Software engineer assessment instructions

Thank you for considering HealthVerity. We value your interest in our company and we understand that your time outside of work is limited. You only need to complete Question 1 and either Question 2 or Question 3. As a general guideline, this should take around 3 hours, but the amount of time you spend on this is up to you. **Please note in your response how much time you spent completing this assessment.**

For this assessment, you'll be evaluated on four key areas: **accuracy, efficiency, clarity, and communication.**

We are most interested in hearing your thoughts and ideas, more so than a perfect response of code. Please show us your own implementation rather than something already existing and please avoid built-in evaluation expressions. Your code should compile and run. **Please do not put your solutions on publicly accessible websites (e.g. GitHub, BitBucket, etc).** Choose the language you feel will best demonstrates your abilities and one which will also run on a Linux box. **Feel free to add comments about what additionally you would do to solve these problems if you had more time.**

Additionally, please also send us, if possible, what you consider to be the best publicly available code you have written (under 200 lines).

If you have any questions at all, please reach out. **Once you complete these questions, please email your submission in one attachment.** Our engineering team will read over your responses and we will do our best to get back to you within 48 hours.

Question 1

You stumble across a random number generator on GitHub. The author believes that their generator will create an infinite sequence of integers that won't repeat. Their random number generator is seeded with a list of size N containing integers between 0 and N-1 (inclusive). It then returns "random" numbers by iterating through the list. The generator starts by returning the value at index 0. **It then uses that value as the index for the next value to return, and so on**. If the generator was seeded with the list [1, 2, 0], the first number it would return would be 1, then 2, then 0, and then it would repeat the sequence. Thus, the number of distinct values would be 3.

Part a:

Write a function that takes as input the seed list of the random number generator of up to 1 million integers and returns the count of distinct integers the random number generator would return. Your function may not modify the seed list. Hint: len(set(input_list)) does not produce the right answer

Sample input: Sample output:
[1, 2, 0] 3
[4, 1, 3, 4, 2] 3

Part b:

Can part a be done with O(1) auxiliary space (i.e. using only a constant amount of additional memory)? If so, write a function that does it. If not, why not? Your function may not modify the seed list.

Question 2

Part a:

Write a program that reads in the names of two files from STDIN and outputs the number of strings they have in common to STDOUT. Each file contains 1 million strings of 32 characters, 1 per line.

Sample input: Sample output:

strings_1.txt 418239

Strings_2.txt

Part b:

If the files contained 1 billion strings instead, would you change your approach? Why?

healthverity

Question 3

Write a program that reads in a CSV file with two columns, “A” and “B”. Column “A” contains 999,998 strings representing unique integers from 1 to 1,000,000 (2 integers are missing). Column “B” contains 999,999 strings representing unique integers from 1 to 1,000,000 (1 integer is missing), and the missing integer is the same as one of the missing integers in Column “A”.

Part a:

Your program should print the integer that exists in column “B”, but not in column “A”.

Part b:

Your program should also print out the integer that’s missing from both columns.

Part c:

Can part b be done with less than O(n) auxiliary space (i.e. if the length of the CSV doubled, your memory usage would not double) and O(n) time complexity? If so, write a program that does it. If not, why not?