**Week 4 Day 2**

### 51: N-Queens I

### Review III

51-1 如果copy 复制同一段代码请一定记住改变量名 比如i改成j。。。

51-2 .不要写成,

51-3 画图分析坐标之间的关系！

### 52 N-Queens II

52-1 My solution is easy to change

### Week 4 Day 3

### 53 Maximum SubArray

53-1 求最大值 可能是dp做法

### 54 Spiral Matrix

### Review I

54-1 for loop 请看本题个人答案为标准请背诵

### 55 JumpGame

55-1 在whileloop 中不断调整curMax可以够到的值，是greedy的做法

### 56 Merge Intervals

### Review III

56-1 sol1:可以用最简单的iterator遍历treeMap。按照Key的从小到大顺序

56-2 sol2: treeMap 重要的函数：

ceilingEntry(K key) return a key-value mapping associated with the least key greater than or equal to the given key, or null if there is no such key

ceilingKey(K key) return the least key greater than or equal to the given key, or null if there is no such key

firstEntry() return the first(lowest) key currently in this map

56-3 Sol 3: 很好的 Collections.sort(collection name, new Comparator<ColectionGenericType>(){

@Override

public int compare(GenericType g1, GenericType g2)

{

}

});

### Week 4 Day 4:

### 57 Insert Interval

### Revisit III

57-1 在一段数组中插入。在开头前和结尾后的特殊情况单独考虑！！！中间有一个交汇的过程。要考虑如果一开始就交汇会怎么样，如果交汇结束后立刻程序停止会怎么样。请阅读我自写代码的含义！

### 58 Length Of Last Word

### 59 Spiral Matrix II

### 60 Permutation Sequence

### Review II

60-1 boolean digits = new Boolean[9]; 声明和定义类型要相同

60-2 为什么要k-- revisit！！

60-3 反复提取数组中的第m个元素 可以用arraylist，提取后删除提取元素，index会自动更新

60-4 阶乘如何用！ 用array[i]存储 array[i+1] = (i+1)\*array[i] !!!

**Week 5 Day 1**

### 61 Rotate List

### 62 Unique Paths

62-1 相比recursion解法，dp解法的好处是同一个点只计算检索一次。

### 63 Unique Paths II

### 64 Minimum Path Sum

### Review I

64-1 请看题目本身 因为已经提供了一个matrix 所以Leetcode solution 3 4 本身就提供了解决问题的方法。

### 65 Valid Number

### Review II

65-1 boolean 数组不需要initialize （default认为都是false）。但是一个boolean单独变量需要initialize

65-2 当可以使用hashSet 如果元素所有的可能性已知，可以使用boolean 变量。变量的个数就是所有可能hashSet取值的个数。

### Week 5 Day 2

### 66 Plus One

66-1 进位题 顺序！！！先取得进位next再对当前为数取余：先做 / 再做 %

66-2 plus one 加1可以认为是初始时next==1 。注意第一位和其他位的区别

66-3 Offical Solution： 思路更清晰：如果不是九加一return，如果是 看下一轮

### 67 Add Binary

### Review I

67-1 StringBuilder 有reverse() 函数，可以最后关心次序。

67-2 string.charAt() 出来的是char 如果想做数学计算一定 -‘0’

67-3 stringbuilder.insert(position,value);

67-4 两个不同长度的数(String)， 我的做法是找出最小长度分别处理。题中最优解把sum的三个变量前面加上if，更优：

int i = a.length() - 1, j = b.length() -1, carry = 0;

while (i >= 0 || j >= 0) {

int sum = carry;

if (j >= 0) sum += b.charAt(j--) - '0';

if (i >= 0) sum += a.charAt(i--) - '0';

sb.append(sum % 2);

carry = sum / 2;

}

### 68 Test Justification

### Review III

68-1 每个单词在选择时候有两种可能：加到此行或者不加，所以使用start end 两个指针。End-1是此行最后一个数。从而分三种讨论： start == (end-1) , end到达所有单词的末尾，还有一般的情况。

68-2 用while loop with start end指针的时候，结尾不要忘了更新： start = end; end = end+1;

**Week 5 Day 3:**

### 69 Sqrt(x)

### Review III

69-1 二分法可以用 但是如何防止越界！！！ if(mid>x/mid) !!!用除法，巧妙回避越界问题！

69-2 bit operation ！！！ 一共三十二位，每位有一个可能。从最高位起开始，每位的和用 | 或运算比较！！

### 70 Climbing Stairs

### 71Simplify Path

### Review II

71-1 String s= "/chaoyang/fan/handsome/"; split(“/”)以后第一个/之前的length为0的string会保留但是最后一个不会！

71-2 stringBuilder.insert(0,str); 在String的前边加东西的方法

71-3 Stack在pop()之前永远要判断是否为空

### 72 Edit Distance

### Review I

72-1 DP关注的是第一个变量到i的结果转化为第二个变量到j的结果

72-2 empty string自己会占数组中的一个位置，所以数组的维度是字符的个数加一

### 73 Set Matrix Zeroes

### Review I

73-1 不占据额外空间，肯定是利用现有空间的特殊位置（如头行头列）做记号。然后用boolean firstRow boolean firstCol单独去记录头行头列的状态

### Week 5 Day 4

### 74 Search a 2D Matrix

74-1 我的解法非常好。但是注意，一个sorted 2D matrix可以永远想成一个sorted 1D matrix，revisit 官方解法1

### 75 Sort Colors

### Review II

### 75-1 双指针站肩 每次可以区分出一个颜色 ，最后剩的可以自动找出。一共三个所以两轮 (n-1)

### **77 Combinations**

77-1 两种tree的分层：1 当前一轮选哪个元素 2 某元素选还是不选

77-2 另一种方法，两个变量n k 看看有无可能dp：

C(n,k) = C(n-1,k-1) + C(n-1,k)

### **76 Minimum Window Substring**

### **Review III**

### 76-1 My Solution: 用两个hashMap比较相对关系。注意while loop结尾的fast++

78 Subsets

79 Word Search

Review II

79-1 利用visited[] 数组表示已经访问过。本题最容易出错的地方！用的是递归，在开始的时候visited[i][j] = true, 在函数结尾别忘了visited[i][j] = false 使用递归的时候，所有的变量最后请复原。

79-2 解法中的fancy solution非常有趣。因为是char每个数在0-255之间，使用其与256抑或，一定不会在其中，所以下一轮一定不会revisit这个点，从而通过改变value为一个不可能的数存储信息在矩阵中。注意，因为是递归，在函数末尾一定要把数值改回去

### **Week 6 Day 1**

### **80 Remove Duplicates From Sorted Array II**

### **Review I**

### **81 Search In Rotated Sorted Array II**

### **Review I**

### **82 Remove Duplicates from sorted List II**

### **83 Remove Duplicates from Sorted List**

### **Week 6 Day 2**

### **84 Largest Rectangle in Histogram**

### **Review III**

### 84-1 Use stack to store “array index” (position) and compare when “pop up”

### **85 Maximum Rectangle**

### 85-1 Based on question 84, think the matrix to an linear array with value

### **Review III**

### **Week 6 Day 3**

### **86 Partition List**

### **87 Scramble String**

### 87-1 如何比较两个string是回文的： toCharArray() 然后 Arrays.sort() 然后再分别new String() 用equals判断

### 87-2 用substring() 来解决string的partition并用于递归。本题是很好的范例。Revisit!

### **88 Merge Sorted Array**

### **89 Gray Code**

### **Review II**

### 89-1 This is a good example to use ArrayList, bit operation to monitor stack and realize recursion

### **90 SubSets II**

### **Review II**

### 90-1 请revisit我的两种做法。

### **Week 6 Day 4**

### **91 Decode Ways**

### **Review II**

### 91-1 为什么本题可以用DP：在解码的过程中有两种可能。后边的可能性取决于前面的可能性相加的和

### **92 Reverse LinkedList II**

### 92-1 规规矩矩分层次写码，目的要明确

### **93 Restore IP Addresses**

### 93-1 因为需要返回所有的结果，所以只能用recursion不能用dp

### 93-2 ‘0’的处理是个重点。 如何将string转为数字: Integer.parseInt(str.substring(start,end));

### **94 Binary Tree Inorder Traversal**

### **Revisit III iterative solution**

### 94-1 Itervation solution !!! Revisit while(root!=null) || !stack.is Empty())

### **95 96. Unique Binary Search Trees I II**

### 95-1 注意本题解法中for loop的用法，本题返回了 group (list) of values

### **Week 7 Day 1**

### **97 Interleaving String**

### **Review II**

### 97-1 用dp时候dp[i+1][j+1]的matrix注意对应str.charAt()时候是i-1 或者 j-1 ！！！

### 97-2 此题是典型的从二维到三维映射关系，一定用dp

### **98 Validate BST**

### 98-1 因为涉及到预先设定的最大值与最小值，sol 1 用了Long.MAX\_VALUE 和 Long.MIN\_VALUE

### 98-2 如果用sol 1，注意题中所有的int都改为long

### 98-3 valid 与否，in order traverse即可知！注意如果初始值边界为Integer Min/ Max 写成如下形式！

### long pre = Long.MIN\_VALUE;

### **99 Recover BST**

### **Review III**

### 99-1 Morris Traversal Revisit!!!

### **100 Same Tree**

### **Week 7 Day 3**

### **101 Symmetric Tree**

### **Review II**

### 101-1 使用pairs input for stack/queue 两两比较方法！！！相对位置不会变

### **102 103 BT Level Order Traverse**

### **Revisit III**

### 102-1 用DFS来解决level order traverse 问题。Level作为一个argument传入每一层函数，从而该cur value可以得知应该去哪一个list上继续加。

### 102-2 用null区分queue的层数。或者因为已知每一层的数目，可以直接count

### **104 Maximum Depths of BT**

### **Week 7 Day 4**