UO Problem Solving Workshop
May 20, 2017

Tech Interviews and Competitive Programming Meetup
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 at eugene.yarovoi AT gmail.com), or ask on Quora.

Guidance: Start by finding any solution you can for a problem, and then try to find an optimized solution.. Use time complexity analysis to help yourself compare different solutions. You can solve the problems in any order (though, roughly, they get a little harder as they go), but attempt the subsections of a single problem in order.

Problem #1, "Flipping Bits"

Given an array of 0s and 1s, and an integer k, find the longest contiguous streak of 1s that you can get by changing any k 0s to 1s.

Example Input: array = [1,1,0,0,1,1,1,0,1,1], k = 1
Input Explanation: We can change one (k=1) 0 to a 1 in the array
Output: 6
Output Explanation: if we change the 0 to a 1 at index 7 (0-indexed), we get a contiguous streak of 1s having length 6. This is the largest streak we can get. No other change gets us a bigger streak. In other words, we changed the bolded value:

[1,1,0,0,1,1,1,**1**,1,1]
| |
-- streak --

Problem #2, "Fuzzy Duplicates"

You're given an array of N integers. In each of the below, you should give a solution that takes in the array and produces a boolean indicating whether the condition was met.

(a) Do there exist two values in the array that have the same value? Good time complexity: O(N log N).
Examples: [3, 4, 5] -> false
[5, 4, 5] -> true

(b) Do there exist two values in the array that have a difference of at most K (K is given as an additional input)? Good time complexity: O(N log N).

Examples: [4, 11, 7], K = 3 -> true (because 7-4 = 3)
[3, 7, 11] -> false

(c) Determine whether there exist two values in the array that are identical, and their index differs by at most D (they are distance D apart or closer in the array). D is an additional integer input. Good time complexity: O(N) or O(N log N), depending on your approach.
Examples: [5, 4, 6, 5], D = 2 -> false (because only the 5s are duplicates, and they're 3 indices apart, > D)
[5, 4, 6, 5], D = 3 -> true (with D = 3, the 5s are close enough)

(d) **[This may be tricky, consider coming back later if you get stuck]** Determine whether there exist two values in the array that satisfy both conditions at once: they have a difference of at most K **and** are at most D distance apart (given both K and D). Target time complexity: O(N log N) or O(N log D). (It's also possible to solve in O(N), independent of K and D, but this is hard, so try it once you've completed everything else.)
Examples: [5, 10, 8, 6], D = 2, K = 1 -> false (only 5 and 6 are close enough by value, and they are too far apart by index)
[5, 7, 8, 6], D = 2, K = 1 -> true (6 and 7 are close enough by index and value).

Problem #3, "Road Repairs"

A road of length N has good parts, and bad parts that need to be repaired. The road is discretized into chunks of size 1. We receive as input a boolean array A of size N, where A[i] is true if the road is good at coordinate i, and A[i] is false if the road is bad there. For example, [true, true, true, false, true, true, false] would be a road of size 7 where the bad parts occur at positions 3 and 6, and the rest of the road is good.

We want to repair all bad parts of the road. A bad chunk gets repaired if at least one repair truck visits it. There are T repair trucks and each one starts with F units of fuel. Each truck starts on some chunk of your choosing, and can move to *adjacent* chunks of road (in the direction of your choice) from there until it runs out of fuel, spending 1 unit of fuel per chunk it visits (including the starting chunk).

(a) In addition to the road array, you're given F, the amount of fuel per truck, as input. Give an efficient algorithm to compute T, the minimum number of repair trucks you need in order to fix all the bad chunks.

Example Input: A = [F, T, T, F, F, F], F = 2
Output: 3
Explanation: You need one truck just to get the bad chunk at the start of the road, since the truck that fixes it can't get to any other bad chunk before running out of fuel. The 3 bad chunks at the end require 2 more trucks to fix, since each truck can visit at most 2 chunks. So, 3 trucks total.

(b) **[This part may be challenging]** Now, in contrast to the previous problem, you're given the road array and T, the number of repair trucks you have, and you're asked to find the minimum F you need in order to be able to fix all the bad chunks.

Guidance: first find the $O(N^2)$ solution, then try to improve to O(N log N), the target time complexity. It's also possible to solve this problem in O(N), but this is hard, so try it once you've completed everything else.

Example Input: A = [F, T, T, T, T, T, T, F, T, F, F, F], T = 2
Output: 5
Explanation: We need at least 5 units of fuel for each truck. If we have 2 trucks with 5 units, we can start one truck at position 0, and the other at position 7 (second F in the array), and drive both forward as far as they go. There's no way to cover all Fs with less than 5 units of fuel per truck. The first truck needs only one unit, but all trucks start with the same amount of fuel, and the second truck needs all 5 units.