http://www.1point3acres.com/bbs/thread-203983-1-1.html. 对code quality要求特别高。

要面FB了，在看面经，顺便把2016年度所有地里的FB面试题都总结了，dirty work，没什么技术含量。希望对大家有帮助。还没offer呢，攒攒人品。另外，好心人给我点货币，新人不知道咋弄货币呢，但是有时候看帖、下载需要。。。

15. 3Sum. Be sure to handle duplicate. Save lastK, lastJ. If same, skip. Also num[i[]== num[i - 1], skip. 3Sum closest: same as 3 sum, except remember the diff for every l++, r--

139. Word Break I/II: (1) dp. If f[j], and substring(j, i) is word. (2). DFS and memorize intermediate result for each substring.

91. Decode Ways: dp. Cases: current char > '2', '0', '1', '2', ways[i] = ways[i + 1] ….

209. Minimum Size Subarray Sum: two pointers, left and right. Moving window and keep sum > s. If there's negative number, see here: check if (sumSoFar - target) in map also see max size

- Map store previous sum values (O(N))
- 把第一题extend到2D。给一个matrix, all elements are positive，问有没有个sub rectangle加起来和等于target。return true/false。sum (0,0) to (i, j) and save in map. Check if sum(i, j) - target in map.
- Lz听到题目有点懵，认真调整心态，解决之。先写了个cumulative sum。把所有从0,0 到i,j的和算在新的matrix的i,j上。方便之后算head到tail的sub rectangle的和。这一步O(n^2)

350. Intersection of Two Arrays II

- Solution 1: Hashmap. Solution 2: sort, then find duplicates
- If only nums2 cannot fit in memory, put all elements of nums1 into a HashMap, read chunks of array that fit into the memory, and record the intersections. If both nums1 and nums2 are so huge that neither fit into the memory, sort them individually (external sort), then read 2 elements from each array at a time in memory, record intersections.

给一个字典包括很多字符串(e.g., abcd, dhfyf)，然后给定一个字符串查看字典中是否包含这个字符串。字符串中可能包括*，*可以匹配任何字符。我用的Trie。

Task那道题，很多面经都提到过。就是比如给你一串task，再给一个cooldown，执行每个task需要时间1，两个相同task之间必须至少相距cooldown的时间，问执行所有task总共需要多少时间。比如执行如下task：12323，假设cooldown是3。总共需要的时间应该是 1 2 3 _ _ 2 3，也就是7个单位的时间。再比如 1242353，假设cool down是4，那总共时间就是 1 2 4 _ _ _ 2 3 5 _ _ _ 3，也就是13个单位的时间. 用个map存了不同type最近的time slot，每碰到相同的type就check一下冷却时间过了没，没过就等待。

- 基于1，给出最优的排列，使得字符串最短。

自然string comparator。不知道的搜下。就是string 比较的时候考虑里面数字的大小，比如 abc9 < abc123 abc > ab9 因为char比digit重要。

117. Populating Next Right Pointers in Each Node II,: each level keep a pre pointer for the previous node on horizontal level,a head pointer for the head of next level; next level, cur pointer move to head pointer. and next pointers I: Connect left child to right child. Follow next link until no next at current level.

- salbring tree，不过没有next指针， 你要用原来的left，right指针
- Level BFS

binary tree转换成doubly linked list

- And revert it back (reverted to balanced tree): 108. convert sorted array to BST, 109. Convert Sorted List to Binary Search Tree: use slow/fast pointer and recursive build BST

75. Sort Colors: index r = 0, b = len -1, loop i towards b, if color[i] blue swap to color[b--], if color[i] red swap to color[r++]

314. Print a binary tree by columns top to bottom. BFS, left col index minus 1, right plus 1. And use a map to store the list for each column index. Get min/max column index during iteration. After tree traverse, from min to max column index, get lists from map.

We're given a sorted array of integers: [-3, -1, 0, 1, 2]. We want to generate a sorted array of their squares: [0, 1, 1, 4, 9]

Longest common substring: dp, dp[m][n] is the length of longest common suffix O(m * n). Or O(m + n) using generalized suffix tree

list of sorted integer arrays，要求找所有的数的median. e.g. [1,3,6,7,9], [2,4, 8], [5], return 5. LC 4, median of 2 sorted arrays. Also see here. If A[mid] < B[mid], can discard A before mid. Use findKthElement(....)

LC 1 two sum + three sum + follow up

Best Time to Buy and Sell Stock, (LC 121, I and LC 122, II), also see LC 123, III and LC 188, IV, dp for k transactions. If k >= n / 2, can trade as many like. Otherwise, for day j, max profit for up to k transaction:

```
        dp[i, j] = max(dp[i, j-1], prices[j] - prices[jj] + dp[i-1, jj]) { jj in range of [0, j-1] }
                 = max(dp[i, j-1], prices[j] + max(dp[i-1, jj] - prices[jj]))
```

- buy and sell stock，每天可以买一股，也可以都卖了，或者不买不卖。
- Find maximum, buy from earlier days, and sell on that day.

LC 309, Buy Sell stock with cool down, dp, 3 states, buy, sell or rest at day j.

```
        buy[i]  = max(rest[i-1]-price, buy[i-1])
        sell[i] = max(buy[i-1]+price, sell[i-1])
        rest[i] = max(sell[i-1], buy[i-1], rest[i-1])
```

Because buy[i] <= rest[i] so means rest[i] = max(sell[i-1], rest[i-1]).so [buy, rest, buy] is never occurred. A further observation is that rest[i] <= sell[i] is also true therefore rest[i] = sell[i-1]. So we get:

buy[i] = max(sell[i-2]-price, buy[i-1])
sell[i] = max(buy[i-1]+price, sell[i-1])

states of day i relies only on i-1 and i-2 we can reduce the O(n) space to O(1).

33. Search in Rotated Sorted Array: binary search. Also see LC 81 where there are duplicates. Need compare num[mid] with num[high]. When equals, high--.

38. Count and Say

sparse vector dot multiplication. LC discuss, and LC 311 List<List<int[]>> rows to store each row as a list, and each list element is an int[2], where first element is index, second one is value.

- 这道题我当时并没有准备到，但是正因为如此，我认为我跟面试官的交流给我加分了不少。面试官首先问我每个vector很大，并不能在内存中存下，该怎么办，我说只需要存下非零的元素和他们的下标就行，然后询问面试官是否可以用预处理后的这两个vector非零元素的index和value作为输入，面试官同意后快速写完O(M*N)的代码，M和N分别是两个vector的长度。面试官说这两个输入如果是根据下标排序好的话应该怎么办，我说可以遍历长度较短的那一个，然后用二分搜索的方法在另一个vector中找index相同的元素，相乘加入到结果中，这样的话复杂度就是O(M*logN)。这时，面试官又问是否可以同时利用两个输入都是排序好这一个特性，我在这个地方有点卡住，但是在白板上写出一个test case，试着用可视化的方法帮助我来进行思考，同时面试官给了一些提醒，最后写出了O(M + N)的双指针方法
- 然后问如果有一个向量比另一个长很多怎么办，遍历短的，对长的二分查找。
- 两个vector相乘

211. Add and Search Word: Trie implementation.

239. Sliding Window Maximum: use Deque to store array index, remove from head all elements which are outside of the window. Remove from tail all numbers which are less than current number. Insert current number to the tail. Poll the head and save to result array after first k elements.

282. Expression Add Operators: be sure to handle “0”, and pass in multiplier for “*”

158. Read N Characters Given Read4 II - Call multiple times: use a buffer to save content read in. LC 157, Read 4 once. Check eof by returned count. If returned more than asked, only return asked.

49. Group Anagrams - use counting sort. Sort the characters array of each string, use it key, value are anagrams.

linked list 反序输出 - 1) LC 206, reverse the list; 2) recursion. Also see LC 92, reverse from m to n. Use fake head usually can simplify.

问题一：flatten an array?

Counting sort: count each character/digit, and then count[i] += count[i - 1], then use the count as index to place the character/digit in the array, and decrease the count.

285. Inorder Successor in BST: loop { if (root.val > p.val) { succ = root; root = root.left; } else { root = root.right; } }. Note: use BST 特性

如果>和<号不再具有transitivity，即A>B=>B<A但是A>B, B>C不等价于A>C。求数组中最大的数。Compare the first two. Discard the worse one. Compare the 'better' with the next element across the list. The one survived needs to be compared with every element from the original list except those that it was already compared with. If it 'loses' even once, return no 'best'. If 'wins' every comparison return it. Similar to LC 277 celebrity

283. Move Zeroes
- Minimizing “writes”: use insert index to place non zero numbers first. Assign 0s at last.

5. Longest Palindromic Substring: use current index as center to extend left and right to find longest palindrome. Check odd length and even length palindrome.

62. Unique Paths I: dp, or actually it’s selecting n - 1 moves from n + m - 2 total moves.
63. Unique Paths II: now there are obstacles. dp[i][j] = (obstacle[i][j] ? 0 : dp[i - 1][j] + dp[i][j - 1]

moving all the nonzeros to the front of a list。similar to 283. Scan backwards.

Prettify JSON: 输入[1,2,3, {"id": 1, "name": "wang", "tag":[1,"home",2], "price":234}]

236. Lowest common ancestor: for BST O(logn), choose root or go down left/right depending on value. For binary tree, nodes p, q: if p or q is root, return root. Else recursive on left and right. If both non-null, return root, else return the one not null.

find distance between two nodes in a binary tree: find LCA of the two nodes, O(n), from this node find distance to both and add up.

Smallest subarray with sum greater than a given value: LC 209. Two pointers, l and r. Advance r, until sum > target. So (r - l) satisfy and keep advancing l while sum > target, and update min(r - l)

flatten nested array: similar to LC 251. Advance to correct place whenever next() called.

215. Kth Largest Element in an Array: use partition. If pivot index less than k, low change to pivot + 1 and repartition, else high change to pivot - 1. Note: kth largest, not smallest, so need k = length - k

114. Flatten Binary Tree to Linked List: recursive dfs, return last node

- Linked list needs to be formed as a cycle

301. Remove Invalid Parentheses (hard)

- Maintain counter and go from left to right, increase 1 when meets ‘(‘, decrease when ‘)’. Continue if count >= 0, otherwise remove first ‘)’ in consecutive ‘)’s, recursive call on the changed string from last ‘(‘ and ‘)’ indexes. If finished, run same on reversed string to remove additional ‘(‘s. If both finished, we get valid string and add to results.
- Easy solution is BFS, each level, each string if invalid, remove one “(“ or “)” and add to queue for next level. If any string is valid, no more process to next level after finish this level. Time: n*C(n, n) + (n-1)*C(n, n-1) + … 1 * C(n, 1) = n * 2^(n-1)

218. The Skyline Problem (hard): put tuples (xStart, -height), (xEnd, height) into list and sort base on x-cor first, height second. Put all into TreeMap one by one, key is height, value is count, and ordered by height descending. In each loop, if it’s xStart, increase the height count, else decrease. Get the first element of the tree map, if height diff from previous height, add (x, height) to result.

278. First Bad Version: binary search on version 1 to n

Min Queue, 跟Min Stack (LC 155)类似， 实现一个Queue， 然后O（1）复杂度获得这个Queue里最小的元素。Min stack : Use two stacks, one stack to store current min value. When push, and value <= current min, push to min value stack too. When pop, if value equals top of min stack, pop that stack too.

interval [startTime, stoptime)   ----integral time stamps
给这样的一串区间 I1, I2......In，找出 一个 time stamp 出现在interval的次数最多。
startTime <= t< stopTime 代表这个数在区间里面出现过。
example :   [1,3), [2, 7),   [4, 8),   [5, 9) 5和6各出现了三次， 所以答案返回5，6。 see here.
Or: make an array of timestamp,add startTime of each interval, add 0 - stopTime. Sort by abs value. Loop through the array, if positive, add 1, else minus one. So the result is the count of intervals at the timestamp. Also See 253. Meeting Rooms II

LC 28 strStr: KMP matching. Construct next[] from pattern. while(str[i]!=pattern[j]) j = next[j]

shortest continuous substring with all characters in input

- 76. Minimum Window Substring. The template in the top vote solution is super. Also see LC 3, longest substring without duplicates, LC 159, longest with at most 2 duplicates

合并邮件列表（后来才知道也是个面经题）
Given 1 million email list:

list 1: a@a.com, b@b.com
list 2: b@b.com, c@c.com
list 3: e@e.com
list 4: a@a.com
...
Combine lists with identical emails, and output tuples:
(list 1, list 2, list 4) (a@a.com, b@b.com, c@c.com)
(list 3) (e@e.com)

79. Word Search: DFS from each (i, j)

输出所有 root - leaf 的路径，递归做完了让迭代。

- Iterative? BFS, when meet leaf, add path to result. Keep a path from root to current level.

17. Letter Combinations of a Phone Number: DFS, pass in a path of translated characters

398. Random Pick Index: reservoir sampling. If is target, count++. Call random.nextInt(count). If 0, result is current index. Loop to end. Return the last result.

37. Sudoku Solver: dfs. For each (i, j) if not occupied, try 1 - 9 and check if it's valid. If valid, go down and recursive call. Otherwise return recover to vacant and false.

一个完全树。node有parent指针。每个node的值为 0或 1. 每个parent的值为两个子node的 "and" 结果. 现在把一个leaf翻牌子（0变1或者1变0）. 把树修正一遍. DFS. fix parent when return.

200. Number of Islands: dfs from each "1" point. Mark visited place in original matrix.

BST to increasing array, see LC 108

- Recursive, iterative. Iterative solution, use 3 stacks, one node stack for all nodes, one left stack to store the left boundary of the corresponding node, right stack to store the right boundary of the node. While node stack is not empty, pop and create left/right node plus their boundary to the stacks.

173. Binary Search Tree Iterator: in constructor, get leftmost leaf, and push all into stack. next() pop from stack, and if right child not empty, get leftmost child of the right child.

BST iterator
Iterator for a list of BSTs (heap contain each BST's iterator)

128. Longest Consecutive Sequence: add all numbers into set. For each number "num", starting from num - 1, check if it's in the set. If in, increase count and remove the number from set. Also, starting from num + 1, do same. Loop through all numbers and get max count.

22. Generate Parentheses: each step, if open "(" less than max, can add one "(" and recursive. If close ")" less than open "(", can append ")" and recurse.

238. Product of Array Except Self: not use additional space. Use output result array as buffer.

191. Number of 1 Bits: mask 1, shift 32 times, AND with the number.

给2D平面上的N个点，求离原点最近的K个点: use heap of K, O(n * logK), or partition, O(n), worst O(n^2), similar to LC 215,

找出两个给出两个string, leetcode, codyabc和一个数字k = 3,问两个string里面存不存在连续的common substring大于等于k.比如这个例子，两个string都有cod,所以返回true。楼主用dp建了一个m*n的table秒了，然后写test case,发现有个小corner case,改了,pass

- Longest common substring

给定一个数列，比如1234，将它match到字母上，1是A，2是B等等，那么1234可以是ABCD但是还可以是12是L，所以1234也可以写作LCD 或者AWD. LC 91, decode ways. Dp backwards.

给出N个序列，比如2个序列A,B,没个序列包含若干的区间，比如
A: [1,5], [10,14], [16,18]
B: [2,6], [8,10], [11,20]
Merge them all: [1,6], [8, 20]. Similar to LC 56, merge intervals

balance parentheses in a string
例子：
"(a)()" -> "(a)()"
"((bc)" -> "(bc)"
")))a((" -> "a"
"(a(b)" ->"(ab)" or "a(b)"
Note: balance的意思就是把原来string里unpaired的括号变成paired的形式。如果有多个可能的结果，比如上述最后一种情况，我们就只需要输出一个对的结果即可，所以这点简化了题目的难度。

感受：遍历string，用一个stack存储每个open parenthesis的index，也就是'('的index, 每当遇到closed parenthesis就执行一次pop操作。
注意两种unbalanced的情况：
1. 出现多余的')': 对应情况就是stack为空，但遇到了一个')'。
2. 出现多余的'(': 对应情况就是遍历结束，stack未空
Simplified LC 301 which ask for all valid results.

get binary tree's next node in inorder
class Node {Node left, Node right, Node parent}
Node getNext (Node current) {}

给一个tree，每个node 有很多children，找到所有最深的nodes 的common ancestor,

- 比如只有一个点最深，那返回他自己。
- Similar to LC 236. Lowest Common Ancestor of a Binary Tree

78. Subsets
90. Subsets II: there are duplicates. Sort numbers first.

341. Flatten Nested List Iterator

102. Binary Tree Level Order Traversal: iterative BFS. add values to the list for this level.

给三个funtions: is_low(), is_mid(), is_high(). 让给一个数组排序, low的放在最前面, mid的放在中间, high的放在最后面.
- Color sort: think about when there are K colors. For K colors, can do counting sort.

39. Combination Sum: recursive DFS.

125. Valid Palindrome: ask interviewer about how to handle empty. Also space, non-alpha-digit
214. Shortest Palindrome: The KMP solution is hard to understand and come up. The easy to think one check at each index to see if can extend left/right as palindrom and reach the left end. If so, add the rest to the end in reverse order.

98. Validate Binary Search Tree

Longest Arithmetic Progression)

10. Regular Expression Matching, also LC 44. Notice the diff of LC 44 and LC 10 in * handling. For LC 44, each time fail, back track to * position and re match.

211. Add and Search Word: simply Trie

138. Copy List with Random Pointer: to avoid using hashmap, first loop and copy nodes, assign original node next to copied node. Next loop and fix random pointers to correct copy. Last, loop again and separate the linked list to original list and copied list.

71. Simplify Path: be sure to handle corner cases, such as "/../" should return "/", multi redundant "/" should be reduced to one.

221. Maximal square: dp

314. Binary Tree Vertical Order Traversal: iterative BFS. also save column number for each node in this level. For left child assign parent.column - 1, right plus one. Get min and max column number during traverse.

198. House Robber: keep two values, robThisHouse and norobThisHouse. At each loop update these 2 values from last loop.

53. Maximum Subarray: keep 2 values, maxEnding which is sum ending at current number, may be previous plus current number if positive, else just current number. maxSum which is the max sum so far.

152. Maximum Product Subarray: keep 2 values for current positive and current negative. Update according to current value.

32. Longest Valid Parentheses: dp, dp[i] is the longest valid string ending at i. So check if charAt(i) == '(', dp[i] = 0; if charAt(i) == ')', 2 cases. Case 1, charAt(i - 1) == '(', then dp[i] = dp[i - 2] + 2; else if charAt(i - dp[i - 1] - 1) == '(', dp[i] = dp[i - 1] + 2 + dp[i - dp[i - 1] - 2]; else 0.

277. Find the Celebrity: first pass, find candidate. Whenever knows(cand, i), change cand to i. Second pass, verify the candidate is the real celebrity.

56. Merge Intervals: sort on start time. Then loop. For each interval, check if end time >= start time of next interval. If so, combine the intervals.

- Variant: 一串start time - end time，格式是Apr 2010 - Mar 2011这种，要求计算出这些时间的总跨度，重叠的跨度不重复计算。举例：["Apr 2010 - Dec 2010", "Aug 2010 - Dec 2010", "Jan 2011 - Mar 2011"]

57. Insert Interval: loop for all intervals whose end < newInterval.start, add to result; loop for all intervals whose start <= newInterval.end, and merge all of them with new interval, and add to result; add the rest intervals to the result.

206. Reverse Linked List: easy iterative or recursive.

implement circular array. Circular buffer

Check big/small endian

第一题：binary tree，给定一个value，return bin tree里面下一个比value大的值
第二题：binary tree的node加一个ptr next，point到inorder traversal的下一个node，比上一个简单

297. Serialize and Deserialize Binary Tree: Need represent null pointer. And splitter to separate nodes. From a list of nodes, remove first node and create root. Then recursive call to create left subtree, then right subtree.

Given a list of number, there is only one peak or one drop. Find the maximum drop.
Exps:
1 -> 2 -> 3 -> 9 -> 3 -> 0 = 9;
10 -> 4 -> 3 -> 8 = 7 ;

给一个array, 然后给一个k, 让你check 连续的k个integer是否含有dulplicate: 用窗口为K的hashset一直扫一遍就行了. Loop if hashset contains >= k numbers, remove nums[low]. If set contains nums[high], then there's duplicate. Advance high and add nums[high] to the set.

300. Longest increasing subsequence: use array tails[n]. Tails[i] store the smallest tail of all increasing sequence of length i + 1. So it's sorted and can do binary search on it. Search the place in tails that current number can put in. update the number. If it's the end, increate size. Finally size is the result.

377. Combination Sum IV. Given an integer array with all positive numbers and no duplicates, find the number of possible combinations that add up to a positive integer target.: dp[target + 1], save the counts of ways to add to target.

224. Basic Calculator: use an operator stack and an operand stack. Or as in one simple solution, keep a sign with "+" or "-". Push and reset result and sign when see "(", pop and calculate result with sign.

43. Multiply Strings: result[m + n]. For digit num1.charAt(i) times digit num2.charAt(j), result put at result[i + j + 1]. Carry from inner loop put in result[i]. To get final result, skip leading "0"s. If all zero, return "0". Else return the rest.

282. Expression Add Operators: dfs. Be sure to pass in multi factor when use "*".

顺时针的print binary tree boundary, 就是从根开始，先打右边界，再打叶子，最后打左边界。

310. Minimum Height Trees: solution 1) find longest path, middle point(s) are the root(s). To find longest path, start from any node do bfs, find most distant node x, then start from x, find most distant node y, x to y is the longest path. Solution 2) build adjacent list for the graph. Put all nodes with 1 adjacent to list of leaves, reduce 1 from all nodes adjacent to the leaves, if the node become leaf, add to new leaves list. Reduce the size n by size of leaves list. Run again on new leaves list until n <= 2. The final leaves list contains the MHT roots.

29. 不用"／", "%"运算符实现division，说了可以用binary search. First turn everything into long to avoid overflow. Recursively.

273. Integer to English Words: group numbers to chunks. Get arrays for below 10, below 20, below 100. Bigger numbers are reduced to above cases. helper(num / 10****0) + " hundred/thousand/million/billion " + helper(num % 10****0). And "0" is special case.

111. Minimum Depth of Binary Tree: recursive DFS or iterative BFS.

找两个字符串中长度为N以上的共同子串: dp[i, j] is the length of longest common string ending at i and j respectly in 2 strings.

一个数组内要是存在至少三个升序的数（array[x] < array[y] < array[z], x < y < z）就返回true. See here(correct?) and here. Save x, and keep minSoFar. If j not set or found current number less than the chosen j, replace j with current number and i with minSoFar. If current number > nums[j], then found result.

161. One Edit Distance: if length diff > 1, false; if same length, check if one character modified; else check if longer string is deleted one char from shorter.

print max depth path of a binary tree

151. Reverse Words in a String: reverse the whole string. Scan the string, reverse characters between spaces.

261. Graph Valid Tree
- any connected graph without simple cycles is a tree. Union find. For edge(i, j), if findRoot(i) == findRoot(j), means there's circle. Else parent[findRoot(i)] = findRoot(j)

给一个linkedlist，里面的element都排序好了，但是是一个blackbox，有三个function可以调用。pop()随机pop出最前面或最后面的element，peek()随机偷看最前面或最后面的element，isEmpty()回传linkedlist是不是空了。问设计一个资料结构，list或是array都可以，把linkedlist里面所有的element都拿出来，并保持他们的排序。followup是如果不能用peek()该怎么做。
- My thinking: if I got element A, and next element B is smaller than A, then A is from the tail of the list; otherwise, A is from the head of the list.

133. Clone Graph: DFS or BFS, keep a map from label to node.

395. Longest Substring with At Least K Repeating Characters: divide & conquer. For string from start to end position, count character first. Find a index mid, where cnt[s.charAt(mid)] < k, recursive call on range (start, mid) and (mid + 1, end).

Erase duplicate in an unsorted array: brutal force, O(n^2), sort O(n logn), hashmap O(n).

几何算法问题。如果给你一堆的矩形， 求重合矩形重合最多的坐标位置。我上过一个算法课，大概思路就是做一个二维的meeting room II

给定N个2D坐标（可以设想为餐厅的位置），要求输入任意坐标，可以返回方圆d距离内的所有餐厅

65. Valid Number: flags to check: numberSeen, eSeen, pointSeen, numberAfterE, sign