Name	Explanation	FollowUps
Valid Triangle Number	1. 就是Sort + Two Sum Less than K 2. 或者用n^2 + Binary Search	

First Missing Number in Sorted Array	给一个sorted array，寻找第k个missing的数值
	1. linear scan 是O(n)
	2. Binary Search 先检查base case
	a) 中点为计算出来这个位置失去的个数 [1,4,6], m = 4, index=1, start=1,
	中间应该有2个数(2和3),但实际只有1-0-1(0个数)，所以确实了
	两个nums. nums[mid] - nums[left] - (mid - left) (实际-现实)
	b) 进行Binary Search
	l,r = 0, len(nums)-1
	while (l + 1 < r):
	if missing < k:
	// when the number is smaller than k, then the index won't be located in [l, m),
	update k -= missing
	l = m k -= missing else:
	// when the number is equal larger than k, then the index won't be located in (m, h]
	r = m return nums[l] += k	

Find Peak Element	1. 在没有adjacent duplicate, 并且左右如果是 peak 都可以算是peak的话， binary search可以往上坡找 思路为控制 [ )区间，最后return l, 不断update正确结果 2. l,r = 0, len(array)-1 while(l < r) (注意最右边不会被选到，是open bound) if array[mid] > array[mid+1] peak = l else: r = m return l	1. Mountain Array 进行Binary Search 2. Recursion (带上l,r就可以了)
Longest Common Prefix	1. Vertical/Horizontal Scanning O(mn) 2. Binary Search O(mn*log(m))	
Largest BST Subtree	1. Use valid binary search, bottom up, use base case as min = float("inf") and max = float ("-inf)	

Power(x,n)	class Solution:
	if n < 0:
	x = 1/x
	n = -n
	ans = 1
	currentProduct = x
	i = n
	while i > 0:
	if i % 2 == 1:
	ans = ans * currentProduct
	currentProduct = currentProduct * currentProduct
	i //= 2
	return ans	

Find In Mountain Array	用三种Binary Search, 1. 先找Peak 2. 左右开工找Element	

Random Pick With Weight	1. Running Sum + Binary Search def binSearch (array, target):
	l,r = 0, len(array)
	while (l < r):
	m = (l + r) // 2
	if array[m] == target:
	return m
	elif array[m] < target:
	l = m + 1
	elif array[m] > target:
	r = m
	return l	

Count Smaller Numbers After Self	1. MergeSort会相对简单 2. binary search 需要binary insert用到树才能达到O(nlogn)	Hard, needs review

Maximum Profit in Job Scheduled	def binarySearch(nums, target):
	'''
	if target in nums: Ret target
	else: return the first number less than target
	'''
	l,r = 0, len(nums) - 1
	while (l <= r):
	m = (l + r) // 2
	if nums[m] == target:
	return target
	elif nums[m] < target:
	l = m + 1
	elif nums[m] > target:
	r = m - 1
	return nums[r]	

Longest Increasing Subsequence	1. 记录Monotone Increase Stack, 找到新的element时可以进行binary search ，将第一个比他大的 replace (bisect_left) 进行replace	
Search in Rotated Sorted Array		
Find Minimum In Rotated Sorted		
Median of Two Sorted Array		
Find First and Last of an Element in a sorted array		
Search 2D Matrix		
isSubSequence	1. 先用Target string 建立dictionary{char: [idx array]}, 然后记录一个pointer， 用于在[idx:array] 里进行binary search找下一个可以放的位置 (greedy + binary search), 这样如果S很短，但是很多的 话可以用达到O(S log T)的runtime	If there are lots of incoming SS, and you want to check one by one to see if TT has its subsequence. In this scenario, how would you change your code
Sqrt(x)	1. 从2到x//2开始压缩binary search	
Find Duplicate Number	1. Tortoise and Hare 2. Binary Search + Count (O N log N)，但是每个level要进行count 3. Sorting + linear Comparison 4. 傻逼hashmap	