给定一个非负数，表示一个数字数组，在该数的基础上+1，返回一个新的数组。

该数字按照大小进行排列，最大的数在列表的最前面。

您在真实的面试中是否遇到过这个题？

Yes

**样例**

给定 [1,2,3] 表示 123, 返回 [1,2,4].

给定 [9,9,9] 表示 999, 返回 [1,0,0,0].

[**标签**](http://www.lintcode.com/zh-cn/problem/plus-one/#tags)

思路：把数组转换成int。然后int变成字符串就行，然后存放就行。

class Solution:

# @param {int[]} digits a number represented as an array of digits

# @return {int[]} the result

def plusOne(self, digits):

# Write your code here

length=len(digits)

sum=0

for i in range(length):

sum=sum+digits[i]\*pow(10,length-i-1)

sum=sum+1

a=[]

while sum>=1:

h=sum

a.append(h)

sum=sum//10

a=a[::-1]

return a

2. 给一个排序数组（从小到大），将其转换为一棵高度最小的排序二叉树。

 注意事项

There may exist multiple valid solutions, return any of them.

您在真实的面试中是否遇到过这个题？

Yes

**样例**

给出数组 [1,2,3,4,5,6,7], 返回

4

/ \

2 6

/ \ / \

1 3 5 7

思路：高度最小，也就是比较平衡，因此递归取每个数组的最中间的值。因此需要下标，返回的条件是前面的序号要比后面的序号要大。

"""

Definition of TreeNode:

class TreeNode:

def \_\_init\_\_(self, val):

self.val = val

self.left, self.right = None, None

"""

class Solution:

"""

@param A: a list of integer

@return: a tree node

"""

def sortedArrayToBST(self, A):

# write your code here

length=len(A)

if length==0:

return None

else:

return newTreeNode (A, 0,length-1)

def newTreeNode (A, start,end) :

if (start > end) :

return None

node = TreeNode(0);

mid = end - (end - start)//2;

node.val = A[mid]

node.left = newTreeNode(A, start, mid - 1);

node.right = newTreeNode(A, mid + 1, end);

return node

3. 给定两个二进制字符串，返回他们的和（用二进制表示）。

您在真实的面试中是否遇到过这个题？

Yes

**样例**

a = 11

b = 1

返回 100

思路：传统的加法的形式只不过是逢二进一。首先把加法各位变成相同的，用零补上空缺的。再做个进位标志。

class Solution:

# @param {string} a a number

# @param {string} b a number

# @return {string} the result

def addBinary(self, a, b):

# Write your code here

lena=len(a);lenb=len(b)

if lena>lenb:

b="0"\*(lena-lenb)+b

elif lenb>lena:

a="0"\*(lenb-lena)+a

i=len(a)-1;j=len(b)-1

result=[]

h=0

s=""

while i>=0 and j>=0:

t=int(a[i])+int(b[j])+h

h=0

if t>=2:

t=t%2

h=1

i-=1;j-=1

result.append(t)

#print(result)

if h !=0:

result.append(h)

for i in range(len(result)):

s=s+str(result[len(result)-i-1])

return s

4．用递归的方法找到从1到最大的N位整数。

 注意事项

用下面这种方式去递归其实很容易：

recursion(i) {

if i > largest number:

return

results.add(i)

recursion(i + 1)

}

但是这种方式会耗费很多的递归空间，导致堆栈溢出。你能够用其他的方式来递归使得递归的深度最多只有 N 层么？

您在真实的面试中是否遇到过这个题？

Yes

**样例**

给出 N = 1, 返回[1,2,3,4,5,6,7,8,9].

给出 N = 2, 返回[1,2,3,4,5,6,7,8,9,10,11,...,99].

思路:每次递归时都要做些循环，通过循环来完成一位数，和二位数乃至多位数的放置

我的想法是多位数组合。不如下面方法好。

class Solution:

# @param n: An integer.

# return : A list of integer storing 1 to the largest number with n digits.

def numbersByRecursion(self, n):

# write your code here

a=[]

if n==0:

return a

digui(i=1,a=a,n=n,h=1)

return a

def digui(i,a,n,h):

if i<=n:

for j in range(h,int(i\*str(9))+1,1):

a.append(j)

return digui(i+1,a,n,h=int(i\*str(9))+1)

5. 给定 *n* 个非负整数 a1, a2, ..., an, 每个数代表了坐标中的一个点 (i, ai)。画 *n* 条垂直线，使得 *i* 垂直线的两个端点分别为(i, ai)和(i, 0)。找到两条线，使得其与 *x* 轴共同构成一个容器，以容纳最多水 。

 注意事项

容器不可倾斜。

您在真实的面试中是否遇到过这个题？

Yes

**样例**

给出[1,3,2], 最大的储水面积是2.

思路：寻找通项假设从i到j的最大值，在i到j-1里产生，和在i到j里最大值。

length=len(heights)

if length==0:

return 0

if length==1:

return heights[-1]

local=[0 for k in range(length)]

a=[0 for l in range(length)]

for j in range(1,length):

max1=0

for i in range(j+1):

leng=min(heights[j],heights[i])

if max1<leng\*(j-i):

max1=leng\*(j-i)

local[j]=max1

a[j]=max(a[j-1],local[j])

return a[-1]

#超时问题出现n2上怎么降低复杂度。应该时越近越好，同时距离还要大些，那么做两个l，r分别移动。谁小移动谁。这样就把复杂度降下来。

class Solution:

# @param heights: a list of integers

# @return: an integer

def maxArea(self, heights):

# write your code here

length=len(heights)

if length==0:

return 0

if length==1:

return heights[-1]

l=0;r=length-1;ret=0

while l<r:

ret = max(ret,min(heights[l],heights[r])\*(r-l))

if(heights[l]<heights[r]):

l+=1

else :

r-=1

return ret

6. 给定一个字符串，请找出其中无重复字符的最长子字符串。

您在真实的面试中是否遇到过这个题？

Yes

**样例**

例如，在"abcabcbb"中，其无重复字符的最长子字符串是"abc"，其长度为 3。

对于，"bbbbb"，其无重复字符的最长子字符串为"b"，长度为1。

思路就是慢慢找吧，把找到的值放进list里，每加入之前判断出现的序号，若是在里面就从序号开始取，一直遍历知道末尾。

class Solution:

# @param s: a string

# @return: an integer

def lengthOfLongestSubstring(self, s):

# write your code here

'''haxibiao'''

'''

#611ms

maxlength=0;first=0;table={}

for i in range(len(s)):

if s[i] in table and first<=table[s[i]]:

first=table[s[i]]+1

else:

maxlength=max(maxlength,i-first+1)

table[s[i]]=i

return maxlength

'''

#759ms

maxlength=0

temp=[]

for j in range(len(s)):

if s[j] not in temp:

temp.append(s[j])

else:

i=temp.index(s[j])

temp=temp[i+1:]

temp.append(s[j])

maxlength=max(maxlength,len(temp))

return maxlength

7. 给定两个整数数组（第一个是数组 A，第二个是数组 B），在数组 A 中取 A[i]，数组 B 中取 B[j]，A[i] 和 B[j]两者的差越小越好(|A[i] - B[j]|)。返回最小差。

您在真实的面试中是否遇到过这个题？

Yes

**样例**

给定数组 A = [3,4,6,7]， B = [2,3,8,9]，返回 0。

思路排序吧，减到符号改变或到list的末尾。

class Solution:

# @param A, B: Two lists of integer

# @return: An integer

def smallestDifference(self, A, B):

# write your code here

A.sort()

B.sort()

length1=len(A)

length2=len(B)

c=[]

i=0

j=0

ret=21474836471

while i<length1 and j<length2:

if A[i]<B[j]:

ret=min(ret,abs(A[i]-B[j]))

i+=1

else:

ret=min(ret,abs(A[i]-B[j]))

j+=1

return ret

8. 请判定一个数独是否有效。

该数独可能只填充了部分数字，其中缺少的数字用 . 表示。

 注意事项

一个合法的数独（仅部分填充）并不一定是可解的。我们仅需使填充的空格有效即可。

您在真实的面试中是否遇到过这个题？

数独是否有效应该是3个方面，行，列，区域块。

对于每一行来说只存在0-9，计算下出现个数就行。列也一样，区域块也一样。

class Solution:

# @param board, a 9x9 2D array

# @return a boolean

def isValidSudoku(self, board):

rows = [list(lst[::]) for lst in board]

columns = [[lst[idx] for lst in board] for idx in range(9)]

blocks\_origin = [board[row][column] for x in [[0, 1, 2], [3, 4, 5], [6, 7, 8]]\

for y in [[0, 1, 2], [3, 4, 5], [6, 7, 8]] for row in x for column in y]

blocks = [[blocks\_origin[row \* 9 + column] for column in range(9)] for row in range(9)]

check = lambda lst: all([lst.count(x) == 1 for x in lst if x != '.'])

return all([check(x) for style in (rows, columns, blocks) for x in style])

9给定一个整数数组（下标由 0 到 n-1， n 表示数组的规模，取值范围由 0 到10000）。对于数组中的每个 ai 元素，请计算 ai 前的数中比它小的元素的数量。

 注意事项

We suggest you finish problem [Segment Tree Build](http://www.lintcode.com/problem/segment-tree-build/), [Segment Tree Query II](http://lintcode.com/en/problem/segment-tree-query-ii/) and [Count of Smaller Number](http://www.lintcode.com/zh-cn/problem/count-of-smaller-number/) first.

您在真实的面试中是否遇到过这个题？

Yes

**样例**

对于数组[1,2,7,8,5] ，返回 [0,1,2,3,2]

没啥说的就是判断吧。

10. 给定一个整数数组 （下标由 0 到 n-1，其中 n 表示数组的规模，数值范围由 0 到 10000），以及一个 查询列表。对于每一个查询，将会给你一个整数，请你返回该数组中小于给定整数的元素的数量。

 注意事项

在做此题前，最好先完成 [线段树的构造](http://www.lintcode.com/problem/segment-tree-build/) and [线段树查询 II](http://lintcode.com/en/problem/segment-tree-query-ii/) 这两道题目。

您在真实的面试中是否遇到过这个题？

**样例**

对于数组 [1,2,7,8,5] ，查询 [1,8,5]，返回 [0,4,2]

思路：排序吧，然后只要找到最小得值就行，但可能不存在所以直接index不行，所以唯有二分查找。找到小于其的最大序号。跳出条件一index满足，另一个时不在里面然后二分查找，查找到最后应该时数组里只有一个元素。

class Solution:

"""

@param A: A list of integer

@return: The number of element in the array that

are smaller that the given integer

"""

def countOfSmallerNumber(self, A, queries):

# write your code here

'''

b=[]

if len(A)==0:

for m in range(len(queries)):

b.append(0)

return b

quicksort(A,0,len(A)-1)

for q in queries:

flag=1

while q not in A and q>= A[0]:

q-=1

flag=0

if q in A :

if flag:

b.append(A.index(q))

else:

q+=1

while q not in A:

q+=1

b.append(A.index(q))

else:

b.append(0)

return b

def quicksort(A,left,right):

low=left

high=right

if low < high:

key=A[low]

while low<high:

while low < high and A[high]>key:

high-=1

A[low]=A[high]

while A[low]<=key and low < high:

low+=1

A[high]=A[low]

A[low]=key

quicksort(A,left,low-1)

quicksort(A,low+1,right)

'''