周日总结！！！！！

Aug 7-2019 beginning from this day

Aug 21- Wed:

EASY Linked List:
Palindrome Linked list:
Points: 1.use stack 2. One plus 1 and one plus 2 to get the mid point (a slow node and a fast node) 3. Compare the second half and stack pop

Linked list cycle: two node, a slow and a fast

Aug 22- TR:
BST property: binary search tree:

- left subtree of a node contains only nodes with keys less than the node's key
- Right subtree of a node contains only nodes with keys greater than the node's key
- Left and right subtree must also be a binary tree
- No duplicate node

Remove Nth node from the end:

- Still need to think!!!!!!! Not that easy

Binary heap:
Is a binary tree with 1) complete tree 2) either Min heap or Max heap

This **google question** is a combination between the property of binary tree and heap.

Balanced tree: (AVL tree) height(higher)- height(lower) <= 1

Aug23: F
Merge two sorted Link List : dummy node for indicate
Reverse linked list

Aug 24: Saturday

BST
Validate binary search tree

- Preorder, inorder, postorder

Convert sorted array to binary search tree

- Python3 use "//" 来获得整数，向前取一位，只用"/" 可能是小数。 5//2 == 2

Merge sorted array:

- **done , analysis later**
- **I use insert sort, not based on the specific question, not good enought**

DayOfYear:

Aug 25, Sunday
Summary for this week: Aug 19 - 25:
7 questions, one per day, not good enough. Should comfire do at least 2 each day.
Tree and linked list

Aug 26: beginning the **Sorting and searching:**

1. **Merge sorted array:**
2. **First bad version**

Aug27: dynamic programming

- **climbing stairs:**
- **Best time to buy and sell stock**
- **Two Amazon OA questions:** ***one is "count number of distinct pairs whose sum exist in given array"; another is "check if a binary tree is subtree of another binary tree " (not solved yet)***
- **Maximum subarray**
- **House Robber: in DP: dp store the value of**

Aug28 : design

- Min stack
- Shuffle an array

Aug 29: Friday: Math

Sep 4: Wednesday: Math

- **Power of three: % get the remainder / result**
- **Roman to integer**

Sept 5: Thursday: Others **bitwise operate**

- Number of 1 bits:
  - In python use “&(and), | (or), ^(XOR), ~(Once complement), <<(shift to left), >>(shift to right) ” to deal with binary value (can directly using on decimal)
  - Shift to left: 2<<2 == 8
    - 2 in binary is 10, left shift 2 is 1000 which is 8
    - Left shift : 向后填 0
  - Shift to right:
    - 向后删除几位
  - 2 ^ 3 = 1 explain
    - 2 = ‘10’, 3 = ‘11’ 2 ^ 3 == “01” which is 1
  - 2 in binary is “00000010”, ~2 is “11111101” which is binary for -3

二进制中， 负数以（正数）原码的补码形式表达： 补码取反再加1

- -1 二进制表达：
  - -1原码： 1000000 00000000 00000000 00000001
  - 得反码： 11111111 11111111 11111111 11111110（除符号位全取反）
  - 得补码： （再加一）11111111 11111111 11111111 11111111

Sept 9: Monday: others

- Missing number
- Reverse bits

Bit manipulation not good at this

Check on some questions on yimusanfendi.

- Leetcode 986: interval List intersections

- !!! dictionary should not has same keys.

Letter combination of a phone number: Leetcode 17. 花花酱的代码

- Backtracking algorithm
- DFS, BFS
  - DFS 里面， python use “ord(“sign”)” to get ASCII number of a sign
    cur=[‘ ’ for _ in range(len(something))] ------- to create a length of something with ‘ ’ list

BFS & DFS:

- From BFS to Djkstra algorithm (黄浩杰的视频)

Sept 10: Tuesday:

Dynamic programming & recursive quesiton

Leetcode recursion, why i see this is because I found DFS use recursion and I almost forget this one.

Recursion:

- Base case
- Recurrence relation that reduces all other cases towards the base case.

Battleship board: leetcode 419: FB interview 偶然在leetcode上看到的， 不是本次search重点

Leetcode39: *Combination Sum(Medium):*----------dynamic programming, search
A big problem is **recursion**

Sept 11: leetcode: permutation:
In a python list, if i want to get a specific index' value, can use "nums[:i]+nums[i+1:]" then ,nums[i] is the value of index "i", other are others value except index i' value

这两天几道题完了一起看一下：premutation one and two.. Combination one and two

- Permutation One:
- Permutation Two:
- Combination One:
- Combination Two:

Sept 12:
Leetcode 22 generate Parentheses: (**medium**)

- Use stack to represent parentheses.
- Beginning parenthese should more than closing parentheses.

Leetcode 784, Letter case permutation: (**easy**)

- If not alphabet, means string.lower() == string.upper(), so do not access just append

Leetcode 37: sudoku Solver: (**Hard**)

- Using hash table ???
- Bit ???

Leetcode 542: 01matrix: (**BFS,medium)**

- Traverse the matrix and store the 0 coordinate to a queue, put float('inf') to value of 1

- BFS from the 0 point
- Not understand very clearly

Leetcode 698: partition to K equal sum subsets: (medium)

- Copy an answer, not very understand

Sept 13 Friday:

Leetcode 70 Climbing stairs: DP, easy
Leetcode 746 Min cost climbing stairs: DP , easy

- Can not understand the meaning of question

Sept 16: Monday
Leetcode 303 range sum query - immutable
Leetcode 53 maximum subarray: 这几个类似的dp题的output都是一个值，（一个最大值或者一个最小值，而不是需要记录下来一个连续的），还有best time to sell stock那个题
Leetcode 198 house robber
Leetcode 213 house robber II: split the initial list into two different list and compare them:

- One is nums[1:] , which exclusive the first item
- One is nums[:-1] , which exclusive the last item

Sell stock sequences:

- Leetcode 309 Best time to buy and sell stock with cooldown
  - 滚动数组 space: O(n) → O(1)????

Sept 17 Tuesday: (**DP**)
Leetcode 740 delete and earn: reduce this question to " house robber " question

- The value of "x" will be assigned to index "x"
  - numsList [2,2,3,3,3,4] ===> pointsList [0,0,4,9,4] this will cost a lot of memory and runtime is slow. BUT, A GOOD EXAMPLE TO PRESENT HOW TO USE "self.function()"
- How to approach most of the DP problem…

Leetcode 300: longest increasing subsequence: (**LIS**)

- Different between subsequence and subarray, substring=====>>
    - Subsequence: no need to be consecutive
    - Substring: need to be consecutive
    - Subarray is in array, not in string: max stock value
- 算法珠玑上有这道题（找到这两点）：
    - 状态转移方程
    - 状态
- Suppose f [i] is the length of the increasing subsequence with tail as i. If
    - i< j and nums[i] < nums[j] =====> f [i] is the prefix of f[j]
    - Thus: f [j] = max{ f [i], 0 <= i <=j && f [i]< f [j] }+1
- *Patience sorting: has princeton_cs_course_slide*

Cracking the coding interview:

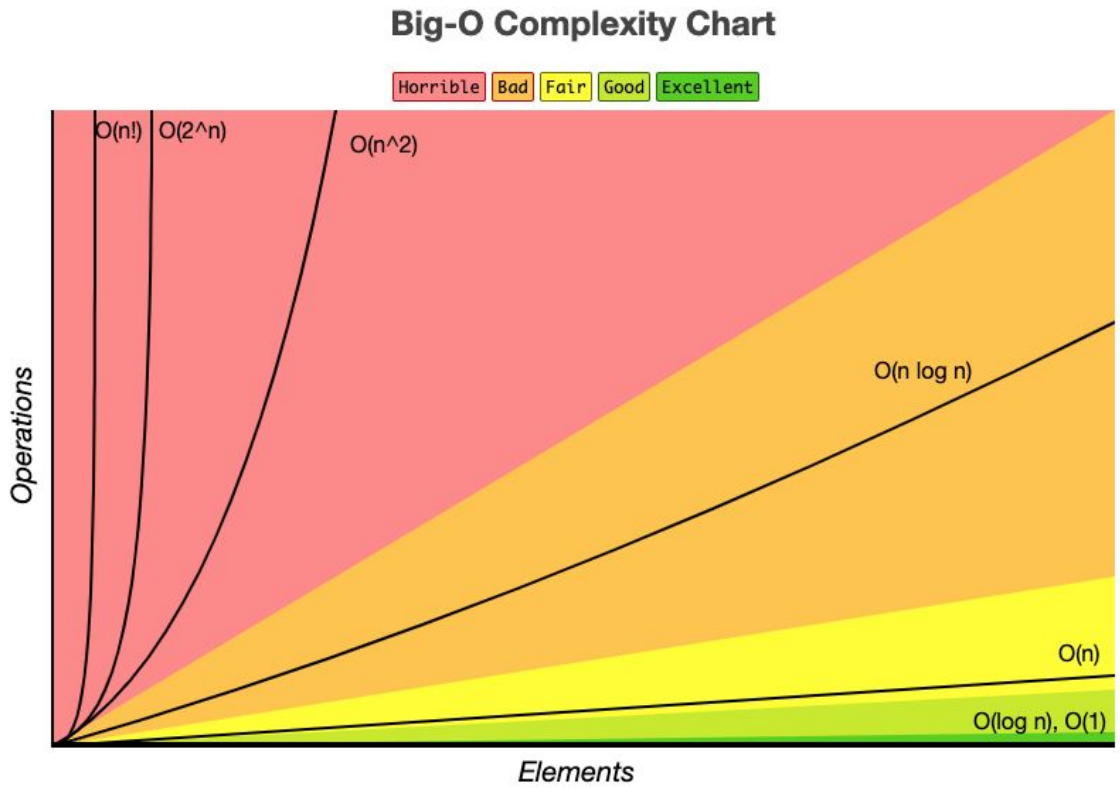
Big- O()

- Amortized time

Suppose an ArrayList(which can resizing their size by insert new element). Means, even the worse case happens every once in a while ,but once it happens, it won't happen again for so long that the cost is "amortized". So time of insertion is O(1).

- log(N)runtime. (binary search)
    - When the number of elements in the problem space get halved each time
- Recursive runtime (fib number)
    - Time complexity is branches to the power of N (branches^N)
- Multi-part algorithmL Add vs Multiply
    - Two different steps(different loop): if is in the form, " do this, then, when you all done, do that" ========= add
    - Two different steps(different loop): if is in the form " do this for each time you do that" ========= multiply

Big-O complexity chart:



Sorting big O

Array Sorting Algorithms

Algorithm	Time Complexity			Space Complexity
	Best	Average	Worst	Worst
Quicksort	Ω(n log(n))	θ(n log(n))	O(n^2)	O(log(n))
Mergesort	Ω(n log(n))	θ(n log(n))	O(n log(n))	O(n)
Timsort	Ω(n)	θ(n log(n))	O(n log(n))	O(n)
Heapsort	Ω(n log(n))	θ(n log(n))	O(n log(n))	O(1)
Bubble Sort	Ω(n)	θ(n^2)	O(n^2)	O(1)
Insertion Sort	Ω(n)	θ(n^2)	O(n^2)	O(1)
Selection Sort	Ω(n^2)	θ(n^2)	O(n^2)	O(1)
Tree Sort	Ω(n log(n))	θ(n log(n))	O(n^2)	O(n)
Shell Sort	Ω(n log(n))	θ(n(log(n))^2)	O(n(log(n))^2)	O(1)
Bucket Sort	Ω(n+k)	θ(n+k)	O(n^2)	O(n)
Radix Sort	Ω(nk)	θ(nk)	O(nk)	O(n+k)
Counting Sort	Ω(n+k)	θ(n+k)	O(n+k)	O(k)
Cubesort	Ω(n)	θ(n log(n))	O(n log(n))	O(n)

Data structure Big O

Common Data Structure Operations

Data Structure	Time Complexity								Space Complexity
	Average				Worst				Worst
	Access	Search	Insertion	Deletion	Access	Search	Insertion	Deletion	
Array	θ(1)	θ(n)	θ(n)	θ(n)	O(1)	O(n)	O(n)	O(n)	O(n)
Stack	θ(n)	θ(n)	θ(1)	θ(1)	O(n)	O(n)	O(1)	O(1)	O(n)
Queue	θ(n)	θ(n)	θ(1)	θ(1)	O(n)	O(n)	O(1)	O(1)	O(n)
Singly-Linked List	θ(n)	θ(n)	θ(1)	θ(1)	O(n)	O(n)	O(1)	O(1)	O(n)
Doubly-Linked List	θ(n)	θ(n)	θ(1)	θ(1)	O(n)	O(n)	O(1)	O(1)	O(n)
Skip List	θ(log(n))	θ(log(n))	θ(log(n))	θ(log(n))	O(n)	O(n)	O(n)	O(n)	O(n log(n))
Hash Table	N/A	θ(1)	θ(1)	θ(1)	N/A	O(n)	O(n)	O(n)	O(n)
Binary Search Tree	θ(log(n))	θ(log(n))	θ(log(n))	θ(log(n))	O(n)	O(n)	O(n)	O(n)	O(n)
Cartesian Tree	N/A	θ(log(n))	θ(log(n))	θ(log(n))	N/A	O(n)	O(n)	O(n)	O(n)
B-Tree	θ(log(n))	θ(log(n))	θ(log(n))	θ(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)
Red-Black Tree	θ(log(n))	θ(log(n))	θ(log(n))	θ(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)
Splay Tree	N/A	θ(log(n))	θ(log(n))	θ(log(n))	N/A	O(log(n))	O(log(n))	O(log(n))	O(n)
AVL Tree	θ(log(n))	θ(log(n))	θ(log(n))	θ(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)
KD Tree	θ(log(n))	θ(log(n))	θ(log(n))	θ(log(n))	O(n)	O(n)	O(n)	O(n)	O(n)

Sept 18: Wednesday DP:
Top-down memoization DP method:
Bottom - up DP method:

DP类型问题的两个性质：

1. 最优子结构：若一个问题的最优解包含其子问题的最优解
2. 子问题重叠

Sept 24: Tuesday
Leetcode 673 Number of longest increasing subsequence

- Length_list and count_list (DP)

- Segment tree: (leetcode recursive approach to segment tree) range searching problems (VisuAlgo.net)
    - Recursive method
    - The node of trees is at index 0, thus tree[0] is the root of our tree.
    - The children of tree[i] are stored at tree[2*i+1] and tree[2*i+2]
    - Lazy propagation

GRAPH:
LEETCODE 133 clone graph (queue + hashtable) hard for me to understand

- Python **deque collections module** !!!!
- https://docs.python.org/zh-cn/3/library/collections.html

Leetcode 138 copy list with random pointer

- Hashmap

Leetcode 200 number of island : bfs , not do it correctly at the first time
    - Leetc*ode 1219: path with maximum gold*

Leetcode 547: friend circles: (compare this with “number of island”)

- Huahua的视频 二维数组和一维数组在内存空间读取的区别
- 加self 的位置，可以参考200题和本体的区别， 200 是 number of island
- Visited来存储已经找到的学生关系

Leetcode 695: Max area of island:

- Why is one i can not use the “number of island”
- Has a very beautiful BFS model solution

Sept 25 Wednesday:

Leetcode 841: Keys and rooms: straightforward and easy

- Check bfs method

Leetcode 207 course schedule (**DFS, topological sorting**)

https://blog.csdn.net/lisonglisonglisong/article/details/45543451

- Khan Academy CS-algorithm course
- Will numCourses > the total number of courses in the “prerequisities” list?? Like the input is 2,[[1,3],[2,3],[4,5]] think this situation
- Topological sorting for DAG(directed acyclic Graph) 拓扑排序版本的dfs
    - !!!!只有有向无环图才有拓扑排序，有无相互关系的问题
    - 花花酱
    - Visited_stack, visiting_list, order_list
- Detect cycle in a directed graph using colors (geeksforgeeks)

leetcode 210: course schedule || ========= kind of hard for me

- **Article** about topological sorted order
- Stemming from 源于。。。
- Defaultdict import from collections
- Python nonlocal, global, local variables

TREE:

Leetcode 94: binary tree inorder traversal

- Three traversal methods, preorder, inorder and postorder binary tree traversal
- Stack

Leetcode 144: binary tree preorder traversal: medium

leetcode 145: binary tree postorder traversal: hard

1. Preorder has root as the first element to search, postorder will has it in the end, inorder will divide the left children and right children by root

Leetcode 105: Construct Binary Tree from Preorder and Inorder Traversal

Leetcode 106:Construct Binary Tree from Inorder and Postorder Traversal

https://blog.csdn.net/woxiaohahaa/article/details/52744072

Sept 26 :thursday:

- Todo :
  - Three binary tree traversal methods
    - Recursive & iterate function
  - Convert between two methods to another one

Iterative的方法都需要借助stack

1. Preorder: 比较简单，但是注意在取出stack中的首个元素后，要再检查一遍是否有此元素，再向res中append其值

```
While stack:
    Cur = stack.pop()
    If cur:
        res.append(cur.val)
        stack.append(cur.right)
        stack.append(cur.left)
```

2. Inorder:
   a. while current is not None == === while current:
3. 总结了recursion的方法在一个py文件叫做binaryTreeTraversal在大电脑上

Leetcode 100: same tree

- deque: from collections import deque (deque双向列表)
- A = [1,2,3], B = [4,5,6]. deq=deque([a,b])
- deq=([1,2,3],[4,5,6])

Leetcode 101 symmetric tree:

- recursion

======== to do!!!
Leetcode 110: **balanced binary tree: has wide use in C++ make sure the time is log(n)**

- The definition of height-balanced tree is **A binary tree in which the depth of the two subtrees of every node never differ by more than 1**
- 题外话： 把一个树变成balanced tree ???
- reduce the time from O(nlog(n)) to O(n)
- return -1 是直接返回当前的recursion 值

Sept 27 friday:
Leetcode 102: binary tree level order traversal

- BFS   should be easy
- DFS :   use level to add new list in the res list

Sept 30 Monday: national Day !!!!!!!! Happy mother land!!

Leetcode 814: binary tree pruning:
Leetcode 112: path sum

In Python:

```
stack= [ (1,2 ) ]                                  c,d = (3,4)
 a,b= stack.pop()                                  ====>> c = 3, d = 4
        ===>>   a = 1, b = 2
```

Oct 1 Tuesday:
Leetcode 124 : Binary tree maximum path sum:
(类似的是： **687 和 543**)

- Python define the lowest and biggest number: float(“inf) or float(“-inf”)
- 543
- 687

Top FB questions:
(23, 31, 42 ,56, 65, 67, 71, 76, 88, 98, 124,125, 133, 158, 161, 173, 199, 211, 238, 253, 269, 273, 278, 282, 297,301, 304, 314, 340, 349, 350, 415, 416, 426, 438, 523, 543, 560, 621,680, 721, 785, 825, 863, 896, 953, 958, 973, 986, 1027)

Leetcode 23:
1.brute force
2. Heap queue (heapq in python) is used to represent a priority queue.

- Import heapq
- **Heap Push** a value (A) into a queue "a = [ ]" : heapq.heappush(a, A), means push value A into a
- Each time use heapq.heappop(a) will **pop** the **smallest** element.
- Dummy = ListNode(-1)

Oct 2 wednesday:

Leetcode 31: next permutation

- Iterate from the end and find the first one A[i]>A[i-1]
- Find the value from A[i-1] to A[end] that is just bigger than the A[i-1]
- Swap just bigger one with A[i-1]
- Sort the rest from A [i] to A [end] ////### or just reverse since this is decrease order

Example is: [1,2,3,5,4] ⇒ [1,2,4,3,5] ⇒ [1,2,4,5,3] ⇒ [1,2,5,3,4]

Leetcode 42: Trapping Rain Water

- **Two pointers**
- Same problem is "leetcode 11: container with most water", also use two pointers and move the pointer of the low height one
- The difference and difficulty of LC 42 is
  - the in LC 11, the first number is actually at index 1, while in LC 42 there is a number represents the first column
  - The loading or water in LC 42 is not simple height*bottom

###redo Oct 24 2019
有点迷糊

Leetcode 56: Merge Intervals (difficult for me to understand: update oct9, not difficult to understand)

- sort the list based on the first element by using lambda :
  - Inter = [[ ], [ ], [ ], [ ], [ ], [ ]]

- Inter.sort(key = lambda interval: interval [0])
- Sort the tuple based on the first element

Redo: Oct 24

- **Leetcode 253 meeting rooms**

Leetcode 65: valid Number : (TBD) float

Leetcode 67: add binary :

- **Python 反转一个string : string[::-1] ⇒ "gnirts"**
- **得到一个list，全是0， 长度为 len(a) :**
  - **res= [0 for _ in range(len(a)+1)]**
- **Python function divmod(8,3) = (2,2) (div, mod)**

Leetcode 71: simplify path :

- **看到这种来来回回，增增删删的题 一般就是 栈**
- **Absolute path VS relative path in Linux/Unix**

When define the stack, if use " stack = list() " will use less runtime and memory than simply use " stack = [ ] " ???

Leetcode 76: minimum window substring: (hard !!!!!!!)

- **From collections import counter**
- **Use counter to counter each item and it number**
  - **myList = [1,1,2,3,4,4,5,5,5]**
  - **counter(myList) = ({1:2, 2:1, 3:1, 4:2, 5:3})**
  - **counter(myList).keys() = [1,2,3,4,5]**
  - **counter(myList).values = [2,1,1,2,3]**

Leetcode 88: Merge sorted array (easy)

- **Interesting sorting method**

Leetcode 98: validate binary search tree:

BST definition: left child should less than his parent, right child should bigger than parent

- **Recursion**
- **Inorder traversal**

Oct 3. Thursday:

Leetcode 124: binary tree maximum path sum: (hard) 不是很明白

- recursion

leetcode 125: valid palindrome:

- newStr = “”.join(newString)
- return newStr == newStr[::-1]

Leetcode 133 clone graph: deep clone 之前做过，还是不会， queue+hashmap

- Simple graph: no repeated edges, and no self-loop in the graph
- Use set() to store the visited node, since this is an undirected graph
- Queue
- Map : 1, store the new and old node’ relationship，2 tell if already get the node into our queue
  ========能想明白过程，体现到代码里还是有问题=====

Leetcode 158: read N characters given read 4 || (hard)

Leetcode 161: One edited distance

- 不是很难，因为顺序也需要一样，顺序不同也不行

Leetcode 173 binary search tree iterator: (system design???)

- Inorder traversal to get the ascending list
- Not hard, but need to have a comprehensive idea

L**eetcode 199 Binary tree right side view:** (不太会啊)

- Python dictionary setdefault ( )
- DFS

Leetcode 211 Add and search word - data structure design: (hard for me, not finished)

Oct 4 friday:

DP

- **存在型动态规划**
  - **确定状态**

        - **研究最优策略的最后一步**
        - **化为子问题**
    - **转移方程**
        - **根据子问题定义直接得到**
    - **初始条件/边界情况**
        - **细心**
    - **计算顺序**
        - **利用之前的计算结果**

Leetcode 322 coin change: **dynamic programming,** bottom-up, long time ,need to update
Watching the 9C DP biblil
Leetcode 62 unique paths:

- Easy way to define a matrix:(two degree)
- dp= [[1 for _ in range(3)] for _ in range(4)]
    - dp= [[1,1,1], [1,1,1], [1,1,1], [1,1,1]]
    - 特别要注意在matrix里面， row and colunm的值

Oct 5: saturday
Leetcode jump game

- Greedy algorithm

Sunday oct6:
Weekly contest 157: not finish one question

- Play with chips(DONE)
- longest arithmetic subsequence of given difference (Not know, dp??) ⇒ collections (dictionary)
- Path with maximum gold
    - VERY careful about the “self, ” should need to check it
    - Leetcode 200: number of island
- count vowels permutation

Leetcode 238 product of array except self:

- No division
- Follow up: O(1) time

Leetcode 253 meeting rooms:

- **Leetcode 56 merge intervals**

- **Use intervals.sort(key = lambda x: x[0])==> sort based on first element**

In the heap(heapq.heappush(), heapq.heappop(),), store the first one to the heap and give a ans with length null, if the subsequence' first is bigger than the end time of last one, can share the same room, otherwise push a new end time into the ans
Finally return the length of ans[]..... time is O(NlogN) for sorting and space is O(N) for heap

Leetcode 269 Alien dictionary: (graph, topological) ======> HARD, and too hard for me

- Check LC 207, course schedule

Leetcode 273 integer to english word: (math, string, Hard) 真麻烦

Leetcode 278 first bad version: (easy, binary search)

- **5%2 = 1 ***** 5/2 = 2.5 ***** 5//2= 2**

Leetcode 282 Expression add operators (Divide and conquer, too hard)

- **Binary operators : "+ , - , *, "**

Leetcode 297 Serialize and Deserialize Binary Tree (Tree, design, hard)
Leetcode 301 remove invalid parentheses (DFS,BFS, hard)

Monday Oct 7

Leetcode 304 : Range Sum Query 2D - Immutable

- Answer is from id "lost world 21" FB engineer

Leetcode 314: Binary tree vertical traversal

- **Leetcode 987: vertical Traversal (root)**
- defaultdict() , belong to the module "collections"
- defaultdict() is simpler and faster than an equivalent technique using dict.setdefault()
- The result of defaultdict() and setdefault() is same.
- example :
  - S = [('yellow', 1), ('blue',2), ('yellow',3'),('blue',4),('red',1)]
    d=collections.defaultdict(list) ## list is the default_factory item
    For k, v, in S:
    d[k].append(v)

  Print (list(d.items()))
  ⇒ [('blue',[2,4]), ('red',[1]), ('yellow', [1,3])

Leetcode 340 longest substring with at most K distinct characters

- Sliding window + hashmap
- hashmap=defaultdict()

Leetcode 349 intersection of two arrays :

Use set to get non-duplicated set1 and set2 then loop through the short one to find if there is one in the longer one.

Follow up: (from leetcoder "hasanseirafi")

- Under these constrains:
    - O(n) time ad O(1) space (need to ask if the lists are sorted)

Leetcode 350 intersection of two arrays ||: easy

Leetcode 415 add strings : recursive method

Leetcode 416 partition equal subset sum

Dp problem, traditional knapsack problem

Backpack: A1,A2...An-1 sum is target ⇒ backpack problem 还得好好看一遍

一个重点是直接开一维的list， 然后，看代码

Leetcode 426 Convert Binary Search Tree to Sorted Doubly Linked List:

- circular doubly linked list

Leetcode 438 find all anagrams in a string:

- Anagram example: abc ⇒ acb, bca, bac, cab,cba
- Sliding-window
- Python ord() function: return the unicode

Oct 8: Tuesday

KMP algorithm for string searching:
https://medium.com/@giri_sh/string-matching-kmp-algorithm-27c182efa387

Leetcode 523 continuous subarray sum:

- 计算前几个数的和：
  - acc=[1,2,3,4]
    - num=acc[:]
    - for i in range(1, len(acc)):
      num[i] = num[i-1]+ acc[i]

num = [1,3,6,10]

Leetcode 543 diameter of binary tree:

- Recursion , review later

Leetcode 560 subarray sum equals K:

- dictionary

Leetcode 621 task scheduler : array, greedy, queue

- Two way to count an object in dictionary
  - Task = {"A","A","A","B","B"}
    1. For k in task:
       Fre[k] = Fre.get(k,0)+1 (Fre.get("A") = 3)
    2. A = Task.count("A") = 3

Leetcode 680 valid palindrome ||:

- While find the different value, skip it and test the rest of them

Leetcode 721 accounts merge: DFS, union find

Create a graph: graph = collections.defaultdict(set)

Redo: Oct 24 2019

- Give each account an ID, based on the index of it within the list of account
- Build emails-accounts-map, map an email to its account
- DFS

Leetcode 785 Is graph bipartite: DFS, BFS, graph

Graph Bipartite 二分图 hard for me

Leetcode 825 friends of appropriate ages:

Create a length of 121 count list and store the count of each age in the index of age.

ageA,countA in enumerate(count)

ageB,countB in enumerate(count)

Leetcode 863 All nodes distance K in binary Tree: 不会

Leetcode 896 Monotonic(单调的，无变化的) array:

First compare the A[0] with A[-1]

Leetcode 953 verifying an alien dictionary:

order_list = {i:c for c, i in enumerate(order)} to get the order of each character

Leetcode 958 check completeness of a binary tree:

- BFS

Leetcode 973 k closest points to origin: heap, D%C, sort

points.sort(key = lambda P: P[0]**2 + P[1]**2)

Leetcode 986 interval list intersections: two points

Leetcode 1027 longest arithmetic sequence: DP

Oct 14 Monday:
A new start after the FB interview at Oct 11:

1 review of the FB question:
Advanced(?)- task schedule: a process scheduler , two same process has cooldown time, return the length of process in order.
Eg: tasks = [A,A,A,B,B,A,A], cooldown = 2
Return tasks = [A,_,_,A,B,_,_,B,A,_,_,A], length = 12

Oct 16 Wednesday
Leetcode group-study-day:
LC953 verifying an Alien dictionary(easy):

- Get the order-index of the order (enumerate)

LC 301 remove invalid parentheses:

LC 273: integer to english words (hard)

LC 238: product of array except self

- Get left of the index and right of index
- get product of left and right

Oct 17 Thursday: leetcode for friday group-study

Leetcode 621 task scheduler: use java for this time, try later , (what if return with tasks in order)
LC 253: meeting rooms:

- Sort based on the first value: intervals.sort(key = lambda x : x[0])

Oct 19 Saturday: weekly contest 159

Oct 20 sunday: for monday group study:
Lc438 find all anagrams in a string:

ord(“a”) to get ASCII of “a”

How to count each character number of a string **p = “abcde**” of the 26 alphabet:

```
Ref = [0] * 26
For c in p:
    res[ord(c) - ord(“a”)] += 1
```

Hash map + sliding window

LC278: first bad version:

binary search, point is how to get the mid, (s+e)//2 or (e-s)//2+s

LC 785: is graph bipartite? (DFS,BFS, graph)

collections.deque([])

Deque is double-ended-queue, use append and pop to take action from both sides

Function has : “ pop(), popleft(), append(), appendleft()

The reason using deque is when we want to use list as the queue(FIFO) we can only use it in deque, otherwise it will be the same as stack

https://docs.python.org/2/tutorial/datastructures.html

LC 340: longest substring with at Most k distinct characters

Key : char, val: freq 注意命名的名称

https://thispointer.com/different-ways-to-remove-a-key-from-dictionary-in-python/

Different way to remove key in dict

Delete a key in dict by using function ***del*** :

del hashmap[s[del_key]] ## this is ok for both normal dict() and defaultdict()

Difference between using defaultdict() and dict()

hashmap.values() / hashmap.keys()

LC 199 binary tree right side view

- **One way is using level(BFS) traversal to get each line and return last one**

LC 124 binary tree Maximum path sum (LC687, LC 543)

- recursion

Oct21 Monday:
Binary tree
LC107 binary tree level order traversal ||

Rabin-karp algorithm

Oct22 Tuesday:
Huahua recursion:

Lc726 number of atoms (hard, string , recursion):

Global variable for all functions

Recursion python version 不会啊，完了解决！！！

LC736 Parse Lisp Expression: （lisp 口齿不清）(hard)
Recursion

LC 394: decode string: 和上面那题很像
One stack to store number and prestring

LC 856: score parentheses:

花花的recursion方法很巧妙，还有一个方法是counting， 花花的，完了check

Oct 23 Wednesday:
What is the purpose of the word “self” in python:

- **Class 类包含**
  - **类的属性：类中所涉及的变量**
  - **类的方法：类中函数**
- **在创建实例类的时候需要加入self到类的方法中。**
- **类似于C++中的self指针和java，C#中的this 参数**

Oct 24 Thursday:
LC34 find first and last position of an element in a sorted Array

Oct 25 Friday:
Nothing

Oct 26 Saturday 160 contest: 一道没做出来

Oct 28 Monday:
LC 215 kth largest element in an Array

Quick sort----

- O(nlogn) as average running time ,O(n^2)--worst running time
- In-place space
- This is better to randomly choose the pivot rather then the last one
- partition(A, start, end) ⇒ A = [7,2,1,6,8,5,3,4]
    - Pivot = A[end]
    - pIndex = A[start]
    - For i from start to end-1:
        - If a[i] <= pivot: < or <= 都可以
            - swap[A[i], A[pIndex]]
            - pIndex = pIndex +1

    swap[A[pIndex], A[end]]

Merge sort

- O(nlogn)-- worst case running time
- O(n) space complexity (not in-place)

QUICK SELECT / QUICK SORT

LC236 Lowest Common ancestor of binary Tree (recursive理解上还是有点困难)

- Binary search tree(BST) and binary tree LC 235 is BST , easy
- 二叉树最近公共祖先详解
- Follow up ⇒ 带有父节点信息的二叉树呢？

LC319 bulb switcher (math):

- **Return int(sqrt(n))**

LC139 word break(DP, not understand)

OCT 29 Tuesday:

LC 935 Knight Dialer : (DP)

- **dp[k][i][j] 表示 numbers of ways to move to (i,j) after k steps.**
- **Two dp array, one is temp to update the original dp**
- **Sum of all values in a matrix:**
    - **Return sum(map(sum, matrix))**

(LC 688 Knight probability in chessboard)

LC1094 car pooling

A very smart way is to label each number of people in and out at each location. Just like put the number on a x axis.

For example the trips = [[2,1,5], [3,5,7]], capacity == 5

So this line would be like +2, +3 -3 and we can loop to capacity -= all number -2

LC270 closest binary search tree value:

对于 min((b,a), key = lambda x : abs(target-x)) 有点不理解，IMAC上有，

```
def iterativeClosestValue(root,target):
    res= root.val
    While root:
        If abs(root.val- target) < (res-target):
            Res = root.val
        root= root.left if root < target else: root.right
    Return res
```

LC 50 Pow (x, n)

N % 2 is equal to N & 1

N //=2 is equal to N >>=1

LC987 vertical order traversal of a binary tree (hash table , tree)

Use defaultdict to store the **coordinates (X , Y)**

Seen = collections.defaultdict(lambda: collections.defaultdict(list))

Seen[0][0].append("yes"), seen[0][1].append("NOOO")

Seen.items() = [(0, defaultdict(<type 'list'>, {0:["yes"], 1: ["NOOO"]})]

Print seen[0][0] = ["yes"]

Difference between the list.append() and list.extend()

- extend() : iterates over its argument and add each element to the list and extend the list. While append() will see a new list as an object and append the new list as a whole at the end of list
- My_list = ['a','b']
- my_list.append('geek') ⇒ ['a','b','geek']
- my_list.extend('geek') ⇒ ['a', 'b','g','e','e','k'']

LC 416 partition equal subset sum: (DP, 0/1 knapsack problem)

dp[i][j] : where we can sum to j by using first i numbers

dp[i][j] = true if dp[i-1][j-num(i)]

Use two dimension array first, then I need to change it to one dimension

OCT 30 Wednesday:

LC 398 Random pick index

- Reservoir sampling(水塘抽样问题)

OCT 31 Thursday: Nothing

November 3:

LC 863 All nodes distance K in binary tree: (上次做不会，这次知道意思了)

- Consider this binary tree as a undirected graph, then find the target, traversal from the target to all the direction and get the Kth node. (Build a Graph + DFS/BFS)
- Create the undirected graph also need the dfs

LC 257 Binary tree path:

- **Easy, but the point is the output are string, so need to use**
    - string +=str(root.val)

LC 540 single element in a sorted array:

- **Brute force is not right, check it out later , omit the note that use it in O(n) and O(1)**
- Binary search

LC 91 Decode ways: (花花有视频)

1. Recursion + memorization
2. Like stair jump problem, one step(one number) or two steps(two numbers)

    - **Similar question:**
        - 62. Unique Paths
        - 70. Climbing Stairs
        - 509. Fibonacci Number
    - LC 639 Decoded ways || (hard)

LC33 search in rotated sorted array

- Find the pivot. Then use binary search.
- One if and 3 elif and one else to decide the location of target, notice is do not use right = len(nums) , left = 0. Mid = (left + right) // 2 while left < right(not <=)

Nov 5 Tuesday:

LC 670 maximum swap:

1. Find the digit that breaks the decreasing trend, careful about then range is to len(s)-1. The index is pivot
2. Find max in s[pivot:] , its index is pivot2
3. From pivot to 0, find the max smaller than maxNum (before pivot is decreasing, so left one is bigger)
4. switch

LC 381 insert delete getRandom O(1) - duplicates allowed (hard):

- LC 380 Similar question: insert delete getRandom O(1) (medium)
- HARD, need to check it out later

Nov 6 Wednesday:

LC 358 rearrange string K distance(hard)

collections.Count(word)

- **LC 621 task schedule**
- **LC 767 reorganized string**

LC 689 Maximum sum of 3 Non-overlapping subarrays (hard):

好好看提交的注释，要再看一下

Nov 7 Thursday:

LC463 island perimeter:

- Perimeter = number of island - number of connections between islands

LC 1004 Max consecutive Ones |||

- Convert this question into “find the longest subarray has most K 0 ”
  Use two pointers , very traditional way
- Similarly questions

1234. Replace the Substring for Balanced String
1004. Max Consecutive Ones III
992. Subarrays with K Different Integers
904. Fruit Into Baskets

LC 78 subsets

- **Combination and permutation**

LC 29 divide two integers: buhui

LC 767 Reorganize String:

- Use heapq: priority queue

Nov 10 Sunday:

LC32. Longest Valid Parentheses(hard):

- Use stack
- Use leftmost to record the valid start point
- Can also use DP

LC 75 sort colors: (two points, array, sort):

- Two point is straight, three parameters, cur, left, right
- If nums[cur] == 0, swap with left
- If nums[cur] == 2, swap with right
- Otherwise just cur++

LC 312 Burst Balloons(hard) .. DP:

- Note said $0 \le n \le 500$, $0 \le nums[i] \le 100$ means you can not use sorting only DP

Nov 11 Monday

LC 341. Flatten Nested List Iterator:

- **Nested List Weight Sum**

LC 449 serialize and deserialize BST

- Serialization is the process of converting a data structure or object into a sequence of bits so that it can be stored in a file or memory buffer, or transmitted across a network connection link to be reconstructed later in the same or another computer environment

Relationship between preorder and inorder traversal:

- https://www.geeksforgeeks.org/construct-tree-from-given-inorder-and-preorder-traversal/
- map function :
  - data = '1 2 3 4 5'
  - res = map(int, data.split())
  - print (res) [1,2,3,4,5]

LC 105 and LC 106 可以看一下

LC 528 random pick with weight:

- Similar question: random pick index; random pick with blacklist; random point in Non-overlapping rectangles
- Binary search carefully
  - Use mid = (L+R)//2
  - L < R: (not <=, careful if only has one variable inside,)
  - Return L (not R)

LC 477 total hamming distance:

- **Hamming distance between two integers is the number of positions at which the corresponding bits are different**
- **Total hamming distance for i-th bit = (the number of zeros in the i-th position) * (the number of ones in the i-th position)**
- Number of 1 bits:
  - In python use “&(and), | (or), ^(XOR), ~(Once complement), <<(shift to left), >>(shift to right) ” to deal with binary value (can directly using on decimal)

- Shift to left: 2<<2 == 8
  - 2 in binary is 10, left shift 2 is 1000 which is 8
  - Left shift : 向后填 0
- Shift to right:
  - 向后删除几位
- 2 ^ 3 = 1 explain
  - 2 = '10', 3 = '11' 2 ^ 3 == "01" which is 1
- 2 in binary is "00000010", ~2 is "11111101" which is binary for -3

二进制中，负数以（正数）原码的补码形式表达： 补码取反再加1

- -1 二进制表达：
  - -1原码： 1000000 00000000 00000000 00000001
  - 得反码： 11111111 11111111 11111111 11111110（除符号位全取反）
  - 得补码：（再加一）11111111 11111111 11111111 11111111

LC394 decode string(did it before)

- **Using stack**

LC 151 reverse words in a string(did it before):

- **Carefully about the duplicate space between two words**
- **不太会**

LC 53 maximum subarray : DP

Nov 12 Tuesday:

LC 37 sudoku solver (hard) hashtable, backtracking:

- Check if it is valid. ⇒ has this value before or not

LC13 roman to integer:

- Similar: integer to roman
  - Minus from biggest value. Which is "M== 1000" to "I == 1"

LC 865 smallest subtree with all the deepest nodes

- **LC 103 binary tree level order traversal**
- **LC 236 lowest common ancestor of a binary tree**

- **Recursion, compare the depth of children**

LC 295 find median from data stream (hard):

- **Design question, use heap , use two heap, should check later**
- **Think about traditional way that i submit**
- **Carefully this is an ordered integer list**
- **Two follow up**

LC 468 validate IP address:

IPV4 and IPV6

- **Word study: pursue simplicty**
-

DFS/BFS binary tree

Nov 13 Wednesday:

LC 311 sparse Matrix Multiplication:

- **locked**
- **Point of sparse**

Nov 14 Thursday :

LC 200 number of islands:(did it before, java for this time)

- Use sink, find a “1” sink it to 0 and then to dfs
- This time code in java

November 16 , saturday

LC 51 N-Queues(hard):

- **Backtracking**
- **N-Queues || . Grid illumination**
- **花花酱**
  - **表示对角线：正对角线，反对角线各（n*2-1）条**

LC 207 course scheduler (之前做过)

- **Do it in java for this time**

November 17 Sunday:
LC 36 valid sudoku:

- Use a set to store each row , col, and small block , when duplicate, return false
- Java 中的Object 是包装类型，Set<Object> set = new HashSet<>();

LC 81 Search in Rotated Sorted Array II

- Similar (LC 33 Search in Rotated Sorted Array: has no duplicate)
- Has duplicate
- Check it later

LC 109 Convert Sorted List to Binary Search Tree:

- This is in a list node
- recursion
- Hard for me to understand explicit

November 8. Monday:

LC. 3. Longest Substring Without Repeating Characters

- Use a set to store value from a sliding window
- Sliding window has two point, l,r starting from left most
- If set has no s.charAt**(l),** add it to ans , r ++
- If not , remove left value, left ++

LC 428 serialize and deserialize N-ary tree(locked):

- **297 serialize and deserialize binary tree:(之前有一道类似的题)**
- **现在理解的有点难**

LC 46 permutations

- https://leetcode.com/problems/permutations/discuss/18239/A-general-approach-to-backtracking-questions-in-Java-(Subsets-Permutations-Combination-Sum-Palindrome-Partioning)

November 19: Tuesday:

LC 339 Nested list weight sum (locked)

- Lintcode 上做了一下，dfs, use stack to store, if is not a integer, change it
- to a list then do the recursion.

LC 25 reverse Nodes in k-group(hard): linked list

- Dummy node : how to create dummy node
- LC 206 reverse linked list: use a prev to store previous node , temp to help to store the next
- Python 中一行实现多个变量的赋值转换
- 好好在看看，变换list的point的顺序不能变，很重要

LC 212 Word search ||(hard):

- **Backtracking, Trie: a tree data structure, also called “prefix tree”**
- **Although balanced trees and hash tables give us the way to search for a word in dataset of string, why still need trie? Table has O(1) time for looking for key, but not efficient in the following operations:**
  - **Finding all keys with common prefix**
  - **Enumerating a dataset of string in lexicographical order**
  - **When hash table increase in size, there are lots of hash collisions and search time could deteriorate to O(n) where n is the number of keys inserted.**
- ***LC 208 implement trie (prefix tree)***

LC 8 string to integer (atoi)

非常无聊的一道题， string

LC 240 search a 2D matrix ||

- **Just binary search**

LC 218 the skyline problem(hard):

- **Similar question: LC 699 falling squares(Hard)**
- **不会，太难**
- **https://leetcode.com/problems/the-skyline-problem/discuss/61194/108-ms-17-lines-body-explained**

- **https://briangordon.github.io/2014/08/the-skyline-problem.html**

LC112 path sum：easy

November 20 2019. Wednesday:
LC 146 LRU catch

- Design problem
- LFU cache; design in-memory file system; design compressed string iterator
- Hard for me for this kind of design problem

LC 480 sliding window median (Hard);

- **LC 295 finding median from data stream**
  - **Use two heaps in following way**
    - **A max-heap to store the smaller half of the input numbers**
    - **A min-heap to store the larger half of the input numbers**
  - **heapq.heappop(nums, target) / heapq.heappush(nums, target)**
  - **In python, heap is defaulted as a min heap, means heappop() will pop out the smallest**

Nov 22 Friday

Chengfei wang FB interview question

1. Dot product. How to use a new data store to store the vertex with lot of consecutive zero:
   a. EG: vec1 = [1,2,3,0,0,0,0,0,4,5]:
      i. Use index-value pair to store without store the 0: vec1' = [(0,1),(1,2),(2,3),(,8,4),(9,5)]
2. How to use a new data store to vertex with lot of consecutive duplication:
   a. EG: vec1 = [1,2,2,2,2,2,2,3,4,5,5,5]
      i. Use start-index, end-index, value to store the vector: vec1' = [(0,0,1),(1,6,2),(7,7,3),(8,8,4),(9,11,5)]
3. Use preorder and inorder to reconstruct the tree
4. Move zero to the back (LC 283)

Tuesday Nov 26:
Nothing

Wednesday Nov 27:
Nothing

December 2, Monday
LC 277 find the celebrity:

- Not difficult.

LC 973 K closest points to origin:

- **So called K-th problem (top discussion)**
  - **First one is sort all points, not sufficient**
  - **Second is based on MAX-HEAP with size K, once this size of heap is greater than K, we are supposed to extract one from the max heap to ensure the size of heap is always K.**
    - **Java -> lambda expression**
    - **Java PriorityQueue: poll and offer**
  - **Last one is Quick sort **
    - **Check quick sort on the book**
    - **Java.util.Arrays.copyOfRange()**

LC560 Subarray sum equals K: (very useful)

- Cummulative sum: sum[i] store cummulative sum upto index i-1, thus, sum[i:j] = sum[j+1] - sum[i]
  - Sum[0] = 0
    - For i in 1 to nums.length:
      - Sum[i] = sum[i-1] + nums[i-1]
- Use HashMap: in java map.put():
  // java getOrDefault: if has key, get value, otherwise is 0

LC 158 read N characters given read4 || - call multiple times (hard, confused)

- Easy 157 read N characters given read4

LC680 valid palindrome(回文) ||:

- Use a recursive to test once one is not same.
- Two pointers, from start and from end

LC 986 interval List Intersections

- In java use ArrayList<>, in java, the size of array cannot be modified (if you want to add or remove elements to/from an array, you have to create a new one) while elements can be added and removed from an ArrayList
- ArrayList need to convert into int[ ][ ]

LC 896 Monotonic Array(easy)

- Use java this time

LC 415 add strings(easy)(very useful a question)

- Convert string to int (both are numbers)
- Use java, StringBuilder (why stringbuilder is better than string)

LC 98 validate binary search tree (DFS, tree)

Dec 7 saturday:
D&C study:
LC 240: Search a 2D matrix ||: (check later)

- Binary search: not really understand.
- D&C:not understand
- What if not return yes or no, but need to get the index of value if exist? DP
- Method 4: search space reduction

Quick sort: average time complexity of quick sort is O(NlogN)

Dec 8 Sunday:
D&C study

- **Master theorem: an advanced technique to estimate the time complexity of certain recursive algorithm.**

Recursion problem:

- **Divide and conquer(DFS, merge sort, quick sort...)**
    - Maximum-subarray problem
        - D & V function need to carefully about the “mid + 1”
        - LC 643 Maximum Average Subarray I

        - LC 53 Maximum Subarray
- **Backtracking**
    - **N- Queen puzzle N-queens ||:**
    - **Robot room cleaner**

```
def backtrack(candidate):
    if find_solution(candidate):
    output(candidate)
    return
    for next_candidate in list_of_candidates:
        if is_valid(next_candidate):
            place(next_candidates)
            backtrack(next_candidate)
            remove(next_candidate)   //backtrack
```

Dec 11. Wednesday: LC explore
LC 498 Robot room cleaner: (hard)

- Constrained programming: means need to mark visited cell
- Backtracking

Dec13 friday:
LC Sudoku Solver: carefully where to find the c; also careful to check small square(正方形), rectangle(矩形)
LC combination:

- k-len(path) > n - index + 1 is to optimize the speed, since if the rest of nums is less than what we need as k-len(path), it won't make a new res

Unfold recursion:

1. We use stack or queue data structure within the function, to replace the role of the system call stack.

D&C with DP:
Dec 15 Sunday:

LC228 summary Ranges (array):

LC 121 best time to buy and sell stock: very popular and must need to know how

- Use a node to store the minPrice
- Can **use DP**
- Can reduce this question to LC **53 maximum subarray(also use DP)**

LC 791 Custom Sort String:

- Use hashmap to store the frequency of string
- Time complexity in count and get the count is constant so is not n to the power of 2

LC 674 longest continuous increasing subsequence: easy, very popular

Dec 16. Monday:

LC 285 inorder successor in BST:

- Should use the property of BST: left child is smaller and right child is bigger

LC 414 Third maximum number: boring

- Heap, priorityqueue

LC 188 Best time to buy and sell stock IV (Hard,DP)

- **LC 122 Best time to buy and sell stock || (multiple times of transactions)**
    - **Sum up all the ascending value**

Dec 17 Tuesday:

LC 188 IV(Hard, DP):

- **t: 0->j-1 max(prices[j]-prices[t] + dp[i-1][t-1])**
    - **== prices[j] + t:0->j-1 max(dp[i-1][t-1] - prices[t])**
- **高票区的答案：java那个**
- **State-machine**
- **用滚动数组降维. Rotating dp array**

LC114 flatten binary tree to linked list:

- **In java stack, peek() and ?? pop() difference ----- this method is not good, from Youtube**
- **Dfs, search right first, then left: prev, left, 那段像是赋值的过程**

LC 145 Binary Tree Postorder traversal(基础题)

Dec 18 Wednesday:

Dec 21 Saturday:

LC 202 happy number

- First, as long as there is a loop after running the process continuously, the number is not a happy number
- Need to justify that a non-happy number will definitely generate a loop
    - Checking if a number is in a HastSet takes O(1) time, whereas for the other data structure it takes O(n) time

- **How to get each digit's power of 2 sumup:**
    - While (n > 0):
        - Int d = n % 10;
        - n = n / 10;
        - totalSum += d * d

      Time complexity == O(logn)
- **In python, get the module and division: n, digit = divmod(n,10)**
  n will be divide, digit is module

  If use same step in Java, TLE

- **Floyd's cycle - finding algorithm:**
    - **LC 141 Linked list cycle**
    - **fastRunner and slowRunner**

LC 235 Lowest common ancestor of a binary search tree:

LC 41 first missing positive:

- Do in constance space and O(n) time

LC 494 target sum(medium):

LC 137 single number ||:(hard for me to understand bitwise)

- Would be easy to use extra space by implementing set or hashmap
- 可以用求和的方法来解决，这个思路也可推广到任意参数为k,p问题：输入数组每个元素都出现了k次，只有一个只出现了p次，求那个单独的只出现p次的数: can use (^: XOR) 亦或操作：
  - **1 ^ 0 = 1**
  - **1 ^ 1 = 0**
  - http://liadbiz.github.io/leetcode-single-number-problems-summary/
- Use hashMap in Java:
  - Initial: **HashMap<Integer, Integer> hashmap = new HashMap<>();**
  - hashmap.getOrDefault(key, 0) ⇒ get the value of key if not exit return 0
  - hashmap.keySet() ⇒ get the key in hashmap
- Linear time, without extra memory
- Bitwise operation:
  - & :and
  - | : or
  - ^ : XOR
  - ~ : NOT, invert all bits
- **In a constance space: *bitwise operation:***
  - ***https://leetcode.com/problems/single-number-ii/discuss/43295/Detailed-explanation-and-generalization-of-the-bitwise-operation-method-for-single-numbers***

Dec 22 Sunday:

LC 430 flatten a multilevel doubly linked list:(DFS, linked list)

- Implement in git branching
- Pre-order DFS traversal
- Use sentinel node
  - LC **146 LRC cache**

- Should make a copy of the curr.next pointer before the first recursive call of dfs(curr, curr.child)
- Only change the pointer ,not return a new result like queue or something

LC 146 LRC cache:

LC 242 valid anagram(颠倒字母顺序构成的单词) rearrangement:

- Hashtable in java
- Int counter = new int[26]
- counter[s.charAt(i) - 'a']++

LC 230 Kth smallest element in a BST:

1. **Three facts to know about BST (write code in big Mac Tree file):**
    a. **Inorder traversal of BST is an array sorted in the ascending order**
    b. **Successor = "after node", i.e. the next node, or the smallest node after the current one (right one more step, then get as left as you can)**
    c. **Predecessor = "before node", i.e. the previous node, or the largest node before the current one**
- Search tree: dfs / bfs
- Use dfs can get the increasing order of BST
    - Recursion
    - Iteration: stack, done, current
- **Follow up: what if the BST is modified(insert/ delete operations) often and you need to find the kth smallest frequently?**
    - **Insert and delete in BST**
    - ***LRU cache***
- **Follow up is a design question: A structure with a BST inside optimises the following operations:**
    - **Search: hugh BST advantage is search for arbitrary in O(logN)**
        -
    - **Insert**
        - **LC 701 insert into a BST**
            - **Iteration**
            - **recursion**

    - **Delete**
        - **LC 450 delete node in a BST**
    - **Find kth smallest**

Dec 23 Monday:

LC 567 permutation in string

- **Two** pointers, sliding window
- Permutation and combination
    - In java, HashMap getOrDefault(key, 0)==> if can find the key, get value in hashmap, otherwise return default value(in this case is 0)
    - In java, HashMap keySet(): the method returns a set having the keys of the hashmap, so if want to loop the key in hashmap, use this: for(char key: map.keySet()){...}
    - In java, use put to add new key and value into hashmap
- Use debugger to check the last java version
- In python, use collections to implement counter , hashmap
    - **In python also can use ord(x) - ord('a') to get the value of x**

LC 49 group anagrams:

- Python tuple: unchangeable collection in order
- Sorted string should be same if they are same

LC 102 binary tree level order traversal (did before)

LC 977 squares of a sorted Array

- In java, sort a array is : Arrays.sort(arrayName)
- Use two pointers:
    - Use two pointer to separate negative number and positive, then use **merge sort** to compare them and sort them

LC 127 word ladder (BFS, graph!, hard for me !!!, but finally understand):

- **Boil down: 归结为**
- **huahua jiang**
- Pre-processing: find generic/ intermediate states for wordList
- **Bidirectional BFS :**
    - BFS: use queue

- IDS dfs
- Disjasik search
- LC word ladder ||

Dec 24 Tuesday:

LC 4 Median of two sorted arrays (hard):

- **Overall run time complexity should be O(log(m+n))**
- Tricky method to find the partition point

Dec 25 Wednesday, christmas

LC 146 LRU cache!!!(design problem):

- LRU aka least recently used cache
- Eviction policy: evicting item with least recently used
- get (); put () in constant time O(1), overall space: O(n)
    a. Fast lookup: maybe **hashtable**
    b. Fast removals: maybe **double linked list**

LC 938 range sum of BST:

- Dfs to find
- Care about how to go to left child or right child
    - If node is less than L: means left may has less value, go to left , vice verse to right

LC 347 top K frequent elements:

- Time complexity is better than O(n log n): time complexity is hard for me to understand
- **Hashmap, heap**
- Questions require similar techniques:
    - LC 692 top K frequent words
    - LC 451 sort characters by frequency

LC 692 top K frequent words:

LC 451 sort characters by frequency

LC 54 Spiral Matrix:

- Very straightforward method
- Change direction, use dir = (dir + 1) % 4
- Later can change my java version into python

LC 344 reverse string (easy, two pointers)

Dec 26: Thursday:

LC 24 swap nodes in pairs(linked list):

- Recursion difficult for me to understand
- In iterative , the process to swap two nodes has principle (order):
    - Dummy.next = head (dummy = ListNode(-1))
    - Prev_node = dummy
    - Prev_node.next = firstNode.next
    - firstNode.next = secondNode.next
    - secondNode.next = firstNode

LC 1021 remove outermost parentheses (stack):

- **Only parenthese**
- Primitive: 原始的，早期的，简单的，粗糙的
- Java : stringBuilder: stringBuilder objects are like string objects but they can be modified
- Use a counter to count if counter > 0

LC 181 employee earning more than their managers:

- MySQL

LC 303 range sum query - immutable (DP)

- Use extra space to store the 前多少的和
- Sum[i, j] = sum[j+1] - sum[i]
- In python, define a global variable, **use self**

LC 739 daily temperatures(hashtable, stack):

- Temperature has range: (30, 100) ⇒ background
- LC 496 next greater element |
- Java, stack.peek()
- **Check later**

Finish the BST TreeNode search, delete, insert on Mac!!!!!!!!!!!!!!!
Finished Python

Dec 27 Friday:
KMP string pattern match algorithm:

Later can find string match problems in LC

Dec 28 Saturday:
BST node delete, insert, search finished in JAVA

LC 189 rotate array

- At least 3 different ways
- Use two pointers to reverse an array , first reverse 0 to len(nums) - k - 1, then reverse other to len(nums), then reverse all

LC 45 jump game ||(hard, array, greedy，DP , BFS):

- Lc 55 jump game (DP, greedy) return type is boolean:
    - Memorized DP
        - Why int nextPosition = position + 1: position is index, so from the consecutive index to the end (which is min of nums[position] and nums.length -1)
        - DP top- down to bottom-up conversion is done by eliminate recursion.
- return type is int
- Dp use cache[i] to represent the minimum number of steps taken to reach the ith index.
- Dp can only implemented in java, python will return TLE

LC 692 Top k frequent words(heap, hashtable, tire):

- Solve in O(n log n) time and O(n) space
- Sort and count
    - Python count = collections.Counter(words)
- Use hashmap + priority queue
- **Trie**
    - **Represent set of words in an alphabet, a tree data structure**

- **Set of linked nodes,each trie has an empty root node, with links to other nodes. connecting back to an empty root node, the number of child nodes in a trie depends upon the total number of values**
- **Difference between trie and hash table**

In java, ⇒ map.Entry<String, Integer> entry: map.entrySet(). This is for i**terative ov**er Map to get both map keys and values in the loop:
Eg: for(Map.Entry<String, Integer> entry: *map*.entrySet()) …….) *map is hashmap name*
Then ,use these two ways to get value and key: value = entry.getValue()
key =enrty.getKey()

Dec 29, sunday:
Nothing

Dec 30 Monday:

LC 239 sliding window maximum (heap, sliding window):

- Solve in linear time
- There are (n - K + 1) sliding windows and there are (K) elements in each window
- In java ,return null can use : " return new int[0]"
- In java , minimum integer is Integer.MIN_VALUE
- Use deque() (geeksforgeeks):
  - pops from/pushes to either side with the same O(1) performance
  - Operations on deque:
    - insertFront(), getFront()
    - insertLast(), geatRear()
    - deleteFront(); isEmpty()
    - deleteLast(); isFull()
  - deque in Java
    - ArrayDeque<Integer> deq = new ArrayDeque<Integer>();
    - push(element): add an element to the head
    - pop(element): removes an element from the head and return it
    - removeFirst/ removeLast
    - poll(): retrieve and remove the head of the deque and return null if deque is empty. Also has pollFirst and pollLast
    - peek(): retrieve but not remove the head of this deque, return null if is empty. Also has peekLast() and peekFirst()

	First Element or Head		Last Element or Tail	
Operations	Throws exception	Special Value	Throws exception	Special Value
Insert	addFirst(element)	offerFirst(element)	addLast(element)	offerLast(element)
Remove	removeFirst()	pollFirst()	removeLast()	pollLast()
Examine	getFirst()	peekFirst()	getLast()	peekLast()

- deque in python
    - Deque can be implemented in python using module “**collections** ”
    - Deque is **O(1) time** for append and pop as compared to list which provides **O(n) time**
        - append(): insert value to right end
        - appendleft()
        - pop()
        - popleft()

- Heap: min heap and max heap

Dec 31 2019: 2019年的最后一天！！！！！！
看 大话数据结构

January 1 2020:
Nothing

January 2 thursday 2020
LC 2 add two numbers (linked list):

- In most of linked list questions, need to have a dummy node point to the start node
- Use a carry to check is submission is bigger than 10(need to use)

LC 17 letter combinations of a phone number (string, backtracking)

- Use stringBuilder to avoid creating many string object

LC 74 search a 2D matrix:

- When transfer the matrix into a sorted array , Row = idx // n col = idx % n
- Binary search, to find a boundary index value to make compare

LC 208 implement trie(prefix tree, design):

- Did this before
- Re-write and store in big mac “/leetcode/design/”

LC 73 set matrix zeros:

- On the fly: 在匆忙中

LC 21 merge two sorted lists(linked list):

- Still, linked list need to use a dummy node to point to head
- ListNode prehead = new ListNode(-1);
- ListNode prev = prehead;

LC 1108 defanging an IP address :

- When use stringBuilder, need to finally convert toString(), if we want string as result
- StringBuilder sb = new StringBuilder();

LC 94 binary Tree Inorder Traversal:

- This time use java, very straightforward method

LC 235 lowest common ancestor of a binary search tree:
Useful one

Install Java EE
Java里面的8种基本类型：
整型（4种）：byte（8位，-128～127）， short（16位）， int（32位）， long（64位）
字符型（1种）：char(16位)，一个字符用单引号， （字符串表示双引号）
浮点型（2种）：float长度32位， double长度64位。 注意默认的小数值是double类型的

- **例如 float f = 32.344 会报错，因为小数默认是double，此时改为 float f = 32.344f 就不会报错**

布尔型（1种）：boolean b1 = true；

String 类型： 不是基本类型， immutable，不可改变
用final 修饰变量时，该变量只有一次赋值机会。

January 3 Friday:
Eclipse J2EE

LC 103 binary tree zigzag level order traversal:

- Stack, tree, BFS

LC 11 container with most water:

- **Two pointer**
- **Not difficult**

LC 9 palindrome number:

- Care about overflow
- To get the last digit, use module 10
    - 1221 % 10 == 1
- To remove the last digit, use divide 10
    - 1221 / 10 == 122
- if we multiply the last digit by 10 and add the second last digit, `1 * 10 + 2 = 12`, it gives us the reverted number we want from “1221” to get “12”

Jan 4 Saturday

LC 153 find minimum in rotated sorted array:

- Binary search:
    - If mid > 0 && nums[mid] > nums[mid - 1] ⇒ return nums[mid]
    - nums[Mid] > nums[start] && nums[mid] > nums[end] ⇒ start = mid + 1
    - Else: end = mid - 1

LC 25 reverse Nodes in k- group(hard)

1. LC 24: swap nodes in pairs (use dummy -> first -> second)
2. LC 92: reverse linked list ||
3. **Constant extra memory**
4. From a comments: CaterJiang: 2 steps:
    a. Test whether we have more than k node left, if less than k node left we return head
    b. Reverse k node at current level

Google most interview questions:

LC 1 two sum:

- Reduce look up time from O(n) to O(1) by trading space for speed⇒ hash table
- One pass hash table

LC 4 median of two sorted Arrays:

- **Overall time should be O(log(m + n))**
- **partitionX + partitionY = (x + y + 1) / 2**

操作符学习：
乘以2不用乘法符号：左移一位： 16<<1 == 32 箭头朝哪边就是往哪边增加，箭头朝右就是往右走，即删掉最右边的位数的值。

带符号右移和无符号右移：
带符号右移：“>>” 移动后符号不变
无符号右移：”>>>” 移动后负变成正的

三元操作符：
表达式 ？ val1：val2
⇒ 表达式为真，val1， 否则返回val2

Java里面读取数据是： Scanner s = new Scanner(System.in);

Use switch instead of if else:

- **Switch need to be used with break**

January 5: monday:
java结束外部循环的方法：

1. **使用boolean值更改**
    a. **Break 只能结束当前循环：**
        i. **for ()**
            for()
            Break
    b. **使用boolean 值更改外部循环：**
        boolean breakout = false;
        for(.....)

```
for(...)
  if(...)
     break
if(breakout)
     break;    ⇒ 通过这个改变外部循环
```

2. **使用标签：**

```
outloop:       // 标签
for(.....)
  for(...)
    if(...)
       break outloop
```

Java 创建数组：

```
int [] a = new int[5];
Int [] a = new int[ ] {1,2,3,4,5};
int [] a = {1,2,3,4,5};
```

January 7 Tuesday:

LC 68 Text justification (hard):

- **Left justified/ fully justified**

LC 380 insert delete gerRandom(1) time : medium :

- Hashtable, design , array
- Insert, delete, getRandom
    - Hashmap (or hashset): java HashMap / Python dictionary (add, delete in average time, but has problem in getRandom)
    - Array List: Java ArrayList / Python list (add, getRandom in constant time, but has problem in delete)
- Combine hashmap + arrayList

jingJing interview with apple: 原题考出来两道

LC 210 Course schedule ||（**BFS**） ⇒ dfs is hard so not think about dfs

- Topologically sorted order
  - // 表示在index位置上后续的课程
  - // 例如如果prereq = [[1,0],[2,0],[3,1],[3,2]]
  - // adjs = [[1,2],[3],[3],[]]
  - // incLinkCounts = [0,1,1,2] index位置的课程的prereq有几个
  -
  - DAG(directed acyclic graph)
  - Use one ArrayList to store

LC 973. K Closest Points to Origin

- If a big data stream, need to use heap, priority queue
  - Java, default is minHeap
  - Java ArrayList : get(index) to get the value on index, add() to add value into arrayList
  - PriorityQueue<int []> pq = new PriorityQueue<>(这里面可以写怎么比较)
- If a stable list, it is good to use binary search

150. Evaluate Reverse Polish Notation:

Do summary about : continue, pass, break:
https://www.digitalocean.com/community/tutorials/how-to-use-break-continue-and-pass-statements-when-working-with-loops-in-python-3

Do summary about BST, treenode search, delete, insert
Summary about LRU cache