		#	Might be able to solve without looking at the solution							
Group	Tag	#		Acceptance	Difficulty					
	hashtable	1	Two Sum							
	two pointers,	F00	W. W. D							
	hashtable two pointers	532 167	K-diff Pairs in an Array Two Sum II - Input array is sorted							
	hashtable	170	Two Sum III - Data structure design							
	two pointers,									
N-Sum	hashtable	15	3Sum							
	two pointers	16	3Sum Closest							
	two pointers two pointers	259 611	3Sum Smaller Valid Triangle Number							
	two pointers,	011	valid mangle number							
	hashtable	18	4Sum							
	hashtable	454	4Sum II							
	queue	346	Moving Average from Data Stream							
	two pointer	643	Maximum Average Subarray I							
	heap, deque heap, median	239 295	Sliding Window Maximum Find Median from Data Stream							
Fixed Length	heap, median	480	Sliding Window Median							
Sliding Window	two pointers,									
	hashmap	438	Find All Anagrams in a String							
	two pointers, hashmap	567	Permutation in String							
	two pointers,	007	- Official of the Control of the Con							
	hashmap	30	Substring with Concatenation of All Words							
	two pointers,	3	Longest Substring Without Repeating Characters							
	regular		Congest educating William Topouring Orlandstore							
	expression	395	Longest Substring with At Least K Repeating Characters							
variant length	two pointers	159	Longest Substring with At Most Two Distinct Characters							
sliding window	two pointers two pointers	340 424	Longest Substring with At Most K Distinct Characters Longest Repeating Character Replacement							
	two pointers,	424	Longest Repeating Character Replacement							
	subsequence	524	Longest Word in Dictionary through Deleting							
	two pointers	76	Minimum Window Substring							
	two pointers	209	Minimum Size Subarray Sum							
	hashmap	325	Maximum Size Subarray Sum Equals k							
subarray sum	hashmap hashmap	525 523	Contiguous Array Continuous Subarray Sum							
	hashmap	560	Subarray Sum Equals K							
	two pointers	26	Remove Duplicates from Sorted Array							
	two pointers	80	Remove Duplicates from Sorted Array II							
Array	two pointers	27	Remove Element							
Organization	two pointers	283	Move Zeroes							
	two pointers	88	Merge Sorted Array							
	pointers, counting	75	Sort Colors							
	two pointers	11 42	Container With Most Water Trapping Rain Water							
Water Catch	two pointers stack	84	Largest Rectangle in Histogram							
	stack	85	Maximal Rectangle							
	heap	407	Trapping Rain Water II							
	fast and slow	141	Linked List Cycle							
Circle	fast and slow	142	Linked List Cycle II							
	fast and slow	287	Find the Duplicate Number			y search, or O(n) by fast and slow two pointers				
		160	Intersection of Two Linked Lists			ntly by connenct AB and BA				
		237 83	Delete Node in a Linked List Remove Duplicates from Sorted List		uo not need to cl	hange every node				
Remove Node		82	Remove Duplicates from Sorted List Remove Duplicates from Sorted List II							
from Linked List		203	Remove Linked List Elements							
		19	Remove Nth Node From End of List							
		61	Rotate Lists		corner case: k =	0, < 0, =m * len(list)				
		86	Partition List							
		328	Odd Even Linked List		can have short, e					
		206	Reverse Linked List		dummy.next, not every time, from	de.next, node = node, dummy.next, node.next the old node, we remove head into the newhead, up	date both head and	d newhead		
		92	Reverse Linked List II			-th, insert (m+1) to n nodes between pre and pre.nex				
move nodes		21	Merge Two Sorted Lists			, , , , , , , , , , , , , , , , , , , ,				
		23	Merge k Sorted Lists							

		147	Insertion Sort List							
	fast and slow	148	Sort List		fact clow points	ers, pay attantion to the fast and slow start position				
	fast and slow	143	Reorder List			reverse right part, insert right into left				
	last and slow	24	Swap Nodes in Pairs			ld be easier, while for iterative, we each time reverse two	and move two	nocition		
		25	Reverse Nodes in k-Group		recursive	ild be easier, writte for iterative, we each time reverse two	b, and move two	position		
		344	Reverse String		recursive					
		541	Reverse String II							
		345	Reverse Vowels of a String							
		151	Reverse Words in a String							
reverse array		186	Reverse Words in a String II							
,		557	Reverse Words in a String III							
		189	Rotate Array		nume = numef	::-1] will not change the nums passed as parameter, but r	nume[:] = nume[::	-11 will do		
		7	Reverse Integer		namo namo _l .	I I I I I I I I I I I I I I I I I I I	namo[.]	1, 00		
	bit	190	Reverse Bits		get the hin in e	each position, and set the result correspondingly				
	backtracking	17	Letter Combinations of a Phone Number		g-1	g,				
	backtracking	401	Binary Watch							
	backtracking	22	Generate Parentheses							
	backtracking	78	Subsets							
	backtracking	90	Subsets II							
	backtracking	491	Increasing Subsequences							
	backtracking	46	Permutations							
tradictional BT	backtracking	47	Permutations II							
	backtracking	77	Combinations							
	backtracking	39	Combination Sum							
	backtracking	40	Combination Sum II							
	backtracking	216	Combination Sum III							
	namic programmi	377	Combination Sum IV							
	backtracking	638	Shopping Offers							
	backtracking	254	Factor Combinations							
	,		Two Pointers							
		5	Longest Palindromic Substring	25.10%	Medium					
		647	Palindromic Substrings	56.00%	Medium	extension of 5, to find the longest, in fact we checked	all the palindrom	ic substrings		
		9	Palindrome Number	34.70%	Easy	corner case, x< 0, x =0 and x is ***0				
		125	Valid Palindrome	25.90%	Easy	string.ascii_letters, digits, str.isalnum()				
		680	Valid Palindrome II	28.10%	Easy					
Palindrome	KMP	214	Shortest Palindrome	23.60%	Hard	only s.startswith can pass, while s[:n-i] == r[i:] will TLE				
1 dillidionic		409	Longest Palindrome	45.10%	Easy	insert or remove from set				
		266	Palindrome Permutation	56.30%	Easy	same as 409, use a set to add odd char and remove i	it when even (me	et again)		
						first determine whether can form as permuation, then				
		267	Palindrome Permutation II	31.50%	Medium	The left problem is the distint permutation of the even		ktracking		
		234	Palindrome Linked List	32.20%	Easy	Two pointer to find the middle, and reverse the right p	part			
		564	Find the Closest Palindrome	14.00%	Hard					
			Binary Search		_					
		35	Search Insert Position	39.40%	Easy	bisect_left				
		34 162	Search for a Range	31.10% 36.70%	Medium Medium	bisect				
		74	Find Peak Element	35.30%	Medium					
		240	Search a 2D Matrix	35.30%	Medium					
		33	Search a 2D Matrix II Search in Rotated Sorted Array	38.10%	Medium					
Deteted Cort-		81	Search in Rotated Sorted Array II	32.10%	Medium	When we find left == mid or right == mid, we have to r	move it towards r	nid until a different value found		
Rotated Sorted Array		153	Find Minimum in Rotated Sorted Array	39.30%	Medium	vinen we find left find of right find, we have to f	IIIOVE IL IUWAIUS I	ina anai a uniciciit value idullu		
-,		154	Find Minimum in Rotated Sorted Array II	36.70%	Hard					
		540	Single Element in a Sorted Array	53.50%	Medium					
		4	Median of Two Sorted Arrays	21.30%	Hard	Median of K sorted Arrays				
		315	Count of Smaller Numbers After Self	34.10%	Hard	binary search				
		374	Guess Number Higher or Lower	34.50%	Easy	, oodion				
		658	Find K Closest Elements	35.30%	Medium					
		278	First Bad Version	24.80%	Easy					
		302	Smallest Rectangle Enclosing Black Pixels	44.80%	Hard					
		441	Arranging Coins	36.10%	Easy					
		475	Heaters	29.60%	Easy	sort and two pointers				
			Backtracking	25.0070	Lusy	22 and the pointere				
		36	Valid Sudoku	34.90%	Medium					
		37	Sudoku Solver	29.20%	Hard					
		488	Zuma Game	36.20%	Hard					
Games		51	N-Queens	30.00%	Hard					
		TC TC	IN-Queens	30.00%	Hard					

		52	N-Queens II	43.80%	Hard				
		351	Android Unlock Patterns	43.20%	Medium				
	string	205	Isomorphic Strings	33.20%	Easy				
ab to 12	string	290	Word Pattern	32.60%	Easy				
	string	291	Word Pattern II	37.80%	Hard	backtracking to test different length			
	string	468	Validate IP Address	20.20%	Medium				
	string	93	Restore IP Addresses	26.60%	Medium				
	string	526	Beautiful Arrangement	54.10%	Medium				
	string	131	Palindrome Partitioning	32.00%	Medium				
	string	132	Palindrome Partitioning II	23.80%	Hard	dp			
	Sung	79	Word Search	26.10%	Medium	can early return instead of set a flag			
		212	Word Search II	22.90%	Hard	can early return instead of set a hag			
		127	Word Ladder	19.30%	Medium				
		126	Word Ladder II	13.90%	Hard				
		422	Valid Word Square	36.30%	Easy				
		425	Word Squares	42.60%	Hard				
		408	<u>Valid Word Abbreviation</u>	27.60%	Easy				
		288	Unique Word Abbreviation	16.10%	Medium	hashtable			
abbreviation		320	Generalized Abbreviation	44.30%	Medium				
		411	Minimum Unique Word Abbreviation	31.80%	Hard	brute force?			
		527	Word Abbreviation	34.80%	Hard	not a bt			
		282	Expression Add Operators	29.30%	Hard				
		679	24 Game	38.60%	Hard				
		5.0	Dynamic Programming (try to save space)	33.3378	. 10.10				
		53	Maximum Subarray	39.20%	Easy				
		152	Maximum Product Subarray	25.10%	Medium				
		238		48.20%		t			
	traverse		Product of Array Except Self		Medium	two pass, pay attention to the index in each side			
	traverse	581	Shortest Unsorted Continuous Subarray	28.30%	Easy				
		303	Range Sum Query - Immutable	28.00%	Easy				
		304	Range Sum Query 2D - Immutable	24.00%	Medium	range matters			
		121	Best Time to Buy and Sell Stock	40.30%	Easy				
-41		122	Best Time to Buy and Sell Stock II	46.30%	Easy				
stock dp		123	Best Time to Buy and Sell Stock III	28.80%	Hard	There is a one pass dp using O(1) space			
		188	Best Time to Buy and Sell Stock IV	24.10%	Hard	Need to think clearly how the dp works			
		309	Best Time to Buy and Sell Stock with Cooldown	40.20%	Medium				
		70	Climbing Stairs	39.30%	Easy	sn = sn-1 + sn-2, s0 = 1, s1 = 0			
		91	Decode Ways	19.30%	Medium	a lot of corner cases			
		639	Decode Ways II	18.90%	Hard				
		198	House Robber	38.20%	Easy				
(n) DP, can do		213	House Robber II	33.50%	Medium				
O(1) Space		337	House Robber III	42.60%	Medium	indeed can do a postorder with another stack			
		256		45.90%		indeed can do a postorder with another stack			
			Paint House		Easy				
		265	Paint House II	37.70%	Hard				
		276	Paint Fence	34.20%	Easy				
		494	Target Sum	43.80%	Medium				
		300	Longest Increasing Subsequence	37.90%	Medium	have a O(nlogn) solution			
		673	Number of Longest Increasing Subsequence	30.80%	Medium				
O(n) DP		96	Unique Binary Search Trees	40.40%	Medium				
but need to check all		95	Unique Binary Search Trees II	31.00%	Medium				
possible		279	Perfect Squares	36.00%	Medium				
position before		322	Coin Change	26.20%	Medium				
		343	Integer Break	45.50%	Medium				
		139	Word Break	29.20%	Medium				
		140	Word Break II	22.70%	Hard				
		120	Triangle	33.10%	Medium				
		62	Unique Paths	40.20%	Medium	O(min(m, n)) space, best performance if we dp on the short direction			
		63	Unique Paths II	31.30%	Medium	, , , , , , , , , , , , , , , , , , ,			
		562	Longest Line of Consecutive One in Matrix	36.30%	Medium				
M*N path O(M) space		221	Maximal Square	28.00%	Medium				
O(N) Space		64	Minimum Path Sum	37.80%	Medium				
		174	<u>Dungeon Game</u>	23.40%	Hard				
		568	Maximum Vacation Days	40.20%	Hard				
			Student Attendance Record I	44.20%	Easy				
		551							
M*N path		551	Student Attendance Record II	27.70%	Hard	N * (2*3)			

	44	Wildcard Matching	19.60%	Hard	might have a quick solution				
	10	Regular Expression Matching	23.90%	Hard	can do a standard dp of (M+1) * (N+1)				
	72	Edit Distance	31.10%	Hard	can do O(M) space				
String Matching (M+1)*(N+1)	87	Scramble String	28.70%	Hard					
()	97	Interleaving String	24.30%	Hard					
	583	Delete Operation for Two Strings	40.80%	Medium					
	115	Distinct Subsequences	31.10%	Hard	can do O(M) space				
	89	Gray Code	40.30%	Medium	no need to do a standard bt, but just extend the resu	It eimilar to eubea			
	375	Guess Number Higher or Lower II	35.60%	Medium	no need to do a standard bi, but just extend the resu	iit, siiriiidi to sabsc			
	629	-	15.90%	Medium					
	600	K Inverse Pairs Array	21.60%	Hard					
		Non-negative Integers without Consecutive Ones							
	678	Valid Parenthesis String	26.20%	Medium					
	241	Different Ways to Add Parentheses	42.70%	Medium					
	312	Burst Balloons	42.20%	Hard					
	321	Create Maximum Number	24.30%	Hard					
	354	Russian Doll Envelopes	32.00%	Hard					
	363	Max Sum of Rectangle No Larger Than K	32.50%	Hard					
	368	Largest Divisible Subset	33.50%	Medium					
	410	Split Array Largest Sum	35.20%	Hard					
	416	Partition Equal Subset Sum	38.50%	Medium					
	403	Frog Jump	31.60%	Hard					
	293	Flip Game	54.90%	Easy					
	293	Flip Game II	45.90%	Medium					
	294 464		45.90% 23.80%	Medium					
		Can I Win							
	486	Predict the Winner	44.40%	Medium					
	413	Arithmetic Slices	54.90%	Medium					
	446	Arithmetic Slices II - Subsequence	25.30%	Hard					
	467	Unique Substrings in Wraparound String	31.40%	Medium					
	466	Count The Repetitions	26.60%	Hard					
	472	Concatenated Words	29.70%	Hard					
	474	Ones and Zeroes	37.70%	Medium					
	514	Freedom Trail	34.90%	Hard					
	516	Longest Palindromic Subsequence	42.40%	Medium					
	517	Super Washing Machines	35.60%	Hard					
	546	Remove Boxes	29.60%	Hard					
	553	Optimal Division	53.70%	Medium					
	471	Encode String with Shortest Length	41.60%	Hard					
	576	Out of Boundary Paths	0.324	Hard					
	370	Design	0.324	Tialu					
	200	Insert Delete GetRandom O(1)		Mardina	list for random O(1), dict for insert and remove O(1)				
list	380								
			38.90%	Medium					
	381	Insert Delete GetRandom O(1) - Duplicates allowed	28.50%	Hard					
	146	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache	28.50% 17.00%	Hard Hard	double linked list for get max, min, most recent O(1),	, need to update b	oth direction		
linked list	146 460	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache LFU Cache	28.50% 17.00% 22.40%	Hard		, need to update b	oth direction		
linked list	146 460 432	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache	28.50% 17.00% 22.40% 27.50%	Hard Hard	double linked list for get max, min, most recent O(1),	need to update be	oth direction		
linked list	146 460	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache LFU Cache	28.50% 17.00% 22.40%	Hard Hard Hard	double linked list for get max, min, most recent O(1),	need to update b	oth direction		
linked list	146 460 432	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache LFU Cache All O'one Data Structure	28.50% 17.00% 22.40% 27.50%	Hard Hard Hard Hard	double linked list for get max, min, most recent O(1),	need to update be	oth direction		
	146 460 432 155	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache LFU Cache All O'one Data Structure Min Stack	28.50% 17.00% 22.40% 27.50% 27.60%	Hard Hard Hard Hard Easy	double linked list for get max, min, most recent O(1),	need to update b	oth direction		
	146 460 432 155 225	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache LFU Cache All O'one Data Structure Min Stack Implement Stack using Queues	28.50% 17.00% 22.40% 27.50% 27.60% 32.10%	Hard Hard Hard Hard Easy Easy	double linked list for get max, min, most recent O(1),	need to update be	oth direction		
	146 460 432 155 225 232	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache LFU Cache All O'one Data Structure Min Stack Implement Stack using Queues Implement Queue using Stacks Elatten 2D Vector	28.50% 17.00% 22.40% 27.50% 27.60% 32.10% 35.90%	Hard Hard Hard Hard Easy Easy	double linked list for get max, min, most recent O(1),	need to update be	oth direction		
stack	146 460 432 155 225 232 251 281	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache LFU Cache All O'one Data Structure Min Stack Implement Stack using Queues Implement Queue using Stacks Elatten 2D Vector Zigzag Iterator	28.50% 17.00% 22.40% 27.50% 27.60% 32.10% 35.90% 39.80% 49.60%	Hard Hard Hard Hard Easy Easy Easy Medium Medium	double linked list for get max, min, most recent O(1),	need to update be	oth direction		
	146 460 432 155 225 232 251 281 284	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache LFU Cache All O'one Data Structure Min. Stack Implement Stack using Queues Implement Queue using Stacks Flatten 2D Vector Zigzag Iterator Peeking Iterator	28.50% 17.00% 22.40% 27.50% 27.60% 32.10% 35.90% 39.80% 49.60% 35.30%	Hard Hard Hard Hard Easy Easy Medium Medium Medium	double linked list for get max, min, most recent O(1),	need to update be	oth direction		
stack	146 460 432 155 225 232 251 281 284 341	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache LFU Cache All O'one Data Structure Min Stack Implement Stack using Queues Implement Queue using Stacks Flatten 2D Vector Zigzag Iterator Peeking Iterator Flatten Nested List Iterator	28.50% 17.00% 22.40% 27.50% 27.60% 32.10% 35.90% 39.80% 49.60% 35.30% 40.30%	Hard Hard Hard Hard Easy Easy Medium Medium Medium Medium	double linked list for get max, min, most recent O(1),	need to update by	oth direction		
stack	146 460 432 155 225 232 251 281 284 341 604	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache All O'one Data Structure Min Stack Implement Stack using Queues Implement Queue using Stacks Elatten 2D Vector Zigzag Iterator Peeking Iterator Elatten Nested List Iterator Design Compressed String Iterator	28.50% 17.00% 22.40% 27.50% 27.60% 32.10% 35.90% 39.80% 49.60% 35.30% 40.30% 30.20%	Hard Hard Hard Hard Easy Easy Easy Medium Medium Medium Medium Easy	double linked list for get max, min, most recent O(1),	need to update be	oth direction		
stack	146 460 432 155 225 232 251 281 284 341 604 353	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache LFU Cache All O'one Data Structure Min Stack Implement Stack using Queues Implement Queue using Stacks Elatten 2D Vector Zigzag Iterator Peeking Iterator Elatten Nested List Iterator Design Compressed String Iterator Design Snake Game	28.50% 17.00% 22.40% 27.50% 27.60% 32.10% 35.90% 39.80% 49.60% 36.30% 40.30% 40.30% 26.10%	Hard Hard Hard Hard Hard Easy Easy Medium Medium Medium Medium Easy Medium Medium	double linked list for get max, min, most recent O(1),	need to update b	oth direction		
stack	146 460 432 155 225 232 251 281 284 341 604 353 379	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache LFU Cache All O'one Data Structure Min Stack Implement Stack using Queues Implement Queue using Stacks Flatten 2D Vector Zigzag Iterator Peeking Iterator Peeking Iterator Design Compressed String Iterator Design Compressed String Iterator Design Snake Game Design Phone Directory	28.50% 17.00% 22.40% 27.50% 27.60% 32.10% 35.90% 39.80% 49.60% 35.30% 40.30% 30.20% 26.10% 31.40%	Hard Hard Hard Hard Easy Easy Medium Medium Medium Easy Medium Medium Medium Medium	double linked list for get max, min, most recent O(1),	need to update b	oth direction		
stack Iterator hashtable	146 460 432 155 225 232 251 281 284 341 604 353 379 631	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache LFU Cache All O'one Data Structure Min Stack Implement Stack using Queues Implement Queue using Stacks Flatten 2D Vector Zigzag Iterator Peeking Iterator Flatten Nested List Iterator Design Compressed String Iterator Design Snake Game Design Phone Directory Design Excel Sum Formula	28.50% 17.00% 22.40% 27.50% 27.60% 32.10% 35.90% 39.80% 49.60% 35.30% 40.30% 30.20% 26.10% 31.40% 17.00%	Hard Hard Hard Hard Easy Easy Easy Medium Medium Medium Medium Medium Medium Hard	double linked list for get max, min, most recent O(1),	need to update be	oth direction		
stack	146 460 432 155 225 232 251 281 284 341 604 353 379 631	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache All Cone Data Structure Min Stack Implement Stack using Queues Implement Queue using Stacks Elatten 2D Vector Zigzag Iterator Peeking Iterator Elatten Nested List Iterator Design Compressed String Iterator Design Compressed String Iterator Design Snake Game Design Phone Directory Design Excel Sum Formula Logger Rate Limiter	28.50% 17.00% 22.40% 27.50% 27.60% 32.10% 35.90% 39.80% 49.60% 35.30% 40.30% 30.20% 26.10% 31.40% 59.00%	Hard Hard Hard Hard Easy Easy Medium Medium Medium Medium Medium Medium Hard Easy	double linked list for get max, min, most recent O(1),	need to update be	oth direction		
stack Iterator hashtable	146 460 432 155 225 232 251 281 284 341 604 353 379 631 359 635	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache All O'one Data Structure Min Stack Implement Stack using Queues Implement Queue using Stacks Elatten 2D Vector Zigzag Iterator Peeking Iterator Elatten Nested List Iterator Design Compressed String Iterator Design Compressed String Iterator Design Phone Directory Design Excel Sum Formula Logger Rate Limiter Design Log Storage System	28.50% 17.00% 22.40% 27.50% 27.60% 32.10% 35.90% 39.80% 49.60% 30.20% 26.10% 31.40% 17.00% 59.00% 42.60%	Hard Hard Hard Hard Easy Easy Medium	double linked list for get max, min, most recent O(1),	need to update b	oth direction		
stack Iterator hashtable	146 460 432 155 225 232 251 281 284 341 604 353 379 631 359 635	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache LFU Cache All O'one Data Structure Min. Stack Implement Stack using Queues Implement Queue using Stacks Elatten 2D Vector Zigzag Iterator Peeking Iterator Peeking Iterator Design Compressed String Iterator Design Compressed String Iterator Design Phone Directory Design Excel Sum Formula Logger Rate Limiter Design Log Storage System Design Hit Counter	28.50% 17.00% 22.40% 27.50% 27.60% 32.10% 35.90% 39.80% 49.60% 35.30% 40.30% 26.10% 31.40% 17.00% 59.00% 42.60% 53.30%	Hard Hard Hard Hard Easy Easy Medium Medium Medium Medium Medium Medium Hard Easy	double linked list for get max, min, most recent O(1), double linked list with ordered dict	need to update be	oth direction		
stack Iterator hashtable	146 460 432 155 225 232 251 281 284 341 604 353 379 631 359 635	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache All O'one Data Structure Min Stack Implement Stack using Queues Implement Queue using Stacks Elatten 2D Vector Zigzag Iterator Peeking Iterator Elatten Nested List Iterator Design Compressed String Iterator Design Compressed String Iterator Design Phone Directory Design Excel Sum Formula Logger Rate Limiter Design Log Storage System	28.50% 17.00% 22.40% 27.50% 27.60% 32.10% 35.90% 39.80% 49.60% 30.20% 26.10% 31.40% 17.00% 59.00% 42.60%	Hard Hard Hard Hard Easy Easy Medium	double linked list for get max, min, most recent O(1),	need to update be	oth direction		
stack Iterator hashtable hashtable	146 460 432 155 225 232 251 281 284 341 604 353 379 631 359 635	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache LFU Cache All O'one Data Structure Min. Stack Implement Stack using Queues Implement Queue using Stacks Elatten 2D Vector Zigzag Iterator Peeking Iterator Peeking Iterator Design Compressed String Iterator Design Compressed String Iterator Design Phone Directory Design Excel Sum Formula Logger Rate Limiter Design Log Storage System Design Hit Counter	28.50% 17.00% 22.40% 27.50% 27.60% 32.10% 35.90% 39.80% 49.60% 35.30% 40.30% 26.10% 31.40% 17.00% 59.00% 42.60% 53.30%	Hard Hard Hard Hard Easy Easy Easy Medium Medium Medium Medium Medium Leasy Medium	double linked list for get max, min, most recent O(1), double linked list with ordered dict	need to update b	oth direction		
stack Iterator hashtable hashtable trie	146 460 432 155 225 232 251 281 284 341 604 353 379 631 359 635	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache All O'one Data Structure Min Stack Implement Stack using Queues Implement Queue using Stacks Flatten 2D Vector Zigzag Iterator Peeking Iterator Peeking Iterator Design Compressed String Iterator Design Snake Game Design Phone Directory Design Log Storage System Design Log Storage System Design Log Storage System Design Log Storage System Design HC Counter Implement Trie (Prefix Tree)	28.50% 17.00% 22.40% 27.50% 27.60% 32.10% 35.90% 39.80% 49.60% 35.30% 40.30% 30.20% 26.10% 31.40% 17.00% 59.00% 42.60% 53.30% 27.00%	Hard Hard Hard Hard Easy Easy Medium Medium Medium Medium Easy Medium	double linked list for get max, min, most recent O(1), double linked list with ordered dict	need to update b	oth direction		
stack Iterator hashtable hashtable trie trie trie	146 460 432 155 225 232 251 281 284 341 604 353 379 631 359 635 362 208 211	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache All O'one Data Structure Min Stack Implement Stack using Queues Implement Queue using Stacks Elatten 2D Vector Zigzag Iterator Peeking Iterator Peeking Iterator Elatten Nested List Iterator Design Compressed String Iterator Design Snake Game Design Phone Directory Design Excel Sum Formula Logger Rate Limiter Design Log Storage System Design Hit Counter Implement Trie (Prefix Tree) Add and Search Word - Data structure design Map Sum Pairs	28.50% 17.00% 22.40% 27.50% 27.60% 32.10% 35.90% 39.80% 49.60% 36.30% 40.30% 30.20% 26.10% 31.40% 59.00% 42.60% 53.30% 27.00% 21.50% 54.00%	Hard Hard Hard Hard Easy Easy Easy Medium Hard Easy Medium Medium Medium Medium	double linked list for get max, min, most recent O(1), double linked list with ordered dict	need to update be	oth direction		
stack Iterator hashtable hashtable trie trie	146 460 432 155 225 232 251 281 284 341 604 353 379 631 359 635 362 208 211 677 642	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache LFU Cache All O'one Data Structure Min. Stack Implement Stack using Queues Implement Queue using Stacks Elatten 2D Vector Zigzag Iterator Peeking Iterator Peeking Iterator Design Compressed String Iterator Design Compressed String Iterator Design Phone Directory Design Excel Sum Formula Logger Rate Limiter Design Log Storage System Design Hit Counter Implement Trie (Prefix Tree) Add and Search Word - Data structure design Map Sum Pairs Design Search Autocomplete System	28.50% 17.00% 22.40% 27.50% 27.60% 32.10% 35.90% 39.80% 49.60% 35.30% 40.30% 26.10% 31.40% 17.00% 59.00% 42.60% 53.30% 27.00% 21.50% 54.00%	Hard Hard Hard Hard Easy Easy Easy Medium Hard Easy Medium	double linked list for get max, min, most recent O(1), double linked list with ordered dict	need to update b	oth direction		
stack Iterator hashtable hashtable trie trie trie trie	146 460 432 155 225 232 251 281 284 341 604 353 379 631 359 635 362 208 211 677 642	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache All O'one Data Structure Min Stack Implement Stack using Queues Implement Queue using Stacks Elatten 2D Vector Zigzag Iterator Peeking Iterator Peeking Iterator Peisign Compressed String Iterator Design Compressed String Iterator Design Snake Game Design Phone Directory Design Excel Sum Formula Logger Rate Limiter Design Log Storage System Design Hit Counter Implement Trie (Prefix Tree) Add and Search Word - Data structure design Map Sum Pairs Design Search Autocomplete System Implement Magic Dictionary	28.50% 17.00% 22.40% 22.40% 27.50% 32.10% 35.90% 39.80% 49.60% 35.30% 40.30% 30.20% 26.10% 31.40% 17.00% 59.00% 42.60% 53.30% 27.00% 21.50% 54.00% 54.00% 55.10%	Hard Hard Hard Hard Easy Easy Medium	double linked list for get max, min, most recent O(1), double linked list with ordered dict	need to update b	oth direction		
stack Iterator hashtable hashtable trie trie trie	146 460 432 155 225 232 251 281 284 341 604 353 379 631 359 635 362 208 211 677 642 676 535	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache All Cone Data Structure Min Stack Implement Stack using Queues Implement Queue using Stacks Elatten 2D Vector Zigzag Iterator Peeking Iterator Elatten Nested List Iterator Design Compressed String Iterator Design Compressed String Iterator Design Phone Directory Design Excel Sum Formula Logger Rate Limiter Design Log Storage System Design Hit Counter Implement Trie (Prefix Tree) Add and Search Word - Data structure design Map Sum Pairs Design Search Autocomplete System Implement Magic Dictionary Encode and Decode TinyURL	28.50% 17.00% 22.40% 22.40% 27.50% 32.10% 35.90% 39.80% 49.60% 35.30% 40.30% 30.20% 26.10% 31.40% 59.00% 42.60% 53.30% 53.30% 54.00% 55.10% 54.00% 57.40%	Hard Hard Hard Hard Hard Easy Easy Medium Hard Easy Medium	double linked list for get max, min, most recent O(1), double linked list with ordered dict	need to update be	oth direction		
stack Iterator hashtable hashtable trie trie trie trie	146 460 432 155 225 232 251 281 284 341 604 353 379 631 359 635 362 208 211 677 642	Insert Delete GetRandom O(1) - Duplicates allowed LRU Cache All O'one Data Structure Min Stack Implement Stack using Queues Implement Queue using Stacks Elatten 2D Vector Zigzag Iterator Peeking Iterator Peeking Iterator Peisign Compressed String Iterator Design Compressed String Iterator Design Snake Game Design Phone Directory Design Excel Sum Formula Logger Rate Limiter Design Log Storage System Design Hit Counter Implement Trie (Prefix Tree) Add and Search Word - Data structure design Map Sum Pairs Design Search Autocomplete System Implement Magic Dictionary	28.50% 17.00% 22.40% 22.40% 27.50% 32.10% 35.90% 39.80% 49.60% 35.30% 40.30% 30.20% 26.10% 31.40% 17.00% 59.00% 42.60% 53.30% 27.00% 21.50% 54.00% 54.00% 55.10%	Hard Hard Hard Hard Easy Easy Medium Hard Medium	double linked list for get max, min, most recent O(1), double linked list with ordered dict	need to update b	oth direction		

		0.40		45.000/						
		348	Design Tic-Tac-Toe	45.60%	Medium					
		588	Design In-Memory File System	30.90%	Hard					
			<u>Tree</u>							
	inorder	94	Binary Tree Inorder Traversal	45.30%	Medium	iterative, Morris traversal				
	preorder	144	Binary Tree Preorder Traversal	44.10%	Medium	iterative				
	postorder	145	Binary Tree Postorder Traversal	39.40%	Hard	iterative, stack with status, inorder and preorder can l	be solved similarly			
	inorder	173	Binary Search Tree Iterator	40.30%	Medium					
	inorder	98	Validate Binary Search Tree	22.90%	Medium					
	inorder	99	Recover Binary Search Tree	29.30%	Hard					
Traversal	inorder	285	Inorder Successor in BST	36.00%	Medium					
	preorder	606	Construct String from Binary Tree	54.90%	Easy					
	inorder	105	Construct Binary Tree from Preorder and Inorder Traversal	31.50%	Medium	hashmap might make it faster				
	inorder	106	Construct Binary Tree from Inorder and Postorder Traversal	31.50%	Medium					
	preorder	255	Verify Preorder Sequence in Binary Search Tree	39.60%	Medium					
	inorder	501	Find Mode in Binary Search Tree	38.40%	Easy					
	inorder	653	Two Sum IV - Input is a BST	50.50%	Easy					
	moraci	102	Binary Tree Level Order Traversal	38.40%	Medium					
		107	Binary Tree Level Order Traversal II	39.10%	Easy					
				60.90%						
		339	Nested List Weight Sum		Easy					
Level Order		364	Nested List Weight Sum II	51.40%	Medium					
Traversal		103	Binary Tree Zigzag Level Order Traversal	33.50%	Medium					
(mostly BFS)		515	Find Largest Value in Each Tree Row	53.90%	Medium					
		116	Populating Next Right Pointers in Each Node	36.90%	Medium	we need to set next, not left, right				
		117	Populating Next Right Pointers in Each Node II	33.60%	Medium					
		199	Binary Tree Right Side View	39.80%	Medium					
		623	Add One Row to Tree	50.10%	Medium					
		637	Average of Levels in Binary Tree	63.20%	Easy					
	postorder	110	Balanced Binary Tree	36.90%	Easy					
		100	Same Tree	45.90%	Easy					
		101	Symmetric Tree	37.90%	Easy					
		226	Invert Binary Tree	50.90%	Easy	recursive, iterative (dfs using stack and bfs using deq	que, appendleft and pop)			
		617	Merge Two Binary Trees	73.40%	Easy					
		314	Binary Tree Vertical Order Traversal	36.20%	Medium					
BFS/DFS (iterative/recursi		104	Maximum Depth of Binary Tree	51.80%	Easy					
(iterative/recursi ve)	DFS	563	Binary Tree Tilt	49.00%	Easy					
,		111	Minimum Depth of Binary Tree	32.70%	Easy					
		257	Binary Tree Paths	36.90%	Easy					
		404	Sum of Left Leaves	46.60%	Easy					
	DFS	112	Path Sum	33.50%	Easy	48.96 iterative				
	DFS	113	Path Sum II	32.60%	Medium	iterative using stack and extra status				
	BFS/DFS	437	Path Sum III	39.30%	Easy	a dfs method with 98.87				
	DFS	124	Binary Tree Maximum Path Sum	25.50%	Hard					
	BFS/DFS	513	Find Bottom Left Tree Value	55.80%	Medium					
	BFS/DFS	129	Sum Root to Leaf Numbers	35.90%	Medium					
	BFS/DFS	298	Binary Tree Longest Consecutive Sequence	40.50%	Medium					
	DFS	250	Count Univalue Subtrees	41.20%	Medium					
	left and right	549	Binary Tree Longest Consecutive Sequence II	36.80%	Medium	two different writing have different performance, savir	ng almost half time			
	left and right	543	Diameter of Binary Tree	42.80%	Easy	the diameter is not the number of nodes	moot name unio			
	on and fight	366	Find Leaves of Binary Tree	58.60%	Medium	and diameter is not the number of nodes				
		310	Minimum Height Trees	28.70%	Medium					
		508	-	52.00%	Medium					
	inorder	508	Most Frequent Subtree Sum		Medium					
	niorder		Convert BST to Greater Tree	52.90% 43.60%						
		156	Binary Tree Upside Down		Medium					
	-41	545	Boundary of Binary Tree	28.20%	Medium					
	stack	114	Flatten Binary Tree to Linked List	34.30%	Medium					
Tree Conversion	stack	536	Construct Binary Tree from String	38.30%	Medium					
COLIVEISION	binary	108	Convert Sorted Array to Binary Search Tree	41.40%	Easy					
	binary	109	Convert Sorted List to Binary Search Tree	33.40%	Medium					
		572	Subtree of Another Tree	43.90%	Easy	serialize, recursive, iterative				
Serialization		331	Verify Preorder Serialization of a Binary Tree	35.70%	Medium					
	stack	297	Serialize and Deserialize Binary Tree	32.60%	Hard	preorder, recover with a stack. or BFS with a list				
		449	Serialize and Deserialize BST	42.20%	Medium					
	binary search	222	Count Complete Tree Nodes	27.10%	Medium					
		133	Clone Graph	25.10%	Medium					
	hashtable									
	hashtable adjacency list	138	Copy List with Random Pointer Number of Connected Components in an Undirected Graph	26.50% 47.50%	Medium Medium	O(1) extra space solution				

	adjacency list	261	Graph Valid Tree	37.30%	Medium				
		230	Kth Smallest Element in a BST	43.00%	Medium				
		235	Lowest Common Ancestor of a Binary Search Tree	38.50%	Easy				
		236	Lowest Common Ancestor of a Binary Tree	29.60%	Medium				
Dinary Coards		270	Closest Binary Search Tree Value	38.90%	Easy				
Binary Search Tree		272	Closest Binary Search Tree Value II	38.40%	Hard				
	neatorder	333		30.20%	Medium				
	postorder		Largest BST Subtree						
		450	Delete Node in a BST	35.80%	Medium				
		530	Minimum Absolute Difference in BST	47.50%	Easy				
		669	Trim a Binary Search Tree	59.60%	Easy				
		207	Course Schedule	31.30%	Medium				
		210	Course Schedule II	26.80%	Medium				
Topological Sort	t	444	Sequence Reconstruction	19.70%	Medium				
	traverse	269	Alien Dictionary	22.80%	Hard				
		329	Longest Increasing Path in a Matrix	35.90%	Hard				
BFS		582	Kill Process	42.50%	Medium				
BFS									
BFS		130	Surrounded Regions	17.90%	Medium				
		200	Number of Islands	33.60%	Medium				
BFS		305	Number of Islands II	38.60%	Hard				
BFS		286	Walls and Gates	43.50%	Medium				
		490	The Maze	42.40%	Medium				
		505	The Maze II	36.10%	Medium	dijkstra Heap			
		499	The Maze III	31.00%	Hard				
		301	Remove Invalid Parentheses	34.90%	Hard				
		317	Shortest Distance from All Buildings	33.60%	Hard				
DFS		332	Reconstruct Itinerary	28.70%	Medium				
D1 3					Medium				
DFS		399	Evaluate Division	40.30%					
		547	Friend Circles	49.00%	Medium				
BFS		417	Pacific Atlantic Water Flow	33.20%	Medium				
BFS		419	Battleships in a Board	61.20%	Medium				
BFS		473	Matchsticks to Square	34.30%	Medium				
BFS		529	Minesweeper	51.80%	Medium				
BFS on all		542	01 Matrix	32.40%	Medium				
			Segment Tree						
		307	Range Sum Query - Mutable	19.60%	Medium				
		308	Range Sum Query 2D - Mutable	21.50%	Hard				
		327	Count of Range Sum	29.30%	Hard				
			Reverse Pairs	19.10%	Hard				
		493							
		493							
			Greedy						
		561	Greedy Array Partition I	73.20%	Easy				
		561 55	Greedy Array Partition I Jump Game	73.20% 29.30%	Easy Medium				
		561 55 45	Greedy Array Partition I Jump Game Jump Game II	73.20% 29.30% 26.10%	Easy Medium Hard				
		561 55 45 45	Greedy Array Partition 1 Jump Game Jump Game II Assign Cookles	73.20% 29.30% 26.10% 47.20%	Easy Medium Hard Easy				
		561 55 45 455 134	Greedy Array Partition I Jump. Game Jump Game II Assign Cookies Gas Station	73.20% 29.30% 26.10% 47.20% 29.00%	Easy Medium Hard Easy Medium				
		561 55 45 455 134 392	Greedy Array Partition 1 Jump Game Jump Game II Assign Cookles	73.20% 29.30% 26.10% 47.20% 29.00% 44.30%	Easy Medium Hard Easy				
		561 55 45 455 134	Greedy Array Partition I Jump. Game Jump Game II Assign Cookies Gas Station	73.20% 29.30% 26.10% 47.20% 29.00%	Easy Medium Hard Easy Medium				
		561 55 45 455 134 392	Greedy Array Partition I Jump Game Jump Game II Assign Cookies Gas Station Is Subsequence	73.20% 29.30% 26.10% 47.20% 29.00% 44.30%	Easy Medium Hard Easy Medium Medium				
		561 55 45 455 134 392 630	Greedy Array Partition 1 Jump Game Jump Game II Assign Cookies Gas Station Is Subsequence Course Schedule III Queue Reconstruction by Height	73.20% 29.30% 26.10% 47.20% 29.00% 44.30% 8.90%	Easy Medium Hard Easy Medium Medium Medium Medium				
		561 55 45 455 134 392 630 406 418	Greedy Array Partition I Jump Game Jump Game II Assign Cookles Gas Station Is Subsequence Course Schedule III Queue Reconstruction by Height Sentence Screen Fitting	73.20% 29.30% 26.10% 47.20% 29.00% 44.30% 8.90% 54.70% 27.40%	Easy Medium Hard Easy Medium Medium Medium Medium Medium				
		561 55 45 455 134 392 630 406 418	Greedy Array Partition I Jump Game Jump Game II Assign Cookies Gas Station Is Subsequence Course Schedule III Queue Reconstruction by Height Sentence Screen Fitting Find Permutation	73.20% 29.30% 26.10% 47.20% 29.00% 44.30% 8.90% 54.70% 27.40% 52.30%	Easy Medium Hard Easy Medium Medium Medium Medium Medium Medium				
		561 55 45 455 134 392 630 406 418 484 452	Greedy Array Partition 1 Jump Game Jump Game II Assign Cookies Gas Station Is Subsequence Course Schedule III Queue Reconstruction by Height Sentence Screen Fitting Find Permutation Minimum Number of Arrows to Burst Balloons	73.20% 29.30% 26.10% 47.20% 29.00% 44.30% 8.90% 54.70% 27.40% 52.30% 43.40%	Easy Medium Hard Easy Medium Medium Medium Medium Medium Medium Medium Medium Medium				
		561 55 45 455 134 392 630 406 418 484 452	Greedy Array Partition 1 Jump Game Jump Game II Assign Cookies Gas Station Is Subsequence Course Schedule III Queue Reconstruction by Height Sentence Screen Fitting Find Permutation Minimum Number of Arrows to Burst Balloons Candy	73.20% 29.30% 26.10% 47.20% 29.00% 44.30% 8.90% 54.70% 27.40% 52.30% 43.40% 24.30%	Easy Medium Hard Easy Medium Medium Medium Medium Medium Medium Medium Medium Medium				
		561 55 45 455 134 392 630 406 418 484 452 135 316	Greedy Array Partition 1 Jump Game Jump Game II Assign Cookies Gas Station Is Subsequence Course Schedule III Queue Reconstruction by Height Sentence Screen Fitting Find Permutation Minimum Number of Arrows to Burst Balloons Candy Remove Duplicate Letters	73.20% 29.30% 26.10% 47.20% 29.00% 44.30% 8.90% 54.70% 27.40% 52.30% 43.40% 24.30% 29.10%	Easy Medium Hard Easy Medium Medium Medium Medium Medium Medium Medium Hard				
		561 55 45 455 134 392 630 406 418 484 452 135 316 330	Greedy Array Partition I Jump Game Jump Game II Assign Cookies Gas Station Is Subsequence Course Schedule III Queue Reconstruction by Height Sentence Screen Fitting Find Permutation Minimum Number of Arrows to Burst Balloons Candy Remove Duplicate Letters Patching Array	73.20% 29.30% 26.10% 47.20% 29.00% 44.30% 8.90% 54.70% 27.40% 52.30% 43.40% 24.30% 29.10% 31.70%	Easy Medium Hard Easy Medium Medium Medium Medium Medium Medium Medium Hard Hard				
		561 55 45 455 134 392 630 406 418 484 452 135 316 330 621	Greedy Array Partition 1 Jump Game Jump Game II Assign Cookies Gas Station Is Subsequence Course Schedule III Queue Reconstruction by Height Sentence Screen Fitting Find Permutation Minimum Number of Arrows to Burst Balloons Candy Remove Duplicate Letters Patching Array Task Scheduler	73.20% 29.30% 26.10% 47.20% 29.00% 44.30% 8.90% 54.70% 27.40% 52.30% 43.40% 24.30% 29.10% 31.70% 38.60%	Easy Medium Hard Easy Medium				
		561 55 45 45 455 134 392 630 406 418 484 452 135 316 330 621 358	Greedy Array Partition 1 Jump Game Jump Game II Assign Cookies Gas Station Is Subsequence Course Schedule III Queue Reconstruction by Height Sentence Screen Fitting Find Permutation Minimum Number of Arrows to Burst Balloons Candy Remove Duplicate Letters Patching Array Task Scheduler Rearrange String k Distance Apart	73.20% 29.30% 26.10% 47.20% 29.00% 44.30% 8.90% 54.70% 52.30% 43.40% 24.30% 29.10% 31.70% 38.60% 31.80%	Easy Medium Hard Easy Medium Medium Medium Medium Medium Medium Hard Hard Hard Hard Hard Hard				
		561 55 45 455 134 392 630 406 418 484 452 135 316 330 621 358 68	Greedy Array Partition 1 Jump Game Jump Game II Assign Cookies Gas Station Is Subsequence Course Schedule III Queue Reconstruction by Height Sentence Screen Fitting Find Permutation Minimum Number of Arrows to Burst Balloons Candy Remove Duplicate Letters Patching Array Task Scheduler Rearrange String k Distance Apart Text Justification	73.20% 29.30% 26.10% 47.20% 29.00% 44.30% 44.30% 54.70% 52.30% 43.40% 24.30% 29.10% 31.70% 38.60% 31.80%	Easy Medium Hard Easy Medium Hard Hard Hard Hard				
		561 55 45 45 455 134 392 630 406 418 484 452 135 316 330 621 358	Greedy Array Partition 1 Jump Game Jump Game II Assign Cookies Gas Station Is Subsequence Course Schedule III Queue Reconstruction by Height Sentence Screen Fitting Find Permutation Minimum Number of Arrows to Burst Balloons Candy Remove Duplicate Letters Patching Array Task Scheduler Rearrange String k Distance Apart	73.20% 29.30% 26.10% 47.20% 29.00% 44.30% 8.90% 54.70% 52.30% 43.40% 24.30% 29.10% 31.70% 38.60% 31.80%	Easy Medium Hard Easy Medium Medium Medium Medium Medium Medium Hard Hard Hard Hard Hard Hard				
		561 55 45 455 134 392 630 406 418 484 452 135 316 330 621 358 68	Greedy Array Partition 1 Jump Game Jump Game II Assign Cookies Gas Station Is Subsequence Course Schedule III Queue Reconstruction by Height Sentence Screen Fitting Find Permutation Minimum Number of Arrows to Burst Balloons Candy Remove Duplicate Letters Patching Array Task Scheduler Rearrange String k Distance Apart Text Justification	73.20% 29.30% 26.10% 47.20% 29.00% 44.30% 44.30% 54.70% 52.30% 43.40% 24.30% 29.10% 31.70% 38.60% 31.80%	Easy Medium Hard Easy Medium Hard Hard Hard Hard				
		561 55 45 455 134 392 630 406 418 484 452 135 316 330 621 358 68	Greedy Array Partition 1 Jump Game Jump Game Jump Game II Assign Cookies Gas Station Is Subsequence Course Schedule III Queue Reconstruction by Height Sentence Screen Fitting Find Permutation Minimum Number of Arrows to Burst Balloons Candy Remove Duplicate Letters Patching Array Task Scheduler Rearrange String k Distance Apart Text Justification Can Place Flowers	73.20% 29.30% 26.10% 47.20% 29.00% 44.30% 44.30% 54.70% 52.30% 43.40% 24.30% 29.10% 31.70% 38.60% 31.80%	Easy Medium Hard Easy Medium Hard Hard Hard Hard	60.79 using defaultdict			
		561 55 45 45 455 134 392 630 406 418 484 452 135 316 330 621 358 68 605	Greedy Array Partition 1 Jump Game Jump Game II Assign Cookies Gas Station Is Subsequence Course Schedule III Queue Reconstruction by Height Sentence Screen Fitting Find Permutation Minimum Number of Arrows to Burst Balloons Candy Remove Duplicate Letters Patching Array Task Scheduler Rearrange String k Distance Apart Text Justification Can Place Flowers Bit Manipulation Single Number	73.20% 29.30% 26.10% 47.20% 29.00% 44.30% 8.90% 54.70% 27.40% 52.30% 43.40% 24.30% 29.10% 31.70% 38.60% 31.80% 18.60% 29.20%	Easy Medium Hard Easy Medium Hard Hard Hard Hard Easy				
		561 55 45 455 134 392 630 406 418 484 452 135 316 330 621 358 68 605	Greedy Array Partition I Jump Game Jump Game II Assign Cookies Gas Station Is Subsequence Course Schedule III Queue Reconstruction by Height Sentence Screen Fitting Find Permutation Minimum Number of Arrows to Burst Balloons Candy Remove Duplicate Letters Patching Array Task Scheduler Rearrange String k Distance Apart Text Justification Can Place Flowers Bit Manipulation Single Number II	73.20% 29.30% 26.10% 47.20% 29.00% 44.30% 44.30% 52.30% 52.30% 43.40% 24.30% 29.10% 31.70% 38.60% 31.80% 18.60% 53.70% 40.80%	Easy Medium Hard Easy Medium Hard Hard Hard Hard Easy	63.24 using defaultdict			
		561 55 45 455 134 496 406 418 484 452 135 316 621 358 68 605	Array Partition I Jump Game Jump Game Jump Game II Assign Cookies Gas Station Is Subsequence Course Schedule III Queue Reconstruction by Height Sentence Screen Fitting Find Permutation Minimum Number of Arrows to Burst Balloons Candy Remove Duplicate Letters Patching Array Task Scheduler Rearrange String k Distance Apart Text Justification Can Place Flowers Bit Manipulation Single Number Single Number Single Number Single Number Single Number Single Number III	73.20% 29.30% 26.10% 47.20% 29.00% 44.30% 8.90% 54.70% 27.40% 52.30% 43.40% 24.30% 29.10% 31.70% 38.60% 31.80% 618.60% 29.20%	Easy Medium Hard Easy Medium Medium Medium Medium Medium Medium Medium Medium Medium Hard Hard Hard Hard Easy Easy Medium Medium	63.24 using defaultdict 86.24 using defaultdict			
		561 55 45 455 134 392 630 406 418 484 452 135 316 330 621 358 68 605	Array Partition I Jump Game Jump Game Jump Game II Assign Cookies Gas Station Is Subsequence Course Schedule III Queue Reconstruction by Height Sentence Screen Fitting Find Permutation Minimum Number of Arrows to Burst Balloons Candy Remove Duplicate Letters Patching Array Task Scheduler Rearrange String k Distance Apart Text Justification Can Place Flowers Bit Manipulation Single Number Single Number III Single Number III Sum of Two Integers	73.20% 29.30% 26.10% 47.20% 29.00% 44.30% 48.99% 54.70% 27.40% 52.30% 43.40% 29.10% 31.70% 38.60% 31.80% 18.60% 49.20%	Easy Medium Hard Easy Medium Hard Hard Hard Hard Medium Hard Medium Hard Medium Hard Medium Hard Medium Hard Medium Hard Hard Medium Hard Hard Medium Hard Hard Hard	63.24 using defaultdict			
		561 55 45 45 455 134 392 630 406 418 484 452 135 316 330 621 358 68 605	Array Partition I Jump Game Jump Game I Assign Cookies Gas Station Is Subsequence Course Schedule III Queue Reconstruction by Height Sentence Screen Fitting Find Permutation Minimum Number of Arrows to Burst Balloons Candy Remove Duplicate Letters Patching Array Task Scheduler Rearrange String k Distance Apart Text Justification Can Place Flowers Bit Manipulation Single Number II Single Number III Sum of Two Integers Number of 1 Bits	73.20% 29.30% 26.10% 47.20% 29.00% 44.30% 8.90% 54.70% 27.40% 52.30% 43.40% 24.30% 29.10% 31.70% 38.60% 31.80% 18.60% 29.20%	Easy Medium Hard Easy Medium Hard Hard Hard Hard Hard Medium Hard Hard Easy Easy Medium Medium	63.24 using defaultdict 86.24 using defaultdict			
		561 55 45 455 4392 630 406 418 484 452 135 316 330 621 358 68 605	Greedy Array Partition 1 Jump Game Jump Game II Assign Cookies Gas Station Is Subsequence Course Schedule III Queue Reconstruction by Height Sentence Screen Fitting Find Permutation Minimum Number of Arrows to Burst Balloons Candy Remove Duplicate Letters Patching Array Task Scheduler Rearrange String k Distance Apart Text Justification Can Place Flowers Bit Manipulation Single Number II Single Number II Single Number III Sum of Two Integers Number of 1 Bits Counting Bits	73.20% 29.30% 26.10% 47.20% 29.00% 43.90% 54.70% 27.40% 52.30% 43.40% 24.30% 31.70% 38.60% 31.80% 53.70% 40.80% 50.50% 51.20% 60.40%	Easy Medium Hard Easy Medium Hard Hard Hard Hard Easy Medium Medium Hard Hard Hard	63.24 using defaultdict 86.24 using defaultdict Have not idea how to solve it			
		561 55 45 455 134 392 630 406 418 484 452 135 316 621 358 68 605	Array Partition I Jump Game Jump Game Jump Game II Assign Cookies Gas Station Is Subsequence Course Schedule III Queue Reconstruction by Height Sentence Screen Fitting Find Permutation Minimum Number of Arrows to Burst Balloons Candy Remove Duplicate Letters Patching Array Task Scheduler Rearrange String k Distance Apart Text Justification Can Place Flowers Bit Manipulation Single Number Single Number Single Number III Single Number III Single Number III Sum of Two Integers Number of 1 Bits Counting Bits Number Complement	73.20% 29.30% 26.10% 47.20% 29.00% 44.30% 8.90% 54.70% 27.40% 52.30% 43.40% 24.30% 29.10% 31.70% 38.60% 31.80% 61.80% 53.70% 40.80% 50.50% 51.20% 39.10% 60.40% 61.30%	Easy Medium Hard Easy Medium Hard Hard Hard Easy Medium Easy Medium Medium	63.24 using defaultdict 86.24 using defaultdict			
		561 55 45 455 4392 630 406 418 484 452 135 316 330 621 358 68 605	Greedy Array Partition 1 Jump Game Jump Game II Assign Cookies Gas Station Is Subsequence Course Schedule III Queue Reconstruction by Height Sentence Screen Fitting Find Permutation Minimum Number of Arrows to Burst Balloons Candy Remove Duplicate Letters Patching Array Task Scheduler Rearrange String k Distance Apart Text Justification Can Place Flowers Bit Manipulation Single Number II Single Number II Single Number III Sum of Two Integers Number of 1 Bits Counting Bits	73.20% 29.30% 26.10% 47.20% 29.00% 43.90% 54.70% 27.40% 52.30% 43.40% 24.30% 31.70% 38.60% 31.80% 53.70% 40.80% 50.50% 51.20% 60.40%	Easy Medium Hard Easy Medium Hard Hard Hard Hard Easy Medium Medium Hard Hard Hard	63.24 using defaultdict 86.24 using defaultdict Have not idea how to solve it			

	477	Total Hamming Biotones	40.400/	Mardina								
	477 201	Total Hamming Distance	46.40% 33.50%	Medium Medium								
	201 318	Bitwise AND of Numbers Range Maximum Product of Word Lengths	33.50% 43.30%	Medium								
	397	Integer Replacement	29.60%	Medium								
	421	Maximum XOR of Two Numbers in an Array	44.50%	Medium								
		Array										
		String Math1										
			07.000/	Mardina	A							
	2 445	Add Two Numbers	27.20% 46.10%	Medium Medium	Append node into the current list							
	67	Add Two Numbers II	31.50%		stack, insert after dummy to form the correct order							
ADD		Add Binary		Easy	consider each list separately as a list, in other words				1			
	66 369	Plus One	37.80%	Easy	Can to similar as 445, while a brilliant way is to deter						-	
	415	Plus One Linked List	53.90% 41.20%	Medium	Either reverse the list using a stack as 445, or can d		lat for each next i	lode, we return v	metner it has a ca	шту		
		Add Strings		Easy	Similar to 2, do not forget to append the carry, and r							
4	43	Multiply Strings	26.50%	Medium Medium	It is more convenient to have a direct index/pointer f	for the array, and i	educe the index e	each time than ca	iculate the index			
Multiplication	537	Complex Number Multiplication	65.40%									
	311	Sparse Matrix Multiplication	50.60% 16.00%	Medium	In (Al) binner and to find all annuits divine + feet		distance Distance	- d / di. d				
Divide	29	Divide Two Integers		Medium	log(N) binary search to find all possible divisor * fact			na / aivisor = quo	tient and remaind	er		
	166	Fraction to Recurring Decimal	17.20%	Medium	Corner Cases: numerator is 0, denominator is 0, ne	-	-					
Square	69	Sqrt(x)	27.50%	Easy	binary search, corner case is mid*mid <= x, but (mid	1+1)^^2 > x, tnen n	iia is the result					
	367	Valid Perfect Square	37.90%	Easy	standard binary search. 1+3+5+(2n-1) =							
	633	Sum of Square Numbers	27.80%	Easy	Come as east but vtt0 uill							
	50	Pow(x, n)	26.70%	Medium	Same as sqrt, but x**2 will overflow while x*x will no There are lots of different solutions. A quick solution	n is recursion, pow	x, n) = pow(x*x r	n/2) * (1 if n % 2 i	lse x)			
	372	Super Pow	33.80%	Medium	- I I I I I I I I I I I I I I I I I I I		, , , , , , , , , , , , , , , , , , , ,	, ,,	, , , , , , , , , , , , , , , , , , ,			
Power	231	Power of Two	39.80%	Easy	Count 1 in binary format, or bit manipulation x & x-1	should be 0						
	326	Power of Three	39.80%	Easy	log10(n)/log10(3) % 1 == 0							
	342	Power of Four	38.00%	Easy	same as power of two, but need to remove those an	e power of 2 but n	of 4					
	8	String to Integer (atoi)	13.90%	Medium	need to pay attention to corner cases, like max/min							
	65	Valid Number	12.70%	Hard	corner cases, while idx < n and (a or b)	int, the lax						
String	12	Integer to Roman	43.80%	Medium	just consider each case							
g	13	Roman to Integer	44.80%	Easy	same to 12							
	273	Integer to English Words	21.70%	Hard	corner case: 0							
	168	Excel Sheet Column Title	25.30%	Easy	idx is n-1 % 26							
	171	Excel Sheet Column Number	46.20%	Easy	IDA D II 1 70 20							
_		<u> </u>	10.2070	Lucy	if num < 0:							
Base	405	Convert a Number to Hexadecimal	40.80%	Easy	num = num + 2**32							
	504	Base 7	45.50%	Easy								
	660	Remove 9	44.70%	Hard								
	6	ZigZag Conversion	26.40%	Medium								
	498	<u>Diagonal Traverse</u>	46.40%	Medium	row+col=s, row in [0, m-1] thus col in [0 or s-m+1, s	or n -1]						
	48	Rotate Image	37.90%	Medium								
	54	Spiral Matrix	25.30%	Medium								
	59	Spiral Matrix II	38.80%	Medium								
	73	Set Matrix Zeroes	35.50%	Medium								
	661	Image Smoother	47.90%	Easy								
	520	Detect Capital	52.50%	Easy								
	616	Add Bold Tag in String	36.20%	Medium								
Fraverse an	289	Game of Life	36.50%	Medium								
rry or matrix	118	Pascal's Triangle	37.70%	Easy								
	119	Pascal's Triangle II	35.90%	Easy								
	566	Reshape the Matrix	67.90%	Easy								
	391	Perfect Rectangle	25.60%	Hard								
	382	Linked List Random Node	46.60%	Medium								
	398	Random Pick Index	41.80%	Medium								
	384	Shuffle an Array	45.90%	Medium								
		Maximum Distance in Arrays	31.00%	Easy								
	624		44.20%	Easy								
	624 628	Maximum Product of Three Numbers	11.2070		str.count							
		Maximum Product of Three Numbers Judge Route Circle	71.90%	Easy	Str.Courit							
	628			Easy Easy	Sil.Count							
	628 657	Judge Route Circle	71.90%		Su.count							
	628 657 674	Judge Route Circle Longest Continuous Increasing Subsequence	71.90% 44.00%	Easy	Su. COURT							
	628 657 674 20	Judge Route Circle Longest Continuous Increasing Subsequence Valid Parentheses	71.90% 44.00% 32.90%	Easy Easy	SIOUIII							
	628 657 674 20 32	Judge Route Circle Longest Continuous Increasing Subsequence Valid Parentheses Longest Valid Parentheses	71.90% 44.00% 32.90% 23.00%	Easy Easy Hard	SULUUII							
	628 657 674 20 32 224	Judge Route Circle Longest Continuous Increasing Subsequence Valid Parentheses Longest Valid Parentheses Basic Calculator	71.90% 44.00% 32.90% 23.00% 26.30%	Easy Easy Hard Hard	SULCOURT							

		150	Evaluate Reverse Polish Notation	26.60%	Medium				
Stack		591	Tag Validator	17.90%	Hard	corner cases			
Stack		385	-	30.00%	Medium	correr cases			
			Mini Parser						
		439	Ternary Expression Parser	50.30%	Medium				
		394 496	Decode String Next Greater Element I	40.90% 57.60%	Medium Easy				
		503		47.20%	Medium				
		556	Next Greater Element II Next Greater Element III	27.20%	Medium				
	travaraa	218 41	The Skyline Problem	26.60% 25.20%	Hard Hard	mark pagetive to 0, and turn value in existing index to pagetive	finally the first positive position is missing		
	traverse	268	First Missing Positive			mark negative to 0, and turn value in exisiting index to negative, 1++n = (n+1)*n/2	, finally the first positive position is missing		
	traverse	169	Missing Number	44.00% 45.80%	Easy	17711 - (1171) 11/2			
	traverse		Majority Element		Easy				
		229 217	Majority Element II	28.20% 44.80%	Medium	set			
missing or extra number		217	Contains Duplicate Contains Duplicate II	32.00%	Easy				
Hamboi	hashtable				Easy	hashtable to store the last idx			
	hashtable	220	Contains Duplicate III	19.20%	Medium				
	traverse	442 448	Find All Duplicates in an Array	53.80%	Medium	add n is a good idea			
	traverse		Find All Numbers Disappeared in an Array	52.50%	Easy				
	hashtable	389	Find the Difference	51.50%	Easy				
	hashtable	645	Set Mismatch	40.60%	Easy				
	traverse	56	Merge Intervals	29.30%	Medium				
	traverse	57	Insert Interval	27.00%	Hard				
	traverse	252	Meeting Rooms	46.50%	Easy				
	greedy	253	Meeting Rooms II	38.70%	Medium				
Interval	binary search	436	Find Right Interval	41.20%	Medium				
		435	Non-overlapping Intervals	40.40%	Medium	greedy			
	traverse	495	Teemo Attacking	52.00%	Medium				
	traverse	163	Missing Ranges	25.50%	Medium				
	traverse	228	Summary Ranges	28.90%	Medium				
		352	Data Stream as Disjoint Intervals	39.50%	Hard				
	hashtable	202	Happy Number	40.00%	Easy				
	math	258	Add Digits	50.70%	Easy				
	math	507	Perfect Number	32.90%	Easy				
	traverse	306	Additive Number	27.30%	Medium				
	traverse	38	Count and Say	33.60%	Easy				
	hashtable	204	Count Primes	26.40%	Easy	corner case, and how to reduce duplicate multiply			
Number	math	246	<u>Strobogrammatic Number</u>	39.40%	Easy				
	math	247	Strobogrammatic Number II	39.10%	Medium	DD.			
	math	248	Strobogrammatic Number III	31.20%	Hard	DP			
	math	263	<u>Ugly Number</u>	38.80%	Easy	<=0			
	math	264	Ugly Number II	32.10%	Medium				
	math	313	Super Ugly Number	37.30%	Medium				
	math	172	Factorial Trailing Zeroes	35.50%	Easy				
	math	625	Minimum Factorization	27.70%	Medium				
		479	Largest Palindrome Product	17.90%	Easy				
	math	357	Count Numbers with Unique Digits	45.60%	Medium				
	math	233	Number of Digit One	27.90%	Hard				
	math	396	Rotate Function	31.50%	Medium				
	math	483	Smallest Good Base	31.40%	Hard				
	math	453	Minimum Moves to Equal Array Elements	46.80%	Easy	min			
	math	462	Minimum Moves to Equal Array Elements II	51.30%	Medium	median			
	math	296	Best Meeting Point	51.30%	Hard	median			
	math	573	Squirrel Simulation	46.70%	Medium				
	traverse	31	Next Permutation	28.50%	Medium				
		386	Lexicographical Numbers	40.50%	Medium				
		60	Permutation Sequence	27.80%	Medium				
Digital or String		440	K-th Smallest in Lexicographical Order	23.30%	Hard				
order		555	Split Concatenated Strings	28.50%	Medium				
		400	Nth Digit	30.10%	Easy				
		179	Largest Number	22.10%	Medium				
	stack	402	Remove K Digits	26.10%	Medium				
	traverse	414	Third Maximum Number	27.50%	Easy	corner cases			
	heap	378	Kth Smallest Element in a Sorted Matrix	43.80%	Medium	can also binary search			
Top N	heap heap	373 215	Find K Pairs with Smallest Sums Kth Largest Element in an Array	30.40% 38.40%	Medium Medium	· ·			

						hucket						
	heap	347	Top K Frequent Elements	47.30%	Medium	bucket LFU/LRU						
		632	Smallest Range	43.50%	Hard							
	heap	659	Split Array into Consecutive Subsequences	33.20%	Medium							
	heap	502	IPO	34.70%	Hard							
	bucket	539	Minimum Time Difference	45.50%	Medium							
	bucket	128	Longest Consecutive Sequence	36.00%	Hard	can solve use hashmap too						
	bucket	164	Maximum Gap	29.10%	Hard							
		463	Island Perimeter	56.80%	Easy							
		492	Construct the Rectangle	49.10%	Easy							
	math	223	Rectangle Area	32.40%	Medium							
Shape	math	335		24.70%	Hard							
опарс	maur	593	Self Crossing Valid Square	36.10%	Medium							
		469		30.60%	Medium							
	math		Convex Polygon									
	math	587 292	Erect the Fence	55.10%	Hard Fasy	100 : 11						
gitic Game			Nim Game			1,2,3 win, 4 loss, 5,6,7 win, 8 loss						
	math	319	Bulb Switcher	42.20%	Medium							
	math	672	Bulb Switcher II	45.60%	Medium							
	dp	376	Wiggle Subsequence	35.20%	Medium							
		280	Wiggle Sort	56.20%	Medium							
2 Pattern		324	Wiggle Sort II	25.50%	Medium							
		334	Increasing Triplet Subsequence	38.60%	Medium							
		456	132 Pattern	28.20%	Medium							
		360	Sort Transformed Array	43.60%	Medium							
		634	Find the Derangement of An Array	26.60%	Medium							
	traverse	370	Range Addition	54.60%	Medium							
		598	Range Addition II	44.40%	Easy	tricky						
		640	Solve the Equation	38.60%	Medium							
		544	Output Contest Matches	72.10%	Medium							
		565	Array Nesting	47.10%	Medium							
	traverse	390	Elimination Game	40.50%	Medium	binary						
	traverse	420	Strong Password Checker	20.10%	Hard	,						
	traverse	393	UTF-8 Validation	34.80%	Medium	do int comparison instead of string						
	traverse	459	Repeated Substring Pattern	38.50%	Easy							
	traverse	465	Optimal Account Balancing	34.00%	Hard							
	traverse	481	Magical String	45.20%	Medium							
	traverse	506	Relative Ranks	47.40%	Easy							
	traverse	521	Longest Uncommon Subsequence I	51.30%	Easy							
	traverse	521	Longest Uncommon Subsequence II	28.70%	Medium							
	traverse	531	Lonely Pixel I	51.60%	Medium							
	traverse	533	Lonely Pixel II	40.00%	Medium							
	traverse	548	Split Array with Equal Sum	29.30%	Medium							
	traverse	277	Find the Celebrity	35.30%	Medium							
	traverse	482	License Key Formatting	41.20%	Medium	replace and upper is faster than do each position						
	traverse	412	Fizz Buzz	58.80%	Easy							
	traverse	14	Longest Common Prefix	31.10%	Easy							
	traverse, KMP	28	Implement strStr()	27.60%	Easy							
	traverse	161	One Edit Distance	30.90%	Medium							
	traverse	58	Length of Last Word	31.50%	Easy							
	traverse	434	Number of Segments in a String	37.00%	Easy							
	traverse	485	Max Consecutive Ones	54.50%	Easy							
	traverse	487	Max Consecutive Ones II	44.40%	Medium	Follow Up: if we can flip at most K zero, we store the	previous at most K	zero index in a queue, s	o that we can calculate	the len when new v	alue come in. Que	eue is better
	traverse	157	Read N Characters Given Read4	29.10%	Easy							
	traverse	158	Read N Characters Given Read4 II - Call multiple times	24.30%	Hard							
	traverse	165	Compare Version Numbers	19.70%	Medium							
			Hash Table									
		349	Intersection of Two Arrays	46.60%	Easy	set.intersection(t)						
		350	Intersection of Two Arrays II	44.30%	Easy	(countera & counterb).elements()						
		599	Minimum Index Sum of Two Lists	57.50%	Easy							
		299	Bulls and Cows	34.00%	Medium							
		49	Group Anagrams	33.30%	Medium	count sort may make the str sort quicker						
		242	Valid Anagram	45.70%	Easy	hashmap is faster than len_26 char						
				46.70%	Easy	nasimap is laster triali lett_20 trial						
		202										
		383	Ransom Note									
		447	Number of Boomerangs	44.10%	Easy							
	gcd					gcd, and re						

	gcd	365	Water and Jug Problem	26.70%	Medium	
	gcd	149	Max Points on a Line	15.40%	Hard	
		249	Group Shifted Strings	40.30%	Medium	
		187	Repeated DNA Sequences	30.60%	Medium	
word distance	travese	243	Shortest Word Distance	51.60%	Easy	just traverse the word list, update the distance, the initial set is important
not edit		244	Shortest Word Distance II	36.30%	Medium	
distance	travese	245	Shortest Word Distance III	49.80%	Medium	only need to slightly modify 243
		387	First Unique Character in a String	46.40%	Easy	
		594	Longest Harmonious Subsequence	36.20%	Easy	
		423	Reconstruct Original Digits from English	43.20%	Medium	
		451	Sort Characters By Frequency	50.60%	Medium	
		500	Keyboard Row	60.30%	Easy	
		575	Distribute Candies	0.648	Easy	
		554	Brick Wall	41.60%	Medium	
		274	H-Index	32.70%	Medium	
		275	H-Index II	33.90%	Medium	
		609	Find Duplicate File in System	54.80%	Medium	
		336	Palindrome Pairs	25.60%	Hard	