Chaînes de caractères [1]

Une chaîne de caractères est une suite de lettres, nombres ou caractères spéciaux. Ce qui permet de stocker du texte. Recopiez et comprenez le code suivant :

```
s="Je m'appelle Dupont"
print(s)
sbis=" et moi Dupond"
ster=s+sbis#Concaténation de deux chaînes
print(ster)
print(s[0])
print(s[2])
print(len(s))
print(s[25])#Provoque une erreur, pourquoi?
print(s*3)
```

Remarquez que, contrairement aux listes, on ne pas modifier les éléments d'une chaîne de caractères [2] :

```
L=[1,2,4,8,14]
L[4]=16
print(L)#L a donc été modifiée
s[4]='e'#Refus de Python de modifier s
```

Exercice 1 (⋆)**.** Écrire une fonction `AfficherChaine` à un paramètre (une chaîne de caractères) qui, à l'aide d'une boucle `for` affiche successivement tous les caractères [3].

Exercice 2 (⋆)**.** Écrire une fonction `Occurrence(C,x)` qui renvoie le nombre d'occurrences de `x` dans `C`. Ainsi, `Occurrence("Babar","a")` vaudra 2. Pour cela, parcourir tous les caractères de `C` et voir s'ils valent `x`.

```
's' in 'ça va ?'#booléen qui vaut False
'a' in 'ça va ?'#booléen qui vaut True
2 in [1,2,4]#booléen qui vaut True
```

Exercice 3 (⋆)**.** Écrire une fonction `CompterVoyelles` qui, à une chaîne de caractère `C`, renvoie le nombre de voyelles dans cette chaîne. Ainsi, `CompterVoyelles("informatique")` vaut six. Pour cela, il suffit de parcourir les caractères de `C` un à un et voir si c'est une voyelle.

Exercice 4 (𝄞⋆⋆)**.** Écrire une fonction `RechercheMot(texte,mot)` où texte et mot sont deux chaînes de caractères qui renvoie `True` si le mot se trouve dans `texte`. Pour cela, on ne fera que des comparaisons lettres à lettres. Pour tous les `i` possibles, on vérifiera si le mot se trouve en position `i`, c'est-à-dire si `texte[i]=mot[0]`, `texte[i+1]=mot[1]` etc. Ainsi, si `texte="À la claire fontaine"` et `mot="clai"` la fonction renvoie `True`. En effet, `texte[5]=mot[0]`, `texte[6]=mot[1]`, `texte[7]=mot[2]`, `texte[8]=mot[3]`. En revanche, si `mot="Claire"`, la fonction renvoie `False` [4].

Exercice 5 (⋆)**.** Écrire une fonction `inverse` qui prend en paramètre une chaîne de caractère `s` et qui renvoie la chaîne de caractère renversée de `s`. Ainsi, `inverse("table")` vaudra "elbat". En déduire une fonction `palindrome` qui prend en paramètre un mot et qui renvoie `True` si ce mot est un palindrome. Les mots `ressasser` et `kayak` sont des palindromes contrairement à `rêver`.

Exercice 6 (⋆)**.** Écrire une fonction `MemeElements` à deux paramètres (deux chaines de caractères) renvoie `True` si elles ont les mêmes caractères (mais pas forcément dans le même ordre ni avec le même nombre d'occurrences) et `False` sinon.

Exercice 7 (⋆⋆)**.** Écrire une fonction `LePlusSouvent(C)` qui, à une chaîne de caractères, renvoie la liste des caractères qui sont le plus apparus. On pourra utiliser la fonction `Occurence`. Par exemple `LePlusSouvent("informatique c'est genial")` renverra la liste `[i,e]`.

Retour sur le TP4

Faire l'exercice sur le tri à bulle du TP4

1. Pas un chêne de caractère
2. Si on veut briller lors d'un dîner, on parle d'immuabilité d'une chaîne de caractères.
3. Remarquez que cette fonction n'a donc pas de `return`.
4. Car, oui Python respecte la casse (les majuscules et minuscules sont différentes).