### **Week 7 Day 4**

### **105 Construct Binary Tree from Preorder and Inorder Traversal**

### **Review III**

### 105-1 要找inorder和preorder的关系，所以必然对于preorder的点我想知道inorder其中的位置，所以sol 2 先traverse一遍inorder的点用hashmap记录点与位置的对应关系。

### 105-2 iteration solution之后再看

### **106 Construct BT from Inorder and Postorder Traversal**

### **Review I**

### **107 Binary Tree Level Order Traversal II**

### **108 Convert Sorted Array To BST**

### **Review III**

### 108-1 为什么mid = start + (end-start)/2 + (end-start)%2 ？ 最后一排数目不一定能达到所有的点，为了a height balanced BST，想让有点的叶子尽量往左靠。

### **Week 8 Day 1**

### **109 Covert Sorted List to BST**

### **Review I**

109-1 快慢指针找mid。要留住pre方便做截断操作

### **110 Balanced BT**

### **111 Minimum Depth Of BT**

### **Review III**

111-1 要考虑特殊情况，只有一个孩子的节点不是叶子，不能停止

### 112. Path Sum

### Review I

### 112-1 任何点的值可以为负数不要思维惯性。认为sum<0就是出界了

### **113 Path Sum II**

### **Week 8 Day 2**

### **114 Flattern BT to LinkedList**

### **Revisit III for both iteration and recursion approach**

### **115 Distinct Subsequence**

### **Revisit I Typical DP string question**

### **116 117 Populating next right pointers in each node**

### **Revisit II**

### **Week 8 Day 3**

### **118 119 Pascal’s Triangle I II**

### **120 Triangle**

### **Revisit III**

### **From bottom to top**

### **121-123 Best Time to Buy and Sell Stock**

### **Review III**

121-1 Greedy

122-1 Always bottom to peak

123-1 the number is two : so from front, from end: two dp then find the cross.两边greedy夹逼法。要对

123-2 初始化数组时候declaration的部分的括号和内容别忘了要和后边defination部分的相同

int[][] dp = new int[t+1][len+1];

123-2 本题最关键的是对dp的优化。一般有两个for loop了对于第三个forloop就要想办法优化为o(1)

Week 8 Day 4

124 BT Maximum Path Sum

Review I

124-1 变量名称要明确，如本题verticalmax, horizontalmax, curmax, res非常好！！！

125 Valid Palindrome

Review I

125-1 分类很重要 isValidChar(char c) isValidNumeric(char c) 分别写

### **126 Word Ladder II**

### **Review III**

126-1 A typical example of Dijkstra algorithm。本来想模仿127的做法。但是发现因为需要所有的可能性。对于1 2 3 4 和1 5 3 4 的情况。因为3 在第一次得到答案的时候已经visited过了，所以会丢解。自然想到dfs。而使用dfs需要bfs的图作为基础。

126-2 在判断相邻的时候不要两两比较（时间复杂度n平方）而应采用替换每个字符的方法（时间复杂度26n）

126-3 本题未在本轮两天内通过解答。下轮一定重新做

127 Word Ladder I

Review III

127-1 只需要一个count，本题用queue代表每一个level。因为只需要一个合适的，所以遍历并更新每一层结果即可。一个visited数组就管用了

### **126 127 先bfs确定每个点的distance来保证接下来的dfs中可以避免重复！revisit！！！**

### **Week 9 Day 1**

### **128 Longest Consecutive Sequence**

### **Review II**

### 128-1 需要时间复杂度O(n) 🡪 hashMap(可以visit任何一个点用O(1)

### 128-1 用hashSet.contains(num+1)来判断是否连续！或者本题也可以用treeset sequence

### **129 Sum Root to Leaf Numbers**

### **Review I for 129-1**

### 129-1 StringBuilder 如何减去最后一个字符： sb.setLength(sb.length()-1);

### Or sb.deleteCharAt(sb.length()-1)

### 129-2 Stringbuilder 其他重要用法复习： sb.insert(0,str); sb.setCharAt(int index, char ch)

### **130 Surrounded Regions**

### Review II

### Sol 1: DFS

### 130-1 Use additional mark for further processing. 本题用\*代表要保留的O。然后遍历数组，把所有的O变为X，再将所有的\*变为O

### 130-2 DFS把握关键点：何时进入下一层循环（recursion）。这个是有边界的。

### 130-3 BFS的做法一定要考虑Queue。对于二维数组则直接把它变回一维即可。

### **Week 9 Day 2**

### 131