Exercices de révision - parcours de listes

Partie 1

Question 1 - Écrire une fonction `afficher_liste(l)` qui affiche chaque élément d'une liste sur une ligne.

```
afficher_liste([2, 5, 8, 12])
# Résultat attendu :
# 2
# 5
# 8
# 12
```

Question 2 - Écrire une fonction `somme_moyenne(l)` qui retourne un tuple `(somme, moyenne)` des éléments d'une liste d'entiers.

```
print(somme_moyenne([3, 7, 1, 9]))
# Résultat attendu : (20, 5.0)
```

Question 3 - Écrire une fonction `idmax_idmin(l)` qui retourne `(indice_du_max, indice_du_min)` d'une liste d'entiers.

```
print(max_min([5, 12, 7, 9]))
# Résultat attendu : (1, 0)
```

Question 4 - Écrire une fonction `trouver(l, valeur)` qui retourne l'indice de valeur dans la liste ou -1 si elle n'est pas présente.

```
print(trouver([3, 7, 5, 9], 7))
# Résultat attendu : 1
print(trouver([3, 7, 5, 9], 4))
# Résultat attendu : -1
```

Question 5 - Écrire une fonction `carres(l)` qui retourne une nouvelle liste contenant les carrés des éléments.

```
print(carres([1, 2, 3, 4]))
# Résultat attendu : [1, 4, 9, 16]
```

Question 6 - Écrire une fonction `filtrer_pairs(l)` qui retourne une nouvelle liste ne contenant que les éléments pairs.

```
print(filtrer_pairs([3, 6, 9, 12]))
# Résultat attendu : [6, 12]
```

Question 7 - Écrire une fonction `inverser(l)` qui retourne la liste inversée.

```
print(inverser([1, 2, 3, 4]))
# Résultat attendu : [4, 3, 2, 1]
```

Partie 2

>>> En réutilisant les fonctions des questions 1 à 7, codez les fonctions suivantes en 3 lignes maximum. <<<

Question 8 - Écrire une fonction `max_min(l)` qui retourne `(maximum, minimum)` d'une liste d'entiers.

```
print(max_min([5, 12, 7, 9]))
# Résultat attendu : (12, 5)
```

Question 9 - Écrire une fonction `compter_pairs(l)` qui retourne le nombre de nombres pairs dans d'une liste d'entiers.

```
print(compter_pairs([4, 7, 10, 3, 8]))
# Résultat attendu : 3
```

Question 10 - Écrire une fonction `pairs_inverses(l)` qui retourne les nombres pairs de la liste dans l'ordre inverse.

```
print(pairs_inverses([2, 5, 6, 9, 8]))
# Résultat attendu : [8, 6, 2]
```

Question 11 - Écrire une fonction `carres_pairs(l)` qui retourne les carrés des nombres pairs.

```
print(carres_pairs([1, 2, 3, 4, 5, 6]))
# Résultat attendu : [4, 16, 36]
```

Question 12 - Écrire une fonction `somme_pairs(l)` qui retourne la somme des nombres pairs.

```
print(somme_pairs([3, 4, 5, 6, 7]))
# Résultat attendu : 10
```

Partie 3 :

>>> Ne pas utiliser de boucles `for`, tous faire avec des boucles `while`. <<<

Question 13 - Écrire une fonction `inverser_while(l)`.

```
print(inverser_while([1, 2, 3, 4]))
# Résultat attendu : [4, 3, 2, 1]
```

Question 14 - Écrire `est_palindrome_while(l)` qui retourne `True` si la liste est un palindrome, sinon `False`.

```
print(est_palindrome_while([1, 2, 3, 2, 1]))
# Résultat attendu : True
print(est_palindrome_while([1, 2, 3]))
# Résultat attendu : False
```

Correction – Partie 1

Question 1

```
def afficher_liste(l):
    for element in l:
        print(element)
```

Question 2

```
def somme_moyenne(l):
    somme = 0
    for x in l:
        somme += x
    moyenne = somme / len(l)
    return somme, moyenne
```

Question 3

```
def idmax_idmin(l):
    idmax = 0
    idmin = 0
    for i in range(len(l)):
        if l[i] > l[idmax]:
            idmax = i
        if l[i] < l[idmin]:
            idmin = i
    return idmax, idmin
```

Question 4

```
def trouver(l, valeur):
    for i in range(len(l)):
        if l[i] == valeur:
            return i
    return -1
```

Question 5

```
def carres(l):
    resultat = []
    for x in l:
        resultat.append(x**2)
    return resultat
```

Question 6

```
def filtrer_pairs(l):
    resultat = []
    for x in l:
        if x % 2 == 0:
            resultat.append(x)
    return  resultat
```

Question 7

```
def inverser(l):
    result = []
    for i in range(len(l)-1, -1, -1):
        result.append(l[i])
    return result
```

Correction – Partie 2

Question 8

```
def max_min(l) :
    ids = idmax_idmin(l)
    return ( l[ids[0]], l[ids[1]])
```

Question 9

```
def compter_pairs(l) :
    return len(filtrer_pairs(l))
```

Question 10

```
def pairs_inverses(l) :
    return inverser(filtrer_pairs(l))
```

Question 11

```
def carres_pairs(l) :
    return carres(filtrer_pairs(l))
```

Question 12

```
def somme_pairs(l) :
    return somme_moyenne(filtrer_pairs(l))[0]
```

Correction – Partie 3

Question 13

```
def inverser_while(l):
    result = []
    i = len(l) - 1
    while i >= 0:
        result.append(l[i])
        i -= 1              # pareil que de faire i = i - 1
    return result
```

Question 14

```
def est_palindrome_while(l) :
    i = 0
    while i < (len(l)//2):
        if l[i] != l[-i-1]:
            return False
        i += 1              # pareil que de faire i = i + 1
    return True
```