

Exercice : Détermination de la plus longue sous-chaine commune à deux chaînes de caractères

En génétique par exemple, il peut s'avérer très important de savoir quels sont les points communs de deux génomes. Le problème de la plus longue sous-chaine permet d'apporter une réponse à cette question.

On appelle **sous-chaine** d'une chaîne de caractères $a = a_1 \dots a_n$ toute chaîne de caractères s extraite de a telle que $s = a_{i_1} \dots a_{i_k}$ où $1 \leq i_1 < \dots < i_k \leq n$ et $\forall p \in \{i_1, \dots, i_k\}, a_p \in a$. Les caractères de s n'apparaissent pas nécessairement de manière consécutive dans la chaîne a .

Soient $a = a_1 \dots a_n$ et $B = b_1 \dots b_m$ 2 chaînes de caractères non vides.

On appelle **plus longue sous-chaine commune** à a et b toute sous-chaine commune à a et b de longueur maximale.

Si l'une des chaînes a ou b est vide ou si a et b n'ont aucune sous-chaine commune, la chaîne vide est alors l'unique plus longue sous-chaine commune à a et b .

Exemple : les chaînes de caractères "AAA" et "TAA" sont les plus longues sous-chaînes communes aux chaînes de caractères "ATAGA" et "TAACA".

1. Pour les chaînes $a = \text{"AATGCG"}$ et $b = \text{"TATTAGC"}$, donner les solutions des plus longues suites communes.
2. Écrire une **fonction de signature** `est_sousChaine(ch : str, sch : str) -> bool` dont les ch et sch paramètres sont deux chaînes de caractères. Cette fonction renvoie True si sch est une sous-chaine de ch , False sinon.
3. Écrire une **fonction de signature** `est_commune(ch1 : str, ch2 : str, sch : str) -> bool` où les paramètres sont des chaînes de caractères. Cette fonction renvoie True si sch est une sous-chaine commune à a et b .
4. Le problème de la plus longue sous-chaine commune entre $ch1$ et $ch2$ est noté $L(ch1, ch2)$, son résultat est la longueur maximale d'une sous-chaine commune à $ch1$ et $ch2$.

Formuler le problème $L(ch1, ch2)$ récursivement afin de pouvoir justifier de sa sous-structure optimale.

5. Écrire la **fonction** `Ig_PlusLgSuiteCommune(ch1 : str, ch2 : str) -> int` qui renvoie la longueur de la plus longue suite commune avec la relation de récurrence du haut vers le bas.
6. Écrire la **fonction** `PlusLgSuiteCommune(ch1 : str, ch2 : str) -> str` renvoie la valeur de la plus longue suite commune, avec la relation de récurrence.
7. Écrire la **fonction** `PlusLgSuiteCommune_mem(ch1, ch2, mem) -> str` qui renvoie la sous-chaine la plus longue, récursivement avec mémoïsation à l'aide d'un dictionnaire.
8. Écrire une version itérative `IgPlusLgSuiteCommune_iter(ch1 : str, ch2 : str) -> int` qui remplit un tableau de taille $n \times m$, avec les longueurs des plus longues sous-séquences communes, en commençant par les cases simples, c'est-à-dire lorsque l'un des chaînes est de longueur nulle.

Exercice : Détermination de la plus longue sous-chaine commune à deux chaînes de caractères

En génétique par exemple, il peut s'avérer très important de savoir quels sont les points communs de deux génomes. Le problème de la plus longue sous-chaine permet d'apporter une réponse à cette question.

On appelle **sous-chaine** d'une chaîne de caractères $a = a_1 \dots a_n$ toute chaîne de caractères s extraite de a telle que $s = a_{i_1} \dots a_{i_k}$ où $1 \leq i_1 < \dots < i_k \leq n$ et $\forall p \in \{i_1, \dots, i_k\}, a_p \in a$. Les caractères de s n'apparaissent pas nécessairement de manière consécutive dans la chaîne a .

Soient $a = a_1 \dots a_n$ et $B = b_1 \dots b_m$ 2 chaînes de caractères non vides.

On appelle **plus longue sous-chaine commune** à a et b toute sous-chaine commune à a et b de longueur maximale.

Si l'une des chaînes a ou b est vide ou si a et b n'ont aucune sous-chaine commune, la chaîne vide est alors l'unique plus longue sous-chaine commune à a et b .

Exemple : les chaînes de caractères "AAA" et "TAA" sont les plus longues sous-chaînes communes aux chaînes de caractères "ATAGA" et "TAACA".

1. Pour les chaînes $a = \text{"AATGCG"}$ et $b = \text{"TATTAGC"}$, donner les solutions des plus longues suites communes.
2. Écrire une **fonction de signature** `est_sousChaine(ch : str, sch : str) -> bool` dont les ch et sch paramètres sont deux chaînes de caractères. Cette fonction renvoie True si sch est une sous-chaine de ch , False sinon.
3. Écrire une **fonction de signature** `est_commune(ch1 : str, ch2 : str, sch : str) -> bool` où les paramètres sont des chaînes de caractères. Cette fonction renvoie True si sch est une sous-chaine commune à a et b .
4. Le problème de la plus longue sous-chaine commune entre $ch1$ et $ch2$ est noté $L(ch1, ch2)$, son résultat est la longueur maximale d'une sous-chaine commune à $ch1$ et $ch2$.

Formuler le problème $L(ch1, ch2)$ récursivement afin de pouvoir justifier de sa sous-structure optimale.

5. Écrire la **fonction** `Ig_PlusLgSuiteCommune(ch1 : str, ch2 : str) -> int` qui renvoie la longueur de la plus longue suite commune avec la relation de récurrence du haut vers le bas.
6. Écrire la **fonction** `PlusLgSuiteCommune(ch1 : str, ch2 : str) -> str` renvoie la valeur de la plus longue suite commune, avec la relation de récurrence.
7. Écrire la **fonction** `PlusLgSuiteCommune_mem(ch1, ch2, mem) -> str` qui renvoie la sous-chaine la plus longue, récursivement avec mémoïsation à l'aide d'un dictionnaire.
8. Écrire une version itérative `IgPlusLgSuiteCommune_iter(ch1 : str, ch2 : str) -> int` qui remplit un tableau de taille $n \times m$, avec les longueurs des plus longues sous-séquences communes, en commençant par les cases simples, c'est-à-dire lorsque l'un des chaînes est de longueur nulle.