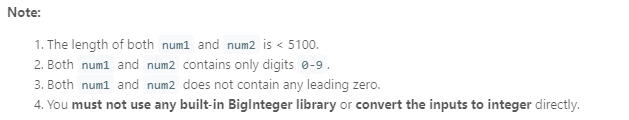


Beats:100%

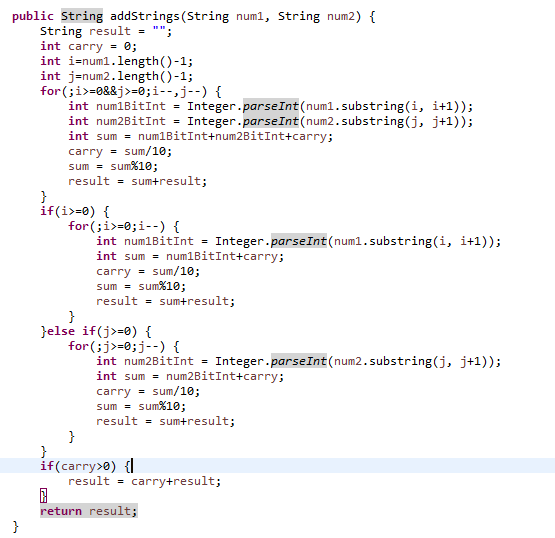


**415. Add Strings**

Given two non-negative integers num1 and num2 represented as string, return the sum of num1 and num2.



Beats:29.93%

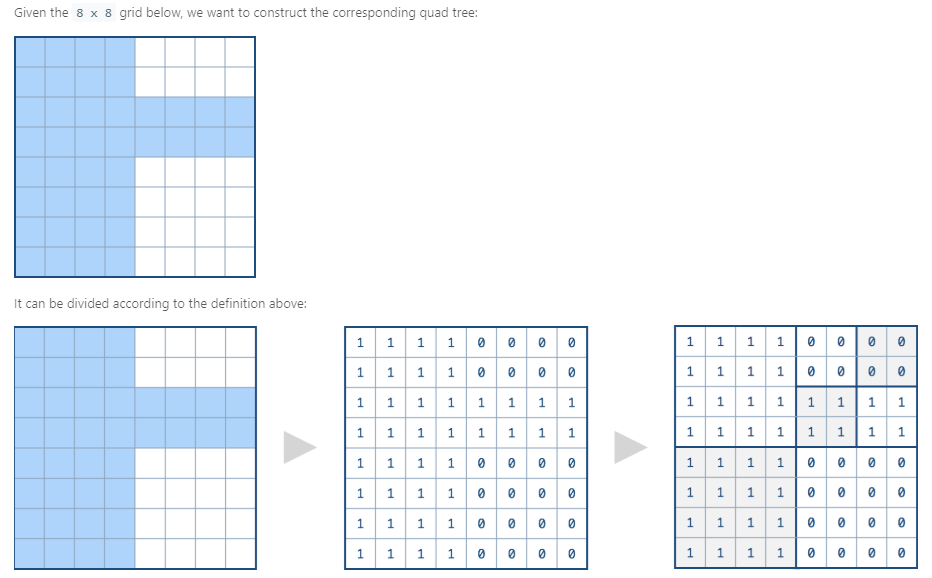


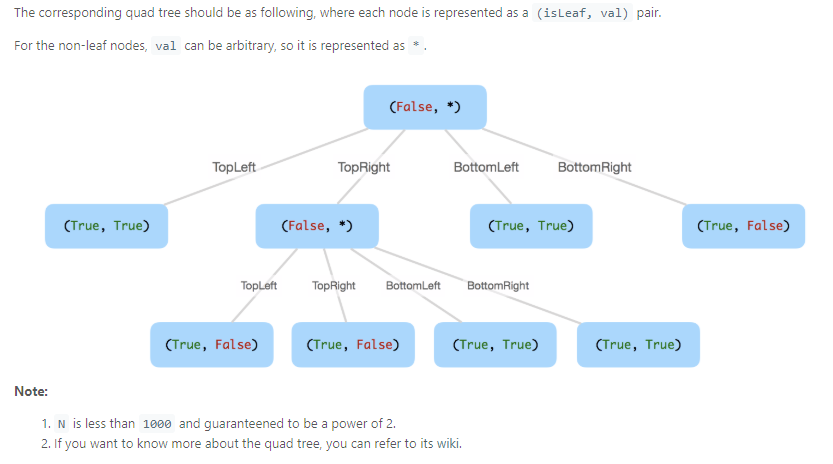
**427. Construct Quad Tree**

We want to use quad trees to store an N x N boolean grid. Each cell in the grid can only be true or false. The root node represents the whole 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.

Your task is to use a quad tree to represent a given grid. The following example may help you understand the problem better:





**434. Number of Segments in a String**

Count the number of segments in a string, where a segment is defined to be a contiguous sequence of non-space characters.

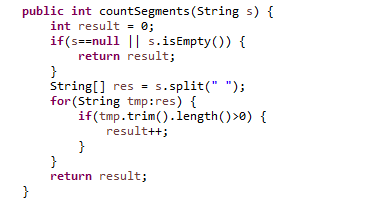
Please note that the string does not contain any **non-printable** characters.

**Example:**

**Input:** "Hello, my name is John"

**Output:** 5

Beats:100%



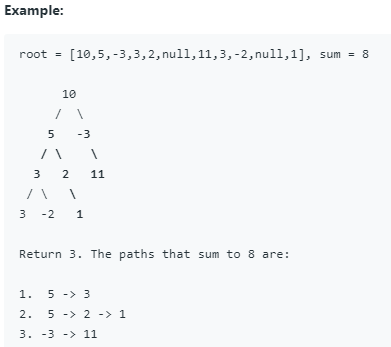
**437. Path Sum III**

You are given a binary tree in which each node contains an integer value.

Find the number of paths that sum to a given value.

The path does not need to start or end at the root or a leaf, but it must go downwards (traveling only from parent nodes to child nodes).

The tree has no more than 1,000 nodes and the values are in the range -1,000,000 to 1,000,000.



Beats:12.69%



**441. Arranging Coins**

You have a total of *n* coins that you want to form in a staircase shape, where every *k*-th row must have exactly *k* coins.

Given *n*, find the total number of **full** staircase rows that can be formed.

*n* is a non-negative integer and fits within the range of a 32-bit signed integer.

**Example 1:**

n = 5

The coins can form the following rows:

¤

¤ ¤

¤ ¤

Because the 3rd row is incomplete, we return 2.

**Example 2:**

n = 8

The coins can form the following rows:

¤

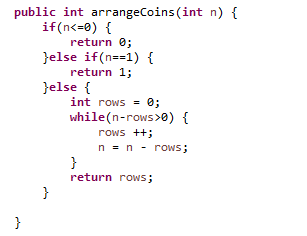
¤ ¤

¤ ¤ ¤

¤ ¤

Because the 4th row is incomplete, we return 3.

Beats:54.77%



**443. String Compression**

Given an array of characters, compress it [**in-place**](https://en.wikipedia.org/wiki/In-place_algorithm).

The length after compression must always be smaller than or equal to the original array.

Every element of the array should be a **character** (not int) of length 1.

After you are done **modifying the input array**[**in-place**](https://en.wikipedia.org/wiki/In-place_algorithm), return the new length of the array.

**Follow up:**  
Could you solve it using only O(1) extra space?

**Example 1:**

**Input:**

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

**Output:**

Return 6, and the first 6 characters of the input array should be: ["a","2","b","2","c","3"]

**Explanation:**

"aa" is replaced by "a2". "bb" is replaced by "b2". "ccc" is replaced by "c3".

**Example 2:**

**Input:**

["a"]

**Output:**

Return 1, and the first 1 characters of the input array should be: ["a"]

**Explanation:**

Nothing is replaced.

**Example 3:**

**Input:**

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

**Output:**

Return 4, and the first 4 characters of the input array should be: ["a","b","1","2"].

**Explanation:**

Since the character "a" does not repeat, it is not compressed. "bbbbbbbbbbbb" is replaced by "b12".

Notice each digit has it's own entry in the array.

**Note:**

1. All characters have an ASCII value in [35, 126].
2. 1 <= len(chars) <= 1000.

Beats:95.15%



**447. Number of Boomerangs**

Given *n* points in the plane that are all pairwise distinct, a "boomerang" is a tuple of points (i, j, k) such that the distance between i and j equals the distance between i and k (**the order of the tuple matters**).

Find the number of boomerangs. You may assume that *n* will be at most **500** and coordinates of points are all in the range **[-10000, 10000]** (inclusive).

**Example:**

**Input:**

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

**Output:**

2

**Explanation:**

The two boomerangs are **[[1,0],[0,0],[2,0]]** and **[[1,0],[2,0],[0,0]]**

Beats:48.7%



**448. Find All Numbers Disappeared in an Array**

Given an array of integers where 1 ≤ a[i] ≤ *n* (*n* = size of array), some elements appear twice and others appear once.

Find all the elements of [1, *n*] inclusive that do not appear in this array.

Could you do it without extra space and in O(*n*) runtime? You may assume the returned list does not count as extra space.

**Example:**

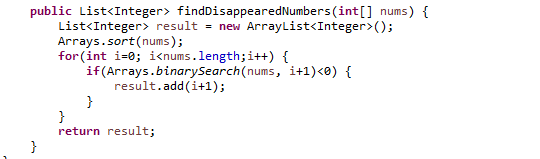
**Input:**

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

**Output:**

[5,6]

Beats:9.61%



**453. Minimum Moves to Equal Array Elements**

Given a **non-empty** integer array of size *n*, find the minimum number of moves required to make all array elements equal, where a move is incrementing *n* - 1 elements by 1.

**Example:**

**Input:**

[1,2,3]

**Output:**

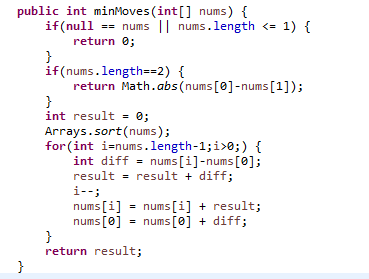
3

**Explanation:**

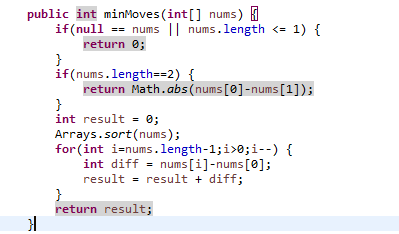
Only three moves are needed (remember each move increments two elements):

[1,2,3] => [2,3,3] => [3,4,3] => [4,4,4]

Beats:10.68%



Also 10.68:



**455. Assign Cookies**

Assume you are an awesome parent and want to give your children some cookies. But, you should give each child at most one cookie. Each child i has a greed factor gi, which is the minimum size of a cookie that the child will be content with; and each cookie j has a size sj. If sj >= gi, we can assign the cookie j to the child i, and the child i will be content. Your goal is to maximize the number of your content children and output the maximum number.

**Note:**  
You may assume the greed factor is always positive.  
You cannot assign more than one cookie to one child.

**Example 1:**

**Input:** [1,2,3], [1,1]

**Output:** 1

**Explanation:** You have 3 children and 2 cookies. The greed factors of 3 children are 1, 2, 3.

And even though you have 2 cookies, since their size is both 1, you could only make the child whose greed factor is 1 content.

You need to output 1.

**Example 2:**

**Input:** [1,2], [1,2,3]

**Output:** 2

**Explanation:** You have 2 children and 3 cookies. The greed factors of 2 children are 1, 2.

You have 3 cookies and their sizes are big enough to gratify all of the children,

You need to output 2.