

# **BACCALAURÉAT**

**SESSION 2023**

---

**Épreuve de l'enseignement de spécialité**

## **NUMÉRIQUE et SCIENCES INFORMATIQUES**

**Partie pratique**

**Classe Terminale de la voie générale**

---

**Sujet n°43**

---

**DURÉE DE L'ÉPREUVE : 1 heure**

**Le sujet comporte 3 pages numérotées de 1 / 3 à 3 / 3  
Dès que le sujet vous est remis, assurez-vous qu'il est complet.**

*Le candidat doit traiter les 2 exercices.*

## EXERCICE 1 (4 points)

Écrire une fonction `ecriture_binaire_entier_positif` qui prend en paramètre un entier positif  $n$  et renvoie une liste d'entiers correspondant à l'écriture binaire de  $n$ .

Ne pas oublier d'ajouter au corps de la fonction une documentation et une ou plusieurs assertions pour vérifier les pré-conditions.

Exemples :

```
>>> ecriture_binaire_entier_positif(0)
[0]
>>> ecriture_binaire_entier_positif(2)
[1, 0]
>>> ecriture_binaire_entier_positif(105)
[1, 1, 0, 1, 0, 0, 1]
```

Aide :

- l'opérateur `//` donne le quotient de la division euclidienne :  $5//2$  donne  $2$  ;
- l'opérateur `%` donne le reste de la division euclidienne :  $5\%2$  donne  $1$  ;
- `append` est une méthode qui ajoute un élément à une liste existante :  
Soit  $T=[5, 2, 4]$ , alors `T.append(10)` ajoute 10 à la liste  $T$ . Ainsi,  $T$  devient  $[5, 2, 4, 10]$ .
- `reverse` est une méthode qui renverse les éléments d'une liste.  
Soit  $T=[5, 2, 4, 10]$ . Après `T.reverse()`, la liste devient  $[10, 4, 2, 5]$ .

On remarquera qu'on récupère la représentation binaire d'un entier  $n$  en partant de la gauche en appliquant successivement les instructions :

```
b = n%2
n = n//2
```

répétées autant que nécessaire.

## EXERCICE 2 (4 points)

La fonction `tri_bulles` prend en paramètre une liste `T` d'entiers non triés et renvoie la liste triée par ordre croissant.

Le tri à bulles est un tri en place qui commence par placer le plus grand élément en dernière position en parcourant la liste de gauche à droite et en échangeant au passage les éléments voisins mal ordonnés (si la valeur de l'élément d'indice  $i$  a une valeur strictement supérieure à celle de l'indice  $i + 1$ , ils sont échangés). Le tri place ensuite en avant-dernière position le plus grand élément de la liste privée de son dernier élément en procédant encore à des échanges d'éléments voisins. Ce principe est répété jusqu'à placer le minimum en première position.

Exemple : pour trier la liste `[7, 9, 4, 3]` :

- première étape : 7 et 9 ne sont pas échangés, puis 9 et 4 sont échangés, puis 9 et 3 sont échangés, la liste est alors `[7, 4, 3, 9]`
- deuxième étape : 7 et 4 sont échangés, puis 7 et 3 sont échangés, la liste est alors `[4, 3, 7, 9]`
- troisième étape : 4 et 3 sont échangés, la liste est alors `[3, 4, 7, 9]`

Compléter le code Python ci-dessous qui implémente la fonction `tri_bulles`.

```
def tri_bulles(T):
    '''
    Renvoie le tableau T trié par ordre croissant
    '''
    n = len(T)
    for i in range(..., ..., -1):
        for j in range(i):
            if T[j] > T[...]:
                ... = T[j]
                T[j] = T[...]
                T[j+1] = temp
    return T
```

Exemples :

```
>>> tri_bulles([])
[]
>>> tri_bulles([7])
[7]
>>> tri_bulles([9, 3, 7, 2, 3, 1, 6])
[1, 2, 3, 3, 6, 7, 9]
>>> tri_bulles([9, 7, 4, 3])
[3, 4, 7, 9]
```