

1. TP 09 - Parcours séquentiel d'un tableau

TD n°9 : Parcours séquentiel d'un tableau	Thème 4 : Langages et Programmation
EXERCICES TYPE EPREUVE PRATIQUE en Terminale	EXERCICES

Lein Capytale

☰ Exercice 9.1 :

Enoncé

Ecrire une fonction qui prend en paramètre un tableau d'entiers non vide et qui renvoie la moyenne de ces entiers. La fonction est spécifiée ci-après et doit passer les assertions fournies.

```
def moyenne (tab):
    """
    moyenne(list) -> float
    Entrée : un tableau non vide d'entiers
    Sortie : nombre de type float
    Correspondant à la moyenne des valeurs présentes dans
    le tableau
    """
```

Solution

```
def moyenne (tab):
    """
    moyenne(list) -> float
    Entrée : un tableau non vide d'entiers
    Sortie : nombre de type float
    Correspondant à la moyenne des valeurs présentes dans
    le tableau
    """
    somme=0
    for elt in tab:
        somme+=elt
    moy=somme/len(tab)
    return moy
```

```
#Jeu de tests - A ne pas modifier
assert moyenne([1]) == 1
assert moyenne([1,2,3,4,5,6,7]) == 4
assert moyenne([1,2]) == 1.5
```

Exercice 9.2 :

Enoncé

Écrire une fonction `recherche` qui prend en paramètres `elt` un nombre et `tab` un tableau de nombres, et qui renvoie le tableau des indices de `elt` dans `tab` si `elt` est dans `tab` et le tableau vide `[]` sinon.

Exemples :

```
>>> recherche(3, [3, 2, 1, 3, 2, 1])
[0, 3]
>>> recherche(4, [1, 2, 3])
[]
```

Solution

```
def recherche(elt, tab):
    reponse=[]
    for indice in range(len(tab)):
        if elt==tab[indice]:
            reponse.append(indice)
    return reponse
```

```
recherche(3, [3, 2, 1, 3, 2, 1])
recherche(4, [1, 2, 3])
```

Exercice 9.3 :

Enoncé

On a relevé les valeurs moyennes annuelles des températures à Paris pour la période allant de 2013 à 2019. Les résultats ont été récupérés sous la forme de deux listes :

- l'une pour les températures, `t_moy = [14.9, 13.3, 13.1, 12.5, 13.0, 13.6, 13.7]`
- l'autre pour les années : `annees = [2013, 2014, 2015, 2016, 2017, 2018, 2019]`

Écrire la fonction `mini` qui prend en paramètres le tableau releve des relevés et le tableau date des dates et qui renvoie la plus petite valeur relevée au cours de la période et l'année correspondante.

```
Exemple : >>> mini(t_moy, annees)
12.5 , 2016
```

Solution

```
def mini(t1,t2):
    minimum=t1[0]
    indicemini=0
    for indice in range(len(t1)):
        if t1[indice]<minimum:
            minimum=t1[indice]
            indicemini=indice
    return minimum,t2[indicemini]

t_moy = [14.9, 13.3, 13.1, 12.5, 13.0, 13.6, 13.7]
annees = [2013, 2014, 2015, 2016, 2017, 2018, 2019]

mini(t_moy, annees)
```

(12.5, 2016)

Exercice 9.4 :

Enoncé

Écrire une fonction `maxi` qui prend en paramètre une liste `tab` de nombres entiers et renvoie un couple donnant le plus grand élément de cette liste, ainsi que l'indice de la première apparition de ce maximum dans la liste. Exemple :

```
>>> maxi([1,5,6,9,1,2,3,7,9,8])  
(9,3)
```

Solution

```
def maxi(tab):  
    maximum=tab[0]  
    indicemaxi=0  
    for indice in range(len(tab)):  
        if tab[indice]>maximum:  
            maximum=tab[indice]  
            indicemaxi=indice  
    return (maximum,indicemaxi)
```

```
maxi([1,5,6,9,1,2,3,7,9,8])  
(9, 3)
```

Exercice 9.5 :

Énoncé

Écrire une fonction `RechercheMinMax` qui prend en paramètre un tableau de nombres non triés `tab`, et qui renvoie la plus petite et la plus grande valeur du tableau sous la forme d'un dictionnaire à deux clés 'min' et 'max'. Les tableaux seront représentés sous forme de liste Python.

Exemples :

```
>>> tableau = [0, 1, 4, 2, -2, 9, 3, 1, 7, 1]
>>> resultat = rechercheMinMax(tableau)
>>> resultat
[ -2, 9]

>>> tableau = []
>>> resultat = rechercheMinMax(tableau)
>>> resultat
[ None, None]
```

Solution Version 1

```
def RechercheMinMax(tab):
    if len(tab)==0:
        return [None, None]
    else:
        minimum=tab[0]
        maximum=tab[0]

        for elt in tab:
            if elt>maximum:
                maximum=elt
            if elt<minimum:
                minimum=elt
        return [minimum, maximum]

tableau = [0, 1, 4, 2, -2, 9, 3, 1, 7, 1]
```

```
RechercheMinMax(tableau)
```

```
[-2, 9]
```

Solution Version 2

```
def maxi(tab):  
    if len(tab)==0:  
        return None  
    else:  
        maximum=tab[0]  
        for elt in tab:  
            if elt>maximum:  
                maximum=elt  
        return maximum  
  
def mini(tab):  
    if len(tab)==0:  
        return None  
    else:  
        minimum=tab[0]  
        for elt in tab:  
            if elt<minimum:  
                minimum=elt  
        return minimum  
  
def rechercheMinMaxbis(tab):  
    return [mini(tab),maxi(tab)]  
  
tableau = [0, 1, 4, 2, -2, 9, 3, 1, 7, 1]  
RechercheMinMax(tableau)  
  
[-2, 9]
```

Exercice 9.6 :

Énoncé

Écrire une fonction `RechercheMin` qui prend en paramètre un tableau de nombres non trié `tab`, et qui renvoie le minimum et l'indice de la première occurrence du minimum de ce tableau. Les tableaux seront représentés sous forme de liste Python.

Exemples :

```
>>> indice_du_min([5])
(5, 0)
>>> indice_du_min([2, 4, 1])
(1, 2)
>>> indice_du_min([5, 3, 2, 2, 4])
(2, 2)
```

Solution

```
def RechercheMin(tab):
    if len(tab)==0:
        return None
    else:
        minimum=tab[0]
        indicemini=0
        for indice in range(len(tab)):
            if tab[indice]<minimum:
                minimum=tab[indice]
                indicemini=indice

        return (minimum, indicemini)
```

```
RechercheMin([2, 4, 1])

(1, 2)
```


Exercice 9.7 :

Enoncé

Écrire une fonction `occurrence_max` prenant en paramètres une chaîne de caractères `chaine` et qui renvoie le caractère le plus fréquent de la chaîne. La chaîne ne contient que des lettres en minuscules sans accent. On pourra s'aider du tableau

```
alphabet=[ 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o',
```

et du tableau `occurrence` de 26 éléments où l'on mettra dans `occurrence[i]` le nombre d'apparitions de `alphabet[i]` dans la chaîne. Puis on calculera l'indice `k` d'un maximum du tableau `occurrence` et on affichera `alphabet[k]`. Exemple :

```
>>> ch='je suis en terminale et je passe le bac et je
souhaite poursuivre des etudes pour devenir expert en
informatique'
>>> occurrence_max(ch)
'e'
```

Solution

```
def occurrence_max(ch):
    alphabet=[ 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o',

    lettremax=''
    maxi=0
    for lettre in alphabet:
        compteur=0
        for caract in ch:
            if lettre==caract:
                compteur+=1
        if compteur>maxi:
            maxi=compteur
```

```
        lettremax=lettre  
    return lettremax,maxi
```

```
ch='je suis en terminale et je passe le bac et je  
souhaite poursuivre des etudes pour devenir expert en  
informatique'  
occurrence_max(ch)  
  
( 'e', 21)
```