Given a sequence $X = [x_1, x_2, ..., x_m]$, another sequence $Z = [z_1, z, ..., z_k]$ is a **subsequence** of X, if there exists a strictly increasing sequence $[i, i_2, ..., i_k]$ of indices of X such that for all $j = 1, 2, ..., k$, we have $x_{i_j} = z_j$.

For example, $Z = [B, C, D, B]$ is a subsequence of $X = [A, B, C, B, D, A, B]$, with the corresponding index sequence $[2, 3, 5, 7]$.

Given two sequences X and Y, we say that a sequence Z is a **common subsequence** of X and Y, if Z is a subsequence of both X and Y.

For example, if $X = [A, B, C, B, D, A, B]$ and $Y = [B, D, C, A, B, A]$, the sequence $[B, C, A]$ is a common subsequence of both X and Y. The sequence $[B, C, A]$ is not a *longest* common subsequence (LCS) of X and Y, however, since it has length 3 and the sequence $[B, C, B, A]$, which is also common to both X and Y, has length 4.

The sequence $[B, C, B, A]$ is an LCS of X and Y, as is the sequence $[B, D, A, B]$, since X and Y have no common subsequence of length 5 or greater.

Write an initial arbitrarily naïve version of an algorithm, that, given two sequences X and Y computes an LCS of X and Y. Estimate the asymptotic worst case runtime in Big-O notation in terms of m and n, where m is the length of sequence X and n is the length of sequence Y.