Index des sujets 2023

Corrigé sujet 12 - Année : 2023

Sujet 12 - 2023

Exercice 1

```python
class ABR:
    def __init__(self, g0, v0, d0):
        self.gauche = g0
        self.cle = v0
        self.droit = d0

    def __repr__(self):
        if self is None:
            return ''
        else:
            return '(' + (self.gauche).__repr__() + ',' + str(self.cle) +
',' +(self.droit).__repr__() + ')'

n0 = ABR(None, 0, None)
n3 = ABR(None, 3, None)
n2 = ABR(None, 2, n3)
abr1 = ABR(n0, 1, n2)


def ajoute(cle,a):
    if a == None:
        return ABR(None,cle,None)
    elif a.cle>cle:
        sag = ajoute(cle,a.gauche)
        return ABR(sag,a.cle,a.droit)
    elif a.cle<cle:
        sad = ajoute(cle,a.droit)
        return ABR(a.gauche,a.cle,sad)
    else:
        return a
```

Attention

- Le sujet est difficile et aborde diverses notions du programme (arbre binaire de recherche, récursivité, programmation objet).
- De façon très inhabituelle pour une exercice 1, une portion de code est fournie, c'est le code de la classe `ABR`, avec la méthode d'affichage. A noter que la `if` de cette méthode n'est jamais executé, en effet `None` n'est pas une instance de la classe `ABR`, par contre `None` a déjà une méthode `__repr__` (ce qui arrête la récursivité)
- La fonction a écrire `ajoute` devrait être une méthode de classe `ABR` et pas une fonction externe, elle devrait donc modifier l'arbre donné en paramètre et pas en créer un nouveau.

Exercice 2

```
1  def empaqueter(liste_masses, c):
2      n = len(liste_masses)
3      nb_boites = 0
4      boites = [0]*n
5      for masse in liste_masses : # 1
6          i=0
7          while i <= nb_boites and boites[i] + masse > c: # 2
8                  i = i + 1
9          if i == nb_boites + 1: # 3
10                 nb_boites = nb_boites + 1
11         boites[i] = boites[i] + masse
12     return nb_boites + 1 # 4
```

1. Parcours de la liste des masses
2. Tant qu'on a pas atteint la première boite vide et que la masse ne rentre pas on avance dans la liste de boites.
3. La masse s'insère dans une boite vide, donc le nombre de boite utilisée augmente
4. Les boites sont comptées à partir de zéro, donc on ajoute un au nombre

Attention

Dans cette correction la fonction renvoie 1 pour si la liste de masse est vide, si on veut que la fonction renvoie `0`, on doit soit introduire un `if` dans le `return` (hors programme), soit modifier le code donné.