Révisions sur les boucles

Exercice 1
On considère la fonction `mystere` ci-dessous :

```
def mystere(n):
        s = 0
        for i in range(n):
                s = s+i
        return s
```

1. Donner la valeur de `s` à chaque itération de la boucle lors de l'exécution de `mystere(5)`.
2. Que calcule cette fonction ? Donner une formule de `s` en fonction de `n`.

Exercice 2
On considère la fonction factorielle $n!$ définie par $n! = 1 \times 2 \times \cdots \times n$ pour n entier positif.
Écrire une fonction `fact(n)` qui calcule $n!$.

Exercice 3
Compléter fonction `recherche` qui prend en paramètres `caractere`, un caractère, et `mot`, une chaîne de caractères, et qui renvoie le nombre d'occurrences de `caractere` dans `mot`, c'est-à-dire le nombre de fois où `caractere` apparaît dans `mot`. **Exemples :**

```
>>> recherche('e', "sciences")
2
>>> recherche('i',"mississippi")
4
>>> recherche('a',"mississippi")
0
```

```
def recherche(caractere, mot):
        compteur = 0
        for lettre in mot:
                if lettre == ...:
                        compteur = ...
        return compteur
```

Exercice 4
Programmer la fonction `moyenne` prenant en paramètre un tableau d'entiers `tab` (type `list`) qui renvoie la moyenne de ses éléments si le tableau est non vide et affiche `'erreur'` si le tableau est vide. **Exemples :**

```
>>> moyenne([5, 3, 8])
5.333333333333333
>>> moyenne([1,2,3,4,5,6,7,8,9,10])
5.5
>>> moyenne([])
'erreur'
```

Exercice 5
Programmer la fonction `recherche`, prenant en paramètre un tableau non vide `tab` (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 `-1`.
Exemples :

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

Exercice 6
Les résultats aux évaluations d'un élève sont regroupés dans une liste composée de couples (`note`,`coefficient`) avec :

- `note` un nombre de type flottant (`float`) compris entre 0 et 20 ;
- `coefficient` un nombre entier positif.

Écrire une fonction `moyenne_ponderee` qui renvoie la moyenne pondérée de cette liste donnée en paramètre.
Par exemple, l'expression `moyenne_ponderee([(15,2),(9,1),(12,3)])` devra renvoyer le résultat du calcul suivant :

$$\frac{2 \times 15 + 1 \times 9 + 3 \times 12}{2 + 1 + 3} = 12{,}5$$

Exercice 7
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)
```

Exercice 8
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])
[]
```

Exercice 9
Écrire une fonction `recherche_dichotomique` qui prend en paramètres un tableau `tab` de nombres entiers triés par ordre croissant et un nombre entier `n`, et qui effectue une recherche **dichotomique** du nombre entier `n` dans le tableau non vide `tab`.
Cette fonction doit renvoyer un indice correspondant au nombre cherché s'il est dans le tableau, -1 sinon.
Exemples :

```
>>> recherche_dichotomique([2, 3, 4, 5, 6], 5)
3
>>> recherche_dichotomique([2, 3, 4, 6, 7], 5)
-1
```

On rappelle l'algorithme de dichotomie vue en cours en première :

```
fonction recherche_par_dichotomie(liste,x):
    gauche ← 0
    droite ← longueur(liste)
    Tant que droite - gauche > 1
        centre = (gauche + droite) // 2
        Si x < liste[centre]
            droite = centre
        Sinon
            gauche = centre
    Si x = liste[gauche]:
        retourner gauche
```