TP n°6 - Plus longue sous-séquence commune

Soit $l = [l_0, \dots, l_{n-1}]$ une liste. On appelle sous-séquence de la liste l toute liste obtenue à partir de l en supprimant éventuellement certains éléments et en laissant les autres dans l'ordre. Autrement dit, une sous-séquence de l est une liste de la forme $[l_{\varphi(0)}; \dots; l_{\varphi(k-1)}]$ avec $0 \leqslant \varphi(0) < \dots < \varphi(k-1) \leqslant n-1$.

Q1 Écrire une fonction `est_sous_sequence` qui prend en argument deux listes l_1 et l_2 et qui teste si l_1 est une sous-séquence de l_2.

Q2 Soit l une liste formée de n éléments distincts. Combien existe-il de sous-séquences de l ?

Q3 Écrire une fonction récursive `sous_sequences` prenant en argument une liste et renvoyant la liste de ses sous-séquences.

Soient $x = [x_0, \dots, x_{n-1}]$ et $y = [y_0, \dots, y_{m-1}]$ deux listes formées d'éléments du même type. On cherche à déterminer la longueur maximale d'une sous-séquence commune à x et y et à trouver une sous-séquence commune de cette longueur. On parle alors de plus longue sous-séquence commune (attention, il n'y a pas forcément unicité : ne pas parler de *la* plus longue sous-séquence commune).

Une première façon de résoudre le problème est de chercher toutes les sous-séquences communes afin de trouver l'une des plus longues.

Q4 En utilisant les questions précédentes, écrire une fonction `plssc1` qui renvoie une plus longue sous-séquence commune de deux listes.

Cet algorithme est en temps exponentiel donc plutôt mauvais. On va essayer de concevoir un algorithme plus rapide.

On note $L(i, j)$ la longueur d'une plus longue sous-séquence commune de $[x_0, \dots, x_{i-1}]$ et $[y_0, \dots, y_{j-1}]$.

Q5 Montrer les relations suivantes :

1. $\forall i \in [\![0, n]\!], L(i, 0) = 0$ et $\forall j \in [\![0, m]\!], L(0, j) = 0$
2. Si $i > 0$ et $j > 0$, alors $L(i, j) = \begin{cases} 1 + L(i-1, j-1) & \text{si } x_{i-1} = y_{j-1} \\ \max\left(L(i-1, j), L(i, j-1)\right) & \text{sinon} \end{cases}$

Q6 En déduire une fonction récursive `lplssc1` qui calcule la longueur d'une plus longue sous-séquence commune de deux listes.
On pourra remarquer que la longueur d'une plus longue sous-séquence commune de deux listes est égale à la longueur d'une plus longue sous-séquence commune des miroirs des deux listes.

On pourra montrer que le nombre d'appels récursifs dans le pire des cas effectués par cette fonction peut être minoré par une exponentielle en $\min(n, m)$.

Q7 Lorsque $n = m = 3$, combien de fois calcule-t-on $L(1, 0)$ dans le pire des cas ?

De nombreuses valeurs sont calculées plusieurs fois. Pour éviter cela, on conçoit un algorithme de programmation dynamique pour le calcul des $L(i, j)$, en stockant leurs valeurs dans une matrice.

Dans cette partie, on pourra convertir les listes en tableaux à l'aide de la fonction `Array.of_list`

Q8 Indiquer dans quel ordre remplir la matrice. Une fois cette matrice calculée, comment déterminer la longueur d'une plus longue sous-séquence commune ?

Q9 Écrire une fonction `lplssc2` qui calcule la longueur d'une plus longue sous-séquence commune de deux listes en implémentant cette méthode. Quelle est sa complexité ?
Pour créer la matrice, on utilisera la fonction `Array.make_matrix`

Q10 Comment retrouver une plus longue sous-séquence commune à partir de la matrice des $L(i, j)$?

Q11 Écrire une fonction `plssc2` qui calcule une plus longue sous-séquence commune.

Q12 L'adapter pour qu'elle donne toutes les plus longue sous-séquence commune.