Exercice : Distance d'édition

Les séquences de caractères peuvent encoder de nombreuses informations de nature différente, par exemple du texte, de la voix ou des séquences ADN. L'alignement de deux chaînes des caractères consiste à comparer deux séquences de caractères afin d'évaluer la similarité entre les deux.

La distance d'édition (ou de Levenshtein, ou d'Ulam) est une mesure de la similarité entre deux chaînes de caractères. Cette distance est le nombre minimal de caractères qu'il faut supprimer, insérer ou substituer pour passer d'une chaîne à l'autre.

La distance d'édition entre 2 mots a et b correspond à **la longueur de la plus courte suite de transformations pour passer de a à b**, avec les transformations suivantes :

- insertion d'une nouvelle lettre
- suppression d'une lettre
- remplacement d'une lettre par une autre

Notation : | a | désigne le cardinal de a, c'est-à-dire le nombre de caractères de la chaîne.
a[0] désigne le premier caractère de la chaîne a.

On suppose que :
• supprimer un caractère,
• insérer un caractère,
• substituer un caractère,
sont des opérations qui ont toute un coût unitaire (1).

Si le caractère est identique, la substitution ne coûte rien (0).

La distance d'édition est définie par induction de la manière suivante :

Entrées : deux chaines a et b
Sortie : la distance d'édition entre a et b

$$d(a,b) = \begin{cases} \max(|a|,|b|) & \text{si } \min(|a|,|b|) = 0 \\ d(a[1:],b[1:]) & \text{si } a[0] = b[0] \\ 1 + \min \begin{cases} d(a[1:],b) \\ d(a,b[1:]) \\ d(a[1:],b[1:]) \end{cases} & \text{sinon} \end{cases}$$

1. La distance d'édition de "chien" à "niche" vaut 4. Expliquer pourquoi

2. La distance d'édition représente-t-elle un problème à sous-structure optimale ? Justifier.

3. Ecrire une **fonction distanceEdition_rec(a,b)** qui calcule la distance d'édition de deux chaînes de manière naïve.

4. Écrire une **fonction distanceEdition_PD_rec(a, b, mem)** en programmation dynamique avec memoïsation à l'aide d'un dictionnaire dont la clé est un tuple d'indices. On pourra tester sur "AGTTC" et "AGCTC", sur "chien" et "niche" ou "sunday" et "saturday".

Si on souhaite programmer dynamiquement par le bas en utilisant un tableau S contenant les distances, il est utile d'exprimer le résultat d'une case en fonction de celles dont elle dépend dans le schéma dynamique :

$$S[i,j] = \begin{cases} \max(i,j) & \text{si } \min(i,j) = 0 \\ S[i-1,j-1] & \text{si } a[i] = b[j] \\ 1 + \min(S[i-1,j], S[i,j-1], S[i-1,j-1]) & \text{sinon} \end{cases}$$

5. On souhaite utiliser la programmation dynamique. Compléter à la main et de bas en haut le tableau suivant, associé à la distance d'édition de "chien" à "niche".

i \ j	0	1,n	2,i	3,c	4,h	5,e
0						
1,c						
2,h						
3,i						
4,e						
5,n						

6. Écrire une **fonction distanceEdition_PD_ite(a,b)** qui calcule la distance d'édition de deux chaînes de caractères par programmation dynamique de bas en haut et comparer les résultats entre les 2 approches.