Nom :	Prénom :	Classe :

NSI 1re — Algorithmes élémentaires sur un tableau indexé

Objectifs :
- Savoir décrire le fonctionnement des algorithmes élémentaires sur un tableau indexé.
- Savoir écrire ces algorithmes en Python.

Introduction

Les algorithmes liés à un tableau indexé (*list*, en Python) abordés dans le cadre de la NSI forment, à grande échelle, un des rouages essentiels de systèmes qui influencent les larges volumes de données constituant l'information, l'économie et la vie quotidienne de milliards de citoyens. Il nous appartient donc de les concevoir avec rigueur et esprit critique.

Dans ce cours, nous aborderons la plupart des algorithmes « élémentaires » suivants :

- Accumulateur (sommes des valeurs)
- Comptage (nombre de valeurs)
- Moyenne
- Recherche d'une valeur
- Comptage du nombre d'occurrences
- Recherche d'un extremum (minimum et maximum)
- Recherche de l'indice d'une valeur
- Recherche de l'indice d'un extremum (minimum et maximum)
- Recherche du premier (ou dernier) indice d'une valeur

Notons que nous n'aborderons pas ici les algorithmes de tris ou de recherche dichotomique, qui font l'objet de cours spécifiques.

Quelques conseils

Tous les algorithmes présentés ici reposent sur l'utilisation de parcours. Parcours par valeur, parcours par indice. Et **pour chaque algorithme**, il convient de **se poser les questions suivantes** :

- Quelle(s) *variable(s) de référence* déclarer en amont ? Et quelles valeurs leurs attribuer *par défaut* ?
- Quel *type de parcours* est le plus adapté ? Et que désirons-nous *faire* au sein de ce parcours ?

Entraînez-vous en ne lisant que la **question associée** à chacun des algorithmes de ce cours.
Puis, pour chacun des algorithmes : 1) Essayez d'en décrire le **fonctionnement** et, seulement après, 2) Essayez de **l'implémenter** en Python.

Tableau utilisé dans ce cours

Pour tous les programmes Python présents dans ce cours, nous utiliserons cet exemple de tableau indexé (*list*) :

```
1  tableau = [23, 3, 5, 8, 5]
```

Accumulateur

Question associée : créer le programme permettant de calculer la somme (accumuler la valeur) de tous les éléments du tableau. *Attention : la fonction sum est interdite.*

1) Description du fonctionnement :

- On crée une variable d'accumulation valant 0 et qui va servir à accumuler la valeur de chaque élément du tableau.
- Puis, au sein d'un parcours par valeur, on ajoute la valeur de chaque élément à notre variable d'accumulation.
- À la fin, on obtient bien la somme de tous les éléments du tableau.

2) Une des implémentations possibles en Python :

```
1  accumulateur = 0  # Initialise la variable à 0
2  for v in tableau: # Lance un parcours par valeur
3      # Ajoute la valeur de l'élément actuel à celle de l'accumulateur :
4      accumulateur = accumulateur + v
```

On peut vérifier manuellement le résultat :

```
1  print(accumulateur) # Affiche « 44 » ; En effet 23+3+5+8+5 = 44
```

Comptage

Question associée : créer le programme permettant de compter le nombre total d'éléments présents dans le tableau. *Attention : la fonction len est interdite.*

1) Description du fonctionnement :

- On crée une variable compteur valant 0 et qui va servir à compter le nombre d'éléments du tableau.
- Puis, au sein d'un parcours par valeur, on incrémente la variable compteur de 1.
- À la fin, on obtient bien le nombre total d'éléments présents dans le tableau.

2) Une des implémentations possibles en Python :

```
1  compteur = 0      # Initialise le décompte à 0
2  for _ in tableau: # Lance un parcours par valeur
3      compteur = compteur + 1 # Incrémente la valeur du « compteur »
```

On peut vérifier manuellement le résultat :

```
1  print(compteur)   # Affiche « 5 » ; En effet le tableau contient bien 5 éléments.
```

Moyenne

Question associée : créer le programme permettant de calculer la moyenne des éléments présents dans le tableau. *Pour rappel, une moyenne se calcule en divisant la somme des valeurs par son nombre. Attention : les fonctions len et sum sont interdites.*

1) Description du fonctionnement :

- On crée une variable d'accumulation valant 0 et qui va servir à accumuler la valeur de chaque élément du tableau.
- On crée une variable compteur valant 0 et qui va servir à compter le nombre d'éléments du tableau.
- Puis, au sein d'un parcours par valeur, on ajoute la valeur de chaque élément à notre variable d'accumulation et on incrémente la variable compteur de 1.
- Après le parcours, on obtient la moyenne en divisant la valeur de la variable d'accumulation avec celle de la variable compteur.

2) Une des implémentations possibles en Python :

```
1  accumulateur = 0  # Initialise la variable à 0
2  compteur = 0      # Initialise le décompte à 0
3  for v in tableau: # Lance un parcours par valeur
4      # Ajoute la valeur de l'élément actuel à celle de l'accumulateur :
5      accumulateur = accumulateur + v
6      compteur = compteur + 1 # Incrémente la valeur du « compteur »
7  # On calcule la moyenne et on stocke le résultat
8  # dans une nouvelle variable « moyenne » :
9  moyenne = accumulateur / compteur
```

On peut vérifier manuellement le résultat :

```
1  print(moyenne) # Affiche « 8.8 » ; En effet 44 / 5 donne bien 8.8
```

Recherche d'une valeur

Question associée : créer le programme permettant de rechercher une valeur spécifique *de votre choix* présente dans le tableau. *Attention : les fonctions count et index sont interdites.*

1) Description du fonctionnement :

- On crée une variable de suivi qui va servir à savoir si une valeur recherchée est présente ou non dans le tableau. Par défaut, on estime que la valeur recherchée n'est pas présente dans le tableau, et on l'initialise à False.
- Puis, au sein d'un parcours par valeur, on compare la valeur de chaque élément à celle de la valeur recherchée. Et, si ces valeurs sont les mêmes, alors on change la valeur de la variable de suivi à True.
- À la fin, on sait si la valeur est présente (True) ou non (False) dans le tableau.

2) Une des implémentations possibles en Python :

```
# On prend comme exemple la recherche de la valeur « 8 »
trouve = False    # Par défaut, on estime que cette valeur
                  # n'a pas encore été trouvée dans le tableau
for v in tableau: # Lance un parcours par valeur
    if v == 8:    # Si la valeur actuelle (celle sur laquelle on itère)
                  # est celle recherchée ...
        trouve = True # Alors on change la valeur de la variable « trouve »
```

On peut vérifier manuellement le résultat :

```
print(trouve) # Affiche « True »
              # En effet, la valeur « 8 » est bien présente dans le tableau
```

Comptage du nombre d'occurrences

Question associée : créer le programme qui compte le nombre d'occurrences d'une valeur spécifique dans un tableau. *Attention : la fonction count est interdite*

1) Description du fonctionnement :

- On crée une variable compteur valant 0 et qui va servir à compter le nombre de fois qu'apparaît une valeur recherchée.
- Puis, au sein d'un parcours par valeur, on compare la valeur de chaque élément à celle de la valeur recherchée. Et, si ces valeurs sont les mêmes, alors on incrémente la valeur de la variable compteur de 1.
- À la fin, on obtient bien le nombre d'occurrences de la valeur recherchée dans le tableau.

2) Une des implémentations possibles en Python :

```
# On prend comme exemple la recherche de la valeur « 5 »
compteur = 0      # Initialise le décompte à 0
for v in tableau: # Lance un parcours par valeur
    if v == 5: # Si la valeur actuelle (celle sur laquelle on itère)
               # est celle recherchée ...
        compteur = compteur + 1 # Alors on incrémente la valeur du « compteur »
```

Vérification du résultat page suivante ...
On peut vérifier manuellement le résultat :

```
print(compteur) # Affiche « 2 »
                # En effet, la valeur « 5 » est bien présente
                # deux fois dans le tableau
```

Recherche d'un extremum

On entend par « recherche d'un extremum » le fait de chercher une valeur minimale ou une valeur maximale dans un tableau.

Recherche d'une valeur minimale

Question associée : créer le programme permettant de chercher la valeur minimale présente dans le tableau. *Attention : la fonction min est interdite.*

1) Description du fonctionnement :

- On crée une variable de suivi de valeur minimum qui prend comme valeur par défaut celle du premier élément du tableau.
- Puis, au sein d'un parcours par valeur, on compare la valeur de chaque élément à celle de la variable de suivi de valeur minimum. Et, si la valeur de l'élément actuel est plus basse que celle de la variable de suivi de valeur minimum, alors on met à jour la variable de suivi de valeur minimum en lui donnant celle de l'élément actuel.
- À la fin, on obtient bien la valeur minimale présente dans le tableau.

2) Une des implémentations possibles en Python :

```
v_min = tableau[0] # On prend comme valeur minimale initiale
                   # celle du premier élément du tableau
for v in tableau:  # Lance un parcours par valeur
    if v < v_min:  # Si la valeur actuelle (celle sur laquelle on itère) est
                   # plus basse que la valeur minimale enregistrée jusqu'ici ...
        v_min = v  # Alors on sauvegarde cette valeur actuelle
                   # comme nouvelle valeur minimale enregistrée
```

On peut vérifier manuellement le résultat :

```
print(v_min) # Affiche « 3 »
             # En effet, la valeur « 3 » est bien la plus basse du tableau.
```

Recherche d'une valeur maximale

Question associée : créer le programme permettant de chercher la valeur maximale présente dans le tableau. *Attention : la fonction max est interdite.*

1) Description du fonctionnement :

- On crée une variable de suivi de valeur maximum qui prend comme valeur par défaut celle du premier élément du tableau.
- Puis, au sein d'un parcours par valeur, on compare la valeur de chaque élément à celle de la variable de suivi de valeur maximum. Et, si la valeur de l'élément actuel est plus élevée que celle de la variable de suivi de valeur minimum, alors on met à jour la variable de suivi de valeur maximum en lui donnant celle de l'élément actuel.
- À la fin, on obtient bien la valeur maximale présente dans le tableau.

2) Une des implémentations possibles en Python :

```
1  v_max = tableau[0] # On prend comme valeur maximale initiale
2                     # celle du premier élément du tableau
3  for v in tableau:  # Lance un parcours par valeur
4      if v > v_max:  # Si la valeur actuelle (celle sur laquelle on itère) est
5                     # plus élevée que la valeur maximale enregistrée jusqu'ici ...
6          v_max = v  # Alors on sauvegarde cette valeur actuelle
7                     # comme nouvelle valeur maximale enregistrée
```

On peut vérifier manuellement le résultat :

```
1  print(v_max) # Affiche « 23 »
2               # En effet, la valeur « 23 » est bien la plus élevée du tableau.
```

Recherche de l'indice d'une valeur

Question associée : créer le programme permettant de rechercher l'indice de la valeur spécifique de votre choix présente dans le tableau. *On suppose la valeur spécifique comme étant unique. Attention : la fonction index est interdite.*

1) Description du fonctionnement :

- On crée une variable de suivi qui va servir à savoir si l'indice correspondant à une valeur recherchée existe ou non dans le tableau. Par défaut, on estime que cet indice n'existe pas, et on l'initialise à False.
- Puis, au sein d'un parcours par indice, on récupère la valeur de l'élément actuel et, si elle est identique à celle de la valeur recherchée, alors on en sauvegarde l'indice dans l'indice correspondant à une valeur recherchée.
- À la fin, si la valeur recherchée existe, on en obtient l'indice. Sinon, on obtient False.

2) Une des implémentations possibles en Python :

```
1  # On prend comme exemple la recherche de la valeur « 8 »
2  i_recherche = False # Par défaut, on estime que l'indice de cette valeur
3                      # n'a pas encore été trouvé dans le tableau
4  for i in range(len(tableau)): # Lance un parcours par indice
5      v = tableau[i]  # On récupère la valeur d'un élément
6                      # à partir de l'indice actuel (celui sur lequel on itère)
7      if v == 8:      # Si cette valeur actuelle est celle recherchée ...
8          i_recherche = i  # Alors on en sauvegarde l'indice
```

On peut vérifier manuellement le résultat :

```
1  print(i_recherche) # Affiche « 3 »
2                     # En effet la valeur « 8 » est bien présente
3                     # dans le tableau à l'indice « 3 ».
```

Recherche de l'indice d'un extremum

On entend par « recherche de l'indice d'un extremum » le fait de chercher une valeur minimale ou une valeur maximale dans un tableau , **et d'en trouver l'indice**.

Recherche de l'indice d'une valeur minimale

Question associée : créer le programme permettant de rechercher **l'indice** de la valeur minimale du tableau. *On suppose la valeur minimale comme étant unique. Attention : les fonctions min et index sont interdites.*

1) Description du fonctionnement :

- Au début du programme, on considère que le premier élément du tableau possède la valeur minimum.
- On crée une variable de suivi d'indice, initialisée à l'indice du premier élément.
- On crée une variable de suivi de valeur minimum, initialisée à la valeur du premier élément.
- Puis, au sein d'un parcours par indice, on récupère la valeur de l'élément actuel : et si elle est plus basse que celle de la variable de suivi de valeur minimum, alors on met à jour :
  - La variable de suivi d'indice en lui donnant l'indice de l'élément actuel.
  - La variable de suivi de valeur minimum en lui donnant la valeur de l'élément actuel.
- À la fin, on obtient bien l'indice de la valeur minimale présente dans le tableau.

2) Une des implémentations possibles en Python :

```
1  i_min = 0              # Le premier élément d'un tableau a pour indice 0
2  v_min = tableau[i_min] # On prend comme valeur minimale initiale
3                         # celle du premier élément du tableau
4  for i in range(len(tableau)): # Lance un parcours par indice
5      v = tableau[i]  # On récupère la valeur d'un élément
6                      # à partir de l'indice actuel (celui sur lequel on itère)
7      if v < v_min:   # Si cette valeur est plus basse
8                      # que la valeur minimale enregistrée jusqu'ici ...
9          v_min = v   # Alors on sauvegarde cette valeur
10                     # comme nouvelle valeur minimale enregistrée
11         i_min = i   # Ainsi que son indice correspondant
```

On peut vérifier manuellement le résultat :

```
1  print(i_min) # Affiche « 1 »
2               # En effet, l'élément d'indice « 1 » possède la valeur « 3 »
3               # qui est bien la plus basse du tableau.
```

Recherche de l'indice d'une valeur maximale

Question associée : créer le programme permettant de rechercher **l'indice** de la valeur maximale du tableau. *On suppose la valeur maximale comme étant unique. Attention : les fonctions max et index sont interdites.*

1) Description du fonctionnement :

- Au début du programme, on considère que le premier élément du tableau possède la valeur maximum.
- On crée une variable de suivi d'indice, initialisée à l'indice du premier élément.
- On crée une variable de suivi de valeur maximum, initialisée à la valeur du premier élément.
- Puis, au sein d'un parcours par indice, on récupère la valeur de l'élément actuel : et si elle est plus haute que celle de la variable de suivi de valeur maximum, alors on met à jour :
  - La variable de suivi d'indice en lui donnant l'indice de l'élément actuel.
  - La variable de suivi de valeur maximum en lui donnant la valeur de l'élément actuel.
- À la fin, on obtient bien l'indice de la valeur maximale présente dans le tableau.

2) Une des implémentations possibles en Python :

```
1   i_max = 0              # Le premier élément d'un tableau a pour indice 0
2   v_max = tableau[i_max] # On prend comme valeur maximale initiale
3                          # celle du premier élément du tableau
4   for i in range(len(tableau)): # Lance un parcours par indice
5       v = tableau[i]  # On récupère la valeur d'un élément
6                       # à partir de l'indice actuel (celui sur lequel on itère)
7       if v > v_max:   # Si cette valeur est plus élevée
8                       # que la valeur maximale enregistrée jusqu'ici ...
9           v_max = v   # Alors on sauvegarde cette valeur
10                      # comme nouvelle valeur maximale enregistrée
11          i_max = i   # Ainsi que son indice correspondant
```

On peut vérifier manuellement le résultat :

```
1   print(i_max) # Affiche « 0 »
2                # En effet, l'élément d'indice « 0 » possède la valeur « 23 »
3                # qui est bien la plus élevée du tableau.
```

Recherche du premier indice d'une valeur

Question associée : créer le programme permettant de rechercher le **premier indice** de la valeur spécifique de votre choix présente dans le tableau. *Attention : la fonction index est interdite.*

1) Description du fonctionnement :

- Par défaut on estime que l'indice correspondant à la valeur recherchée n'existe pas, et on l'initialise à False.
- Puis, au sein d'un parcours par indice, on récupère la valeur de l'élément actuel et, si elle est identique à celle de la valeur recherchée, et si l'indice correspondant à la valeur recherchée n'a pas encore été trouvé, alors on en sauvegarde l'indice dans la l'indice correspondant à la valeur recherchée.
- À la fin, si la valeur recherchée existe, on en obtient l'indice. Sinon, on obtient False.

2) Une des implémentations possibles en Python :

```
# On prend comme exemple la recherche de la valeur « 5 »
i_recherche = False # Par défaut, on estime que l'indice de cette valeur
                    # n'a pas encore été trouvé dans le tableau
for i in range(len(tableau)): # Lance un parcours par indice
    v = tableau[i]  # On récupère la valeur d'un élément
                    # à partir de l'indice actuel (celui sur lequel on itère)
    if v == 5:      # Si cette valeur actuelle est celle recherchée ...
        if i_recherche == False: # Et si un indice
                                 # n'a pas encore été sauvegardé ...
            i_recherche = i      # Alors on en sauvegarde l'indice
```

On peut vérifier manuellement le résultat :

```
print(i_recherche) # Affiche « 2 »
                   # En effet, la valeur « 5 » apparaît bien la première fois
                   # dans le tableau à l'indice « 2 ».
```

Recherche du dernier indice d'une valeur

Question associée : créer le programme permettant de rechercher le **dernier indice** de la valeur spécifique de votre choix présente dans le tableau. *Attention : la fonction index est interdite.*

1) Description du fonctionnement :

- Par défaut on estime que l'indice correspondant à la valeur recherchée n'existe pas, et on l'initialise à False.
- Puis, au sein d'un parcours par indice, on récupère la valeur de l'élément actuel et, si elle est identique à celle de la valeur recherchée, alors on en sauvegarde l'indice dans la l'indice correspondant à la valeur recherchée. Et si jamais la valeur apparaît plusieurs fois dans le tableau, la boucle écrasera la valeur de l'indice précédemment sauvegardé.
- À la fin, si la valeur recherchée existe, on en obtient le dernier indice où elle apparaît. Sinon, on obtient False.

2) Une des implémentations possibles en Python :

```
v_recherche = 5     # On prend comme exemple la recherche de la valeur « 5 »
i_recherche = False # Par défaut, on estime que l'indice de cette valeur
                    # n'a pas encore été trouvé dans le tableau
for i in range(len(tableau)): # Lance un parcours par indice
    v = tableau[i]  # On récupère la valeur d'un élément
                    # à partir de l'indice actuel (celui sur lequel on itère)
    if v == v_recherche: # Si cette valeur actuelle est celle recherchée ...
        i_recherche = i  # Alors on en sauvegarde l'indice
```

On peut vérifier manuellement le résultat :

```
print(i_recherche) # Affiche « 4 »
                   # En effet la valeur « 5 » apparaît bien
                   # dans le tableau, en dernière position, à l'indice « 4 ».
```