Activité : Arbre Binaire de Recherche (ABR)

Introduction

Un arbre binaire de recherche (ABR) est un arbre binaire dont les nœuds contiennent des valeurs que l'on peut comparer comme des nombres entiers ou les lettres de l'alphabet.

Un ABR respecte les règles suivantes :

1) Pour un nœud donné :

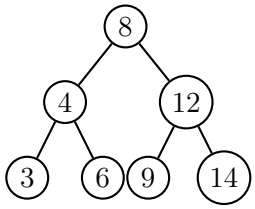
- Toutes les valeurs situées dans l'arbre fils gauche sont inférieures à la valeur du nœud ;
- Toutes les valeurs situées dans l'arbre fils droit sont supérieures à la valeur du nœud ;

2) Toutes les valeurs d'un ABR sont distinctes, pas de doublon.

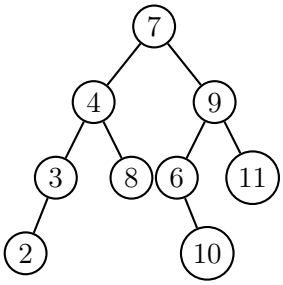
Partie 1

1) Les arbres ci-dessous sont-ils des arbres binaires de recherche (ABR) ? Justifier votre réponse.

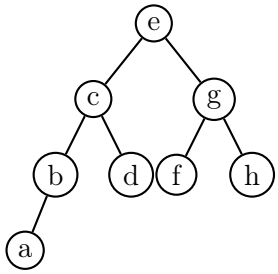
-a-



-b-



-c-



2) Un arbre binaire de recherche contient les nombres entiers de 1 à 10. Représenter cet arbre binaire sachant que la racine est de hauteur 1 et que :

a) la racine de l'arbre est le nombre 5 et sa hauteur est minimale.

b) la racine de l'arbre est le nombre 3 et sa hauteur est 4.

c) la racine de l'arbre est le nombre 9 et la hauteur est maximale.

Partie 2

On reprend l'implémentation des arbres binaires vus plus tôt dans l'année que l'on rappelle. On dispose de deux classes pour implémenter les ABR : la classe Arbre et la classe Nœud .

❑ La classe Arbre a un attribut racine.

- L'attribut racine initialisé à **None** définit un arbre vide ;
- Pour un arbre non vide, la racine prend la valeur d'un Nœud .

Nous avons les méthodes des arbres binaires :

- La méthode est_vide() renvoie un booléen, True pour un arbre vide et False pour un arbre non vide.
- La méthode fils_gauche() qui renvoie l'arbre gauche d'un Nœud .
- La méthode fils_droit() qui renvoie l'arbre droit d'un Nœud .

❑ La classe Nœud contient 3 attributs : valeur, gauche et droit.

- L'attribut valeur contient la valeur d'un nœud de l'arbre ;
- Les attributs gauche et droit représentent des arbres binaires qui sont des Nœud éventuellement vides.

Elle ne contient pas de méthode particulière.

1) Soit a un ABR tel que a=Nœud (5, Nœud (3, None, None), Nœud (6, None, None)).

a) Représenter schématiquement l'ABR a.

b) On ajoute dans cet ABR le nœud de valeur 4. Compléter la description récursive de a.

2) On définit avec l'implémentation donnée ci-dessus notre ABR en python.

a) Quelle instruction Python construit l'arbre a ?

b) Soit b = a.fils_gauche() ? Quel est le type de la variable b ?

c) Que renvoie b.est_vide() ?

Partie 3

Dans cette partie, nous allons créer deux **fonctions** pour rechercher et ajouter une valeur dans un ABR.

1) La recherche d'une valeur dans un ABR nécessite de parcourir l'arbre jusqu'à trouver la valeur si elle est présente. Le parcours d'un arbre est récursif, donc la recherche d'une valeur s'appuie sur la récursivité.

On donne ci-après l'algorithme de recherche d'une valeur `x` dans un arbre :

```
si arbre vide:
    valeur de x non présente
sinon:
    si x < valeur du Noeud visité:
        on renvoie la recherche de x avec arbre gauche
    sinon si x > valeur Noeud visité:
        on renvoie la recherche de x avec arbre droit
    sinon:
        valeur de x présente
```

a) Écrire la fonction **appartient** qui prend en paramètre la valeur x cherchée et un arbre binaire de recherche. Cette fonction est de type booléen, elle renvoie **True** si la valeur est dans l'arbre, **False** sinon.

b) Vérifier votre fonction et votre méthode avec l'arbre `a` créé dans la première partie.

2) L'ajout d'une valeur dans un ABR nécessite aussi le parcours récursif de l'arbre et l'ajout d'un Nœud .
On donne ci-après l'algorithme de l'ajout d'une valeur `x` dans un arbre `a`

```
si arbre a est vide:
    a = Arbre(x)
sinon:
    si x < valeur du noeud visité:
        si le noeud gauche vide:
            noeud gauche = Noeud(x)
        sinon:
            on ajoute x dans arbre gauche
    si x > valeur noeud visité:
        si le droit vide:
            noeud droit = Noeud(x)
        sinon:
            on ajoute x dans arbre droit
    renvoi de arbre a
```

a) Écrire la fonction **ajoute** qui prend en paramètre la valeur `x` à ajouter et un arbre binaire de recherche. Elle renvoie un arbre binaire avec la valeur ajoutée.

b) Vérifier votre fonction avec l'arbre `a` créé dans la première partie en y ajoutant différentes valeurs.