# 线性表

## 数组

### 26. Remove Duplicates from Sorted Array

STL：

unique(Arr, Arr+n) 去除矩阵中相邻元素的重复值，返回地址

去除重复元素后的数组的个数count为count= unique(Arr, Arr+n)-Arr

distance(Arr, Arr+n) 返回两个迭代器之间的距离

### 80. Remove Duplicates from Sorted Array II

### 33. Search in Rotated Sorted Array

用二分法，O(logn)

### 81. Search in Rotated Sorted Array II

用二分法，O(logn)，对于相同的元素，直接跳过

### 4. Median of Two Sorted Arrays

用二分法，O(logn)

### 128. Longest Consecutive Sequence

用hashmap记录两端的值

### 1. Two Sum

O(n),用hashmap

### 15. 3Sum

先排序，然后左右夹逼，可以计算k个值的和

复杂度为O(max{nlogn, nk-1})

### 16. 3Sum Closest

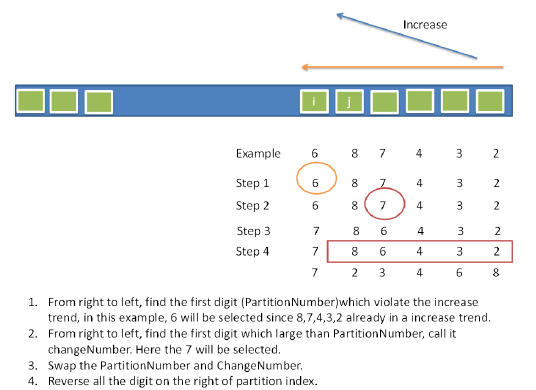
先排序，然后左右夹逼，复杂度O(n2)

### 18.4Sum

先排序，然后左右夹逼，复杂度O(n3)，可以先用一个hashmap先缓存两个数的和，最终复杂度为O(n2)

### 27. Remove Element

### 31. Next Permutation



### 60. Permutation Sequence

采用康托编码的思路，假设n个不重复的元素，第k个排列是a1,a2,….an,那么a1是第k/(n-1)!个元素，同理，k2 = k%(n-1)!， a2 = k2/(n-2)!，……

### 36. Valid Sudoku

细节实现，检查行，检查列，检查9个小方格

### 42. Trapping Rain Water

思路1：记录每个位置处的左边起的最大值和右边起的最大值，遍历数组，如果某位置处的左右最大值中的最小值大于该高度，则水量+=(min-height)，表示该位置处可以蓄水

时间O(n),空间O(n)

思路2：以最大值将数组分成两半，从左边到最大值位置，记录最大的元素，如果当前元素小于最大元素，则可以蓄水，水量+=(max-height)，时间O(n),空间O(n)

思路3：利用栈，每次将栈内小于当前高度的全部出栈，时间O(n),空间O(n)

### 48. Rotate Image

1.从外到内，一圈圈转。

2.沿着副对角线翻转一次，然后沿着水平中线转一次

3.沿着水平中线转一次，沿着主对角线翻转一次

时间复杂度O(n^2), 空间O(1)

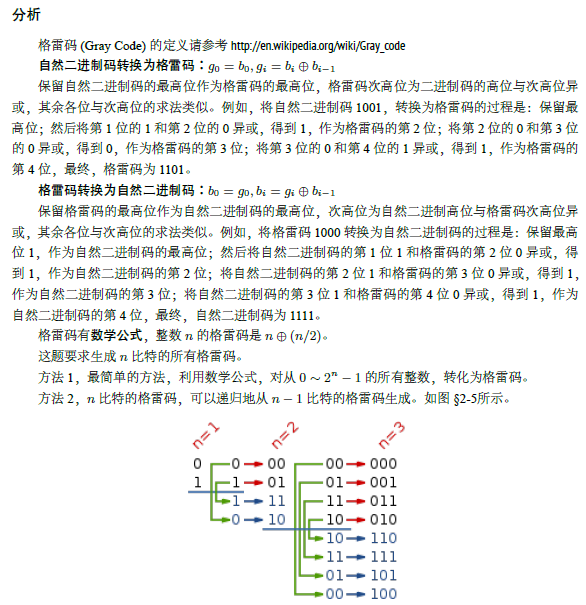
### 66. Plus One

时间O(n),空间O(1)

### 70. Climbing Stairs

斐波那契数列，时间O(n),空间O(1)

### 89. Gray Code



### 73. Set Matrix Zeroes

用两个数组记录行和列是否存在.时间O(m\*n)，空间O(m+n)

### 134. Gas Station

时间O(n),空间O(1)

只要sum(gas)>=cost(gas)，就可以走一个循环，至于循环开始的位置，记录最后一次当前和<0时的位置i,返回i+1即可

### 135. Candy

方法1：左右各扫描一次，初始值都为1，从左扫描，如果是递增，则值加1，否则归位为1，从右往左扫描，如果是递减，则值加1，否则归位为1，存储矢量中记录每个位置处的最大值，时间O(n),空间O(n)

方法2，递归版，麻烦，繁琐，思想为用一个矢量记录结果，i位置处的值是在该值大于左值时，左值结果+1与当前值的最大值，该值大于右值时，右值结果+1与当前值的最大值，所以每个位置处最多计算两次，且不会重复计算。时间O(n),空间O(n)

方法3：自己编写，用scount和lcount记录上升序列数和下降序列数，结果res在每点直接累加res+=(scount+1)或res=(lcount+1)，表示序列长度变化时，序列中的所有人的糖果数都需要增加1，需要处理的是在下降序列的第一个点时用temp记录当前的上升序列数，然后在上升序列的第一个点时判断temp，当temp不为0时，需要把res值减去min(temp,lcount)，因为在局部极大值处重复加了两次。时间O(n),空间O(1).

### 136. Single Number

用异或处理，厉害，可以处理任意偶数次

### 137. Single Number II

还是考虑位运算

方法1：用一个长度为sizeof(int)的count数组记录第i位出现1的次数。把次数不是3的整数倍的位组合起来构成结果。

方法2：用one记录二进制1出现“1次”的有哪些二进制位，用two记录二进制1出现“2次”的有哪些二进制位，当某一位的one和two同时为1时，将其归0.最终one记录的是最终的结果。本题的思路是用二进制模拟三进制运算。

## 单链表

### 2. Add Two Numbers

类似二进制加法，时间O(n)，空间O(1)

### 92. Reverse Linked List II

时间O(n)，空间O(1)

### 86. Partition List

时间O(n)，空间O(1)

### 83. Remove Duplicates from Sorted List

时间O(n)，空间O(1)

### 82. Remove Duplicates from Sorted List II

时间O(n)，空间O(1)

### 61. Rotate List

时间O(n)，空间O(1)

### 19. Remove Nth Node From End of List

用两个节点p,q,q先走n步，然后两个节点一起走，直到q走到尾节点

### 24. Swap Nodes in Pairs

时间O(n)，空间O(1)

### 25. Reverse Nodes in k-Group

时间O(n)，空间O(1)

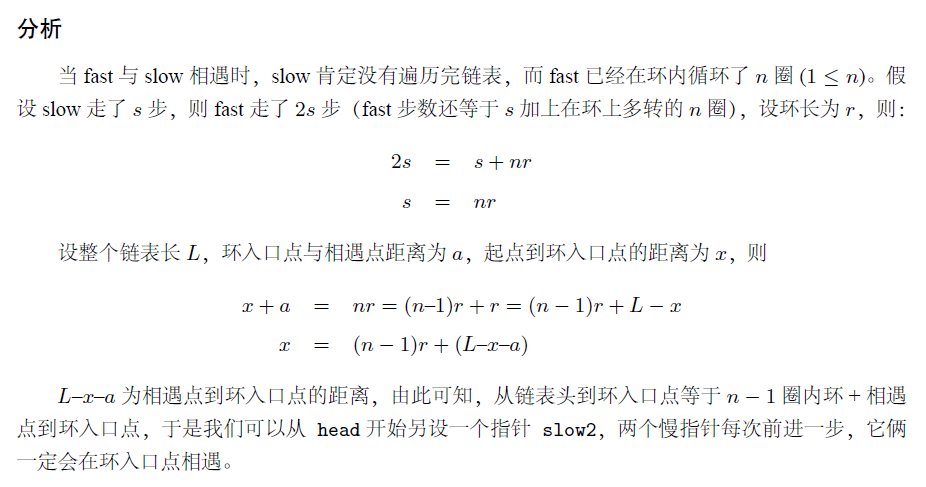
### 138. Copy List with Random Pointer

三遍扫描，第一遍新建节点，使构成a1,b1,a2,b2的形式，第二遍确定random节点，第三遍分拆节点。时间O(n)，空间O(1)

### 141. Linked List Cycle

设置慢指针和快指针，一个每次走一步，一个每次走两步，如果两个指针相遇，则说明有环。

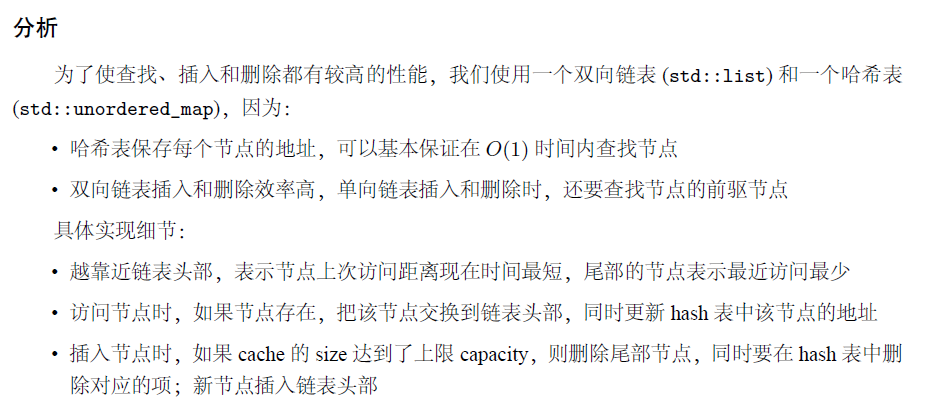
### 142. Linked List Cycle II



### 143. Reorder List

找到中间节点，断开，把后半截单链表reverse，然后合并。时间O(n)，空间O(1)

### 146. LRU Cache



# 字符串

### 125. Valid Palindrome

isalnum(int c)判断是否是字母或者数字

### 28. Implement strStr()

暴力算法，时间O(m\*n)，空间O(1)

KMP,实际O(n+m),空间O(M)

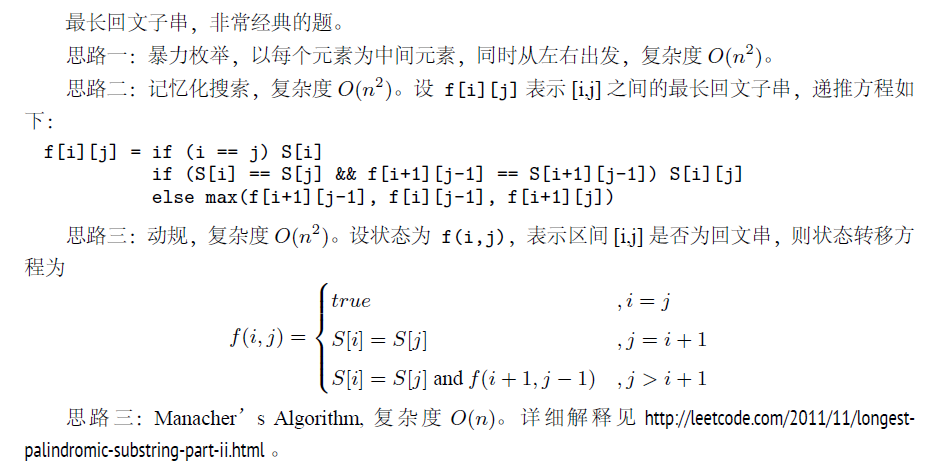
还有Boyer-Mooer,Rabin-Karp算法

### 8. String to Integer (atoi)

细节题

### 67. Add Binary

### 5. Longest Palindromic Substring



动态规划，时间O(n^2),空间O(n^2)

Manacher算法，时间O(n)，空间O(n)

### 10. Regular Expression Matching

递归版，时间O(n)，空间O(1)

### 44. Wildcard Matching

### 14. Longest Common Prefix

纵向扫描，时间复杂度O(n1+n2+…)，对每个字符串同时从0,1,2个字符开始比较

横向扫描，每个字符串与第0个字符串比较，直到遇到不匹配的，时间复杂度O(n1+n2+…)

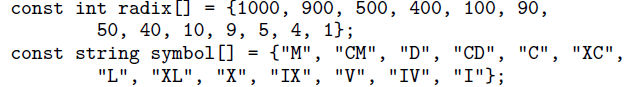
### 65. Valid Number

1.用有限状态机，时间O(n),空间O(n)

2.用strtod()

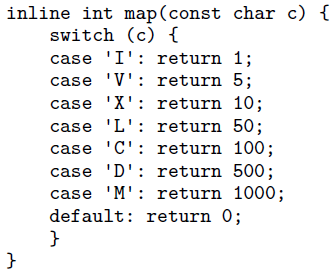
### 12. Integer to Roman

用两个数组



### 13. Roman to Integer

建立一个map



### 38. Count and Say

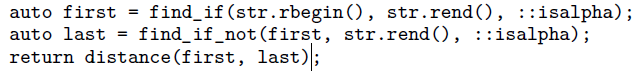
### 49. Group Anagrams

建立一个string到vector<string>的map

### 71. Simplify Path

### 58. Length of Last Word

1.用STL



isalpha判断是否是英文字母

2.顺序扫描

# 栈和队列

## 栈

### 20. Valid Parentheses

### 32. Longest Valid Parentheses

使用栈，时间O(n),空间O(n)

两遍扫描，时间O(n),空间O(1)

### 84. Largest Rectangle in Histogram

使用栈，时间O(n),空间O(n)

### 150. Evaluate Reverse Polish Notation

## 队列

# 树

## 二叉树遍历

Morris遍历

### 144. Binary Tree Preorder Traversal

栈，时间O(n),空间O(n)

### 145. Binary Tree Postorder Traversal

栈，时间O(n),空间O(n)

### 94. Binary Tree Inorder Traversal

栈，时间O(n),空间O(n)

### 102. Binary Tree Level Order Traversal

递归版，时间O(n),空间O(n)

迭代版，时间O(n),空间O(1)

### 107. Binary Tree Level Order Traversal II

同上

### **103. Binary Tree Zigzag Level Order Traversa**

递归版，时间O(n),空间O(n)

迭代版，时间O(n),空间O(n)

### 99. Recover Binary Search Tree

中序遍历，找到最前和最后的逆序的数对即可

### 100. Same Tree

递归版，时间O(n),空间O(logn)

迭代版，时间O(n),空间O(logn)

### 110. Balanced Binary Tree

时间O(n),空间O(logn),定义函数，若树是平衡的，返回高度，否则返回-1

### 101. Symmetric Tree

递归版，时间O(n),空间O(logn)

迭代版，时间O(n),空间O(logn)

### 114. Flatten Binary Tree to Linked List

递归，时间O(n),空间O(logn)

迭代，时间O(n),空间O(logn)

### 117. Populating Next Right Pointers in Each Node II

递归，时间O(n),空间O(1)

## 二叉树构建

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

时间O(n),空间O(logn)

### 106. Construct Binary Tree from Inorder and Postorder Traversal

时间O(n),空间O(logn)

## 二叉查找树

### 96. Unique Binary Search Trees

动态规划，时间O(n^2),空间O(n)

### 95. Unique Binary Search Trees II

动态规划，时间O(n^2),空间O(n)

### 98. Validate Binary Search Tree

时间O(n),空间O(logn)

### 108. Convert Sorted Array to Binary Search Tree

时间O(n),空间O(logn)

### 109. Convert Sorted List to Binary Search Tree

1,自顶向下方法，记录链表长度，时间O(n^2),空间O(logn)

自底向上方法，每次修改list指针，先创建左树，然后创建root,然后修改list，然后创建右树

## 二叉树的递归

### 111. Minimum Depth of Binary Tree

时间O(n),空间O(logn)

### 104. Maximum Depth of Binary Tree

时间O(n),空间O(logn)

### 112. Path Sum

时间O(n),空间O(logn)

### 113. Path Sum II

时间O(n),空间O(logn)

### 124. Binary Tree Maximum Path Sum

时间O(n),空间O(logn)

MPS思路是求左子树的MPS和右子树的MPS与包含root节点的左右子树最大Path Sum的最大值。可以用一个函数实现求MPS和Path Sum，其返回Path Sum，函数递归调用左子树l和右子树r，将最大的Path Sum返回，然后对于l，r大于0，则将其与root->val相加，然后对result求最大值。

### 116. Populating Next Right Pointers in Each Node

1，递归

2，BFS，队列

3，BFS，循环

### 129. Sum Root to Leaf Numbers

时间O(n),空间O(logn)

# 排序

### 88. Merge Sorted Array

从后到前，时间O(m+n),空间O(1)

### 21. Merge Two Sorted Lists

时间O(min(m,n)),空间O(1)

### 23. Merge k Sorted Lists

复用mergeTwo, 时间O(n1+n2+n3+..),空间O(1)

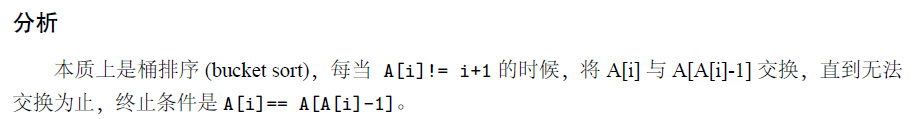
### 147. Insertion Sort List

时间O(n^2),空间O(1)

### 148. Sort List

用归并排序，利用mergeTwo,时间O(nlogn),空间O(1)

### 41. First Missing Positive



时间O(n),空间O(1)

### 75. Sort Colors

1.记录三种颜色出现的次数，需要两遍扫描，时间O(n),空间O(1)

2.双指针，时间O(n),空间O(1)

3.用快排的partition思想

# 查找

### 34. Search for a Range

二分查找, 时间O(logn),空间O(1)

STL,lower\_bound, upper\_bound

### 35. Search Insert Position

用lower\_bound，时间O(logn),空间O(1)

### 74. Search a 2D Matrix

二分法，时间O(logn),空间O(1)

# 暴力枚举法

### 78.Subsets

1.增量构造法，每个元素都可以选或者不选，时间O(2^n)，空间O(n)

2.位向量法，每个元素都可以选或者不选，时间O(2^n)，空间O(n)

3.二进制法，在元素数不超过int位数时可用

### 90. Subsets II

用增量构造法，需要复位和判断是否是重复元素，时间O(2^n)，空间O(n)

### **46. Permutation**s

用next\_permutation()

### 47. Permutations II

同46题

### 77. Combinations

1.深搜，递归，时间O(n!),空间O(n)

2.用prev\_permutation,将k个1，n-k个0的向量进行permutation，每次根据1的位置创建当前矢量。

### 17. Letter Combinations of a Phone Number

深搜，递归，时间O(3^n),空间O(n)

# 广度优先搜索

### 127. Word Ladder

广度优先算法，时间O(n),空间O(n)

### 126. Word Ladder II

### 130. Surrounded Regions

# 深度优先搜索

### 131. Palindrome Partitioning

1.dfs,时间O(2^n),空间O(n)

2,动态规划法，时间O(n^2),空间O(1)

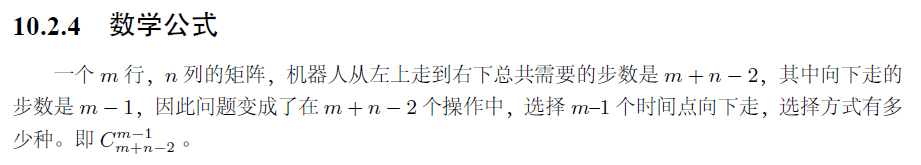
### 62. Unique Paths

1.深度搜索，时间O(n^4),空间O(n)

2.深搜加缓存，时间O(n^2),空间O(n^2)

3.动态规划，自顶向下，时间O(n^2),空间O(n)

4.



### 63. Unique Paths II

动态规划，时间O(n^2),空间O(n)

### 51. N-Queens

深搜+剪枝，时间O(n!),空间O(n)

### 52. N-Queens II

深搜+剪枝，时间O(n!),空间O(n)

### 93. Restore IP Addresses

时间O(n^4),空间O(n)

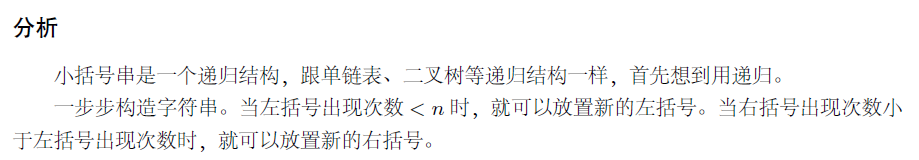
### 39. Combination Sum

时间O(n!),空间O(n)

### 40. Combination Sum II

时间O(n!),空间O(n)

### 22. Generate Parentheses



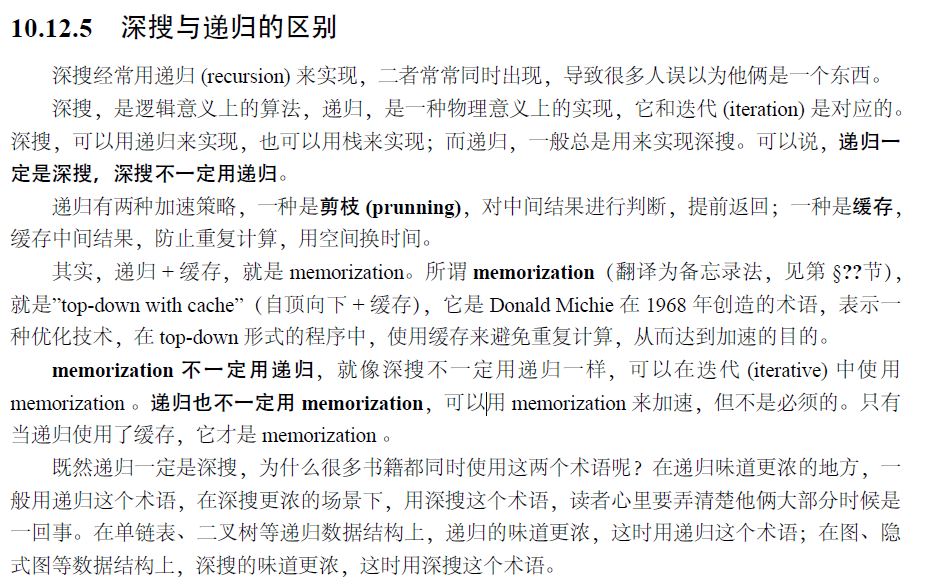
### 37. Sudoku Solver

### 79. Word Search

深度搜索，时间O(n^2m^2)，空间O(n^2)

### 212. Word Search II





# 分治法

### 50. Pow(x, n)

二分法，时间O(logn)，空间O(1)

### 69. Sqrt(x)

二分法，时间O(logn)，空间O(1)

# 贪心法

### 55. Jump Game

时间O(n)，空间O(1)

### 45. Jump Game II

时间O(n)，空间O(1)

### 121. Best Time to Buy and Sell Stock

时间O(n)，空间O(1)

### 122. Best Time to Buy and Sell Stock II

时间O(n)，空间O(1)

### 2. Longest Substring Without Repeating Characters

时间O(n)，空间O(1)

### 11. Container With Most Water

从两边往中间走，时间O(n)，空间O(1)

# 动态规划

### 120.Triangle

时间O(n^2)，空间O(1)

我做的为时间O(n^2)，空间O(n)

### 53. Maximum Subarray

时间O(n)，空间O(1)

### 131. Palindrome Partitioning

时间O(n^2)，空间O(n^2)

### 132. Palindrome Partitioning II

时间O(n^2)，空间O(n^2)

### 85. Maximal Rectangle

时间O(n^2)，空间O(n^2)

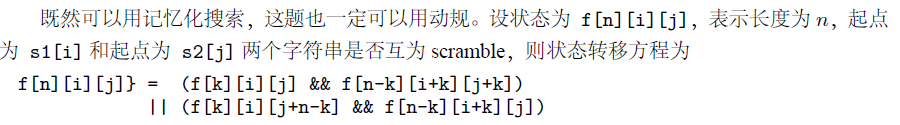
### 123. Best Time to Buy and Sell Stock III

时间O(n)，空间O(n)

### 97. Interleaving String

时间O(n^2)，空间O(n^2)

### 87. Scramble String



### 64. Minimum Path Sum

### 72. Edit Distance

时间O(n\*m)，空间O(n\*m)

### 91. Decode Ways

时间O(n)，空间O(1)

### 115. Distinct Subsequences

时间O(n)，空间O(1)

### 139. Word Break

时间O(n^2)，空间O(n)

### 140. Word Break II

时间O(n^2)，空间O(n^2)

# 图

### 133. Clone Graph

广度遍历或者深度遍历都可以，时间O(n)，空间O(n)

深度遍历往往用递归，宽度遍历往往用队列

# 细节实现题

### 7. Reverse Integer

时间O(logn)，空间O(1)

### 9. Palindrome Number

时间O(1)，空间O(1)

### 57. Insert Interval

时间O(n)，空间O(1)

### 56. Merge Intervals

时间O(n^2)，空间O(n)

### 76. Minimum Window Substring

时间O(n)，空间O(1)

### 43. Multiply Strings

将字符串转换成int数组，然后对应相乘，时间O(n\*m)，空间O(n+m)

### 30. Substring with Concatenation of All Words

时间O(n\*m)，空间O(m)

### 118. Pascal's Triangle

时间O(n^2)，空间O(n)

### 119. Pascal's Triangle II

从后往前滚动，时间O(n^2)，空间O(n)

### 54. Spiral Matrix

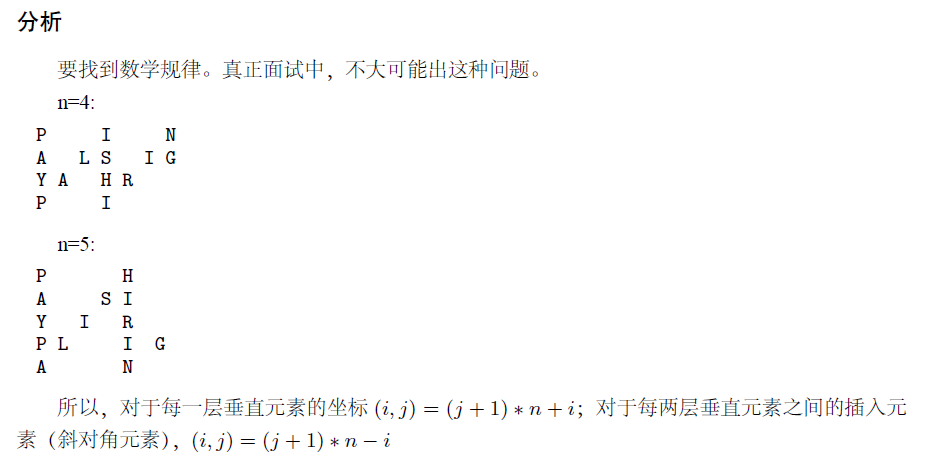
时间O(n^2)，空间O(n)

### 59. Spiral Matrix II

时间O(n^2)，空间O(n^2)

### 6. ZigZag Conversion

时间O(n)，空间O(1)



### 29. Divide Two Integers

时间O(logn)，空间O(1)

### 68. Text Justification

时间O(n)，空间O(1)

### 149. Max Points on a Line

1.暴力枚举法，以边为中心，每两个点组合，判断其它点是否与它构成直线，时间O(n^3)，空间O(1);

2.以点为中心，遍历剩余的点，找到所有斜率，如果斜率相同，则一定共线。以哈希表记录key,value，key为斜率，value为点数，找到最大的值存储

时间O(n^2),空间O(n)