剑指offer(python版) 53-56

53表示数值的字符串

题目：请实现一个函数用来判断字符串是否表示数值（包括整数和小数）。例如，字符串"+100","5e2","-123","3.1416"和"-1E-16"都表示数值。 但是"12e","1a3.14","1.2.3","+-5"和"12e+4.3"都不是

思路：

定义两个标志位，分别表示E或者e是否出现过，以及小数点.是否出现过。   
1. 以e（或E）为分隔，获得两个子字符串；e之前的字符串小数点只能出现一次；e之后的字符串不允许出现小数点；   
2. 符号位+或-只可能出现在两个子字符串的首位；   
3. e（或E）、小数点.不能出现在末尾

代码：

1. # -\*- coding:utf-8 -\*-
2. **class** Solution:
3. # s字符串
4. **def** isNumeric(self, s):
5. # write code here
6. isAllowDot = True
7. isAllowE = True
8. **for** i **in** range(len(s)):
9. **if** s[i] **in** "+-" **and** (i==0 **or** s[i-1] **in** "eE") **and** i < len(s)-1:
10. **continue**
11. **elif** isAllowDot **and** s[i] == ".":
12. isAllowDot = False
13. **if** i >= len(s)-1 **or** s[i+1] **not** **in** "0123456789":
14. **return** False
15. **elif** isAllowE **and** s[i] **in** "Ee":
16. isAllowDot = False
17. isAllowE = False
18. **if** i >= len(s)-1 **or** s[i+1] **not** **in** "0123456789+-":
19. **return** False
20. **elif** s[i] **not** **in** "0123456789":
21. **return** False
22. **return** True

54字符流中第一个不重复的字符

题目：请实现一个函数用来找出字符流中第一个只出现一次的字符。例如，当从字符流中只读出前两个字符"go"时，第一个只出现一次的字符是"g"。当从该字符流中读出前六个字符“google"时，第一个只出现一次的字符是"l"。

思路：本题是字符流，也就是会增长的，每次字符串多一个字符，就要重新判断是哪个只出现一次的字符

因为牛客网里剑指offer的python只有2.7，没有3.0以上的版本，而python2.7的字典遍历通常不是有序的（python3通常有序），所以只能再借助一个列表来存储全部字符串，遍历字符串从而寻找

代码：

1. # -\*- coding:utf-8 -\*-
2. **class** Solution:
3. # 返回对应char
4. **def** \_\_init\_\_(self):
5. self.charlist=[]
6. **def** FirstAppearingOnce(self):
7. # write code here
8. **for** key **in** self.charlist:
9. **if** self.charlist.count(key)==1:
10. **return** key
11. **return** '#'
12. **def** Insert(self, char):
13. # write code here
14. self.charlist.append(char)

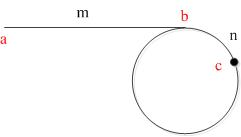
55链表中环的入口结点

题目：给一个链表，若其中包含环，请找出该链表的环的入口结点，否则，输出null。

思路：

假设有p，q两指针，p每步跨一个结点，q每步跨两个结点。那么经过k步之后q比p多走过的结点数为k。如果链表中没有环，那么q永远在p的前面，两指针不会相遇。如果链表有环，且环中结点个数为r，那么q会进入环内绕圈。如果两指针会在相遇，一定是在环内，并且q比p多绕n圈；也就是说一定存在k，只要满足k=n \* r，就能使两指针相遇。

如下图所示，假设链表起点为a，入口结点为b，相遇点为c，ab段长度为m，bc段长度为n，环长度为r。经过上面的证明，我们知道c点一定存在，并且根据两指针走过的路程，存在等式2(m+n+h\*r)= m+n+g\*r，这样我们得到m=(g-2h)\*r-n，也就是说ac段长度是环长度的整数倍。如果p指针重新从链表头出发，q指针从相遇点c出发，步伐一致，那么当p走到b点，q也正好走到b。



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** EntryNodeOfLoop(self, pHead):
8. # write code here
9. **if** **not** pHead **or** **not** pHead.next **or** **not** pHead.next.next:
10. **return** None
11. slow=pHead.next
12. fast=pHead.next.next
13. **while** slow!=fast **and** fast.next:
14. slow=slow.next
15. fast=fast.next.next
16. **if** slow==fast:
17. slow=pHead
18. **while** slow!=fast:
19. slow=slow.next
20. fast=fast.next
21. **return** slow
22. **return** None

56删除链表中重复的结点

题目：在一个排序的链表中，存在重复的结点，请删除该链表中重复的结点，重复的结点不保留，返回链表头指针。 例如，链表1->2->3->3->4->4->5 处理后为 1->2->5

思路：要删除有序链表中所有的重复节点，而头结点有可能就是重复节点。这样的比较好的解决方式就是新建头结点，然后往后遍历，同样的值就全部略过

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** deleteDuplication(self, pHead):
8. # write code here
9. first=ListNode(-1)
10. first.next=pHead
11. cur=pHead
12. last=first
13. **while** cur **and** cur.next:
14. **if** cur.val != cur.next.val:
15. cur = cur.next
16. last = last.next
17. **else**:
18. val = cur.val
19. **while** cur **and** cur.val == val:
20. cur = cur.next
21. last.next = cur
22. **return** first.next