Problem Solving Workshop #32 Tech Interviews and Competitive Programming Meetup
February 11, 2018 https://www.meetup.com/tech-interviews-and-competitive-programming/
Instructor: Eugene Yarovoi (can be contacted through the group Meetup page above under Organizers)

More practice questions: leetcode.com, glassdoor.com, geeksforgeeks.org
Books: Elements of Programming Interviews, Cracking the Coding Interview
Have questions you want answered? Contact the instructor, or ask on Quora. You can post questions and follow the instructor and other people who write about algorithms.

Try to find optimized solutions, and provide a time and space complexity analysis with every solution.

Problem #1, "Shortest Subarray With Sum"

You're given an array of integers. Find the position of the shortest contiguous subarray such that its sum is >= K, for a given integer K > 0.

Example Input: array = [4, 2, 3, 6, 1, 2], K = 11
Output: starting index = 1, ending index = 3
Explanation: the contiguous subarray [2, 3, 6], which has starting index 1 and ending index 3, has a sum of 11. There is no shorter (length < 3) subarray that has a sum of at least 11.

(a) **[Easy-Medium]** Solve the case where all integers in the array are positive. An optimized solution will have O(n) time complexity.
(b) **[Medium-Hard]** Now solve when the integers in the array can be negative. O(n log n) is a good time complexity, but O(n) is possible.

Problem #2, "Planetary Network" (Medium)

You have a bunch of research centers on a newly colonized planet. You want to set up Internet connectivity between all of the research centers. We say that two research centers A and B are connected if either there is the ability to transmit directly between A and B, or if there is an indirect way to get a message from A to B, for example, it could be that A can transmit to C, which can transmit to D, which can transmit to B (a path like A -> C -> D -> B). All connections are bidirectional.

There's two ways a direct line of transmission can be established between two research centers. One way, if the distance between the two is lower than a certain threshold T and there are no obstacles in the way blocking it, is short-range radio transmission. This method is cheap to set up and the colonists can make as many short-range transmitters as they need. The second method can connect **any** two centers anywhere, but uses a special technology that the space colonists can't build, and they only have a limited number S of such super-transmitters with them.

Your goal is to determine the necessary threshold T (how strong the short-range trasnmitters must be) in order to be able to connect all research centers with only S super-transmitters available.

For each research center, you will be given a list of other research centers that, given sufficiently strong short-range transmission (large enough T) can be connected directly via short-range transmission (though the super-transmitters can always connect any two centers anywhere). Not all centers are reachable from a given center for purposes of short-range transmission (maybe there's a mountain in the way, or the planet's curvature, or they're simply too far for all intents and purposes). For each such center you will aso receive the distance to it. Finally, you will receive the parameter S, how many super-transmitters are available. Return the required T in order to be able to connect all centers.

Example Input:
3 centers
Center 1: [Center 2, distance 4], [Center 3, distance 10]
Center2: [Center 1, distance 4]
Center 3: [Center 1, distance 10]
K= 1
Output: 4
Explanation: We can connect centers 1 and 3 via the 1 super-link, so we need T=4 to connect centers 1 and 2. If K=0 instead, the output would be 10, as we would need T=10 to connect everything without any super-links.

Problem #3, "Longest Subsequence From Dictionary"

You're given a string S and a dictionary D of words. Find the longest word in D that is a subsequence of S.

The definition of "W is a subsequence of S" is that if we delete zero or more characters from S, we can get W with no further rearrangement of the characters. For example, "aba" is a subsequence of "adba" (the "d" can be deleted"), but not of "aabcd".

Example Input:
S = "bappbable"
D = ["able", "apple", "kangaroo"]

Output: "apple"
Explanation: "able" and "apple" are both subsequences of S. Of these two, "apple" is the longer word. "kangaroo" would have been even longer, but is not a subsequence of S.

(a) **[Easy]** Any solution to the problem.

(b) **[Easy-Medium]** Suppose there are a total of L characters in the dictionary (sum over all words), W words in the dictionary, and S is of length N. Get O(NW) complexity.

(c) **[Medium-Hard]** Improve your complexity further in the case where N >> average word size. In this case, O(NW) is not very good. Can you get a solution that is closer to the ideal O(L + N)?