

## TD: arbres binaires de recherche:

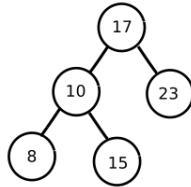


Figure 1: Arbre binaire de recherche

**Exercice 1: Arbre binaire de recherche (fonctions):** On s'intéresse au programme suivant:

```
0 def creer_arbre(r = None):
1     """renvoie un arbre vide ou un arbre de racine r"""
2     if r:
3         return [r, [], []]
4     else:
5         return []
6
7 def arbre_vide(a):
8     return a == []
9
10 def fils_gauche(a):
11     if not arbre_vide(a):
12         return a[1]
13
14 def fils_droit(a):
15     if not arbre_vide(a):
16         return a[2]
17
18 def insere(a, val):
19     """fonction recursive"""
20     if arbre_vide(a):
21         a.append(val)
22         a.append([])
23         a.append([])
24     elif val <= a[0]:
25         insere(a[1], val)
26     else:
27         insere(a[2], val)
28
```

arbre\_binaire\_relation\_ordre.py

1. Expliquer avec précision le rôle de chacune de ces fonctions. En quoi la fonction `insere()` est-elle récursive?
2. En quoi l'arbre de la figure 1 est-il un arbre binaire de recherche?
3. A l'aide de l'éditeur Python, implémenter l'arbre de la figure 1 et le représenter.

**Exercice 2: Arbre binaire de recherche (classe v1):** On s'intéresse au programme suivant:

```
0 class Noeud:
1     """Arbre binaire de recherche implemente par ses noeuds"""
2     def __init__(self, valeur):
3         self.valeur = valeur
4         self.parent = None
5         self.gauche = None
6         self.droite = None
7
8     def __str__(self):
9         return str(self.valeur)
10
11    def ajoute(self, valeur):
12        if valeur < self.valeur:
13            if self.gauche is None:
14                self.gauche = Noeud(valeur)
15                self.gauche.parent = self
16            else:
17                self.gauche.ajoute(valeur)
18        elif valeur > self.valeur:
19            if self.droite is None:
20                self.droite = Noeud(valeur)
21                self.droite.parent = self
22            else:
23                self.droite.ajoute(valeur)
```

arbre\_binaire\_recherche1.py

1. Expliquer avec précision le rôle de chacune des méthodes de la classe Noeud. Une des fonctions est-elle récursive?
2. A l'aide de l'éditeur Python, implémenter l'arbre de la figure 1 et le représenter.

**Exercice 3: Arbre binaire de recherche (classe v2):** On s'intéresse au programme suivant:

```
0 class Arbre:
1     def __init__(self, val):
2         self.valeur = val
3         self.gauche = None
4         self.droit = None
5
6     def insere(self, val):
7         if val < self.valeur:
8             if self.gauche != None:
9                 self.gauche.insere(val)
10            else:
11                self.gauche = Arbre(val)
12        else:
13            if self.droit != None:
14                self.droit.insere(val)
15            else:
16                self.droit = Arbre(val)
```

arbre\_binaire\_recherche2.py

1. Expliquer avec précision le rôle de chacune des méthodes de la classe Noeud. Une des fonctions est-elle récursive?

2. A l'aide de l'éditeur Python, implémenter l'arbre de la figure 1 et le représenter.

**Exercice 4: Arbre binaire de recherche:**

Dessiner un arbre binaire de recherche contenant les valeurs 11, 13, 14, 15, 17, 18, 19 satisfaisant à la condition donnée dans chaque cas:

1. la hauteur de l'arbre est 3.
2. ressemblant à l'arbre suivant:

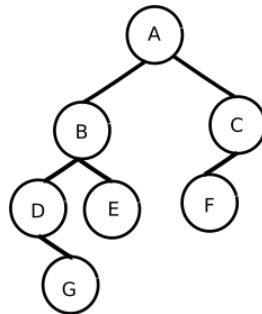


Figure 2: Arbre binaire de recherche