

Séance 3 : BOUCLES FOR ET SÉQUENCES

L1 – Université Côte d'Azur

Exercice 1 – Calcul de la fréquence des mots d'un texte

Pour chaque fonction de l'exercice, écrire des tests.

1. Créez une chaîne `alphabet="abcdefghijklmnopqrstuvwxyz"` contenant toutes les lettres de l'alphabet, dans l'ordre et en minuscule. Vérifiez avec `assert` que sa taille est bien de 26.
2. Écrivez une fonction `indice(lettre)` qui donne l'indice d'un caractère dans la chaîne `alphabet`. On renverra 26 si la lettre n'est pas dans l'alphabet. Ainsi, `indice('a')` vaut 0 et `indice('z')` vaut 25
3. Écrire une fonction `statistiques(texte)` qui renvoie un tableau de taille 27 à partir d'un texte non vide. Ce tableau contient dans chaque case d'indice `i` le nombre de fois où la lettre d'indice `i` est apparue dans le texte.

```

1 >>> tableau = statistiques("salut à toi mon ami")
2 >>> tableau[0] # nombre de a (« à » ne compte pas)
3 2
4 >>> tableau[1] # nombre de b
5 0
6 >>> tableau[26] # nombre d'autres symboles (4 espaces et le « à »)
7 5

```

4. Écrire une fonction `affichage(tableau)` qui affiche un version lisible du tableau, pour chaque case non vide, on affichera la lettre ainsi que son nombre d'occurrence.

```

1 >>> affichage(tableau)
2 a : 2
3 i : 2
4 l : 1
5 m : 2
6 n : 1
7 o : 2
8 s : 1
9 t : 2
10 u : 1
11 Autres : 5

```

5. Écrire une fonction `plus_fréquent(tableau)` qui renvoie la lettre la plus fréquente du texte. Si deux lettres ont la même fréquence, on choisira la première dans l'ordre alphabétique.

```

1 >>> plus_fréquent(tableau)
2 'a'

```

6. Comment modifier la fonction pour renvoyer, dans le cas où plusieurs lettres ont la même fréquence, la dernière dans l'ordre alphabétique? On appellera cette nouvelle variante `plus_fréquent_bis(texte)`.

```

1 >>> plus_fréquent_bis(tableau)
2 't'

```

7. Enfin, écrivez une fonction, `tous_les_plus_fréquents(texte)` qui renvoie la chaîne contenant toutes les lettres de l'alphabet ayant la fréquence maximale.

```

1 >>> tous_les_plus_fréquents(tableau)
2 'aimot'

```

Exercice 2 – Concaténer

Écrivez et tester une fonction `concatenation(L1, L2)` qui à partir de deux listes L1 et L2 renvoie la liste L1+L2. On ne pourra pas utiliser l'opération + de Python. Il faudra créer un tableau de la bonne taille avant de le remplir.

Exercice 3 – Tables de vérité

1. Écrivez une fonction `car` qui transforme chaque booléen en caractère. On aura, `car(True)=='T'` et `car(False)=='F'`.
2. Écrivez `table_2_vérité` qui prend une fonction booléenne à deux arguments et trace sa table de vérité. On utilisera **obligatoirement** des boucles `for` sur la liste `L = [True, False]`.

```
1 def formule(a,b):
2     return not (a and b)
```

```
1 >>> tables_2_vérité(formule)
2 a b P
3 T T F
4 T F T
5 F T T
6 F F T
```

3. Même question avec `table_3_vérité` qui prend des fonctions booléennes à trois arguments. Testez sur la formule du TD 0 : `(a or not c) and (not a or b) and (not b or c)`.
4. Démontrer à la main des égalités mathématiques, c'est dépassé. Écrivez une fonction `égalité_formules_2(f1, f2)` qui prend des fonctions booléennes à deux arguments et revoie `True` si elles sont égales (et `False` sinon).
5. Utilisez votre fonction pour démontrer les lois de Morgan de manière automatique.
`not (a and b) = not a or not b` ainsi que `not (a or b) = not a and not b`

Exercice 4 – Print Proust

Écrivez une fonction `justification_à_gauche(s, n)` qui prend une chaîne de caractères `s` contenant une phrase (sans symbole `'\n'`) et qui l'affiche en mettant au plus `n` caractères par ligne, en justifiant à gauche (si un mot est trop grand il sera seul sur sa ligne).

```
1 >>> justification_à_gauche("Mais au lieu de la simplicité, c'est le faste que..." , 12)
2 Mais au lieu
3 de la
4 simplicité,
5 c'est le
6 faste que...
```

Testez votre programme en affichant cette phrase¹ de Proust en entier² sur 80 caractères par ligne.

Mais au lieu de la simplicité, c'est le faste que je mettais au plus haut rang, si, après que j'avais forcé Françoise, qui n'en pouvait plus et disait que les jambes « lui rentraient », à faire les cent pas pendant une heure, je voyais enfin, débouchant de l'allée qui vient de la Porte Dauphine – image pour moi d'un prestige royal, d'une arrivée souveraine telle qu'aucune reine véritable n'a pu m'en donner l'impression dans la suite, parce que j'avais de leur pouvoir une notion moins vague et plus expérimentale – emportée par le vol de deux chevaux ardents, minces et contournés comme on en voit dans les dessins de Constantin Guys, portant établi sur son siège un énorme cocher fourré comme un cosaque, à côté d'un petit groom rappelant le « tigre » de « feu Baudenord », je voyais – ou plutôt je sentais imprimer sa forme dans mon coeur par une nette et épuisante blessure – une incomparable victoria, à desseins un peu haute et laissant passer à travers son luxe « dernier cri » des allusions aux formes anciennes, au fond de laquelle reposait avec abandon Mme Swann, ses cheveux maintenant blonds avec une seule mèche grise ceints d'un mince bandeau de fleurs, le plus souvent des violettes, d'où descendaient de longs voiles, à la main une ombrelle mauve, aux lèvres un sourire ambigu où je ne voyais que la bienveillance d'une Majesté et où il y avait surtout la provocation de la cocotte, et qu'elle inclinait avec douceur sur les personnes qui la saluaient.

Exercice 5 – Tapis : le retour !

À faire chez vous : refaire les tapis du TP 2 mais en utilisant uniquement des boucles `for` (donc sans `while`) et sans utiliser de fonctions auxiliaires (donc vous êtes obligés de faire une boucle `for` dans une boucle `for`).

1. oui, c'est une seule phrase!

2. Si nécessaire le texte est disponible à l'adresse : <https://upinfo.univ-cotedazur.fr/~obaldellon/L1/py/tp3/marcel-proust.html>