Fiche d'exercice : Les tuples et les listes

Attention

Tous les exercices, s'ils sont réalisés par des boucles `for`, doivent être faits avec une boucle sur les indices et une boucle sur les valeurs.

```
l = [1,2,3,4]
# for par valeur:
for elt in l:
   print(l)


# for par indice (utilisation de len() qui permet d'avoir la taille d'une séquence):
for i in range(len(l)):
   print(l[i])
```

Exercices d'introduction

1. **Créer un tuple nommé `mon_tuple` qui contient les éléments 1,2,3,4,5 et une liste `ma_liste` qui contient les éléments suivants 'a','b','c','d','e'. **
   Les afficher dans le terminal.
2. **Afficher dans le terminal le troisième élément de `mon_tuple` et le premier élément de `ma_liste`.**
3. **Modifier le deuxième élément de `ma_liste` par 'z'.**
   Vérifier la modification en affichant la liste dans le terminal.
4. **Ajouter l'élément 'f' dans `ma_liste`.**
   Supprimer le premier élément de `ma_liste`.
   Vérifier les modifications en affichant la liste dans le terminal.
5. **Afficher dans le terminal tous les éléments de `mon_tuple` et `ma_liste` un à un à l'aide d'une boucle `for`.**
6. **Créer une liste `nombres` qui contient les chiffres allant de 1 à 9. (Proposer une version par compréhension).**
   Afficher dans le terminal les 5 premiers éléments de la liste en utilisant une boucle while.
   Afficher les éléments du quatrième au huitième en utilisant une boucle for.
   Afficher les éléments de la liste `nombres` dans le sens inverse en utilisant une boucle while.

Tuples et fonctions

{Algorithme à savoir}

Écrire une fonction qui prend en paramètre une valeur et une liste et renvoie `True` si la valeur demandée est dans la liste, `False` sinon.

Écrire une fonction `moyenne` qui prend en paramètre une liste d'entiers et renvoie la moyenne de tous les nombres présents dans ce liste.

Écrire une fonction `somme` qui prend en paramètre une liste d'entiers et renvoie la somme des éléments de cette liste.

Écrire une fonction `produit` qui prend en paramètre une liste d'entiers et renvoie la produit des éléments de cette liste.

Cet exercice met en oeuvre une modification par **effet de bord.**

Écrire une fonction `echange` qui prend en paramètres une liste et deux indices et échange les valeurs aux positions i et j dans la liste passée en paramètres.

Écrire une fonction `inverser_tableau` qui prend en paramètre un liste et renvoie une autre liste qui contient tous les éléments de celui en paramètre mais dans le sens inverse.

On souhaite modéliser un jeu de cartes. Chaque carte sera instanciée par un liste (nombre, couleur). On souhaite vérifier qu'une carte créée soit valide. Une carte est valide si le nombre est compris entre 1 et 13 et si la couleur est soit "Coeur", "Trèfle", "Pique" ou "Carreaux".

Écrire une fonction `est_valide` qui prend en paramètre une liste correspondant à une carte et renvoie True si la carte est valide, False sinon.

Écrire une fonction `compter_occurrences` qui prend en paramètre une liste et un élément, et renvoie le nombre de fois que cet élément apparaît dans la liste.

Écrire une fonction `rangement_valeurs` qui prend en paramètre une liste et un élément et renvoie 3 listes : une liste contenant les valeurs inférieure à la valeur passée en paramètre, une liste contenant uniquement la valeur passée en paramètre si elle est présente, une dernière contenant tous les éléments supérieurs à la valeur passée en paramètre.
Exemple:

```
>>> rangement_valeurs([1,7,4,3,6,2,8],5)
[1,4,3,2],[],[7,6,8]
>>> rangement_valeurs([1,2,4,3,6,2,8],2)
[1],[2,2],[4,3,6,8]
```

{Algorithme à savoir}

Écrire une fonction `minimum_liste` qui prend en paramètre une liste d'entiers et renvoie la valeur minimum de la liste et son indice.

{Algorithme à savoir}

Écrire une fonction `maximum_liste` qui prend en paramètre une liste d'entiers et renvoie la valeur maximum de la liste et son indice.

Écrire une fonction `fusionner_sans_doublons` qui prend en paramètres deux listes et renvoie une nouvelle liste contenant tous les éléments des deux listes sans doublons.

Écrire une fonction `diviseurs` qui prend un entier en paramètre et renvoie la liste de ses diviseurs.

Écrire une fonction `est_croissante` qui prend une liste d'entiers en paramètre et renvoie True si les éléments de la liste sont dans l'ordre croissant, False sinon.

On souhaite réaliser des opérations sur des points d'un plan. On va représenter un point comme étant une liste (abscisse, ordonnée).

1. Instancier un point A d'abscisse -3 et d'ordonnée 2.
   Instancier un point B d'abscisse 4 et d'ordonnée 4.
2. **Écrire une fonction `milieu` qui prend 2 points en paramètres et renvoie le point situé au milieu des 2 en paramètres.**
   On rappelle que la formule pour trouver le point du milieu est `milieu`$(\frac{x_A+x_B}{2}, \frac{y_A+y_B}{2})$.
3. **Écrire une fonction `distance` qui prend en paramètres deux points et renvoie la distance entre les deux points.**
   On rappelle la formule pour trouver la distance $\sqrt{((x_A-x_B)^2+(y_A-y_B)^2)}$.

Matrices

On rappelle qu'une matrice est un tableau à deux dimensions. En python, pour les représenter, on réalise des listes de listes.

1. On dispose de la matrice 1,2,3],[4,5,6],[7,8,9.
   i. Donner la position (indice de la ligne et de la colonne) de la valeur 2.
   ii. Donner la valeur à la ligne 1 et colonne 2.
   iii. Comment en python retrouver la valeur cherchée en question 2.
2. On dispose de la matrice 1,2,3],[4,5,6],[7,8,9.
   i. Écrire un programme qui permet de donner la somme de tous les nombres de cette matrice.
   ii. Tester avec d'autres matrices plus grandes.
3. Créer une matrice par compréhension qui contient tous les nombres de 0 à 11. Chacune des lignes de la matrice doivent avoir une taille de 4.

Listes par compréhension

1. On dispose de la compréhension suivante [i for i in range(10)].
   i. Quelle liste est créée par cette compréhension?
   ii. Modifier cette compréhension pour donner le carré de chaque nombre.
2. On peut rajouter des conditions dans des compréhensions pour éviter certaines valeurs.
   i. Quelle liste est créée par la compréhension suivante : [i for i in range(20) if i %2 == 0] ?
   ii. Modifier cette compréhension pour quelle fasse l'inverse.
3. Créer une liste par compréhension qui contient les racines carrées des nombres allant de 1 jusque 20. On rappelle que le module `math` dispose de la fonction `sqrt` qui permet de calculer les racines carrées.