BACCALAURÉAT

SESSION 2025

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°37

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 (10 points)

On considère dans cet exercice une représentation binaire d'un entier non signé en tant que tableau de booléens.

Si

```
tab = [True, False, True, False, False, True, True]
```

est un tel tableau, alors l'entier qu'il représente est $2^6 + 2^4 + 2^1 + 2^0 = 83$. Cette représentation consistant à placer en premier le booléen indiquant la puissance la plus élevée de 2 est dite *big-endian* ou grand-boutiste.

Écrire une fonction `gb_vers_entier` qui prend en paramètre un tel tableau et renvoie l'entier qu'il représente.

Exemple :

```
>>> gb_vers_entier([])
0
>>> gb_vers_entier([True])
1
>>> gb_vers_entier([True, False, True,
        False, False, True, True])
83
>>> gb_vers_entier([True, False, False, False,
        False, False, True, False])
130
```

EXERCICE 2 (10 points)

La fonction `tri_insertion` suivante prend en argument un tableau `tab` (type `list`) et trie ce tableau en utilisant la méthode du tri par insertion. Compléter cette fonction pour qu'elle réponde à la spécification demandée.

On rappelle le principe du tri par insertion : on considère les éléments à trier un par un, le premier élément constituant, à lui tout seul, un tableau trié de longueur 1. On range ensuite le second élément pour constituer un tableau trié de longueur 2, puis on range le troisième élément pour avoir un tableau trié de longueur 3 et ainsi de suite…

A chaque étape, le premier élément du sous-tableau non trié est placé dans le sous-tableau des éléments déjà triés de sorte que ce sous-tableau demeure trié.

Le principe du tri par insertion est donc d'insérer à la n-ième itération, le n-ième élément à la bonne place.

```
def tri_insertion(tab):
    '''Trie le tableau tab par ordre croissant
    en appliquant l'algorithme de tri par insertion'''
    n = len(tab)
    for i in range(1, n):
        valeur_insertion = ...
        # la variable j sert à déterminer
        # où placer la valeur à ranger
        j = ...
        # tant qu'on n'a pas trouvé la place de l'élément à
        # insérer on décale les valeurs du tableau vers la droite
        while j > ... and valeur_insertion < tab[...]:
            tab[j] = tab[j-1]
            j = ...
        tab[j] = ...
```

Exemple :

```
>>> tab = [98, 12, 104, 23, 131, 9]
>>> tri_insertion(tab)
>>> tab
[9, 12, 23, 98, 104, 131]
```