

A BETTER VERSION --> https://neetcode.io/			
Category	Name	Link	Notes
https://youtu.be/KLlXCFG5nA	Two Sum	https://leetcode.com/problems/two-sum/	use hash map to instantly check for difference value, map will add index of last occurrence of a num, don't use same element twice;
https://youtu.be/1pk0XD63yU	Best Time to Buy and Sell Stock	https://leetcode.com/problems/best-time-to-buy-and-sell-stock/	find local min and search for local max, sliding window;
https://youtu.be/3OamN90kPa	Contains Duplicate	https://leetcode.com/problems/contains-duplicate/	hashset to get unique values in array, to check for duplicates easily
https://youtu.be/bNvYQI2wAlk	Product of Array Except Self	https://leetcode.com/problems/product-of-array-except-self/	make two passes, first in-order, second in-reverse, to compute products
https://youtu.be/5WZt3MMT06c	Maximum Subarray	https://leetcode.com/problems/maximum-subarray/	pattern: prev subarray cant be negative, dynamic programming; compute max sum for each prefix
https://youtu.be/DXV6YfWcRm	Maximum Product Subarray	https://leetcode.com/problems/maximum-product-subarray/	dp: compute max and max-abs-val for each prefix subarr;
https://youtu.be/nVnW4P8b1VA	Find Minimum in Rotated Sorted Array	https://leetcode.com/problems/find-minimum-in-rotated-sorted-array/	check if half of array is sorted in order to find pivot, arr is guaranteed to be in at most two sorted subarrays
https://youtu.be/UX8ENwh8oY8	Search in Rotated Sorted Array	https://leetcode.com/problems/search-in-rotated-sorted-array/	at most two sorted halves, mid will be apart of left sorted or right sorted, if target is in range of sorted portion then search it, otherwise search other half
https://youtu.be/jz2zG8nZ89A	3Sum	https://leetcode.com/problems/3sum/	sort input, for each first element, find next two where a + b+c, if a=prevA, skip a, if b=prevB skip b to elim duplicates; to find b,c use two pointers, left/right on remaining list;
https://youtu.be/UuItKWpPaAo	Container With Most Water	https://leetcode.com/problems/container-with-most-water/	shrinking window, left/right initially at endpoints, shift the pointer with min height;
https://youtu.be/gYUrDV42fY	Sum of Two Integers	https://leetcode.com/problems/sum-of-two-integers/	add bit by bit, be mindful of carry, after adding, if carry is still 1, then add it as well;
https://youtu.be/5Km3utkw7e	Number of 1 Bits	https://leetcode.com/problems/number-of-1-bits/	modulo, and dividing n; mod and div are expensive, to divide use bit shift, instead of mod to get 1's place use bitwise & 1;
https://youtu.be/8y8M56SRlWw	Counting Bits	https://leetcode.com/problems/counting-bits/	write out result for num=16 to figure out pattern; res[i] = res[i - offset], where offset is the biggest power of 2 <= i;
https://youtu.be/WnPLSRLSANE	Missing Number	https://leetcode.com/problems/missing-number/	compute expected sum - real sum; xor n with each index and value;
https://youtu.be/UcoN6UAI6A	Reverse Bits	https://leetcode.com/problems/reverse-bits/	reverse each of 32 bits;
https://youtu.be/Y0IT9f3c7aI	Climbing Stairs	https://leetcode.com/problems/climbing-stairs/	subproblem find (n-1) and (n-2), sum = n;
https://youtu.be/H9bfaeqjoqs	Coin Change	https://leetcode.com/problems/coin-change/	top-down: recursive dfs, for amount, branch for each coin, cache to store prev coin_count for each amount; bottom-up: compute coins for amount = 1, up until n, using for each coin (amount - coin), cache prev values
https://youtu.be/cWnW0hdFY	Longest Increasing Subsequence	https://leetcode.com/problems/longest-increasing-subsequence/	recursive: foreach num, get subseq with num and without num, only include num if prev was less, cache solution of each; dp=subseq length which must end with each num, curr num must be after a prev dp or by itself;
https://youtu.be/Ua0Gh3iSWM	Longest Common Subsequence	https://leetcode.com/problems/longest-common-subsequence/	recursive: if first chars are equal find lcs of remaining of each, else max of: lcs of first and remain of 2nd and lcs of 2nd remain of first, cache result; nested forloop to compute the cache without recursion;
https://youtu.be/5x9N9nlnc3A	Word Break Problem	https://leetcode.com/problems/word-break/	for each prefix, if prefix is in dict and wordbreak(remaining str)=True, then return True, cache result of wordbreak;
https://youtu.be/G8BK7V5k6dG	Combination Sum	https://leetcode.com/problems/combination-sum/	visualize the decision tree, base case is curSum > > target, each candidate can have children of itself or elements to right of it in order to elim duplicate solutions;
https://youtu.be/773zK8wEkyg	House Robber	https://leetcode.com/problems/house-robber/	for each num, get max of prev subarr, or num + prev subarr not including last element, store results of prev, and prev not including last element
https://youtu.be/rWJ4CYYOxM	House Robber II	https://leetcode.com/problems/house-robber-ii/	subarr = arr without first & last, get max of subarr, then pick which of first should be added to it
https://youtu.be/6aEYtOWuJ	Decode Ways	https://leetcode.com/problems/decode-ways/	can cur char be decoded in one or two ways? Recursion -> cache -> iterative dp solution, a lot of edge cases to determine, 52, 31, 29, 10, 20 only decoded one way, 11, 26 decoded two ways
https://youtu.be/IIEdxauD4ly	Unique Paths	https://leetcode.com/problems/unique-paths/	work backwards from solution, store paths for each position in grid, to further optimize, we don't store whole grid, only need to store prev row;
https://youtu.be/yAn0Czclv8	Jump Game	https://leetcode.com/problems/jump-game/	visualize the recursive tree, cache solution for O(n) time/mem complexity, iterative is O(1) mem, just iterate backwards to see if element can reach goal node, if yes, then set it equal to goal node, continue;
https://youtu.be/mQeF6bN8hmK	Clone Graph	https://leetcode.com/problems/clone-graph/	recursive dfs, hashmap for visited nodes
https://youtu.be/EqI5nU9etnU	Course Schedule	https://leetcode.com/problems/course-schedule/	build adjacency_list with edges, run dfs on each V, if while dfs on V we see V again, then loop exists, otherwise V visit in a loop, 3 states= not visited, visited, still visiting
https://youtu.be/s-YkqHqGI	Pacific Atlantic Water Flow	https://leetcode.com/problems/pacific-atlantic-water-flow/	dfs each cell, keep track of visited, and track which reach pac, atl; dfs on cells adjacent to pac, atl, find overlap of cells that are visited by both pac and atl cells;
https://youtu.be/pV2kpPD66eI	Number of Islands	https://leetcode.com/problems/number-of-islands/	foreach cell, if cell is 1 and unvisited run dfs, increment count and marking each contiguous 1 as visited
https://youtu.be/68KZ7Mu_muU	Longest Consecutive Sequence	https://leetcode.com/problems/longest-consecutive-sequence/	use bruteforce and try to optimize, consider the max subseq containing each num; add each num to hashset, for each num if num-1 doesn't exist, count the consecutive nums after num, ie num+1; there is also a union-find solution;
https://youtu.be/6kTZnNnPy5	Alien Dictionary (Leetcode Premium)	https://leetcode.com/problems/alien-dictionary/	chars of a word not in order, the words are in order, find adjacency list of each unique char by iterating through adjacent words and finding first chars that are different, run topSort on graph and do topSort detection;
https://youtu.be/7bXsUuwmnQo	Graph Valid Tree (Leetcode Premium)	https://leetcode.com/problems/graph-valid-tree/	union find, if union return false, loop exists, at end size must equal n, or it's not connected; dfs to get size and check for loop, since each edge is double, before dfs on neighbor of N, remove N from neighbor list of neighbor;
https://youtu.be/8F1XPM4WOUc	Number of Connected Components	https://leetcode.com/problems/number-of-connected-components/	dfs on each node that hasn't been visited, increment component count, adjacency list; bfs and union find are possible;