Minimum & maximum

Exercice 1 Maximum

Programmer la fonction `maxi`, prenant en paramètre un tableau non vide `tab` (de type `list`) d'entiers et qui renvoie le plus grand entier du tableau.
<sl-icon name="exclamation-triangle"></sl-icon> Il ne faut pas utiliser la fonction `max()`

```
>>> maxi([8, 0, 12, 15, 13, 18, 10])
18

>>> maxi([-2, -3, -1, -4, -15])
-1
```

Correction

```
def maxi(tab):
    m = tab[0]
    for v in tab:
        if v > m:
            m = v
    return m
```

Exercice 2 Minimum

Programmer la fonction `mini`, prenant en paramètre un tableau non vide `tab` (de type `list`) d'entiers et qui renvoie le plus petit entier du tableau.
<sl-icon name="exclamation-triangle"></sl-icon> Il ne faut pas utiliser la fonction `min()`

```
>>> mini([8, 0, 12, 15, 13, 18, 10])
0

>>> mini([2, 3, -1, 4, 15])
-1
```

Correction

```
def mini(tab):
    m = tab[0]
    for v in tab:
        if v < m:
            m = v
    return m
```

Exercice 3 Somme

Programmer la fonction **somme**, prenant en paramètre un tableau non vide **tab** (de type **list**) d'entiers et qui renvoie la somme des nombres de la liste.

```
>>> somme([8, 10, 12, 15, 13, 18, 10])
86
>>> somme([5,15])
20
```

Correction

```
def somme(tab):
    s = 0
    for v in tab:
        s = s + v
    return s
```

Exercice 4 Moyenne

Programmer la fonction **moyenne**, prenant en paramètre un tableau non vide **tab** (de type **list**) d'entiers et qui renvoie la moyenne des éléments du tableau.

```
>>> moyenne([8, 10, 12, 15, 13, 18, 10])
12.28571428571429

>>> moyenne([5,15])
10
```

Correction

```
def moyenne(tab):
    s = 0
    for v in tab:
        s = s + v
    return s / len(tab)
```

Exercice 5 Moyenne 2

Dans cet exercice, les nombres sont des entiers ou des flottants.
Écrire une fonction **moyenne2** renvoyant la moyenne pondérée de deux listes non vide, passée en paramètre, la première nommée **notes** contient les notes des devoirs, la seconde nommée **coefs** contient les coefficients des devoirs.

- si la somme des coefficients est nulle, la fonction renvoie `None`,
- sinon la fonction renvoie, sous forme de flottant, la moyenne des valeurs affectées de leur coefficient.

```
>>> moyenne2([8,12, 13.5, 5],[2, 0 ,1,0.5])
9.142857142857142
>>> moyenne2([3, 5],[0, 0])
None
```

Dans le premier exemple la moyenne est calculée par la formule :

$$\frac{8\times 2+12\times 0+13,5\times 1+5\times 0,5}{2+0+1+0,5}$$

Correction

```
def moyenne2(notes, coeffs):
    somme_note = 0
    somme_coeff = 0
    for i in tab:
        somme_note = somme_note + notes[i] *coeffs[i]
        somme_coeff = somme_coeff + coeffs[i]
    if somme_coeff == 0:
        return None
    else:
        return somme_note / somme_coeff
```

Algorithme de recherche

Exercice 6 Recherche

Programmer la fonction `recherche`, prenant en paramètre un tableau non vide `tab` (de type `list`) d'entiers et un entier `elmt`, et qui renvoie l'indice de la premiere occurrence de l'élément cherché.
Si l'élément n'est pas présent, la fonction renvoie la longueur du tableau.

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

Correction

```
def recherche(tab, elmt):
    for i in range(tab):
        if tab[i] == elmt:
            return i
    return len(tab)
```

Exercice 7 Recherche 2

Programmer la fonction **recherche2**, prenant en paramètre un tableau non vide **tab** (de type **list**) d'entiers et un entier **n**, et qui renvoie l'indice de la dernière occurrence de l'élément cherché.
Si l'élément n'est pas présent, la fonction renvoie la longueur du tableau.

```
>>> recherche2([5, 3], 1)
2
>>> recherche2([2, 4], 2)
0
>>> recherche2([2, 3, 5, 2, 1],2)
3
```

Correction

```
def recherche2(tab, elmt):
    indice = len(tab)
    for i in range(tab):
        if tab[i] == elmt:
            indice = i
    return indice
```

Exercice 8 Recherche 2

Programmer la fonction **recherche3**, prenant en paramètre un tableau non vide **tab** (de type list) d'entiers et un entier **n**, et qui renvoie une liste contenant tous les indices des occurrences de l'élément cherché.
Si l'élément n'est pas présent, la fonction renvoie une liste vide.

```
>>> recherche3([5, 3], 1)
[]
>>> recherche3([2, 4], 2)
[0]
>>> recherche3([2, 3, 2, 5, 2, 4], 2)
[0, 2, 4]
```

Correction

```python
def recherche3(tab, elmt):
    indices = []
    for i in range(tab):
        if tab[i] == elmt:
            indices.append(i)
    return indices
```