SYNVERT DATA INSIGHTS - TECHNICAL TEST (CODE SECTION)

Hey, thanks for applying to our company :) Here are five code questions, ordered from easy to hard, to give us an idea of your ability. Don't worry if you can't finish them all, there are meant to be more than anyone can reasonably finish in an hour. We ask that you include the code for all questions in a single file (Python, Java, C/C++, MatLab), and send it to us within 1 hour. Best of luck!

Note: we will cross-check your solutions against ones found on Google/GitHub searches, so no copying and variable-renaming, please! Also, **please do not use ChatGPT, Copilot or anything similar** since we want to see your logic and programming style.

QUESTION 1

Given 2 strings, find the starting position of string 1 inside string 2. Please implement the logic by yourself without using built-in functions or methods.

QUESTION 2

Given 2 lists, create a new list by taking alternate items from the lists.

Example Input:

A = [1,2,3,4,5]

B = [6,7,8,9,10,11,12]

Output:

[1,6,2,7,3,8,4,9,5,10,11,12]

QUESTION 3

Given a string A and a list of shorter strings B, and the longest string in B that can be created by removing characters from string A (without rearranging characters).

Example Input:

A = "peanut butter jelly is the best!"

B = ["bus","pets","bully","jester"]

Output:

"bully"

QUESTION 4

For n cities, create a randomized $n \times n$ matrix to represent the cost of traveling between the cities. Then find the cheapest route that starts from the first city, visits all other cities, and returns to the first city (a classic traveling salesman problem). Brute force is fine.

Example Input:

n = 4

This will create a 4×4 matrix with random values, e.g.:

$$\begin{pmatrix} 0 & 0.914 & 0.136 & 0.507 \\ 0.118 & 0 & 0.259 & 0.389 \\ 0.876 & 0.664 & 0 & 0.734 \\ 0.097 & 0.475 & 0.310 & 0 \end{pmatrix}$$

in which entry (i, j) is the cost of going from city i to city j (indices in our matrix start from the top-left).

Output:

The cheapest route is (1,3,2,4,1) which costs 1.286.

QUESTION 5

If you get this far, good job, you've already done very well. This one is pretty difficult. Create a program that verifies if a number is a Carmichael Number, that is, a number n that is not prime, but satisfies the relationship:

$$b^n \equiv b \pmod{n}$$

For all integers b which are coprime[1] with n.

Hint: The first Carmichael Number is 561.

Hint 2: $b^n \bmod n$ will cause errors for most n, as b^n gets huge fast. Instead perform the operation by repeating $b \times b \bmod n$ a total of n times.

[1] b and n are coprime if the only factor they share is 1. For example, 14 and 15 are coprime (the only number that divides both of them is 1), while 14 and 21 are not coprime
(they can both be divided by 7).