剑指offer（python版）57-60

57二叉树的下一个结点

题目：给定一个二叉树和其中的一个结点，请找出中序遍历顺序的下一个结点并且返回。注意，树中的结点不仅包含左右子结点，同时包含指向父结点的指针。

思路：

1.二叉树为空，则返回空；

2.节点右子树存在，则设置一个指针从该节点的右子树出发，一直沿着指向左子结点的指针找到的叶子节点即为下一个节点；

3.节点不是根节点。如果该节点是其父节点的左子树，则返回父节点；否则继续向上遍历其父节点的父节点，重复之前的判断，返回结果

代码：

1. # -\*- coding:utf-8 -\*-
2. # class TreeLinkNode:
3. #     def \_\_init\_\_(self, x):
4. #         self.val = x
5. #         self.left = None
6. #         self.right = None
7. #         self.next = None
8. **class** Solution:
9. **def** GetNext(self, pNode):
10. # write code here
11. **if** **not** pNode:
12. **return** pNode
13. **if** pNode.right:
14. current=pNode.right
15. **if** current.left:
16. current=current.left
17. **return** current
18. **while** pNode.next:
19. parent=pNode.next
20. **if** parent.left==pNode:
21. **return** parent
22. pNode=parent

58

题目：请实现一个函数，用来判断一颗二叉树是不是对称的。注意，如果一个二叉树同此二叉树的镜像是同样的，定义其为对称的。

思路：运用递归的方法，就好像一串葡萄有左右两小串葡萄，把两小串葡萄拆下来对比葡萄大小。

代码：

1. # -\*- coding:utf-8 -\*-
2. # class TreeNode:
3. #     def \_\_init\_\_(self, x):
4. #         self.val = x
5. #         self.left = None
6. #         self.right = None
7. **class** Solution:
8. **def** isSymmetrical(self, pRoot):
9. # write code here
10. **if** **not** pRoot:
11. **return** True
12. **return** self.recursivet(pRoot.left,pRoot.right)
13. **def** recursivet(self,left,right):
14. **if** **not** left **and** **not** right:
15. **return** True
16. **if** **not** left **or** **not** right:
17. **return** False
18. **if** left.val==right.val:
19. **return** self.recursivet(left.left,right.right) **and** self.recursivet(left.right,right.left)
20. **return** False

59按之字形顺序打印二叉树

题目：请实现一个函数按照之字形打印二叉树，即第一行按照从左到右的顺序打印，第二层按照从右至左的顺序打印，第三行按照从左到右的顺序打印，其他行以此类推

思路：第一种是：获取到所有节点的值后，将偶数层的节点值倒序

代码：

1. # -\*- coding:utf-8 -\*-
2. # class TreeNode:
3. #     def \_\_init\_\_(self, x):
4. #         self.val = x
5. #         self.left = None
6. #         self.right = None
7. **class** Solution:
8. **def** Print(self, pRoot):
9. # write code here
10. resultArray = []
11. **if** **not** pRoot:
12. **return** resultArray
13. curLayerNodes = [pRoot]
14. isEvenLayer = True
15. **while** curLayerNodes:
16. curLayerValues = []
17. nextLayerNodes = []
18. isEvenLayer = **not** isEvenLayer
19. **for** node **in** curLayerNodes:
20. curLayerValues.append(node.val)
21. **if** node.left:
22. nextLayerNodes.append(node.left)
23. **if** node.right:
24. nextLayerNodes.append(node.right)
25. curLayerNodes = nextLayerNodes
26. resultArray.append(curLayerValues[::-1]) **if** isEvenLayer **else** resultArray.append(curLayerValues)
27. **return** resultArray

60把二叉树打印成多行

题目：从上到下按层打印二叉树，同一层结点从左至右输出。每一层输出一行

思路：层序遍历就使用队列的那种形式即可，只是在每一层遍历之前，要先保存这一层的节点个数

代码：

1. # -\*- coding:utf-8 -\*-
2. # class TreeNode:
3. #     def \_\_init\_\_(self, x):
4. #         self.val = x
5. #         self.left = None
6. #         self.right = None
7. **class** Solution:
8. # 返回二维列表[[1,2],[4,5]]
9. **def** Print(self, pRoot):
10. # write code here
11. result=[]
12. **if** **not** pRoot:
13. **return** result
14. A=[]
15. A.append(pRoot)
16. **while** A:
17. size=len(A)
18. temp=[]
19. **for** node **in** A:
20. temp.append(node.val)
21. result.append(temp)
22. **for** i **in** range(size):
23. node =A.pop(0)
24. **if** node.left:
25. A.append(node.left)
26. **if** node.right:
27. A.append(node.right)
28. **return** result