Les différentes structures de données

Tableaux(listes)

Les tableaux (ou listes) se définissent en Python avec des crochets : `[]`.
Par exemple : `l1 = [4,5,11]`.

- **Accès**: L'accès d'un élément se fait à partir de son indice.
    - `l1[1]` donne `5`
- **Ajout**: Il est possible d'ajouter un élément à une liste en utilisant la méthode `.append()` ou en concaténant 2 listes ensemble avec l'opérateur `+`.
    - l'instruction `l1.append(7)` modifie la valeur de `l1` : `[4,5,11,7]`
    - l'instruction `l1+[7]` créer une nouvelle liste contenant les valeurs `[4,5,11,7]`
- **Modification** : On peut modifier un élément d'une liste à partir de son indice.
    - Après l'instruction `l1[2] = 6`, on aura `l1` qui vaut `[4,5,6]`.
- **Parcours** :
    - Parcours par élément : `for element in l1 :`, la variable `element` va prendre les valeurs `4`, `5` puis `11`.
    - Parcours par indice : `for i in range(len(l1)) :`, la variable `i` va prendre les valeurs `0`, `1` puis `2`.

Tuples

Les tuples se définissent en Python avec des parenthèses : `()`.
Par exemple : `t1 = (4,5,11)`.

- **Accès**: L'accès d'un élément se fait à partir de son indice.
    - `t1[1]` donne `5`
- **Ajout**: Il est possible de concaténer 2 tuples ensemble avec l'opérateur `+`.
    - l'instruction `t1+(7,)` créer une nouvelle liste contenant les valeurs `(4,5,11,7)`.
    - ! note : si on souhaite concaténer un seul élément à notre tuple il est oligatoire de mettre une virgule juste après !
- **Modification** : Il est impossible de modifier un élément dans un tuple. On dit que les tuples sont **immuables**.
- **Parcours** : Le parcour d'un tuple se fait de la même manière que pour une liste.
    - Par élément : `for element in t1 :`
    - Par indice : `for i in range(len(t1)) :`

Dictionnaires

Les dictionnaires se définissent en Python avec des accolades : `{}`.
Un dictionnaire est constitué de couples `cles : valeurs`.
! Attention ! : Les clés d'un dictionnaire doivent être non mutables(qu'on ne peut pas modifier).
Par exemple : `d1 = {"a": 1, "b" : 2, "c" : 3}`.

- **Accès**: On peut accéder aux valeurs des dictionaiires à partir de leurs clés.
    - Exemple : `d1["b"]` donne `2`
- **Ajout**: Il est possible d'ajouter un couple `cle : valeur` si la clé n'est pas déjà présente dans le dictionnaire.
    - Exemple : Après l'instruction `d1["d"] = 4`, `d1` vaut `{"a": 1, "b" : 2, "c" : 3, "d" : 4}`
- **Modification** : Il est possible de modifier la valeur d'un dictionnaire à partir de sa clé, si la clé est déjà présente dans le dictionnaire.
    - Exemple : Après l'instruction `d1["a"] = 27`, `d1` vaut `{"a": 27, "b" : 2, "c" : 3}`
- **Parcours** : Il est possible de parcourir un dictionnaire de différentes façon.
    - Parcours par clé avec avec l'instruction `for cle in d1 :` ou `for cle in d1.keys() :`
    - Parcours par valeurs avec l'instruction `for val in d1.values() :`
    - Parcours par clé et valeurs avec l'instruction `for cle, val in d1.items() :`

Tableau récapitulatif

	Tableaux(listes)	Tuples	Dictionnaires
Constructeur	`[]` ou `list()`	`()` ou `tuple()`	`{}` ou `dict()`
Ordonnée(possède des			

indices)	Oui	Oui	Non
	Tableaux(listes)	**Tuples**	**Dictionnaires**
Mutable(modifiable)	Oui	Non	Oui