

## EXERCICES SUITE AU COURS SUR LES SÉQUENCES LIST ET STR

### COMPREHENSION DE LISTE

Q1- Quelle instruction permet d'affecter la liste `[0,1,4,9,16]` à la variable `tableau` ?

Réponses :

A- `tableau = [ i**2 for i in range(4) ]`

B- `tableau = [ i**2 for i in range(5) ]`

C- `tableau = [ i**2 for i in range(16) ]`

D- `tableau = [ i**2 for i in range(17) ]`

Q2- Que vaut l'expression `[2 * k for k in range(5)]` ?

Réponses :

|                              |                               |
|------------------------------|-------------------------------|
| A- <code>[0,2,4,6,8]</code>  | C- <code>[1,2,4,8,16]</code>  |
| B- <code>[2,4,6,8,10]</code> | D- <code>[2,4,8,16,32]</code> |

### Mot-clé `in` et test d'appartenance

Q1- Quelle instruction permet de tester si la liste `[0,2,4,6,8]` contient le nombre 4 ?

Q2- Que retourne l'instruction: `25 in [ i**2 for i in range(16) ]` ?

### TABLEAUX 2 DIMENSIONS

Q1- On définit : `matrice = [[1,2,3], [4,5,6], [7,8,9], [10,11,12]]`

Quelle est la valeur de `matrice[1][2]` ?

Réponses :

|      |      |
|------|------|
| A- 2 | C- 6 |
| B- 4 | D- 8 |

Q2- `n` définit la liste `L` ainsi : `L = [ [1], [1,2], [1,2,3] ]`

Des égalités suivantes, une seule est fausse. Laquelle ?

Réponses :

|                                |                                |
|--------------------------------|--------------------------------|
| A- <code>len(L[0]) == 1</code> | C- <code>len(L[2]) == 3</code> |
| B- <code>len(L) == 6</code>    | D- <code>L[2][2] == 3</code>   |

Q3- On définit ainsi une liste `M` :

`M = [['A','B','C','D'], ['E','F','G','H'], ['I','J','K','L']]`

Quelle expression vaut la chaîne de caractères 'H' ?

Réponses :

A- `M[1][3]`

B- `M[3][1]`

C- `M(7)`

D- `M(8)`

---

## Algorithmes

```
def occurrences(mot, caractere):  
    s = 0  
    for c in mot :  
        if c == caractere:  
            s = s + 1  
    return s
```

Q1- Adapter le programme de recherche d'occurrences pour trouver le nombre de voyelles dans un mot. Appeler cette fonction **nombre\_de\_voyelles**

Q2- Adapter le programme de recherche d'occurrences pour compter les nombres pairs dans une liste numérique. Appeler cette fonction **nombres\_pairs**

Q3- Compléter la fonction qui recherche un caractère précis dans une liste:

```
def recherche(L, caractere):  
    trouve = False  
    for c in L:  
        if ...  
            trouve = ...  
    return ...
```

```
> recherche(['a','b','c','d'], 'b')  
True  
> recherche(['a','b','c','d'], 'e')  
False
```

Q4- Adapter ce script pour écrire la fonction **recherche\_car\_speciaux**. Cette fonction retournera True si la liste proposée contient un caractère parmi (@, &, \_, -, ?, /, +)

```
def recherche_car_speciaux(L):  
    trouve = ...  
    for c in L:  
        if ...  
            trouve = ...  
    return ...
```

```
> recherche_car_speciaux(['@','b','c','d'])  
True  
> recherche_car_speciaux(['a','b','c','d'])  
False
```