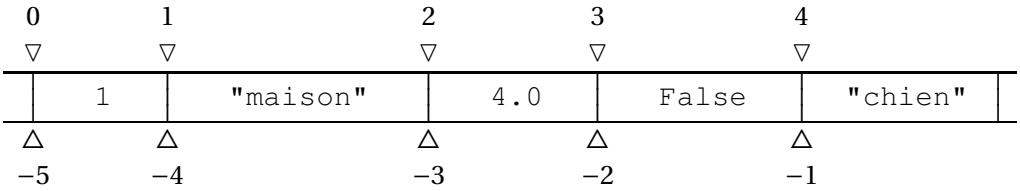
Séance 10

# LES LISTES (1/2)

## L'essentiel

- Une liste est une structure qui permet de stocker des données : `>>>L=[1,"maison",4.0,False,"chien"]`
- Pour connaître la longueur d'une liste `L` : `>>>len(L)`.
- Pour accéder à l'élément d'indice `i` de la liste `L` : `>>>L[i]`.
  Attention :
  - Une liste est indexée à partir de 0.
  - Le dernier élément a pour indice `len(L)-1`.
  - `L[len(L)]` renvoie une erreur (`list index out of range`).




- On peut parcourir la liste `L` dans l'ordre décroissant : `>>>L[-i]`.
- On peut extraire une sous-liste de la liste `L` : `>>>L[1:3]` renvoie `["maison",4.0]`.
- Pour parcourir une liste, on peut :
  - parcourir les indices :
  ```
  for i in range(len(L))
      print(L[i])
  ```
  - parcourir les éléments :
  ```
  for el in L :
      print(el)
  ```
- Création de liste en compréhension :
  On peut créer une liste à partir d'une fonction `f` : `>>>[f(k) for k in range(3,6)]`.
  Remarque : on peut également utiliser une autre liste `l` : `>>>[f(k) for k in l]`
- Attention : l'opérateur + pour des listes correspond à la concaténation (mise bout à bout).

---

## Exercices du jour

**Exercice 1:** 1. Que fait la fonction suivante?

```
def pouet(L, element):
    for e in L:
        if e == element:
            return True
    return False
```

2. Modifier la fonction précédente pour qu'elle ait le même effet, mais en réalisant une boucle sur les indices plutôt que sur les valeurs.
3. Modifier alors la fonction en une fonction `indice` pour qu'elle renvoie l'indice de `element` dans la liste `L` et -1 si `element` n'est pas présent. (*On donnera l'indice de la première apparition si l'élément est présent plusieurs fois*).
4. En s'inspirant du code précédent, réaliser une fonction `occurence` avec les mêmes paramètres et qui renvoie le nombre d'occurrences de `element` dans la liste `L`, c'est-à-dire le nombre de fois à l'élément apparait.

**Exercice 2:** On considère dans cet exercice que `L` est nécessairement une liste composée d'entiers. La fonction suivante doit servir à vérifier si tous les éléments de `L` sont pairs.

```
def VerifPair(L) :
    for e in L:
        if e%2 == 0 :
            return True
        else :
            return False
```

Pourtant `VerifPair([2,3,4,6])` renvoie `True`. Pourquoi? Corriger le problème.

**Exercice 3:**

1. Créer une fonction `MaximumListe` qui prend en entrée une liste non vide de nombres `L` et renvoie le plus grand nombre présent dans cette liste. *Remarque : cette fonction existe déjà sous python et s'appelle* `max`*... bien sûr dans cet exercice, on s'interdit de l'utiliser!*
2. Modifier la fonction précédente afin qu'elle renvoie le plus grand nombre ainsi que son indice sous forme d'un couple. (*On donnera l'indice de la première apparition si le plus grand nombre est répété dans la liste*).

## Exercices en autonomie

**Exercice 4:** Écrire une fonction Python qui prend en entrée une liste de nombres entiers `L` et renvoie le nombre de multiple de 3 présents dans cette liste.

**Exercice 5:** Écrire une fonction Python qui prend en paramètres deux listes et affiche les éléments communs aux deux listes (*On pourra utiliser la fonction de l'exercice 1*).

## Aide pour les exercices

## Solutions des exercices

***Correction 4*** Sur demande, envoyez un mail à `vincent.maille@ac-amiens.fr`.

***Correction 5*** Sur demande, envoyez un mail à `vincent.maille@ac-amiens.fr`.