BACCALAURÉAT

SESSION 2025

Épreuve de l'enseignement de spécialité

NUMÉRIQUE et SCIENCES INFORMATIQUES

Partie pratique

Classe Terminale de la voie générale

Sujet n°17

DURÉE DE L'ÉPREUVE : 1 heure

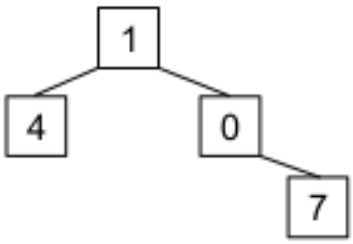
Le sujet comporte 3 pages numérotées de 1 / 3 à 3 / 3
Dès que le sujet vous est remis, assurez-vous qu'il est complet.

Le candidat doit traiter les 2 exercices.

EXERCICE 1 (10 points)

Un arbre binaire est soit vide, représenté en Python par la valeur None, soit un nœud, contenant une étiquette et deux sous-arbres gauche et droit et représenté par une instance de la classe Noeud donnée ci-dessous.

```
class Noeud:
    def __init__(self, etiquette, gauche, droit):
        self.v = etiquette
        self.gauche = gauche
        self.droit = droit
```



L'arbre ci-dessus sera donc implémenté de la manière suivante :

```
a = Noeud(1, Noeud(4, None, None),
             Noeud(0, None,
                      Noeud(7, None, None)))
```

Écrire une fonction récursive `taille` prenant en paramètre un arbre a et qui renvoie la taille de l'arbre que cette instance implémente.

Écrire de même une fonction récursive `hauteur` prenant en paramètre un arbre a et qui renvoie la hauteur de l'arbre que cette instance implémente.

On considère que la hauteur d'un arbre vide est -1 et la taille d'un arbre vide est 0.

Exemples :

```
>>> hauteur(a)
2
>>> taille(a)
4
>>> hauteur(None)
-1
>>> taille(None)
0
>>> hauteur(Noeud(1, None, None))
0
>>> taille(Noeud(1, None, None))
1
```

EXERCICE 2 (10 points)

On rappelle que les tableaux sont représentés par des listes en Python du type `list`.

Le but de cet exercice est d'écrire une fonction `ajoute` qui prend en paramètres trois arguments `indice`, `element` et `tab` et renvoie un tableau `tab_ins` dans lequel les éléments sont ceux du tableau `tab` avec, en plus, l'élément `element` à l'indice `indice`.

On considère que les variables `indice` et `element` sont des entiers positifs et que les éléments de `tab` sont également des entiers.

En réalisant cette insertion, Les éléments du tableau `tab` dont les indices sont supérieurs ou égaux à `indice` apparaissent décalés vers la droite dans le tableau `tab_ins`.

Si `indice` est égal au nombre d'éléments du tableau `tab`, l'élément `element` est ajouté dans `tab_ins` après tous les éléments du tableau `tab`.

Exemples :

```
>>> ajoute(1, 4, [7, 8, 9])
[7, 4, 8, 9]
>>> ajoute(3, 4, [7, 8, 9])
[7, 8, 9, 4]
>>> ajoute(0, 4, [7, 8, 9])
[4, 7, 8, 9]
```

Compléter et tester le code ci-dessous :

```
def ajoute(indice, element, tab):
    '''Renvoie un nouveau tableau obtenu en insérant
    element à l'indice indice dans le tableau tab.'''
    nbre_elts = len(tab)
    tab_ins = [0] * (nbre_elts + 1)
    for i in range(indice):
        tab_ins[i] = ...
    tab_ins[...] = ...
    for i in range(indice + 1, nbre_elts + 1):
        tab_ins[i] = ...
    return tab_ins
```