TD : Les tableaux en Python

L'objectif de ce TD est vous familiariser avec la notion de tableau.

1) Création d'un tableau

a) Premiers exemples

Question 1. Créer une variable pour représenter chacun des tableaux suivants :

- Un tableau contenant les nombres 3, 1, 4, 5, 9, 2, 6, 5, 3
- Un tableau contenant toutes les voyelles de l'alphabet (en minuscule)
- Un tableau contenant les multiples de 11 inférieurs à 100

Question 2. Modifier le premier tableau de la question 1 pour obtenir 3, 1, 4, 5, 2, 2, 6, 5, 0

b) Avec des fonctions

Question 3. Créer une fonction qui renvoie un tableau contenant les `n` premiers nombres entiers positifs (à partir de 0), où `n` est l'argument de la fonction.

Question 4. Créer une fonction qui renvoie un tableau contenant les `n` premiers nombres entiers impairs, où `n` est l'argument de la fonction.

c) Génération aléatoire

Question 5. Créer une fonction qui génère et renvoie un tableau aléatoire à partir des éléments passés en arguments :

- la taille du tableau à créer
- la valeur minimale autorisée (incluse)
- la valeur maximale autorisée (incluse)

Rappel : Pour obtenir des nombres aléatoires en Python, on utilise les lignes de codes suivantes :

- Au début du fichier : `import random` (import des fonctions de génération aléatoire)
- Au début du fichier : `random.seed()` (initialisation du générateur aléatoire)
- Dans le code : `random.randint(a, b)` (tirage aléatoire entre `a` et `b` inclus)

2) Manipulation sur les tableaux

a) Premières fonctions

Question 6. Ecrire une fonction qui prend en argument un tableau et affiche successivement chacun des éléments de ce tableau, en passant à la ligne à chaque fois.

Question 7. Ecrire une fonction qui prend en argument un tableau de nombres entiers et renvoie le nombre d'éléments impairs dans ce tableau

Question 8. Ecrire une fonction qui prend en argument un tableau de nombres et renvoie la moyenne des éléments contenus dans ce tableau.

b) Valeur et position du maximum

Question 9. Ecrire une fonction qui renvoie le maximum du tableau passé en argument (c'est-à-dire la plus grande valeur contenue dans ce tableau).

Question 10. Ecrire une fonction qui renvoie la position du maximum du tableau passé en argument (c'est-à-dire l'indice de la case où apparaît ce maximum).

Remarque : Si ce maximum apparaît plusieurs fois, la fonction doit renvoyer la position de la première occurrence.

c) Recherche d'un élément dans un tableau non trié

Question 11. Ecrire une fonction qui prend en argument un élément `x` et un tableau `t`, et qui détermine (en renvoyant un booléen) si l'élément `x` apparait dans le tableau `t`.

Remarque : Il existe en Python le mot-clef `break`. Lorsque le programme atteint ce mot-clef lors de son exécution, il sort immédiatement de la boucle dans laquelle il se trouvait.

d) Ordre croissant

Question 12. Ecrire une fonction qui prend en argument un tableau et détermine s'il est trié par ordre croissant (chaque élément doit être inférieur ou égal au suivant).

3) Recherche dans un tableau trié

On suppose désormais qu'on travaille avec des tableaux triés par ordre croissant : on va voir que cela permet d'écrire des algorithmes de recherche beaucoup plus rapides.

a) Recherche entre deux indices

★ **Question 13.** Ecrire une fonction qui prend en argument :

- un tableau `t` (qu'on suppose trié)
- une valeur `x`
- un premier indice `a`
- un deuxième indice `b`

et qui détermine (en renvoyant un booléen) si l'élément `x` apparaît dans le tableau `t` entre les indices `a` et `b` (inclus).

Remarque : On suppose que $a \leq b$: si ce n'est pas le cas, la fonction doit renvoyer `False`.

Indice : Pensez récursif !

b) Encapsulation

Dès lors qu'on a implémenté cette fonction auxiliaire, il suffit de l'exécuter avec `a=0` et `b=len(t)-1` (ces indices couvrent l'intégralité du vecteur v).

Néanmoins, l'utilisateur final n'a pas à avoir accès à la fonction auxiliaire : la seule fonction qui l'intéresse est celle qui indique si l'élément recherché apparait dans le vecteur.

On utilise donc le procédé dit d' "encapsulation" : on déclare la fonction auxiliaire à l'intérieur de la fonction principale pour ne présenter que cette dernière à l'utilisateur.

★ **Question 14.** Utiliser l'encapsulation pour écrire une fonction qui prend en argument un tableau `t` (qu'on suppose trié) et une valeur `x` et qui détermine (en renvoyant un booléen) si l'élément `x` apparaît dans le tableau `t`.

c) Premières notions de complexité

★ **Question 15.** Comparer les performances des fonctions écrites aux questions 11 et 14 en les testant sur des tableaux de plus en plus longs. Que remarquez-vous ? Comment expliquez-vous ces résultats ?