TP 2 – Rechercher et ordonner

Objectifs de la séance :

- Intégrer les idées permettant d'automatiser
  - des recherches dans une liste
  - un rangement de liste selon un ordre
- Manipuler des boucles imbriquées

1 Comptage d'éléments dans un tableau

Réaliser une fonction `comptage` qui compte le nombre de fois que l'élément `x` est présent dans une liste `L`.
On pourra la tester en tapant dans la console d'exécution la ligne :
`comptage(5,[1,2,5,3,5,10,5,0,5])` ou la ligne :
`comptage('s','Pour qui sont ces serpents qui sifflent sur vos têtes ?')`

Combien d'instructions de base sont effectuées dans ce programme ?

2 Recherche d'un mot dans un texte

Quelques rappels sur les chaînes de caractères
En python les chaînes de caractère sont des données de type `string`, elles sont délimitées par `'` ou `"`. Elles se manipulent comme des listes.

`len(X)`	longueur de la chaîne X ; nombre de caractères dans la chaîne X
`X+Y`	concaténation des chaînes X et Y c'est-à-dire X suivie de Y
`X[i]`	le $(i+1)$-ème caractère de la chaîne X
`X[i:j]`	caractères d'indices i à j dans la chaîne (si i manque : depuis le début, si j manque, jusqu'à la fin)
`int("23")`	conversion de "23" en l'entier 23

Le problème de la recherche d'un mot `m` dans un texte `t` est fondamental dans de nombreux domaines : écriture de document, recherche de gène sur des chromosomes, recherche d'information via un moteur de recherche, etc ...
Une méthode assez naturelle de recherche consiste à comparer une à une les lettre du mot recherché et les lettres du texte `t` (en partant d'une position k que l'on fait varier)

				b	o	n	b	o	n			
							b	o	n	b	o	n
q	u	e	l	b	o	n	b	o	n	b	o	n

Dans la suite, les mots et les textes sont représentés par des chaînes de caractères (type `string`).

1. Que fait l'algorithme suivant ?

```
def Préfixe(m,t):
    # Entrée : deux chaines de caracteres avec len(m) <= len(t)
    n = len (m)
    for i in range(n):
        if m[i] != t[i]:
            return False
    return True
```

2. Expliquer en quoi l'indentation de la dernière ligne de ce programme est très importante.

3. Écrire une fonction `Présent(m,t)` qui décide si le mot `m` est présent dans le texte `t` à l'aide d'une boucle imbriquée dans une autre boucle.

4. Proposer d'écrire la fonction `Présent(m,t)` de façon plus simple à l'aide des fonctions de chaînes de caractère.

5. Combien d'instructions de base sont effectuées lorsque l'on utilise `Present(m,t)` ?

6. Si on souhaite minimiser le nombre d'instructions effectuées, vaut il mieux utiliser des boucles `for` ou des boucles `while` ?

3 Le tri bulle

1. Que fait le code suivant ? On pourra commencer par faire un exemple

```
def Mystere(L,k):
    # Entree : une liste d'entiers L et un indice k < len(L)
    for j in range (k):
        if L[j +1] < L[j]:
            L[j], L[j+1] = L[j+1], L[j]
    return L
```

2. En déduire un algorithme de tri par ordre croissant de listes d'entiers, que vous implémenterez en Python (cet algorithme de tri s'appelle s'appelle le tri bulle, « bubble sort » en anglais).