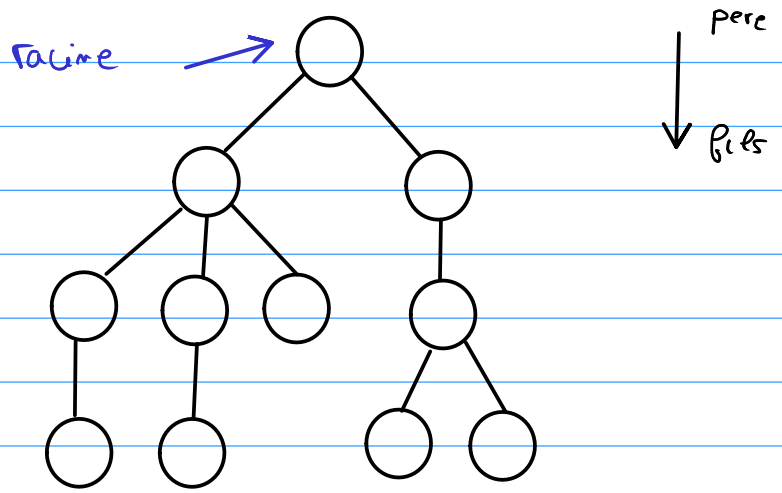
Chapitre V : Arbres

1. Généralités

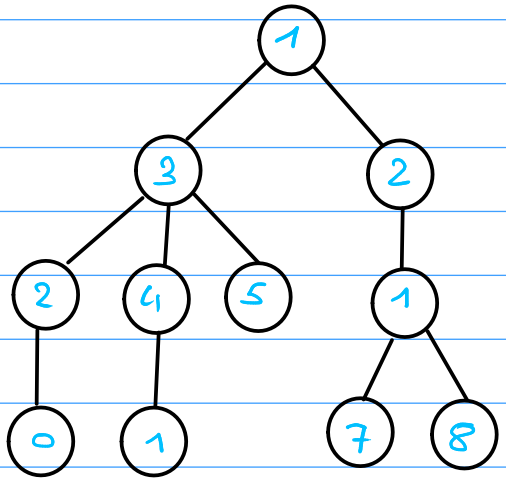
Def: Un arbre est une structure de donnée composée de noeuds partageant des liens de parenté :

- chaque noeud peut avoir un nombre quelconque de fils
- si l'arbre est non vide, exactement un noeud n'a pas de père, la racine
- tous les noeuds, à l'exception de la racine, possèdent exactement un père

Exemple:

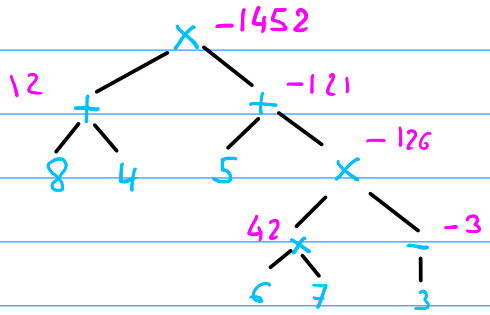


Les noeuds peuvent contenir de l'information.



Remarque : A toute expression algébrique on peut associer un unique arbre

Exemple : $(8+4)\times(5+(6\times7)\times(-3))$

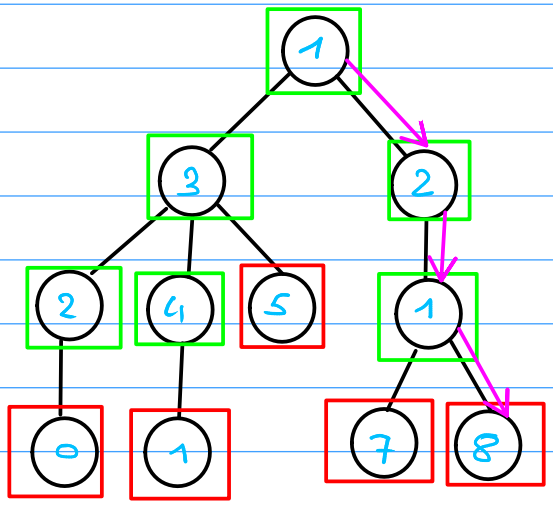


Def : . Un noeud sans fils est une feuille

. Un noeud qui n'est pas une feuille est un noeud interne

. La hauteur d'un arbre est la longueur maximale d'un chemin allant de la racine à une feuille.

Exemple :

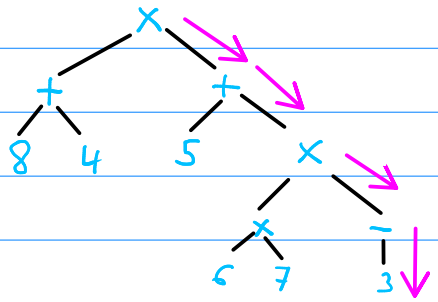


11 noeuds dont

5 feuilles et

6 noeuds internes

la hauteur est 3



12 noeuds dont 6 feuilles et 6 noeuds internes

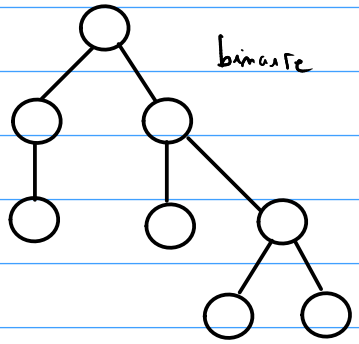
la hauteur est 4

Remarque : la hauteur de l'arbre à un seul noeud est 0

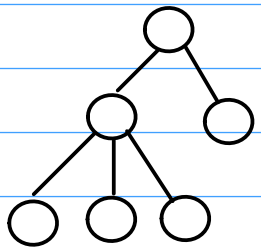
La hauteur de l'arbre vide est -1 par convention.

2. Arbres binaires

Def : Un arbre binaire est un arbre dont chacun des noeuds a au plus 2 fils.



binaire



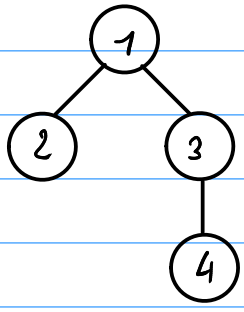
pas binaire

En python, un arbre binaire peut être représenté par la structure Arbre composée :

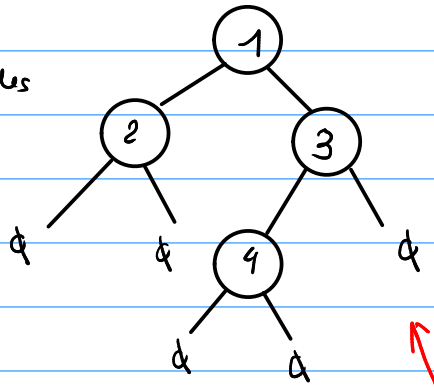
- val : stocke la valeur du noeud
- gauche : qui pointe (désigne) vers le sous-arbre gauche
- droite : qui pointe vers le sous-arbre droite

Le symbole ϕ désigne l'arbre binaire vide

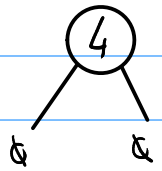
Exemple :



on ajoute les sous arbres vides

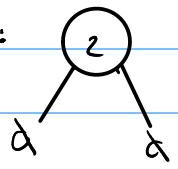


Arbre 4 :



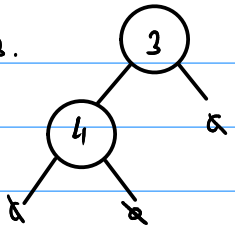
Val = 4
gauche = ϕ
droite = ϕ

Arbre 2 :



Val = 2
gauche = ϕ
droite = ϕ

Arbre 3.



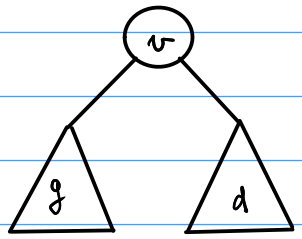
Val = 3
gauche = Arbre 4
droite = ϕ

Arbre 1 :

Val = 1
gauche = Arbre 2
droite = Arbre 3

Un arbre binaire est donc une structure définie récursivement.

- arbre binaire vide : ϕ
- arbre binaire non vide :

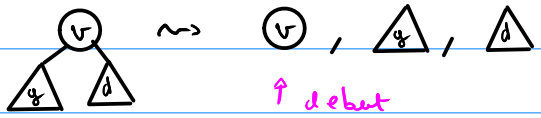


où g et d sont des arbres binaires (vide ou pas)

Parcours sur un arbre binaire Il existe 3 parcours sur les arbres binaires en fonction de quand le noeud est traité par rapport aux sous-arbres.

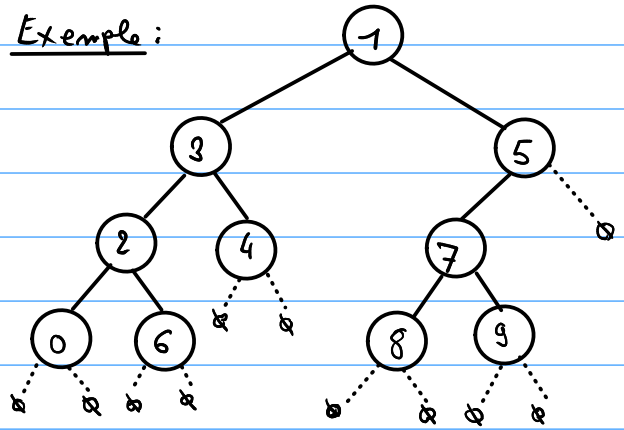
Infixé : On traite le sous arbre gauche puis le noeud puis le sous arbre droit

Préfixé :



Post fixé :

Exemple :



Infixé : 0, 2, 6, 3, 4, 1, 8, 7, 9, 5

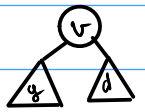
Préfixé : 1, 3, 2, 0, 6, 4, 5, 7, 8, 9

Post fixé : 0, 6, 2, 4, 3, 8, 9, 7, 5, 1

Algorithmes classiques

Nombres de noeuds

$\phi \longrightarrow 0$

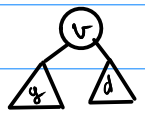


$\longrightarrow 1 + \text{noeuds}(g) + \text{noeuds}(d)$

```
noeuds ( A ) :
    Si  A = ϕ :
        retourne 0
    Sinon
        retourne 1 + noeuds( A.gauche ) + noeuds( A.droite )
```

Nombre de feuilles

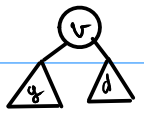
$\phi \longrightarrow 0$



$$\longrightarrow \begin{cases} 1 & \text{si } g = \phi \text{ et } d = \phi \\ \text{feuilles}(g) + \text{feuilles}(d) \end{cases}$$

Nombre de noeuds internes

$\phi \longrightarrow 0$

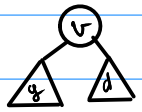


$$\longrightarrow \begin{cases} 0 & \text{si } g = \phi \text{ et } d = \phi \\ 1 + \text{internes}(g) + \text{internes}(d) \end{cases}$$

Hauteur

Rappels $h(\phi) = -1$ $h(0) = 0$ $h\left(\right) = 2$

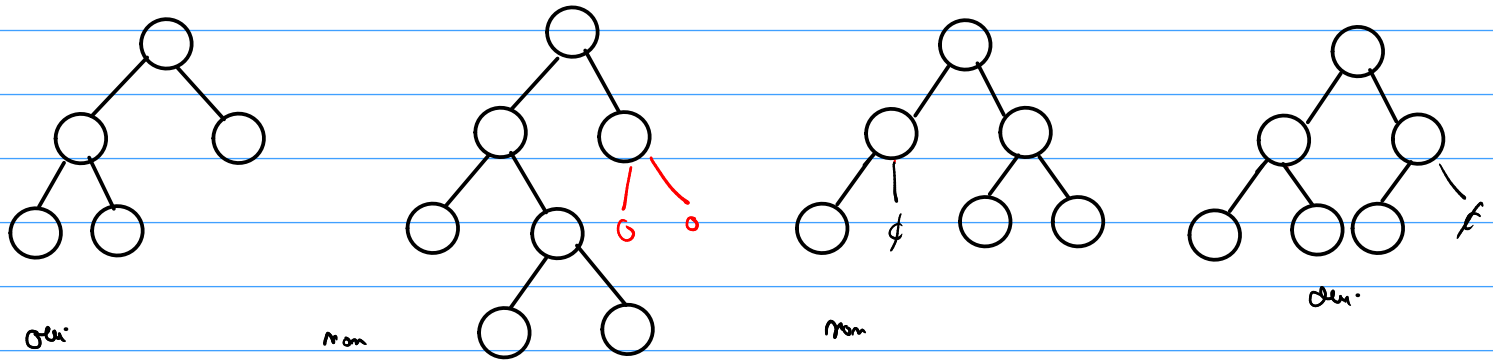
$\phi \longrightarrow -1$



$\longrightarrow 1 + \max(\text{hauteur}(g), \text{hauteur}(d))$

3. Tas

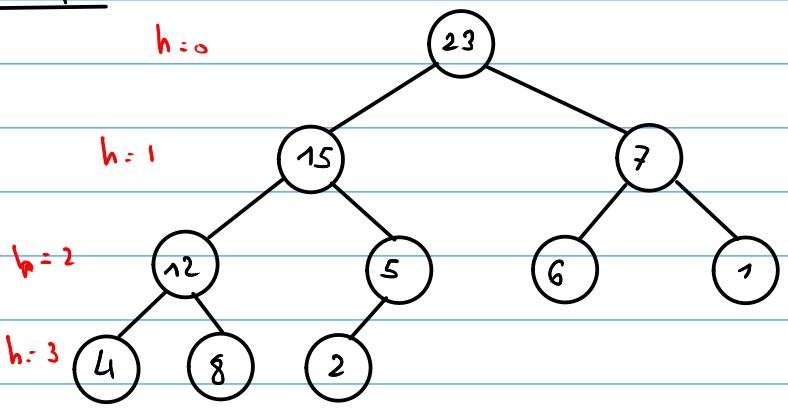
Def: Un arbre binaire est presque complet lorsque tous ses niveaux sont pleins à l'exception éventuelle du dernier et dans ses feuilles sont alignés à gauche.



Def: Un tas est un arbre binaire presque complet à valeurs dans V, muni d'un poids $\omega : V \to \mathbb{Z}$ tel que pour tout noeud (v), les poids des éléments du sous arbre  sont tous inférieur ou égal à $\omega(v)$

Exemple:

$V = \mathbb{N}$ $\quad \omega : V \to \mathbb{Z}$, $m \mapsto m$



c'est un tas !!

Rq L'élément de poids maximal se trouve à la racine

Représentation d'un tas en python : 23, 15, 7, 12, 5, 6, 1, 4, 8, 2

h=0 h=1 h=2 h=3

→ tableau

23	15	7	12	5	6	1	4	8	2
0	1	2	3	4	5	6	7	8	9

Indices

Indice des fils du noeud d'indice i : $2i+1$ et $2i+2$

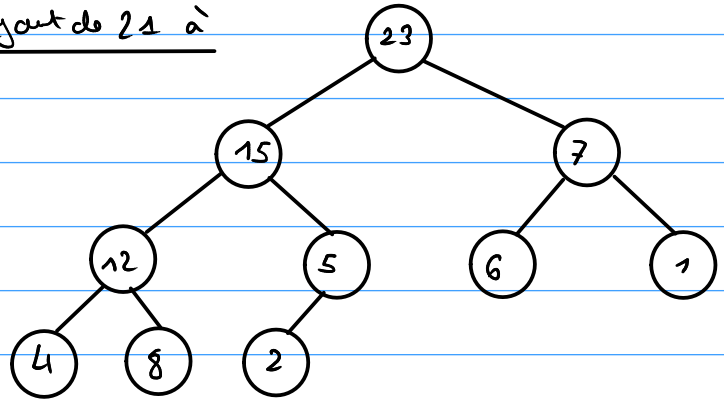
Ex: 12 a pour indice 3, ses fils ont indices 7, 8

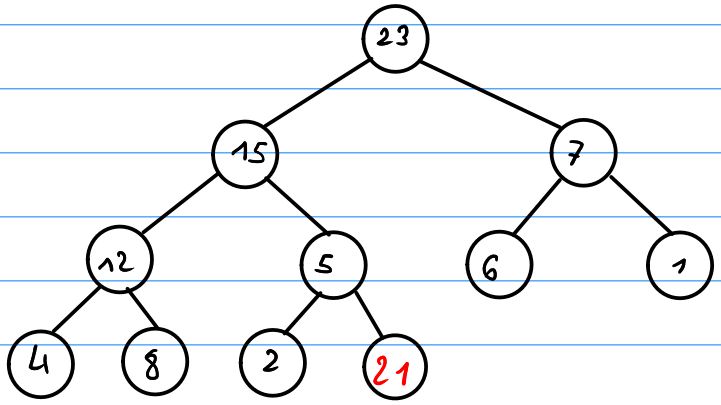
Indice du père du noeud d'indice j : $\left\lfloor \frac{j-1}{2} \right\rfloor$

Insertion d'un élément v :

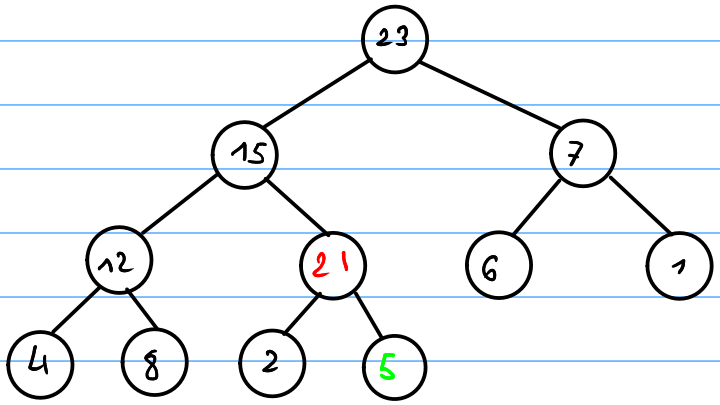
- On ajoute v à la fin du tas (du tableau)
- tant que w(v) est supérieur à w(père(v))
  On échange v avec son père

Ex : ajout de 21 à

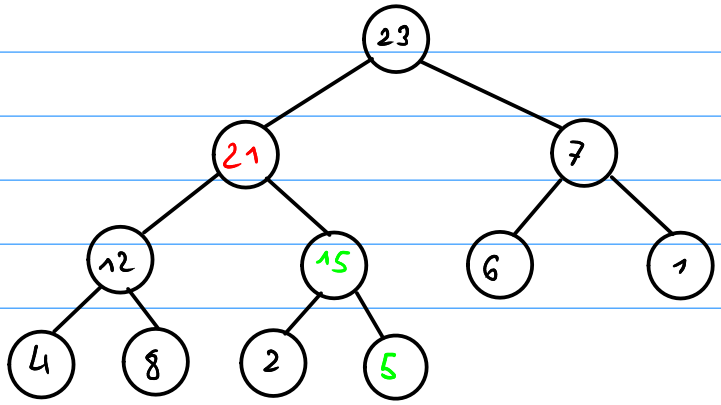




w(21) > w(5)



w(21) > w(15)

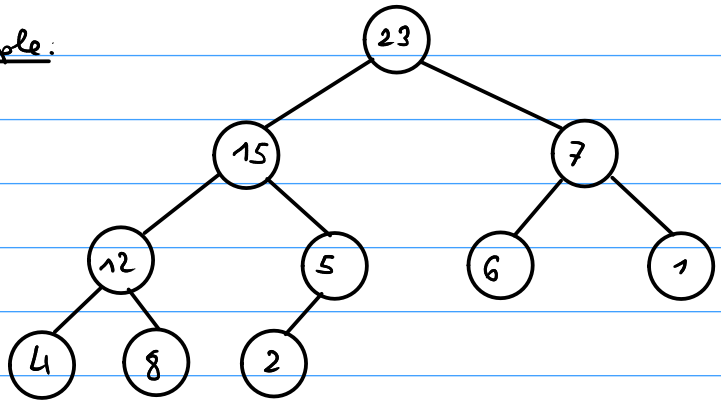


au final on obtient un tas !

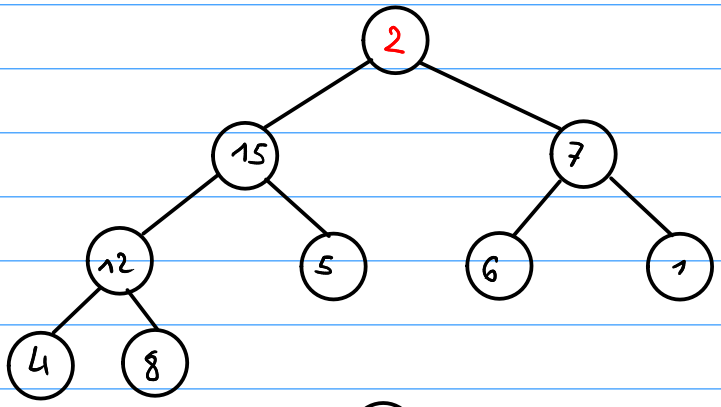
Suppression de l'élément maximal

- On remplace la racine par le dernier élément v
- Tant que v est de poids inférieur à celui de l'un de ses fils
  On échange v avec son fils de poids maximal

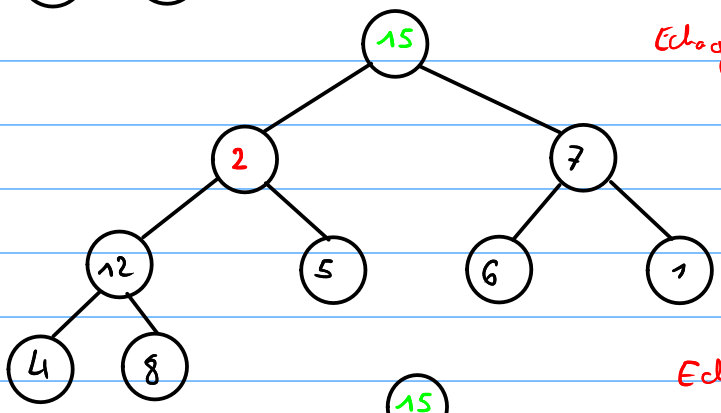
Exemple :



Elément maximal : 23

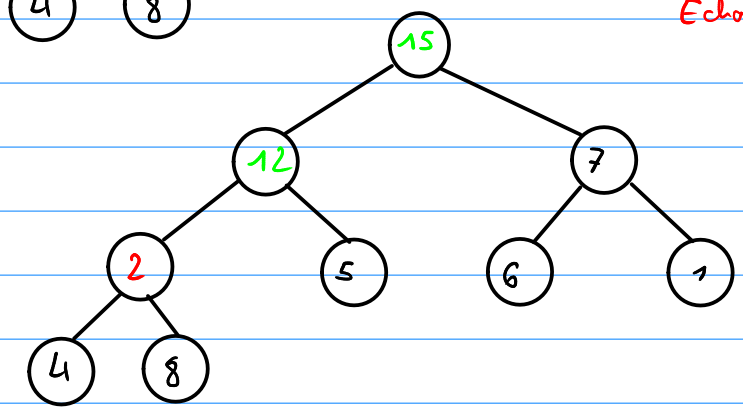


Echange 2 avec 15

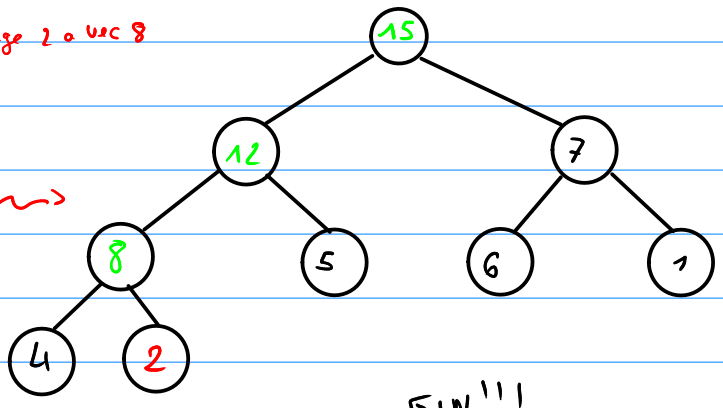


Echange 2 avec 12

Un tas



Echange 2 avec 8



FIN !!!

Rq : Les tas sont utilisés par le tri par tas, les files de priorité, ...