Boucles imbriquées

1 Recherche d'un mot dans un texte

On cherche ici à écrire une fonction `find(mot, texte)` qui retourne le premier indice de la sous-chaîne `mot` dans la chaîne `texte`. Une telle fonction existe dans la plupart des logiciels modernes (raccourci `Ctrl+F` = Find) et permet de trouver dans un document l'emplacement de toutes les occurrences d'un mot donné (par exemple pour le remplacer par un autre).

Exemple. L'éditeur de texte de Spyder possède une telle fonction. En appuyant sur `Ctrl+F` on trouve toutes les occurrences d'un mot, et en appuyant sur `Ctrl+R` (= Replace) on peut par exemple remplacer facilement un nom de variable mal choisi par un autre dans tout un script en une seule opération. Ces deux fonctions existent un peu partout : logiciels Office, LibreOffice, Notepad++, lecteurs de PDF, etc.

Question 1. Écrire une fonction `sontEgales(ch1, ch2)` qui retourne `True` si les deux chaînes `ch1` et `ch2` sont égales, et `False` sinon. La fonction devra impérativement parcourir les deux chaînes (on s'interdit d'utiliser directement `ch1 == ch2`). Quel est le coût de cette fonction ?

Question 2. En déduire une fonction `find(mot, texte)` qui retourne l'indice de la première occurrence de la sous-chaîne `mot` dans la chaîne `texte`. Proposer une valeur qui code l'absence de `mot` dans `texte`. Quel est le coût de cette fonction ?

Exemple. `find('ra', 'abracadabra')` doit retourner `2`. `find('ra', 'bra')` doit retourner `1`.

Question 3. BONUS En déduire une fonction `find_all(mot, texte)` qui renvoie la liste des indices de toutes les occurrences de `mot` dans `texte`.

2 Recherche des deux valeurs les plus proches d'une liste

On considère une liste d'entiers positifs quelconques. On cherche à écrire une fonction `voisins(li)` qui retourne le couple d'entiers appartenant à la liste précédente dont la distance est minimale.

Exemple. `voisins([13, 11, 21, 29, 8, 6, 10, 4, 23, 16])` doit retourner `(11, 10)`.

Question 4. Écrire une fonction `alea(n, N)` qui retourne une liste de `n` entiers aléatoires tous différents compris entre `0` et `N` (inclus). On utilisera la fonction `randint` du module `random`.

Question 5. Écrire la fonction `voisins` et la tester en utilisant la fonction `alea` pour produire automatiquement une liste de test. Quel est le coût de la fonction `voisins` ?

3 Tri à bulles

Problème : on cherche à trier dans l'ordre croissant une liste de `n` entiers positifs tous différents (comme celle retournée par la fonction `alea`). Cette opération est fondamentale en informatique, dans la mesure où elle permet de résoudre une grande quantité d'autres problèmes (recherche d'un élément, plus proches voisins, détermination de la médiane, etc).

L'algorithme du *tri à bulle* est un moyen simple de résoudre ce problème. Principe : on répète `n` fois l'opération qui consiste à parcourir toute la liste et à échanger deux valeurs consécutives si elles ne sont pas dans l'ordre.

Question 6. Écrire une fonction `tamis(li)` qui applique le principe qui consiste à parcourir toute la liste et à échanger deux valeurs consécutives si elles ne sont pas dans l'ordre.

Exemple. `tamis([3, 4, 1, 2])` doir retourner `[3, 1, 2, 4]`.

Question 7. On constate sur l'exemple précédent que la fonction `tamis` place le max de la liste en dernière position. Démontrer par récurrence que ce résultat est vrai pour toute itération `i` en utilisant la propriété : « À l'itération `i`, le max de la sous-liste `li[:i+2]` est à l'indice `i+1` (soit en dernière position) ».

Question 8. En déduire une fonction `bubble_sort(li)` qui retourne la liste `li` triée dans l'ordre croissant. Tester cette fonction sur quelques listes aléatoires générées par la fonction `alea`. Quelle est le coût de la fonction `bubble_sort` ?