Travail TP 01 : Problème de la Plus Longue Sous-Séquence Commune (LCS)

1. Développement d’Algorithmes
2. Analyser la complexité en temps et en espace de premier algorithme

Complexité en temps :

Meilleur cas : Lorsque toutes les lettres de X et Y correspondent, chaque appel récursif réduit la taille de l'entrée de 1. Le nombre d'appels récursifs est n (où n est la longueur de la plus petite chaîne).  
Pire cas : Lorsque aucune lettre de X et Y ne correspond, chaque appel récursif génère deux sous-problèmes (un pour n-1, m et un pour n, m-1). Cela crée un arbre de récursion binaire. Le nombre total de sous-problèmes est donc **2^(min(n, m))**, où n et m sont les longueurs des deux chaînes.  
  
Donc la complexité en temps de cet algorithme est de **O(2^(min(n, m)))**

Complexité en espace :

À chaque appel récursif, l'algorithme utilise de l'espace pour les paramètres et pour chaque appel dans la pile. La profondeur maximale de la pile d'appels est égale à la longueur de la plus grande chaîne (n ou m).Le tableau LCSResult pour stocker la séquence de caractères de la LCS. Sa taille est de **O(n + m)**, où n et m sont les longueurs des chaînes X et Y.

La complexité en espace de cet algorithme est de **O(n + m)**