剑指offer（python版） 33-36

33丑数

题目：把只包含质因子2、3和5的数称作丑数（Ugly Number）。例如6、8都是丑数，但14不是，因为它包含质因子7。 习惯上我们把1当做是第一个丑数。求按从小到大的顺序的第N个丑数。

思路：因为丑数只包含质因子2，3，5，假设我们已经有n-1个丑数，按照顺序排列，且第n-1的丑数为M。那么第n个丑数一定是由这n-1个丑数分别乘以2，3，5，得到的所有大于M的结果中，最小的那个数

事实上我们不需要每次都计算前面所有丑数乘以2，3，5的结果，然后再比较大小。因为在已存在的丑数中，一定存在某个数T2T2，在它之前的所有数乘以2都小于已有丑数，而T2×2T2×2的结果一定大于M，同理，也存在这样的数T3，T5T3，T5，我们只需要标记这三个数即可

代码：

1. # -\*- coding:utf-8 -\*-
2. **class** Solution:
3. **def** GetUglyNumber\_Solution(self, index):
4. # write code here
5. **if** index==0:
6. **return** 0
7. baselist=[1]
8. min2=min3=min5=0
9. curnum=1
10. **while** curnum<index:
11. minnum=min(baselist[min2]\*2,baselist[min3]\*3,baselist[min5]\*5)
12. baselist.append(minnum)
13. **while** baselist[min2]\*2<=minnum:
14. min2+=1
15. **while** baselist[min3]\*3<=minnum:
16. min3+=1
17. **while** baselist[min5]\*5<=minnum:
18. min5+=1
19. curnum+=1
20. **return** baselist[-1]

34第一个只出现一次的字符

题目：在一个字符串(0<=字符串长度<=10000，全部由字母组成)中找到第一个只出现一次的字符,并返回它的位置, 如果没有则返回 -1（需要区分大小写）

思路: 利用Python中的字典。字典的键（Key）一定唯一，每个键对应的值（Value）对应该键Key出现的次数。

1. 代码：# -\*- coding:utf-8 -\*-
2. **class** Solution:
3. **def** FirstNotRepeatingChar(self, s):
4. # write code here
5. dict={}
6. **for** i **in** s:
7. dict[i]=1 **if** i **not** **in** dict **else** dict[i]+1
8. **for** i **in** range(len(s)):
9. **if** dict[s[i]]==1:
10. **return** i
11. **return** -1

35数组中的逆序对

题目：在数组中的两个数字，如果前面一个数字大于后面的数字，则这两个数字组成一个逆序对。输入一个数组,求出这个数组中的逆序对的总数P。并将P对1000000007取模的结果输出。 即输出P%1000000007

输入描述：

题目保证输入的数组中没有的相同的数字

数据范围：

对于%50的数据,size<=10^4

对于%75的数据,size<=10^5

对于%100的数据,size<=2\*10^5

思路：拷贝该数组后对拷贝的数组排序。计算数组中的最小值在原始数组中出现的位置，统计原始数组中最小值前面的个数，之后在原始数组中去掉最小值。重复上述步骤

代码：（只对了一半，递归超时了）

1. # -\*- coding:utf-8 -\*-
2. **class** Solution:
3. **def** InversePairs(self, data):
4. # write code here
5. sortdata=sorted(data)
6. count=0
7. **for** i **in** sortdata:
8. pos=data.index(i)
9. count+=pos
10. data.pop(pos)
11. **return** count

36两个链表的第一个公共结点

题目：输入两个链表，找出它们的第一个公共结点

思路：利用两个链表的长度差来做。如果两个链表有公共节点，那两个链表共用公共节点之后的部分。计算两个链表的长度差diff，让较长的链表前进diff后，这两个链表同时同步向后移动，直至两个链表的节点相等

代码：

1. # -\*- coding:utf-8 -\*-
2. # class ListNode:
3. #     def \_\_init\_\_(self, x):
4. #         self.val = x
5. #         self.next = None
6. **class** Solution:
7. **def** FindFirstCommonNode(self, pHead1, pHead2):
8. # write code here
9. len1=self.getChainLen(pHead1)
10. len2=self.getChainLen(pHead2)
11. **if** len2>len1:
12. pHead1,pHead2=pHead2,pHead1
13. diff=abs(len2-len1)
14. **while** diff>0:
15. pHead1=pHead1.next
16. diff-=1
17. **while** pHead1!=pHead2:
18. pHead1=pHead1.next
19. pHead2=pHead2.next
20. **return** pHead1
21. **def** getChainLen(self,head):
22. chainlen=0
23. **while** head:
24. head=head.next
25. chainlen+=1
26. **return** chainlen