		#
Group	Tag	#
N-Sum	hashtable	1
	hashtable	532
	two pointers	167
	hashtable	170
	hashtable	15
	two pointers	16
	two pointers	259
	two pointers	611
	hashtable	18
	hashtable	454
Fixed Length Sliding Window	queue	346
	two pointer	643
	heap, deque	239
	heap, median	295
	heap, median	480
	hashmap	438
	hashmap	567
	hashmap	30
variant length sliding window	hashmap	3
	expression	395
	two pointers	159
	two pointers	340
	two pointers	424
	subsequence	524
	two pointers	76
	two pointers	209
subarray sum	hashmap	325
	hashmap	525
	hashmap	523
	hashmap	560
Array Organization	two pointers	26
	two pointers	80
	two pointers	27
	two pointers	283
	two pointers	88
	pointers, counting s	75
Water Catch	two pointers	11
	two pointers	42
	stack	84
	stack	85
	heap	407
Circle	fast and slow	141
	fast and slow	142

Circle	fast and slow	287
		160
Remove Node from Linked List		237
		83
		82
		203
		19
move nodes		61
		86
		328
		206
		92
		21
		23
		147
	fast and slow	148
	fast and slow	143
		24
		25
reverse array		344
		541
		345
		151
		186
		557
		189
		7
	bit	190
tradictional BT	backtracking	17
	backtracking	401
	backtracking	22
	backtracking	78
	backtracking	90
	backtracking	491
	backtracking	46
	backtracking	47
	backtracking	77
	backtracking	39
	backtracking	40
	backtracking	216
	ynamic programmin	377
	backtracking	638
	backtracking	254
		5
		647
		9
		125
		680

Palindrome	**KMP**	214
		409
		266
		267
		234
		564
Rotated Sorted Array		35
		34
		162
		74
		240
		33
		81
		153
		154
		540
		4
		315
		374
		658
		278
		302
		441
		475
Games..		36
		37
		488
		51
		52
		351
ab to 12	string	205
	string	290
	string	291
	string	468
	string	93
	string	526
	string	131
	string	132
		79
		212
		127
		126
		422
		425
		408
		288
abbreviation		320
		411
		527
		282

		679
		53
		152
	traverse	238
	traverse	581
		303
		304
		121
stock dp		122
		123
		188
		309
		70
		91
		639
O(n) DP, can do O(1) Space		198
		213
		337
		256
		265
		276
		494
		300
		673
O(n) DP but need to check all possible position before		96
		95
		279
		322
		343
		139
		140
		120
		62
		63
		562
M*N path O(M) space		221
		64
		174
		568
		551
		552
M*N path M*N Space		361
		44
		10
String Matching (M+1)*(N+1)		72
		87
		97
		583
		115
		89

		375
		629
		600
		678
		241
		312
		321
		354
		363
		368
		410
		416
		403
		293
		294
		464
		486
		413
		446
		467
		466
		472
		474
		514
		516
		517
		546
		553
		471
		576
list		380
		381
linked list		146
		460
		432
stack		155
		225
		232
Iterator		251
		281
		284
		341
		604
		353
		379
	hashtable	631
	hashtable	359
		635
		362
	trie	208
	trie	211

	trie	677
	trie	642
		676
	hashtable	535
		271
		355
		348
		588
Traversal	inorder	94
	preorder	144
	postorder	145
	inorder	173
	inorder	98
	inorder	99
	inorder	285
	preorder	606
	inorder	105
	inorder	106
	preorder	255
	inorder	501
	inorder	653
Level Order Traversal (mostly BFS)		102
		107
		339
		364
		103
		515
		116
		117
		199
		623
		637
BFS/DFS (iterative/recursive)	postorder	110
		100
		101
		226
		617
		314
		104
	DFS	563
		111
		257
		404
	DFS	112
	DFS	113
	BFS/DFS	437
	DFS	124
	BFS/DFS	513
	BFS/DFS	129
	BFS/DFS	298
	DFS	250

	left and right	549
	left and right	543
		366
		310
		508
	inorder	538
		156
		545
Tree Conversion	stack	114
	stack	536
	binary	108
	binary	109
Serialization		572
		331
	stack	297
		449
	binary search	222
	hashtable	133
	hashtable	138
	adjacency list	323
	adjacency list	261
Binary Search Tree		230
		235
		236
		270
		272
	postorder	333
		450
		530
		669
Topological Sort		207
		210
		444
	traverse	269
		329
BFS		582
BFS		130
BFS		200
BFS		305
BFS		286
		490
		505
		499
		301
		317
DFS		332
		399
DFS		547
BFS		417
BFS		419
BFS		473

BFS	529
BFS on all	542
	307
	308
	327
	493
	561
	55
	45
	455
	134
	392
	630
	406
	418
	484
	452
	135
	316
	330
	621
	358
	68
	605
	136
	137
	260
	371
	191
	338
	476
	461
	477
	201
	318
	397
	421
ADD	2
	445
	67
	66
	369
	415
Multiplication	43
	537
	311

Divide	29
	166
Square	69
	367
	633
Power	50
	372
	231
	326
	342
String	8
	65
	12
	13
	273
Base	168
	171
	405
	504
	660
Traverse an arry or matrix	6
	498
	48
	54
	59
	73
	661
	520
	616
	289
	118
	119
	566
	391
	382
	398
	384
	624
	628
	657
	674
	20
	32
	224
	227
Stack	71
	388
	150
	591
	385
	439
	394

		496
		503
		556
		218
	traverse	41
	traverse	268
	traverse	169
	traverse	229
missing or extra number	hashtable	217
	hashtable	219
	hashtable	220
	traverse	442
	traverse	448
	hashtable	389
	hashtable	645
	traverse	56
	traverse	57
	traverse	252
	greedy	253
Interval	binary search	436
		435
	traverse	495
	traverse	163
	traverse	228
		352
	hashtable	202
	math	258
	math	507
	traverse	306
	traverse	38
	hashtable	204
Number	math	246
	math	247
	math	248
	math	263
	math	264
	math	313
	math	172
	math	625
		479
	math	357
	math	233
	math	396
	math	483
	math	453
	math	462
	math	296
	math	573
	traverse	31
		386
		60
Digital or String		440

order		555
		400
		179
	stack	402
	traverse	414
	heap	378
Top N	heap	373
	heap	215
	heap	347
		632
	heap	659
	heap	502
	bucket	539
	bucket	128
	bucket	164
		463
		492
	math	223
Shape	math	335
		593
	math	469
	math	587
Logitic Game		292
	math	319
	math	672
132 Pattern	dp	376
		280
		324
		334
		456
		360
		634
	traverse	370
		598
		640
		544
		565
	traverse	390
	traverse	420
	traverse	393
	traverse	459
	traverse	465
	traverse	481
	traverse	506
	traverse	521
	traverse	522
	traverse	531
	traverse	533
	traverse	548
	traverse	277
	traverse	482

	traverse	412
	traverse	14
	traverse, KMP	28
	traverse	161
	traverse	58
	traverse	434
	traverse	485
	traverse	487
	traverse	157
	traverse	158
	traverse	165
		349
		350
		599
		299
		49
		242
		383
		447
		356
	gcd	592
	gcd	365
	gcd	149
		249
		187
word distance not edit distance	travese	243
		244
	travese	245
		387
		594
		423
		451
		500
		575
		554
		274
		275
		609
		336

Might be able to solve without looking at the solution

	Acceptance	Difficulty
Two Sum		
K-diff Pairs in an Array		
Two Sum II - Input array is sorted		
Two Sum III - Data structure design		
3Sum		
3Sum Closest		
3Sum Smaller		
Valid Triangle Number		
4Sum		
4Sum II		
Moving Average from Data Stream		
Maximum Average Subarray I		
Sliding Window Maximum		
Find Median from Data Stream		
Sliding Window Median		
Find All Anagrams in a String		
Permutation in String		
Substring with Concatenation of All Words		
Longest Substring Without Repeating Characters		
Longest Substring with At Least K Repeating Characters		
Longest Substring with At Most Two Distinct Characters		
Longest Substring with At Most K Distinct Characters		
Longest Repeating Character Replacement		
Longest Word in Dictionary through Deleting		
Minimum Window Substring		
Minimum Size Subarray Sum		
Maximum Size Subarray Sum Equals k		
Contiguous Array		
Continuous Subarray Sum		
Subarray Sum Equals K		
Remove Duplicates from Sorted Array		
Remove Duplicates from Sorted Array II		
Remove Element		
Move Zeroes		
Merge Sorted Array		
Sort Colors		
Container With Most Water		
Trapping Rain Water		
Largest Rectangle in Histogram		
Maximal Rectangle		
Trapping Rain Water II		
Linked List Cycle		
Linked List Cycle II		

Find the Duplicate Number		n(logn) by binary se
Intersection of Two Linked Lists		can solve elegently
Delete Node in a Linked List		do not need to char
Remove Duplicates from Sorted List		
Remove Duplicates from Sorted List II		
Remove Linked List Elements		
Remove Nth Node From End of List		
Rotate Lists		corner case: k = 0,
Partition List		
Odd Even Linked List		can have short, ele
Reverse Linked List		dummy.next, node.
Reverse Linked List II		keep, pre and m-th,
Merge Two Sorted Lists		
Merge k Sorted Lists		
Insertion Sort List		
Sort List		fast slow pointers,
Reorder List		find right part, rever
Swap Nodes in Pairs		recursive should be
Reverse Nodes in k-Group		recursive
Reverse String		
Reverse String II		
Reverse Vowels of a String		
Reverse Words in a String		
Reverse Words in a String II		
Reverse Words in a String III		
Rotate Array		nums = nums[::-1]
Reverse Integer		
Reverse Bits		get the bin in each
Letter Combinations of a Phone Number		
Binary Watch		
Generate Parentheses		
Subsets		
Subsets II		
Increasing Subsequences		
Permutations		
Permutations II		
Combinations		
Combination Sum		
Combination Sum II		
Combination Sum III		
Combination Sum IV		
Shopping Offers		
Factor Combinations		
Two Pointers		
Longest Palindromic Substring	25.10%	Medium
Palindromic Substrings	56.00%	Medium
Palindrome Number	34.70%	Easy
Valid Palindrome	25.90%	Easy
Valid Palindrome II	28.10%	Easy

Shortest Palindrome	23.60%	Hard
Longest Palindrome	45.10%	Easy
Palindrome Permutation	56.30%	Easy
Palindrome Permutation II	31.50%	Medium
Palindrome Linked List	32.20%	Easy
Find the Closest Palindrome	14.00%	Hard
Binary Search		
Search Insert Position	39.40%	Easy
Search for a Range	31.10%	Medium
Find Peak Element	36.70%	Medium
Search a 2D Matrix	35.30%	Medium
Search a 2D Matrix II	38.10%	Medium
Search in Rotated Sorted Array	32.10%	Medium
Search in Rotated Sorted Array II	32.80%	Medium
Find Minimum in Rotated Sorted Array	39.30%	Medium
Find Minimum in Rotated Sorted Array II	36.70%	Hard
Single Element in a Sorted Array	53.50%	Medium
Median of Two Sorted Arrays	21.30%	Hard
Count of Smaller Numbers After Self	34.10%	Hard
Guess Number Higher or Lower	34.50%	Easy
Find K Closest Elements	35.30%	Medium
First Bad Version	24.80%	Easy
Smallest Rectangle Enclosing Black Pixels	44.80%	Hard
Arranging Coins	36.10%	Easy
Heaters	29.60%	Easy
Backtracking		
Valid Sudoku	34.90%	Medium
Sudoku Solver	29.20%	Hard
Zuma Game	36.20%	Hard
N-Queens	30.00%	Hard
N-Queens II	43.80%	Hard
Android Unlock Patterns	43.20%	Medium
Isomorphic Strings	33.20%	Easy
Word Pattern	32.60%	Easy
Word Pattern II	37.80%	Hard
Validate IP Address	20.20%	Medium
Restore IP Addresses	26.60%	Medium
Beautiful Arrangement	54.10%	Medium
Palindrome Partitioning	32.00%	Medium
Palindrome Partitioning II	23.80%	Hard
Word Search	26.10%	Medium
Word Search II	22.90%	Hard
Word Ladder	19.30%	Medium
Word Ladder II	13.90%	Hard
Valid Word Square	36.30%	Easy
Word Squares	42.60%	Hard
Valid Word Abbreviation	27.60%	Easy
Unique Word Abbreviation	16.10%	Medium
Generalized Abbreviation	44.30%	Medium
Minimum Unique Word Abbreviation	31.80%	Hard
Word Abbreviation	34.80%	Hard
Expression Add Operators	29.30%	Hard

24 Game	38.60%	Hard
Dynamic Programming (try to save space)		
Maximum Subarray	39.20%	Easy
Maximum Product Subarray	25.10%	Medium
Product of Array Except Self	48.20%	Medium
Shortest Unsorted Continuous Subarray	28.30%	Easy
Range Sum Query - Immutable	28.00%	Easy
Range Sum Query 2D - Immutable	24.00%	Medium
Best Time to Buy and Sell Stock	40.30%	Easy
Best Time to Buy and Sell Stock II	46.30%	Easy
Best Time to Buy and Sell Stock III	28.80%	Hard
Best Time to Buy and Sell Stock IV	24.10%	Hard
Best Time to Buy and Sell Stock with Cooldown	40.20%	Medium
Climbing Stairs	39.30%	Easy
Decode Ways	19.30%	Medium
Decode Ways II	18.90%	Hard
House Robber	38.20%	Easy
House Robber II	33.50%	Medium
House Robber III	42.60%	Medium
Paint House	45.90%	Easy
Paint House II	37.70%	Hard
Paint Fence	34.20%	Easy
Target Sum	43.80%	Medium
Longest Increasing Subsequence	37.90%	Medium
Number of Longest Increasing Subsequence	30.80%	Medium
Unique Binary Search Trees	40.40%	Medium
Unique Binary Search Trees II	31.00%	Medium
Perfect Squares	36.00%	Medium
Coin Change	26.20%	Medium
Integer Break	45.50%	Medium
Word Break	29.20%	Medium
Word Break II	22.70%	Hard
Triangle	33.10%	Medium
Unique Paths	40.20%	Medium
Unique Paths II	31.30%	Medium
Longest Line of Consecutive One in Matrix	36.30%	Medium
Maximal Square	28.00%	Medium
Minimum Path Sum	37.80%	Medium
Dungeon Game	23.40%	Hard
Maximum Vacation Days	40.20%	Hard
Student Attendance Record I	44.20%	Easy
Student Attendance Record II	27.70%	Hard
Bomb Enemy	38.60%	Medium
Wildcard Matching	19.60%	Hard
Regular Expression Matching	23.90%	Hard
Edit Distance	31.10%	Hard
Scramble String	28.70%	Hard
Interleaving String	24.30%	Hard
Delete Operation for Two Strings	40.80%	Medium
Distinct Subsequences	31.10%	Hard
Gray Code	40.30%	Medium

Guess Number Higher or Lower II	35.60%	Medium
K Inverse Pairs Array	15.90%	Medium
Non-negative Integers without Consecutive Ones	21.60%	Hard
Valid Parenthesis String	26.20%	Medium
Different Ways to Add Parentheses	42.70%	Medium
Burst Balloons	42.20%	Hard
Create Maximum Number	24.30%	Hard
Russian Doll Envelopes	32.00%	Hard
Max Sum of Rectangle No Larger Than K	32.50%	Hard
Largest Divisible Subset	33.50%	Medium
Split Array Largest Sum	35.20%	Hard
Partition Equal Subset Sum	38.50%	Medium
Frog Jump	31.60%	Hard
Flip Game	54.90%	Easy
Flip Game II	45.90%	Medium
Can I Win	23.80%	Medium
Predict the Winner	44.40%	Medium
Arithmetic Slices	54.90%	Medium
Arithmetic Slices II - Subsequence	25.30%	Hard
Unique Substrings in Wraparound String	31.40%	Medium
Count The Repetitions	26.60%	Hard
Concatenated Words	29.70%	Hard
Ones and Zeroes	37.70%	Medium
Freedom Trail	34.90%	Hard
Longest Palindromic Subsequence	42.40%	Medium
Super Washing Machines	35.60%	Hard
Remove Boxes	29.60%	Hard
Optimal Division	53.70%	Medium
Encode String with Shortest Length	41.60%	Hard
Out of Boundary Paths	0.324	Hard
Design		
Insert Delete GetRandom O(1)	38.90%	Medium
Insert Delete GetRandom O(1) - Duplicates allowed	28.50%	Hard
LRU Cache	17.00%	Hard
LFU Cache	22.40%	Hard
All O`one Data Structure	27.50%	Hard
Min Stack	27.60%	Easy
Implement Stack using Queues	32.10%	Easy
Implement Queue using Stacks	35.90%	Easy
Flatten 2D Vector	39.80%	Medium
Zigzag Iterator	49.60%	Medium
Peeking Iterator	35.30%	Medium
Flatten Nested List Iterator	40.30%	Medium
Design Compressed String Iterator	30.20%	Easy
Design Snake Game	26.10%	Medium
Design Phone Directory	31.40%	Medium
Design Excel Sum Formula	17.00%	Hard
Logger Rate Limiter	59.00%	Easy
Design Log Storage System	42.60%	Medium
Design Hit Counter	53.30%	Medium
Implement Trie (Prefix Tree)	27.00%	Medium
Add and Search Word - Data structure design	21.50%	Medium

Map Sum Pairs	54.00%	Medium
Design Search Autocomplete System	26.70%	Hard
Implement Magic Dictionary	51.10%	Medium
Encode and Decode TinyURL	74.60%	Medium
Encode and Decode Strings	26.20%	Medium
Design Twitter	25.00%	Medium
Design Tic-Tac-Toe	45.60%	Medium
Design In-Memory File System	30.90%	Hard
Tree		
Binary Tree Inorder Traversal	45.30%	Medium
Binary Tree Preorder Traversal	44.10%	Medium
Binary Tree Postorder Traversal	39.40%	Hard
Binary Search Tree Iterator	40.30%	Medium
Validate Binary Search Tree	22.90%	Medium
Recover Binary Search Tree	29.30%	Hard
Inorder Successor in BST	36.00%	Medium
Construct String from Binary Tree	54.90%	Easy
Construct Binary Tree from Preorder and Inorder Traversal	31.50%	Medium
Construct Binary Tree from Inorder and Postorder Traversal	31.50%	Medium
Verify Preorder Sequence in Binary Search Tree	39.60%	Medium
Find Mode in Binary Search Tree	38.40%	Easy
Two Sum IV - Input is a BST	50.50%	Easy
Binary Tree Level Order Traversal	38.40%	Medium
Binary Tree Level Order Traversal II	39.10%	Easy
Nested List Weight Sum	60.90%	Easy
Nested List Weight Sum II	51.40%	Medium
Binary Tree Zigzag Level Order Traversal	33.50%	Medium
Find Largest Value in Each Tree Row	53.90%	Medium
Populating Next Right Pointers in Each Node	36.90%	Medium
Populating Next Right Pointers in Each Node II	33.60%	Medium
Binary Tree Right Side View	39.80%	Medium
Add One Row to Tree	50.10%	Medium
Average of Levels in Binary Tree	63.20%	Easy
Balanced Binary Tree	36.90%	Easy
Same Tree	45.90%	Easy
Symmetric Tree	37.90%	Easy
Invert Binary Tree	50.90%	Easy
Merge Two Binary Trees	73.40%	Easy
Binary Tree Vertical Order Traversal	36.20%	Medium
Maximum Depth of Binary Tree	51.80%	Easy
Binary Tree Tilt	49.00%	Easy
Minimum Depth of Binary Tree	32.70%	Easy
Binary Tree Paths	36.90%	Easy
Sum of Left Leaves	46.60%	Easy
Path Sum	33.50%	Easy
Path Sum II	32.60%	Medium
Path Sum III	39.30%	Easy
Binary Tree Maximum Path Sum	25.50%	Hard
Find Bottom Left Tree Value	55.80%	Medium
Sum Root to Leaf Numbers	35.90%	Medium
Binary Tree Longest Consecutive Sequence	40.50%	Medium
Count Univalue Subtrees	41.20%	Medium

Binary Tree Longest Consecutive Sequence II	36.80%	Medium
Diameter of Binary Tree	42.80%	Easy
Find Leaves of Binary Tree	58.60%	Medium
Minimum Height Trees	28.70%	Medium
Most Frequent Subtree Sum	52.00%	Medium
Convert BST to Greater Tree	52.90%	Medium
Binary Tree Upside Down	43.60%	Medium
Boundary of Binary Tree	28.20%	Medium
Flatten Binary Tree to Linked List	34.30%	Medium
Construct Binary Tree from String	38.30%	Medium
Convert Sorted Array to Binary Search Tree	41.40%	Easy
Convert Sorted List to Binary Search Tree	33.40%	Medium
Subtree of Another Tree	43.90%	Easy
Verify Preorder Serialization of a Binary Tree	35.70%	Medium
Serialize and Deserialize Binary Tree	32.60%	Hard
Serialize and Deserialize BST	42.20%	Medium
Count Complete Tree Nodes	27.10%	Medium
Clone Graph	25.10%	Medium
Copy List with Random Pointer	26.50%	Medium
Number of Connected Components in an Undirected Graph	47.50%	Medium
Graph Valid Tree	37.30%	Medium
Kth Smallest Element in a BST	43.00%	Medium
Lowest Common Ancestor of a Binary Search Tree	38.50%	Easy
Lowest Common Ancestor of a Binary Tree	29.60%	Medium
Closest Binary Search Tree Value	38.90%	Easy
Closest Binary Search Tree Value II	38.40%	Hard
Largest BST Subtree	30.20%	Medium
Delete Node in a BST	35.80%	Medium
Minimum Absolute Difference in BST	47.50%	Easy
Trim a Binary Search Tree	59.60%	Easy
Course Schedule	31.30%	Medium
Course Schedule II	26.80%	Medium
Sequence Reconstruction	19.70%	Medium
Alien Dictionary	22.80%	Hard
Longest Increasing Path in a Matrix	35.90%	Hard
Kill Process	42.50%	Medium
Surrounded Regions	17.90%	Medium
Number of Islands	33.60%	Medium
Number of Islands II	38.60%	Hard
Walls and Gates	43.50%	Medium
The Maze	42.40%	Medium
The Maze II	36.10%	Medium
The Maze III	31.00%	Hard
Remove Invalid Parentheses	34.90%	Hard
Shortest Distance from All Buildings	33.60%	Hard
Reconstruct Itinerary	28.70%	Medium
Evaluate Division	40.30%	Medium
Friend Circles	49.00%	Medium
Pacific Atlantic Water Flow	33.20%	Medium
Battleships in a Board	61.20%	Medium
Matchsticks to Square	34.30%	Medium

Minesweeper	51.80%	Medium
01 Matrix	32.40%	Medium
Segment Tree		
Range Sum Query - Mutable	19.60%	Medium
Range Sum Query 2D - Mutable	21.50%	Hard
Count of Range Sum	29.30%	Hard
Reverse Pairs	19.10%	Hard
Greedy		
Array Partition I	73.20%	Easy
Jump Game	29.30%	Medium
Jump Game II	26.10%	Hard
Assign Cookies	47.20%	Easy
Gas Station	29.00%	Medium
Is Subsequence	44.30%	Medium
Course Schedule III	8.90%	Medium
Queue Reconstruction by Height	54.70%	Medium
Sentence Screen Fitting	27.40%	Medium
Find Permutation	52.30%	Medium
Minimum Number of Arrows to Burst Balloons	43.40%	Medium
Candy	24.30%	Hard
Remove Duplicate Letters	29.10%	Hard
Patching Array	31.70%	Hard
Task Scheduler	38.60%	Medium
Rearrange String k Distance Apart	31.80%	Hard
Text Justification	18.60%	Hard
Can Place Flowers	29.20%	Easy
Bit Manipulation		
Single Number	53.70%	Easy
Single Number II	40.80%	Medium
Single Number III	50.50%	Medium
Sum of Two Integers	51.20%	Easy
Number of 1 Bits	39.10%	Easy
Counting Bits	60.40%	Medium
Number Complement	61.30%	Easy
Hamming Distance	70.70%	Easy
Total Hamming Distance	46.40%	Medium
Bitwise AND of Numbers Range	33.50%	Medium
Maximum Product of Word Lengths	43.30%	Medium
Integer Replacement	29.60%	Medium
Maximum XOR of Two Numbers in an Array	44.50%	Medium
Array String Math1		
Add Two Numbers	27.20%	Medium
Add Two Numbers II	46.10%	Medium
Add Binary	31.50%	Easy
Plus One	37.80%	Easy
Plus One Linked List	53.90%	Medium
Add Strings	41.20%	Easy
Multiply Strings	26.50%	Medium
Complex Number Multiplication	65.40%	Medium
Sparse Matrix Multiplication	50.60%	Medium

Divide Two Integers	16.00%	Medium
Fraction to Recurring Decimal	17.20%	Medium
Sqrt(x)	27.50%	Easy
Valid Perfect Square	37.90%	Easy
Sum of Square Numbers	27.80%	Easy
Pow(x, n)	26.70%	Medium
Super Pow	33.80%	Medium
Power of Two	39.80%	Easy
Power of Three	39.80%	Easy
Power of Four	38.00%	Easy
String to Integer (atoi)	13.90%	Medium
Valid Number	12.70%	Hard
Integer to Roman	43.80%	Medium
Roman to Integer	44.80%	Easy
Integer to English Words	21.70%	Hard
Excel Sheet Column Title	25.30%	Easy
Excel Sheet Column Number	46.20%	Easy
Convert a Number to Hexadecimal	40.80%	Easy
Base 7	45.50%	Easy
Remove 9	44.70%	Hard
ZigZag Conversion	26.40%	Medium
Diagonal Traverse	46.40%	Medium
Rotate Image	37.90%	Medium
Spiral Matrix	25.30%	Medium
Spiral Matrix II	38.80%	Medium
Set Matrix Zeroes	35.50%	Medium
Image Smoother	47.90%	Easy
Detect Capital	52.50%	Easy
Add Bold Tag in String	36.20%	Medium
Game of Life	36.50%	Medium
Pascal's Triangle	37.70%	Easy
Pascal's Triangle II	35.90%	Easy
Reshape the Matrix	67.90%	Easy
Perfect Rectangle	25.60%	Hard
Linked List Random Node	46.60%	Medium
Random Pick Index	41.80%	Medium
Shuffle an Array	45.90%	Medium
Maximum Distance in Arrays	31.00%	Easy
Maximum Product of Three Numbers	44.20%	Easy
Judge Route Circle	71.90%	Easy
Longest Continuous Increasing Subsequence	44.00%	Easy
Valid Parentheses	32.90%	Easy
Longest Valid Parentheses	23.00%	Hard
Basic Calculator	26.30%	Hard
Basic Calculator II	28.70%	Medium
Simplify Path	24.70%	Medium
Longest Absolute File Path	35.90%	Medium
Evaluate Reverse Polish Notation	26.60%	Medium
Tag Validator	17.90%	Hard
Mini Parser	30.00%	Medium
Ternary Expression Parser	50.30%	Medium
Decode String	40.90%	Medium

Next Greater Element I	57.60%	Easy
Next Greater Element II	47.20%	Medium
Next Greater Element III	27.20%	Medium
The Skyline Problem	26.60%	Hard
First Missing Positive	25.20%	Hard
Missing Number	44.00%	Easy
Majority Element	45.80%	Easy
Majority Element II	28.20%	Medium
Contains Duplicate	44.80%	Easy
Contains Duplicate II	32.00%	Easy
Contains Duplicate III	19.20%	Medium
Find All Duplicates in an Array	53.80%	Medium
Find All Numbers Disappeared in an Array	52.50%	Easy
Find the Difference	51.50%	Easy
Set Mismatch	40.60%	Easy
Merge Intervals	29.30%	Medium
Insert Interval	27.00%	Hard
Meeting Rooms	46.50%	Easy
Meeting Rooms II	38.70%	Medium
Find Right Interval	41.20%	Medium
Non-overlapping Intervals	40.40%	Medium
Teemo Attacking	52.00%	Medium
Missing Ranges	25.50%	Medium
Summary Ranges	28.90%	Medium
Data Stream as Disjoint Intervals	39.50%	Hard
Happy Number	40.00%	Easy
Add Digits	50.70%	Easy
Perfect Number	32.90%	Easy
Additive Number	27.30%	Medium
Count and Say	33.60%	Easy
Count Primes	26.40%	Easy
Strobogrammatic Number	39.40%	Easy
Strobogrammatic Number II	39.10%	Medium
Strobogrammatic Number III	31.20%	Hard
Ugly Number	38.80%	Easy
Ugly Number II	32.10%	Medium
Super Ugly Number	37.30%	Medium
Factorial Trailing Zeroes	35.50%	Easy
Minimum Factorization	27.70%	Medium
Largest Palindrome Product	17.90%	Easy
Count Numbers with Unique Digits	45.60%	Medium
Number of Digit One	27.90%	Hard
Rotate Function	31.50%	Medium
Smallest Good Base	31.40%	Hard
Minimum Moves to Equal Array Elements	46.80%	Easy
Minimum Moves to Equal Array Elements II	51.30%	Medium
Best Meeting Point	51.30%	Hard
Squirrel Simulation	46.70%	Medium
Next Permutation	28.50%	Medium
Lexicographical Numbers	40.50%	Medium
Permutation Sequence	27.80%	Medium
K-th Smallest in Lexicographical Order	23.30%	Hard

Split Concatenated Strings	28.50%	Medium
Nth Digit	30.10%	Easy
Largest Number	22.10%	Medium
Remove K Digits	26.10%	Medium
Third Maximum Number	27.50%	Easy
Kth Smallest Element in a Sorted Matrix	43.80%	Medium
Find K Pairs with Smallest Sums	30.40%	Medium
Kth Largest Element in an Array	38.40%	Medium
Top K Frequent Elements	47.30%	Medium
Smallest Range	43.50%	Hard
Split Array into Consecutive Subsequences	33.20%	Medium
IPO	34.70%	Hard
Minimum Time Difference	45.50%	Medium
Longest Consecutive Sequence	36.00%	Hard
Maximum Gap	29.10%	Hard
Island Perimeter	56.80%	Easy
Construct the Rectangle	49.10%	Easy
Rectangle Area	32.40%	Medium
Self Crossing	24.70%	Hard
Valid Square	36.10%	Medium
Convex Polygon	30.60%	Medium
Erect the Fence	0.226	Hard
Nim Game	55.10%	Easy
Bulb Switcher	42.20%	Medium
Bulb Switcher II	45.60%	Medium
Wiggle Subsequence	35.20%	Medium
Wiggle Sort	56.20%	Medium
Wiggle Sort II	25.50%	Medium
Increasing Triplet Subsequence	38.60%	Medium
132 Pattern	28.20%	Medium
Sort Transformed Array	43.60%	Medium
Find the Derangement of An Array	26.60%	Medium
Range Addition	54.60%	Medium
Range Addition II	44.40%	Easy
Solve the Equation	38.60%	Medium
Output Contest Matches	72.10%	Medium
Array Nesting	47.10%	Medium
Elimination Game	40.50%	Medium
Strong Password Checker	20.10%	Hard
UTF-8 Validation	34.80%	Medium
Repeated Substring Pattern	38.50%	Easy
Optimal Account Balancing	34.00%	Hard
Magical String	45.20%	Medium
Relative Ranks	47.40%	Easy
Longest Uncommon Subsequence I	51.30%	Easy
Longest Uncommon Subsequence II	28.70%	Medium
Lonely Pixel I	51.60%	Medium
Lonely Pixel II	40.00%	Medium
Split Array with Equal Sum	29.30%	Medium
Find the Celebrity	35.30%	Medium
License Key Formatting	41.20%	Medium

Fizz Buzz	58.80%	Easy
Longest Common Prefix	31.10%	Easy
Implement strStr()	27.60%	Easy
One Edit Distance	30.90%	Medium
Length of Last Word	31.50%	Easy
Number of Segments in a String	37.00%	Easy
Max Consecutive Ones	54.50%	Easy
Max Consecutive Ones II	44.40%	Medium
Read N Characters Given Read4	29.10%	Easy
Read N Characters Given Read4 II - Call multiple times	24.30%	Hard
Compare Version Numbers	19.70%	Medium
Hash Table		
Intersection of Two Arrays	46.60%	Easy
Intersection of Two Arrays II	44.30%	Easy
Minimum Index Sum of Two Lists	57.50%	Easy
Bulls and Cows	34.00%	Medium
Group Anagrams	33.30%	Medium
Valid Anagram	45.70%	Easy
Ransom Note	46.70%	Easy
Number of Boomerangs	44.10%	Easy
Line Reflection	30.20%	Medium
Fraction Addition and Subtraction	46.80%	Medium
Water and Jug Problem	26.70%	Medium
Max Points on a Line	15.40%	Hard
Group Shifted Strings	40.30%	Medium
Repeated DNA Sequences	30.60%	Medium
Shortest Word Distance	51.60%	Easy
Shortest Word Distance II	36.30%	Medium
Shortest Word Distance III	49.80%	Medium
First Unique Character in a String	46.40%	Easy
Longest Harmonious Subsequence	36.20%	Easy
Reconstruct Original Digits from English	43.20%	Medium
Sort Characters By Frequency	50.60%	Medium
Keyboard Row	60.30%	Easy
Distribute Candies	0.648	Easy
Brick Wall	41.60%	Medium
H-Index	32.70%	Medium
H-Index II	33.90%	Medium
Find Duplicate File in System	54.80%	Medium
Palindrome Pairs	25.60%	Hard

earch, or O(n) by fast and slow two pointers
by connenct AB and BA
nge every node

< 0, =m * len(list)

gent solution
next, node = node, dummy.next, node.next every time, from the old node, we remove head into the
, insert (m+1) to n nodes between pre and pre.next

pay attantion to the fast and slow start position
rse right part, insert right into left
e easier, while for iterative, we each time reverse two, and move two position

will not change the nums passed as parameter, but nums[:] = nums[::-1] will do

position, and set the result correspondingly

extension of 5, to find the longest, in fact we checked all the palindromic substrings
corner case, x< 0, x =0 and x is ***0
string.ascii_letters, digits, str.isalnum()

only s.startswith can pass, while s[:n-i] == r[i:] will TLE
insert or remove from set
same as 409, use a set to add odd char and remove it when even (meet again)
first determine whether can form as permuation, then if the odd exist, it must be in the middle The le
Two pointer to find the middle, and reverse the right part

bisect.bisect_left
bisect

When we find left == mid or right == mid, we have to move it towards mid until a different value four

Median of K sorted Arrays
binary search

sort... and two pointers

backtracking to test different length

dp
can early return instead of set a flag

hashtable

brute force?
not a bt

two pass, pay attention to the index in each side

range matters

There is a one pass dp using O(1) space
Need to think clearly how the dp works

sn = sn-1 + sn-2, s0 = 1, s1 = 0
a lot of corner cases

indeed can do a postorder with another stack

have a O(nlogn) solution

O(min(m, n)) space, best performance if we dp on the short direction

N * (2*3)

might have a quick solution
can do a standard dp of (M+1) * (N+1)
can do O(M) space

can do O(M) space
no need to do a standard bt, but just extend the result, similar to subset

list for random O(1), dict for insert and remove O(1)

double linked list for get max, min, most recent O(1), need to update both direction
double linked list with ordered dict

dict is faster than a list in OJ

iterative, Morris traversal
iterative
iterative, stack with status, inorder and preorder can be solved similarly

hashmap might make it faster

we need to set next, not left, right

recursive, iterative (dfs using stack and bfs using deque, appendleft and pop)

48.96 iterative
iterative using stack and extra status
a dfs method with 98.87

two different writing have different performance, saving almost half time
the diameter is not the number of nodes

serialize, recursive, iterative

preorder, recover with a stack. or BFS with a list

O(1) extra space solution

dijkstra.. Heap

60.79 using defaultdict
63.24 using defaultdict
86.24 using defaultdict
Have not idea how to solve it...

101 to 010 can be done by 101 ^ 111

Append node into the current list
stack, insert after dummy to form the correct order
consider each list separately as a list, in other words, solve the problem as 2
Can to similar as 445, while a briliant way is to determine whether the current digit is 9, if not, then v
Either reverse the list using a stack as 445, or can do a recursion so that for each next node, we ret
Similar to 2, do not forget to append the carry, and reverse the result
It is more convenient to have a direct index/pointer for the array, and reduce the index each time tha

log(N) binary search to find all possible divisor * factor that is less than dividend. Dividend / divisor =
Corner Cases: numerator is 0, denominator is 0, negative number and sign
binary search, corner case is mid*mid <= x, but (mid+1)**2 > x, then mid is the result
standard binary search. 1+3+5+(2n-1) = ...

Same as sqrt, but x**2 will overflow while x*x will not There are lots of different solutions. A quick s

Count 1 in binary format, or bit manipulation x & x-1 should be 0
log10(n)/log10(3) % 1 == 0
same as power of two, but need to remove those are power of 2 but not 4
need to pay attention to corner cases, like max/min int, the idx
corner cases, while idx < n and (a or b)
just consider each case...
same to 12
corner case: 0
idx is n-1 % 26

if num < 0:   num = num + 2**32

row+col=s, row in [0, m-1] thus col in [0 or s-m+1, s or n -1]

str.count

corner cases

mark negative to 0, and turn value in exisiting index to negative, finally the first positive position is n
1+..+n = (n+1)*n/2

set
hashtable to store the last idx

add n is a good idea

greedy

corner case, and how to reduce duplicate multiply

DP
<=0

min
median
median

corner cases
can also binary search

bucket LFU/LRU

can solve use hashmap too

1,2,3 win, 4 loss, 5,6,7 win, 8 loss

tricky..

binary

do int comparison instead of string

replace and upper is faster than do each position

Follow Up: if we can flip at most K zero, we store the previous at most K zero index in a queue, so t

set.intersection(t)
(countera & counterb).elements()

count sort may make the str sort quicker..
hashmap is faster than len_26 char

gcd, and re

just traverse the word list, update the distance, the initial set is important

only need to slightly modify 243

newhead, update both head and newhead

eft problem is the distint permutation of the even chars using backtracking

nd

we do not have carry and can return early
turn whether it has a carry

an calculate the index

= quotient and remainder

olution is recursion, pow(x, n) = pow(x*x, n/2) * (1 if n % 2 else x) nissing

:hat we can calculate the len when new value come in. Queue is better