

Exercice 9

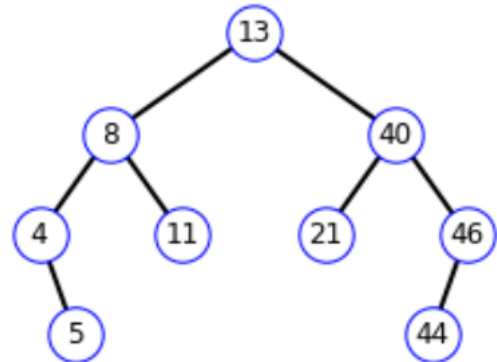
ABR

Un arbre binaire de recherche (ABR) est une structure de donnée composée de nœuds. Chaque nœud a au plus 2 enfants ordonnés d'une manière particulière :

- les enfants à gauche d'un nœud ont des valeurs inférieures à lui.
- les enfants à droite d'un nœud ont des valeurs supérieures à lui.
- Et cela doit être vrai pour chaque nœud de l'arbre.

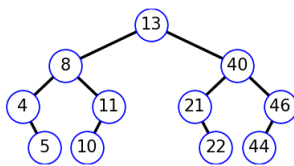
Ci-contre on a représenté un arbre binaire de recherche à partir de la liste d'entier:

[13, 40, 8, 4, 5, 11, 46, 21, 44]



? **QUESTION 1:**

1. Quelle est la taille de cet arbre? 9
2. Reproduire cet arbre en y insérant les entiers 10 et 22. *En respectant les règles d'un ABR.*



? **QUESTION 2:**

Parmi les trois parcours en profondeur (prefixe, infixe et postfixe (ou suffixe)), lequel permet d'obtenir la liste triée en ordre croissant des entiers de l'arbre de la question 1? **infixe**

On a choisit de représenter les arbres à partir d'une classe Noeud.

```
class Noeud:
    def __init__(self, valeur, gauche = None, droit = None):
        self.valeur = valeur
        self.gauche = gauche
        self.droit = droit
```

*gauche et droit à défaut d'être **None** sont eux-mêmes des instances de la classe Noeud.*

? QUESTION 3:

Voici une fonction récursive qui permet d'insérer une valeur dans un ABR non vide.

```
def inserer(arbre,valeur):
    if valeur < arbre.valeur:
        if arbre.gauche is None:
            arbre.gauche = Noeud(valeur)
        else:
            inserer(arbre.gauche,valeur)
    elif valeur > arbre.valeur:
        if arbre.droit is None:
            arbre.droit=Noeud(valeur)
        else:
            inserer(arbre.droit,valeur)
```

1. Recopier cette fonction en remplaçant les ??? par les bonnes instructions.
2. Justifier que cette fonction ne permet pas d'insérer deux fois la même valeur.

? QUESTION 4:

On considère un arbre binaire de recherche constitués d'entiers comme dans la question

1.

1. Décrire une méthode permettant d'atteindre le minimum des entiers de l'arbre. **il faut atteindre la valeur la plus à gauche**
2. Écrire une fonction minimum, prenant en paramètre un ABR et renvoyant le minimum.

```
def minimum(arbre):
    if arbre:
        if arbre.gauche==None:
            return arbre.valeur
        else:
            return minimum(arbre.gauche)
```