Lycée

Informatique
Première

Traveaux Pratiques : les tableaux

Exercice 1 (*Mise en bouche*)**.** Dans l'interpréteur Python, taper l'instruction suivante :

```
t = [4, 2, 5, 1, 3].
```

1. Quelle est la taille de ce tableau ? Quel fonction Python pouvez-vous utiliser ?
2. Quel en est le premier élément, le dernier ? Quelle instruction Python allez-vous utiliser pour les faire afficher dans l'interpréteur ?
3. On tape dans l'interpréteur successivement les commandes suivantes (on appuie sur entrée entre chacunes d'entres elles) :

   ```
   t[1]      t[4]      t[5]
   ```

   Quels résultats attendez vous pour chacune des commandes ? Vérifier vos hypothèses.
4. Quelle instruction Python devez vous utiliser pour transformer le tableau `t` en le tableau : `[4,2,8,1,3]`

Exercice 2 (*Remplacement et ajouts*)**.** On décide de stocker dans un tableau `liste_prenoms` les prénoms d'une bande d'amis.

1. Dans l'interpréteur, taper l'instruction Python qui permet de créer un tableau `liste_prenoms` de taille 4 et d'y stocker les chaînes de caractères `"Thomas"`, `"Sam"`, `"Line"`, `"Jules"`.
2. Jules quitte le groupe d'ami, et Kinan le remplace. Dans l'interpréteur, taper l'instruction qui permet de mettre à jour le tableau.
3. Antoine, l'ami de Kinan, souhaite aussi se joindre au groupe. Dans l'interpréteur, taper l'instruction qui permet de rajouter (à la fin...) `"Antoine"` au tableau `liste_prenoms`.

Exercice 3 (*Mais que se passe-t-il ?*)**.**

1. Dans l'interpréteur, taper les commandes suivantes :

   Console Python

   ```
   >>> t1 = [5, 7, 8, 2]
   >>> t2 = t1
   >>> print(t2)
   >>> t1[0] = 100
   >>> print(t2)
   ```

   Que s'est-il passé ? Commenter.
2. Dans la partie éditeur de texte de votre IDE, taper en première ligne `t1 = [5, 7, 8, 2]`. À la seconde ligne, créer une variable de type tableau `t2`, de même taille que `t1` mais ne contenant que des 0.

3. Recopier et compléter à la suite de votre code le code suivant pour que celui-ci recopie les valeurs présentes dans `t1` dans `t2` :

Code Source

```
for i in ......................:
    t2[i] = ...................
```

4. Recopier à la fin de votre fichier les instructions 3 à 5 de la question 1. Exécuter votre code puis commenter.
5. Écrire une fonction `recopier` qui prend en entrée un tableau `t` et qui renvoie en sortie un tableau `copie` qui est une copie identique indépendante de `t`. Vous testerez votre fonction :

Console Python

```
>>> t1 = [5, 7, 8, 2]
>>> t2 = recopier(t1)
>>> print(t2)
[5, 7, 8, 2]
>>> t1[0] = 100
>>> print(t2)
[5, 7, 8, 2]
```

Exercice 4.

1. Créer un tableau `t` de taille 157 et dont le contenu de toutes les cases est égal à 0.
2. En utilisant une boucle `for` :
   (a) remplir le tableau avec les entiers de 1 à 157. À la fin de l'exécution de votre code `t` est le tableau : `[1,2,3, ..., 157]`
   (b) remplir le tableau avec la liste des 157 premiers nombres pairs positifs. À la fin de l'exécution de votre code `t` est le tableau : `[0, 2, ..., 312]`
   (c) remplir le tableau avec les valeurs $+1$ et -1 de la manière suivante : si l'indice de la case est pair, la case contient la valeur $+1$, si l'indice de la case est impair, la case contient la valeur -1. À la fin de l'excécution de votre code `t` est le tableau `[1, -1, 1, ..., 1]`

Traveaux Pratiques : réalisation d'un module

Introduction. L'objectif de cette partie est la réalisation d'un module Python de manipulation de tableaux. Un module est un fichier qui contient une liste de fonctions et de variables en rapport avec un même thème. Vous connaissez déjà le module `math`, il contient des fonctions comme `cos`, `sqrt`, des variables comme `pi`, `e`, etc.

Ouvrir un nouveau fichier intitulé `tableaux.py`. La première fonction utile de ce module sera la fonction `recopier`. Copier-coller la fonction que vous avez écrite à l'intérieur de ce fichier.

Exercice 1 (*Sous-tableaux*)**.** Soit `t` un tableau. On appelle **sous-tableau** de tranche `i` à `j` le tableau constitué des éléments de `t` d'indices `i`, `i+1`, ... `j-1` (`j` est **exclu**).

1. `t` est le tableau `[4, 5, 8, 9, 6, 3, 56, 2, -4, 13]`.
   (a) Écrire le sous-tableau de `t` de tranche 2 à 5. .....
   (b) Écrire le sous-tableau de `t` de tranche 0 à 6. .....
   (c) Écrire le sous-tableau de `t` de tranche 3 à 10. .....
2. Donner la taille de chacun des sous-tableaux précédents. Si `n` est la taille d'un sous-tableau de `t` de tranche `i` à `j`, quelle est la relation entre `n`, `i`, et `j` ?
3. Écrire une fonction `sous_tableau` qui prend en entrée un tableau `t` et deux entiers `i` et `j` et renvoie en sortie le sous-tableau de `t` de tranche `i` à `j`. Vous testerez votre fonction avec les sous-tableaux calculés en question 1.

Console Python

```
>>> t = [4, 5, 8, 9, 6, 3, 56, 2, -4, 13]
>>> sous_tableau(t, 2, 5)
>>> sous_tableau(t, 0, 6)
>>> sous_tableau(t, 3, 10)
```

Exercice 2 (*Concaténation*)**.** Soient `t1` et `t2` deux tableaux. La concaténation de `t1` puis de `t2` est le tableau constitué des éléments de `t1` puis des éléments de `t2` **dans cet ordre**.

1. Soient les tableaux `t1 = [1, 3, 5]` et `t2 = [2, 4, 6]`.
   (a) Écrire la concaténation de `t1` puis de `t2`. .....
   (b) Écrire la concaténation de `t2` puis de `t1`. .....
   (c) Écrire la concaténation de `t1` puis de `t1`. .....
   (d) Écrire la concaténation de `t2` puis de `[]`. .....
2. Donner la taille de chacunes des concaténation de tableaux précédentes. Si `n` est la taille du tableau concaténé de `t1` puis de `t2`, `n1` et `n2` les tailles respectives de `t1` et `t2`, quelle est la relation entre `n`, `n1` et `n2` ?

3. Écrire une fonction `concatener` qui prend en entrée deux tableaux `t1` et `t2` et renvoie en sortie le le tableau résultant de la concaténation de `t1` puis de `t2`. Vous testerez votre fonction en comparant avec vos réponses à la question 1.

Console Python

```
>>> t1 = [1, 3, 5]
>>> t2 = [2, 4, 6]
>>> concatener(t1, t2)
>>> concatener(t2, t1)
>>> concatener(t1, t1)
>>> concatener(t2, [])
```

Exercice 3 (*Insérer un élément*)**.** Soient `t` un tableau, `i` un nombre entier inférieur ou égal à la taille du tableau, et `e` un élément. On souhaite construire un nouveau tableau `tprime` copie de `t` à ceci près qu'à l'indice `i` se trouve l'élément `e`, les éléments suivants se trouvant décalés.

1. Soient le tableau `t = [5, 7, 9, 5, 3]`, et l'élément `e = 666`.
   (a) Écrire le tableau `t` dans lequel on a inséré l'élément `e` à l'indice 3. .....
   (b) Écrire le tableau `t` dans lequel on a inséré l'élément `e` à l'indice 0. .....
   (c) Écrire le tableau `t` dans lequel on a inséré l'élément `e` à l'indice `5` .....
2. Donner la taille de chacuns des tableau précédents. Si `n` est la taille du tableau `t`, quelle est la taille du tableau `tprime` ?
3. Écrire une fonction `inserer` qui prend en entrée un tableau `t`, un indice `i` et un élémént `e` et qui renvoie en sortie le tableau `tprime` dans lequel a été inséré l'élément `e` a l'indice `i`. Vous testerez vos fonctions en comparant vos réponses à celles de la question 1.

Console Python

```
>>> t = [5, 7, 9, 5, 3]
>>> e = 666
>>> insere(t, 3, e)
>>> insere(t, 0, e)
>>> insere(t, 5, e)
```

4. Pouvez-vous écrire une fonction `inserer2` qui utilise les fonctions `sous_tableau` et `concatener` qui insère l'élément `e` à l'indice `i` dans le tableau `t`. Vous testerez votre fonction. Selon vous, laquelle est la plus efficace ? La plus lisible ?

Devoir Maison.

Exercice 4.

1. Importer la variable `crypte` depuis le module `tp_informatique`. Quelle est la taille de ce tableau ? Quelle est le type des éléments qui le composent ?
2. Écrire le code Python qui remplace chaque élément de ce tableau par son code ASCII correspondant. Vous utiliserez pour cela la fonction `ord` (cf. internet pour la documentation).
3. (a) Écrire en Python la fonction `tourne` telle que pour tout `i` entier positif :

$$\texttt{tourne(i)} = \begin{cases} i+3, & \text{si } 65 \leq i \leq 87 \\ i-23, & \text{si } 87 < i \leq 90 \\ 33, & \text{sinon.} \end{cases}$$

   (b) Écrire le code Python qui remplace chaque élément du tabeau obtenu en question 2 par son image par `tourne`.
4. Écrire le code Python qui remplace chaque élément du tabeau obtenu à la question 3b par le caractère ASCII correspondant.Vous utiliserez pour cela la fonction `chr` (cf. internet pour la documentation)

   Faites afficher le tableau.
5. (Bonus) Écrire une fonction Python `decrypte` qui prend en entrée un tableau `t` chiffré avec la même méthode que le tableau `crypte` de la question 1 et qui renvoie en sortie le tableau correspondant au message déchiffré.

Codes.

Code Source - Recopier

```
def recopier(t):
    # Recopie le tableau t dans une nouvelle variable.
    n = len(t)
    copie = [0]*n
    for i in range(n):
        copie[i] = t[i]
    return copie
```

Console Python

```
>>> t1 = [5, 7, 8, 2]
>>> t2 = recopier(t1)
>>> print(t2)
[5, 7, 8, 2]
>>> t1[0] = 100
>>> print(t2)
[5, 7, 8, 2]
```

Code Source - Sous-tableaux

```
def sous_tableau(t, i, j):
    # Extrait le sous tableau constitué des éléments
    # d'indice compris entre i et j (exclu) du tableau
    # 0 < i < j <= len(t) : sinon IndexError
    n = j - i
    tprime = [0]*n
    for k in range(n):
        tprime[k] = t[k + i]
    return tprime
```

Console Python

```
>>> t = [4, 5, 8, 9, 6, 3, 56, 2, -4, 13]
>>> sous_tableau(t, 2, 5)
[8, 9, 6]
>>> sous_tableau(t, 0, 6)
[4, 5, 8, 9, 6, 3]
>>> sous_tableau(t, 3, 10)
[9, 6, 3, 56, 2, -4, 13]
```

Code Source - Concaténer

```
def concatener(t1, t2):
    # Renvoie le tableau constitué des éléments de la
    # liste t1 puis des éléments de la liste t2.
    # Marche aussi avec des listes vides (merci Python)
    n1 = len(t1)
    n2 = len(t2)
    t = [0]*(n1 + n2)
    for i in range(n1):
        t[i] = t1[i]
    for j in range(n2):
        t[n1 + j] = t2[j]
    return t
```

Console Python

```
>>> t1 = [1, 3, 5]
>>> t2 = [2, 4, 6]
>>> concatener(t1, t2)
[1, 3, 5, 2, 4, 6]
>>> concatener(t2, t1)
[2, 4, 6, 1, 3, 5]
>>> concatener(t1, t1)
[1, 3, 5, 1, 3, 5]
>>> concatener(t2, [])
[2, 4, 6]
```

Code Source - Insérer

```
def insere(t, i, e):
    # Insère l'élément e à l'indice i dans le tableau t.
    # i <= len(t) : sinon IndexError.
    # i = 0 : e inséré en début de tableau
    # i = len(t) : e inséré en fin de tableau
    n = len(t)
    tprime = [0]*(n+1)
    for k in range(i):
        tprime[k] = t[k]
    tprime[i] = e
    for k in range(i+1, n+1):
        tprime[k] = t[k-1]
    return tprime
```

Console Python

```
>>> t = [5, 7, 9, 5, 3]
>>> e = 666
>>> insere(t, 3, e)
[5, 7, 9, 666, 5, 3]
>>> insere(t, 0, e)
[666, 5, 7, 9, 5, 3]
>>> insere(t, 5, e)
[5, 7, 9, 5, 3, 666]
```