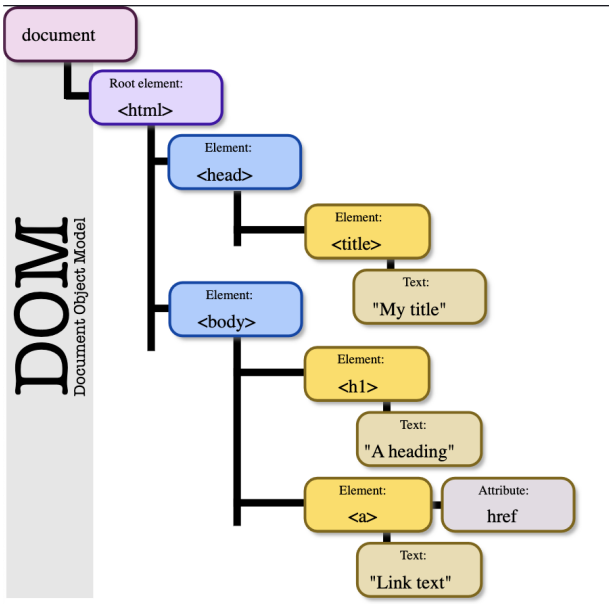
Les arbres

Table des matières

1] Présentation

Dans ce chapitre, on présente une nouvelle structure de donnée : les arbres qui sont particulièrement adaptés à la représentation des données hiérarchiques comme un arbre généalogique ou encore le DOM d'une page html.



2] Les arbres

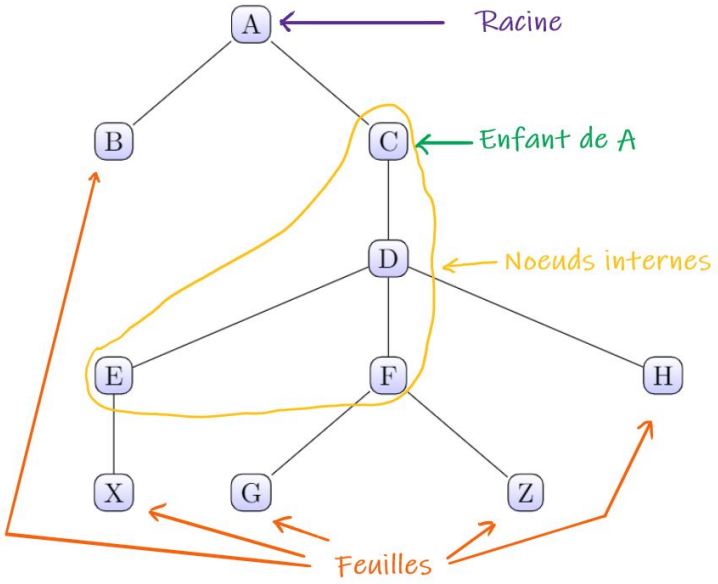
2.1) Définition

Nous allons travailler sur la structure de donnée "arbre". Cette structure n'est pas linéaire mais hiérarchique.

Définition

Un **arbre** est une structure de donnée constituée de nœuds. Chaque noeud peut être étiqueté par une information.
Le **sommet** de l'arbre s'appelle la racine.
Le **nœud** B situé sous un nœud A est appelé enfant du nœud A
Un nœud qui ne possède pas d'enfant est appelé **feuille**.
Les nœuds autre que la racine et les feuilles sont appelés **nœuds internes**.
Une **branche** est une suite de nœud consécutifs de la racine vers une feuille.

Exemple :

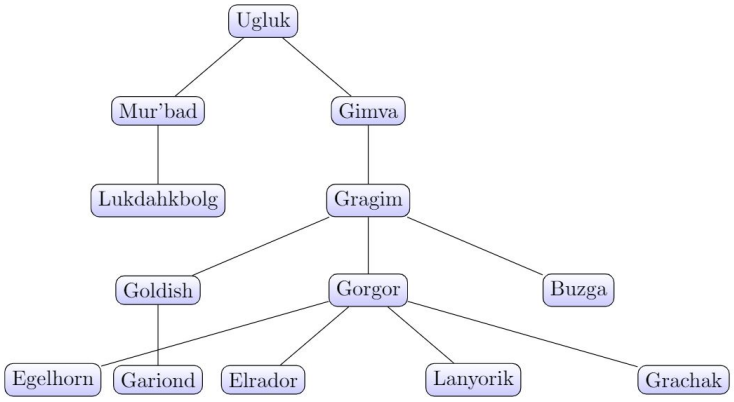


Dans cette arbre :

- il y a 10 nœuds.
- Le nœud A est la racine
- Il y a 5 feuilles donc 5 branches.
- Le nœud D est l'enfant du nœud C.
- Il y a 4 nœuds internes.

Exercice 1 :

On donne l'arbre suivant :



1. Quelle est la racine de cette arbre ?
2. Combien y-a-t-il de nœuds ?
3. Combien y-a-t-il de feuilles ?
4. Combien y-a-t-il de branches ?
5. L'ensemble des nœuds internes compte combien d'éléments ?

6. Quels sont les enfants de Gragim ?

2.2) Caractéristiques

Caractéristiques

On peut caractériser un arbre par différentes caractéristiques :

- **Son arité** : le nombre maximal d'enfants qu'un nœud peut avoir.
- **Sa taille** : le nombre de nœud qui le compose.
- **Sa hauteur** : le nombre de nœud que constitue la branche contenant le plus de nœuds (racine non comprise).

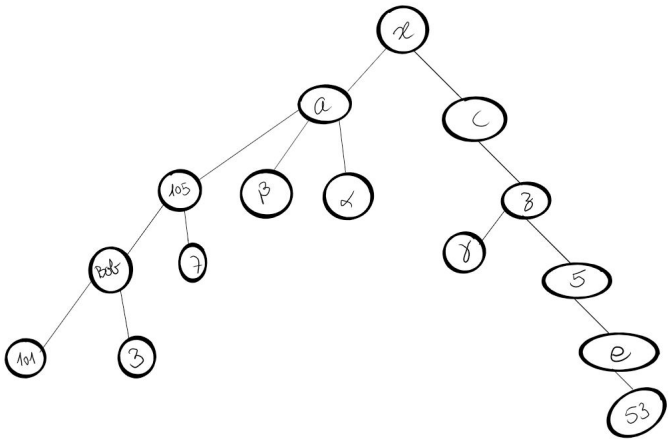
Exemple :

Si on reprend l'arbre de la définition précédente.

- L'arité de cet arbre est 3.
- La taille de cette arbre est 10.
- La hauteur de cet arbre est 4.

Exercice 2 :

On donne l'arbre suivant :



Déterminer l'arité, la taille et la hauteur de cette arbre.

2.3) Utilisation des arbres

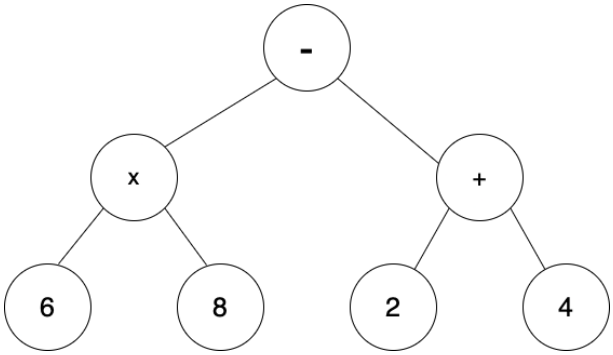
a) Les arbres de jeux

Un arbres de jeu promet de représenter toutes les positions et tous les coups possibles d'un jeu à l'aide d'un arbre.

b) Arbres représentant des expressions arithmétiques

Les quatres opération +,-,*,/ sont des opérations binaires. Une expression ne comportant que ces opérations peut être représentée sous forme d'arbre binaire comportant deux sortes de noeuds : les noeuds internes sont des opérateurs et les feuilles sont les nombres.

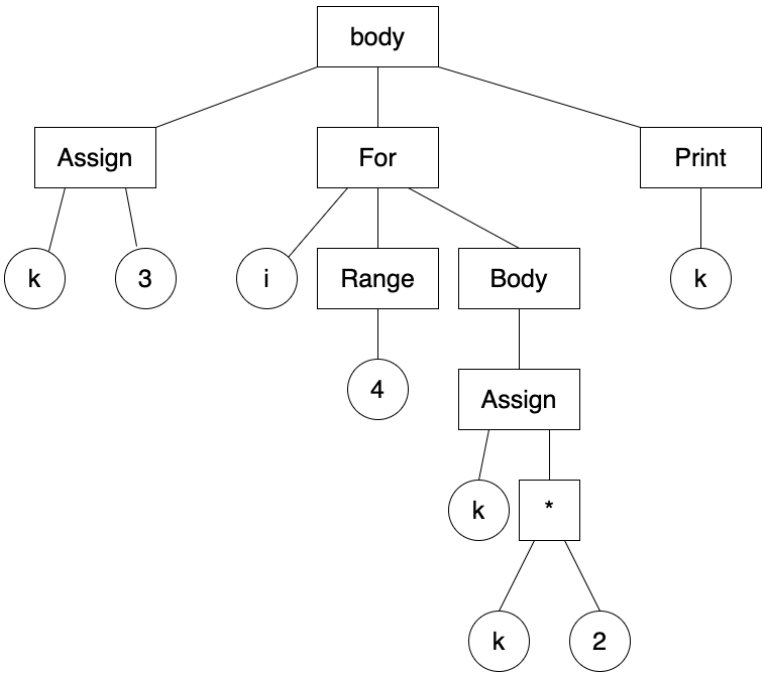
Dans un tel arbre, l'arrangement des noeuds joue le rôle de parenthèses auxquelles nous sommes habitués. Ainsi, l'arbre ci-dessous représente l'expression (6*8)-(2+4).



c) Arbres syntaxiques

Comme tous les langages informatiques, Python dispose d'une grammaire qui indique comment former des programmes corrects. Lorsque l'interpréteur Python lit du code source, il construit tout d'abord l'arbre syntaxique du code. En simplifiant un peu, le code source suivant pourrait représenté par l'arbre syntaxique ci-dessous.

```
1 k = 3
2 for i in range(4):
3     k = k * 2
4 print(k)
```



3] Arbres binaires

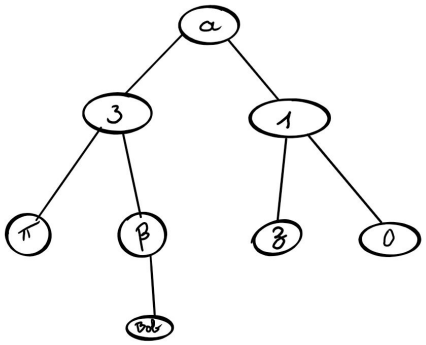
3.1) Définition

Arbre biniare

On appelle arbre binaire un arbre d'arité 2.

Les arbres binaires sont constitués de nœuds de 0, 1 ou 2 enfants.

Exemple :



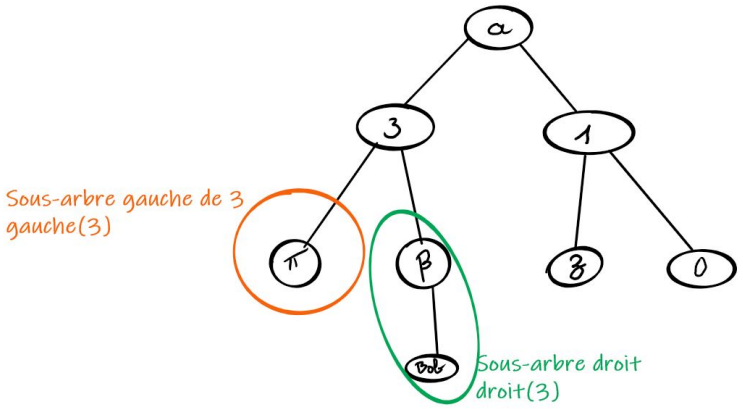
Sous-arbre

Quand un nœud a deux enfants, il possède un sous-arbre gauche et un sous-arbre droit.

Pour un nœud α, on notera :

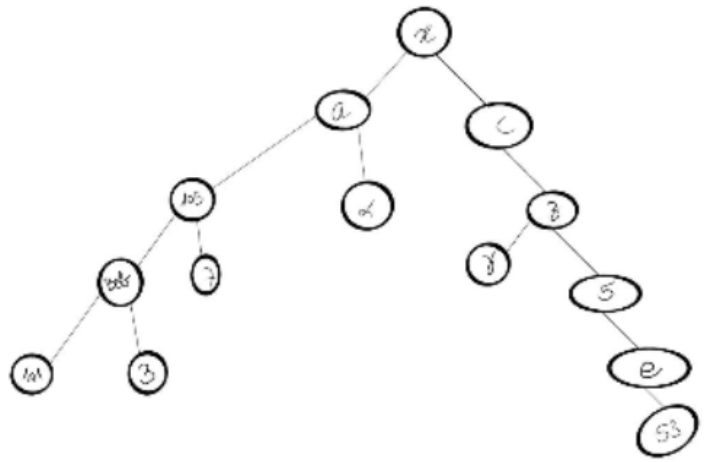
- gauche(α) : le sous-arbre gauche du nœud α ;
- droit(α) : le sous-arbre droit du nœud α.

Exemple :

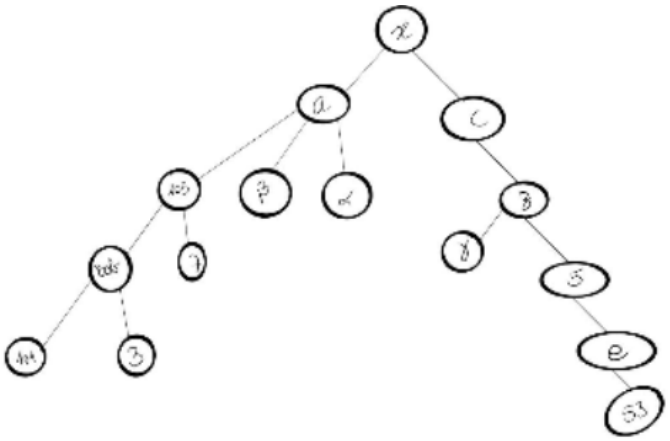


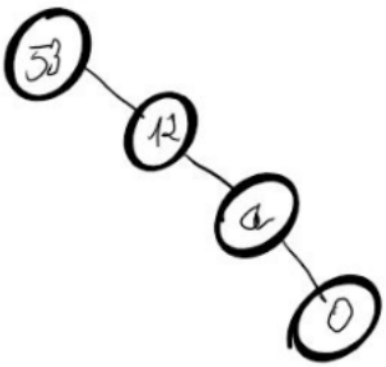
Exercice 3 : Parmi les arbres suivants lesquels sont binaires ?

1.



2.





3.2) Mesures sur les arbres binaires

Les arbres binaires étant particuliers, on peut calculer un certains nombres de caractéristiques d'un arbre binaire.

a) Taille d'un arbre

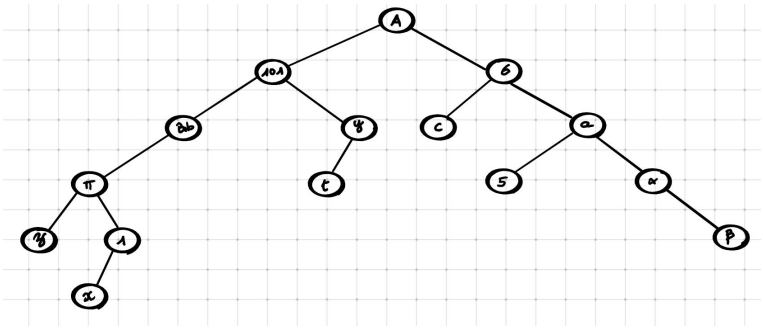
Taille d'un arbre B

La taille d'un arbre B correspond au nombre de ses nœuds, elle est définie par :

- Taille(B)=0, si B est un arbre vide
- Taille(B)=1+Taille(gauche(B))+Taille(droit(B)) sinon

Exercice 4 :

On donne l'arbre suivant :



Déterminer la taille de cette arbre.

b) Hauteur d'un nœud ou profondeur d'un nœud

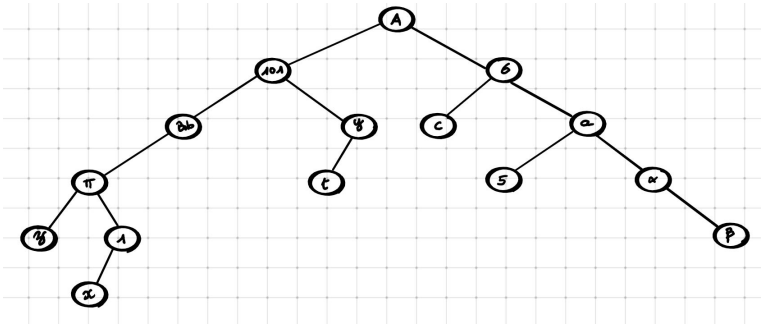
Hauteur d'un nœud ou profondeur d'un nœud

La hauteur d'un nœud ou la profondeur de x correspond au nombre d'arêtes au dessus x pour revenir à la racine, elle est définie par :

- HauteurDeNoeud(x)=0, si x est la racine de l'arbre.
- HauteurDeNoeud(x)=1+HauteurDeNoeud(y) si y est le père de x. sinon

Exercice 5 :

On donne l'arbre suivant :



Déterminer la hauteur des nœuds bob et α

c) Hauteur d'un arbre

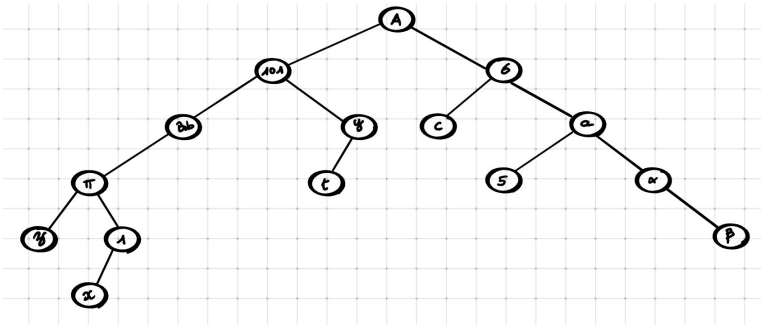
Hauteur d'un arbre

La hauteur d'un arbre B correspond au nombre d'arêtes entre la racine et la feuille la plus éloignée :

- Hauteur(B)=Max(HauteurDeNoeud(x)), où x décrit l'ensemble des nœud de B.

Exercice 6 :

On donne l'arbre suivant :



Déterminer la hauteur de cet arbre.

d) Longueur de cheminement d'un arbre

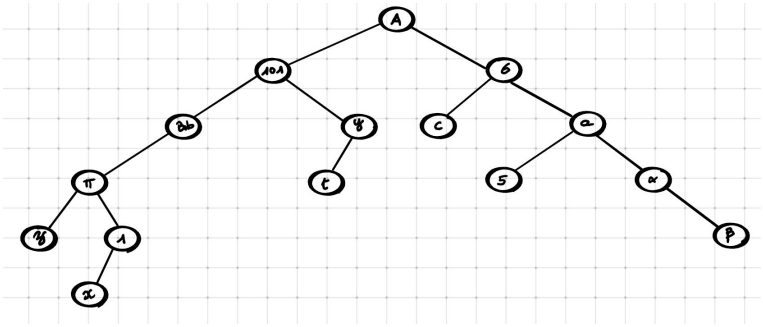
Longueur de cheminement d'un arbre

La longueur de cheminement d'un arbre B correspond à la somme des hauteurs de chacun des nœuds :

- $LC(B) = \sum_{i=1}^{T(B)} H(x_i)$

Exercice 7 :

On donne l'arbre suivant :



Déterminer la longueur de cheminement de cet arbre.

e) Longueur de cheminement externe d'un arbre

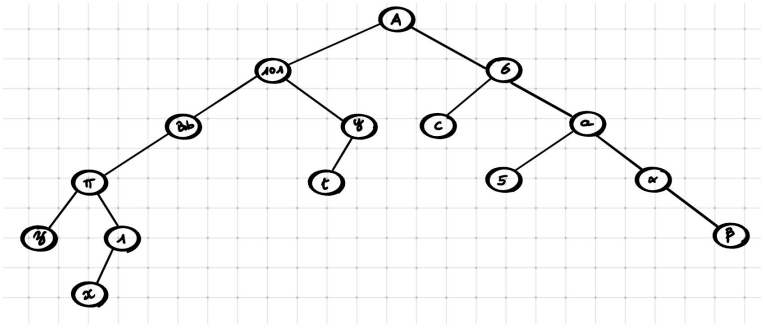
Longueur de cheminement externe d'un arbre

La longueur de cheminement externe d'un arbre B correspond à la somme des hauteurs de chacune des feuilles :

- $LCE(B) = \sum_{i=1}^{NF} H(f_i)$ où NF est le nombre de feuilles

Exercice 8 :

On donne l'arbre suivant :



Déterminer la longueur de cheminement externe de cet arbre.

f) Longueur de cheminement interne d'un arbre

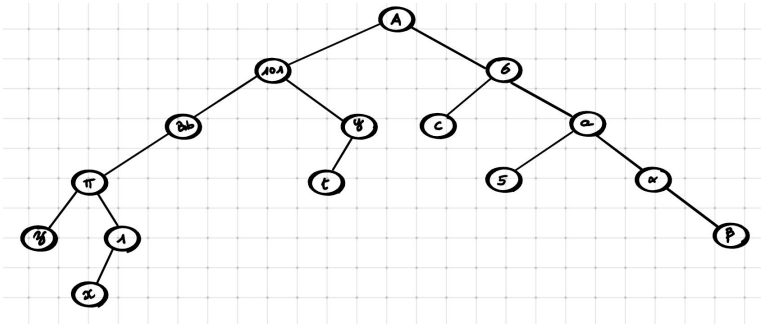
Longueur de cheminement interne d'un arbre

La longueur de cheminement interne d'un arbre B correspond à la somme des hauteurs des noeuds internes de B :

- $LCI(B) = \sum_{i=1}^{T(B)-NF} H(x_i)$ où x_i est un nœud interne.

Exercice 9 :

On donne l'arbre suivant :



Déterminer la longueur de cheminement interne de cet arbre.

Exercice 10 :

Déterminer une relation entre $LC(B)$,$LCE(B)$ et $LCI(B)$

g) Profondeur moyenne d'un arbre

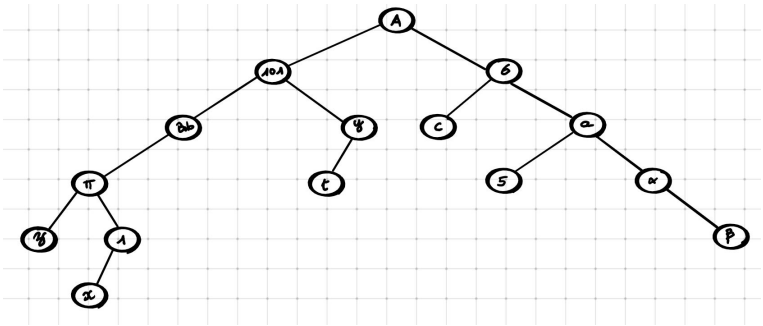
Profondeur moyenne d'un arbre

La profondeur moyenne d'un arbre B est définie par :

$$PM(B) = \frac{LC(B)}{T(B)}$$

Exercice 11 :

On donne l'arbre suivant :



Déterminer la profondeur moyenne de cet arbre.

h) Profondeur moyenne externe d'un arbre

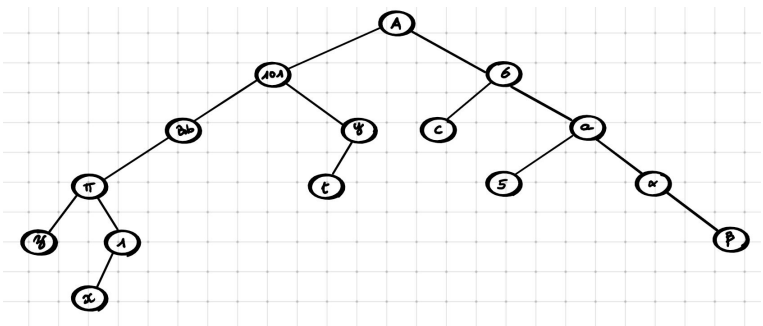
Profondeur moyenne externe d'un arbre

La profondeur moyenne externe d'un arbre B est définie par :

$$PME(B) = \frac{LCE(B)}{NF}$$

Exercice 12 :

On donne l'arbre suivant :



Déterminer la profondeur moyenne externe de cet arbre.

i) Profondeur moyenne interne d'un arbre

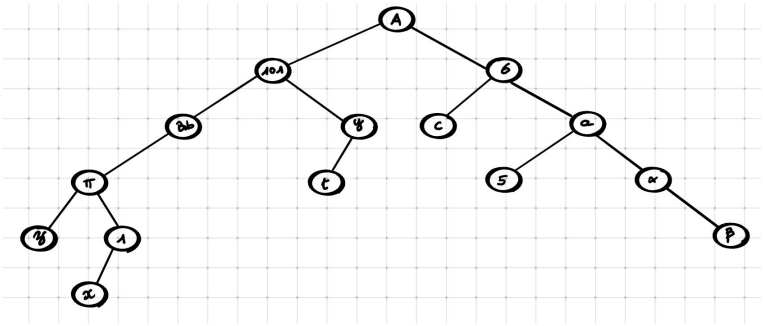
Profondeur moyenne interne d'un arbre

La profondeur moyenne interne d'un arbre B est définie par :

$$PMI(B) = \frac{LCI(B)}{T(B) - NF}$$

Exercice 13 :

On donne l'arbre suivant :

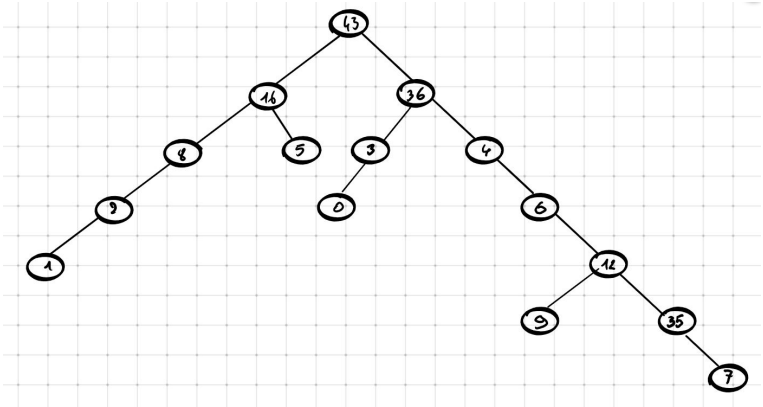


Déterminer la profondeur moyenne interne de cet arbre.

Remarque : La longueur de cheminement et la profondeur moyenne seront utiles pour les algorithmes sur les arbres binaires.

Exercice 14 :

On donne l'arbre B suivant :

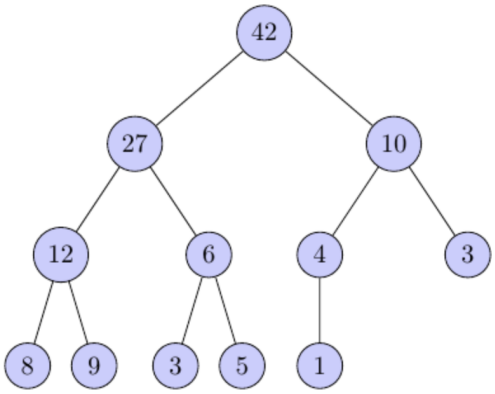


Déterminer : la racine ; le nombre de feuilles ; le nombre de branches ; l'arité ; sa taille : $T(B)$; la hauteur de B : $H(B)$; $LC(B)$; $LCE(B)$; $LCI(B)$; $PM(B)$; $PME(B)$; $PMI(B)$.

3.3) utilisation : tas binaires

Un tas binaire est un arbre binaire complet à gauche, tel que le nœud racine porte une donnée supérieure à celle de tous les autres noeuds et tel que ses deux sous-arbres soient aussi des tas. L'arbre est alors un tas-max.

Exemple de tas-max :



Un tas-min a la même structure, mais chaque noeud porte une donnée plus petite que tous ses descendants.

La structure de tas binaire est utilisée : dans un algorithme de tri (tri par tas), dans la gestion des files de priorité.

4] Implémentation des arbres binaires

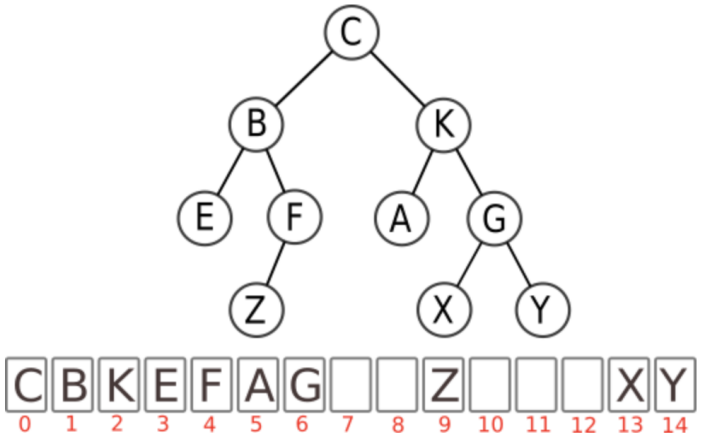
4.1) A partir des listes

Il existe une méthode pour implémenter un arbre binaire (qui est une structure hiérarchique) avec une liste (qui est une structure linéaire). Ceci peut se faire par le biais d'une astuce sur les indices :

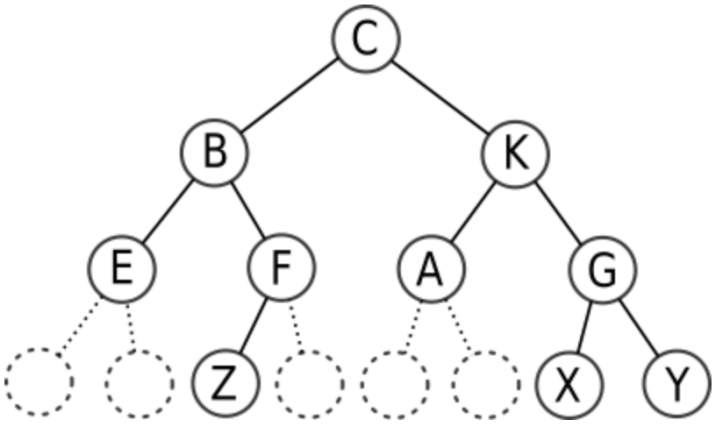
Les fils du nœud d'indice i sont placés aux indice $2i + 1$ et $2i + 2$.

Cette méthode est connue sous le nom de «méthode d'Eytzinger», et utilisée notamment en généalogie pour numéroter facilement les individus d'un arbre généalogique.

Exemple :



Pour comprendre facilement la numérotation, il suffit de s'imaginer l'arbre complet (en rajoutant les fils vides) et de faire une numérotation en largeur, niveau par niveau :

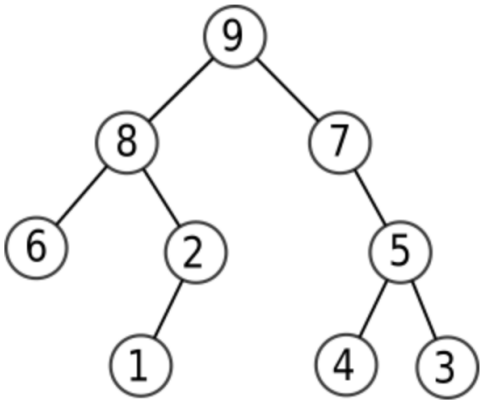


Exercice 15 :

Si on note Δ le sous-arbre vide, dessiner l'arbre représenté par la liste :
a = [3, 4, Δ, 7, 5, Δ, Δ]

Exercice 16 :

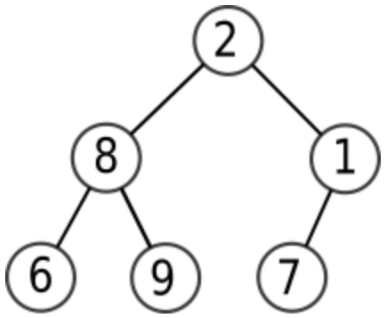
Déterminer l'implémentation de l'arbre suivant par une liste selon la méthode d'Eytzinger.



4.2) A partir de tuples

Considérons qu'un arbre peut se représenter par le tuple (valeur, sous-arbre gauche, sous-arbre droit).

L'arbre ci-dessous :



peut alors être représenté par le tuple :

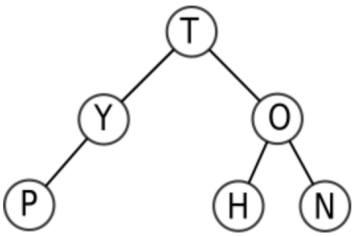
```
>>> a = (2, (8, (6,(),()), (9,(),())), (1, (7, (),()), ()))
```

Le sous-arbre gauche est alors a[1] et le sous-arbre droit est a[2].

```
>>> a[1]
(8, (6, (), ()), (9, (), ()))
>>> a[2]
(1, (7, (), ()), ())
```

Exercice 17 :

Écrire le tuple représentant l'arbre ci-dessous.



5] Arbre binaire de recherche

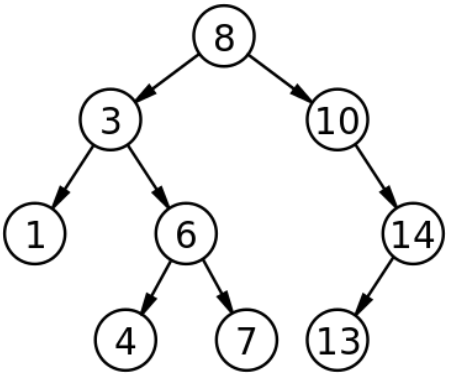
5.1) Définition

Arbre binaire de recherche

Un arbre binaire de recherche est un arbre binaire dans lesquels l'étiquette d'un nœud est appelé clé et est un entier. L'arbre binaire vérifie deux propriétés :

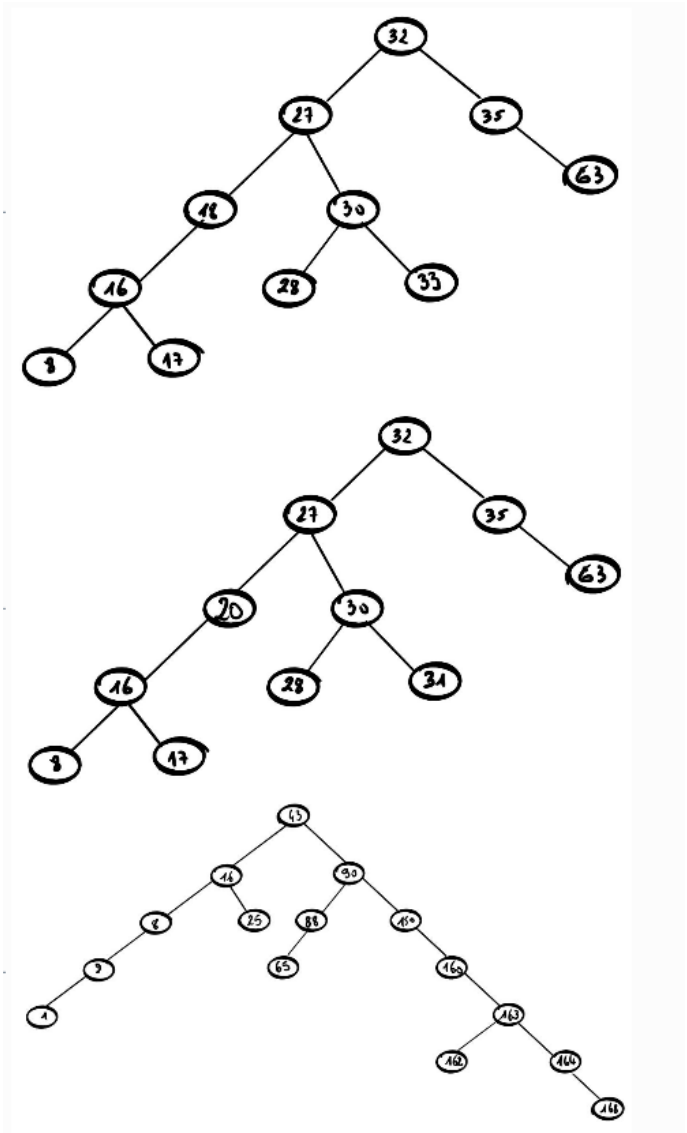
- Les clés de tous les nœuds du sous-arbre gauche gauche d'un n œud x sont inférieures ou égales à la clé de x.
- Les clés de tous les nœuds du sous-arbre droit d'un nœud x sont strictement supérieures à la clé de x.

Exemple :



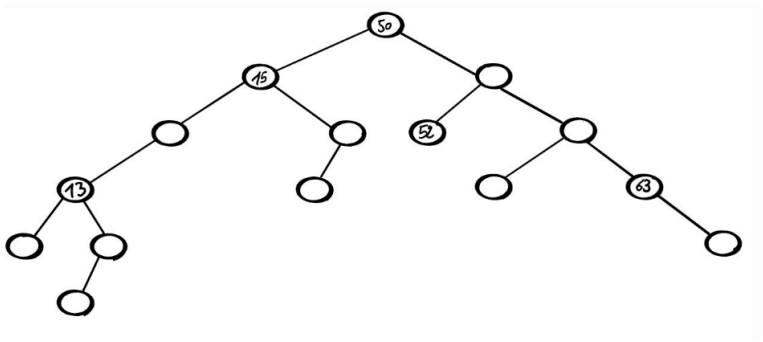
Exercice 18 :

Quels sont parmi les arbres proposés ci-dessous les arbres binaires de recherche ?



Exercice 19 :

Complétez cet ABR.

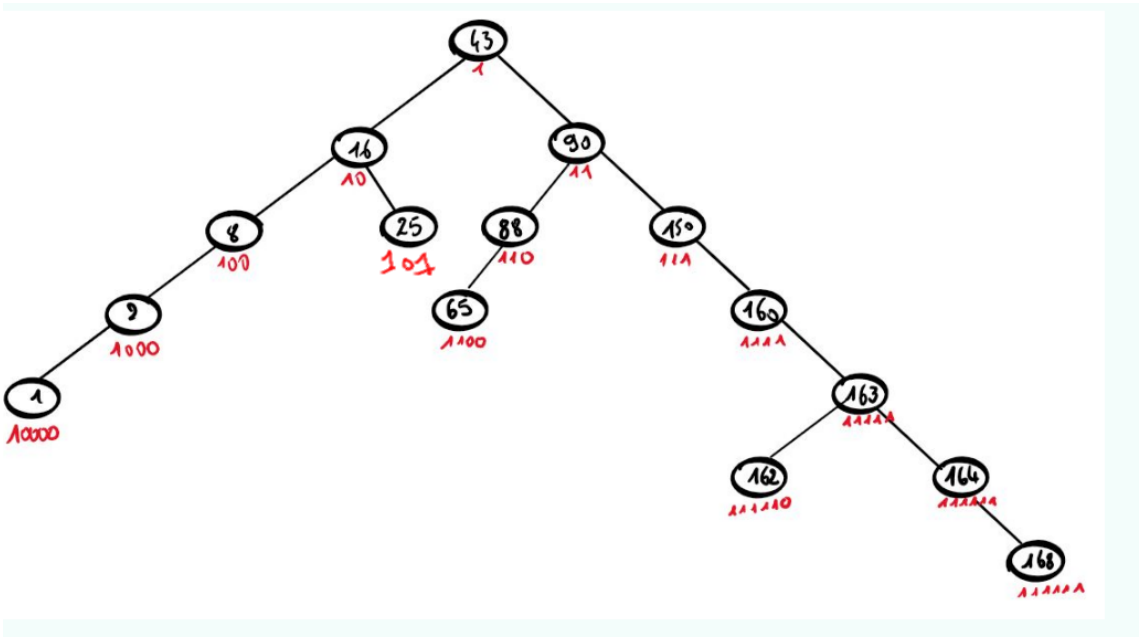


5.2) Propriétés

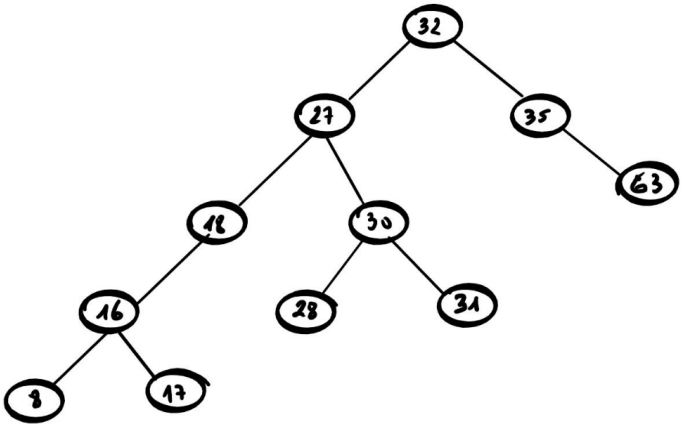
Un étiquetage intéressant d'un arbre de recherche est le suivant :

- La racine est étiquetée 1 ;
- La premier nœud du sous arbre gauche prend l'étiquette de son père auquel on ajoute un 0 ;
- La premier nœud du sous arbre droit prend l'étiquette de son père auquel on ajoute un 1.

Exemple :



Exercice 20 : Étiqueter comme l'exemple précédent l'arbre suivant :



Remarque : On observe que les étiquettes des nœuds de profondeur p sont constituées de p + 1 bits, et sont deux à deux distinctes. On en déduit que toutes les étiquettes des nœuds d'un même arbre sont deux à deux distinctes.

Les étiquettes peuvent être vues comme les écritures de numéros en base 2 : on a ainsi numéroté les différents nœuds de l'arbre.

5.3) Utilisation

Les arbres binaires de recherche permettent de rechercher une élément dans un arbre de n noeuds en temps $O(h)$ où h est la hauteur de l'arbre binaire de recherche. Si l'arbre est équilibré, on a $h = log_2(n)$ et la recherche a une complexité en temps logarithmique. Si le l'arbre de recherche est déséquilibré (comme un peigne)) la complexité en temps peut être linéaire.

5.4) Une implémentation de l'objet arbre binaire de recherche en Python

Voici l'implémentation d'un arbre binaire de recherche en python.

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

    def inserer(self,x):
        if x<self.valeur:
            if self.gauche!=None:
                self.gauche.inserer(x)
            else:
                self.gauche=ABR(x)
        else:
            if self.droit!=None:
                self.droit.inserer(x)
            else:
                self.droit=ABR(x)
```

Exercice 21 : Créer l'arbre de l'exercice 18 avec cette implémentation

Exercice 22 : Créer une méthode `taille()`

Exercice 23 : Créer une méthode `hauteur()`