Math93.com

Cours - NSI
Les arbres

1 Quelques Exemples

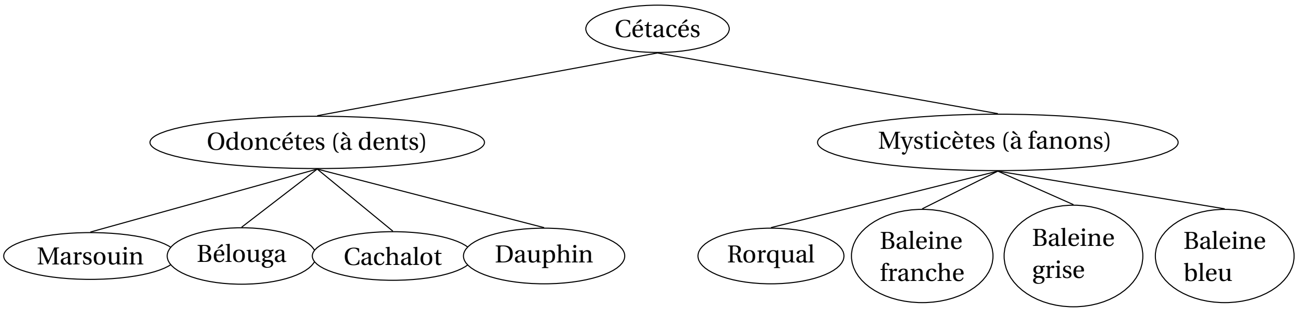


FIGURE 1 – Arbre phylogénétique

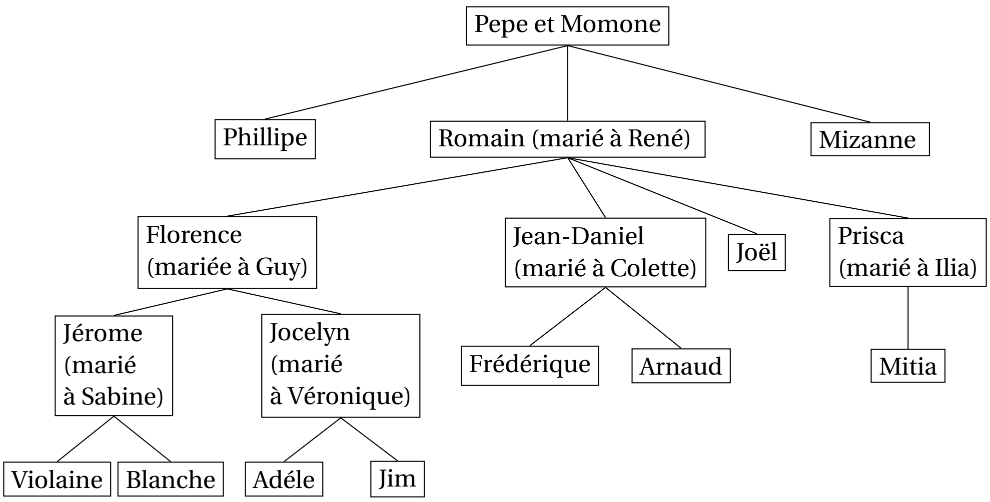


FIGURE 2 – Arbre Généalogique

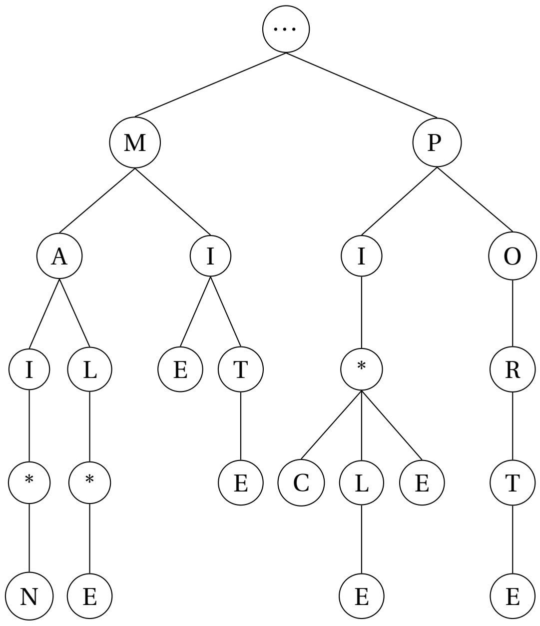


FIGURE 3 – Arbre Léxicographique

Exercice 1

Dans l'arbre lexicographique, introduire les mots MALLE, PORTAIL ET POLAR

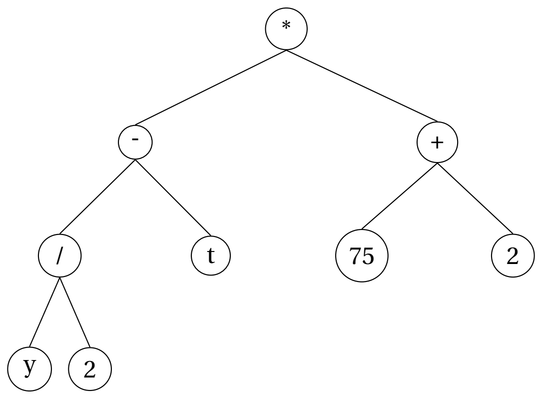


FIGURE 4 – Arbre d'une expression mathématiques $\left(\frac{y}{2} - t\right)(75+2)$ utilisant la priorité des opérations

Exercice 2

Représenter l'expression : $3 + (73 - 1)^3$ avec un arbre.

Les arbres permettent de :

- **hiérarchiser** les informations ;
- la représentation sous forme arborescente ;
- Rendre efficace l'accés aux aux informations de données volumineuses.

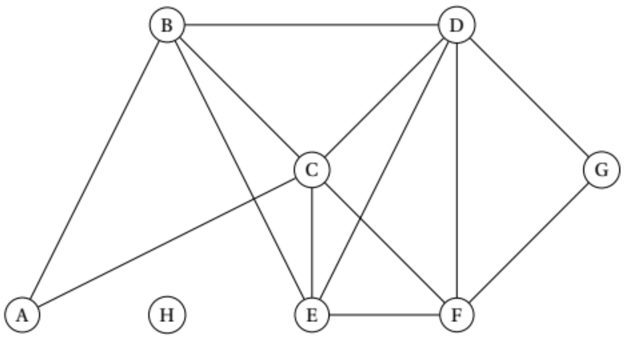
2 Définitions

2.1 Les graphes

Définition 1 (Graphe)

- Soit V un ensemble de sommets (aussi appelés nœuds, points ou vertex) ;
- Soit E un ensemble d'arêtes (ou arcs) qui relient des sommets de V : $E \subseteq \{(x, y) \in V^2 \,|\, x \neq y\}$

Le couple (V, E) est un graphe



Remarque

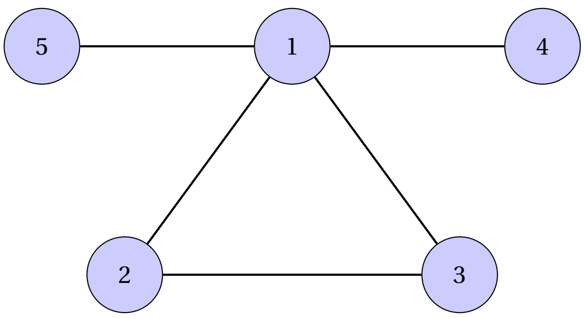
Une arête (ou arc) est un couple de sommets.

Définition 2 (Etiquette)

L'étiquette (ou nom du sommet) représente la "valeur" du noeud. Un arbre dont tous les noeuds sont nommés est dit étiqueté.

Définition 3 (Degré d'un noeud et degré d'un graphe)

Le **degré** d'un noeud est égal au nombre d'arréte qui partent de ce noeud.
Le **degré** d'un graphe est égal au plus grand des degrés de ses noeud.
Le **degré** d'un graphe vide est égal à 0.



Noeud	1	2	3	4	5	Degré du graphe
Degré	4	2	2	1	1	4

Définition 4 (Chaîne ou Chemin)

Un chemin (chaîne) d'origine x et d'extrémité y , noté $\mu[x, y]$ est défini par une suite finie d'arcs consécutifs, reliant x à y.
Une chaîne simple est une chaîne ne passant pas deux fois par une même arête, c'est-à-dire dont toutes les arêtes sont distinctes.

Définition 5 (Cycle)

Un cycle est une chaîne simple dont l'origine est égale à l'extrémité.

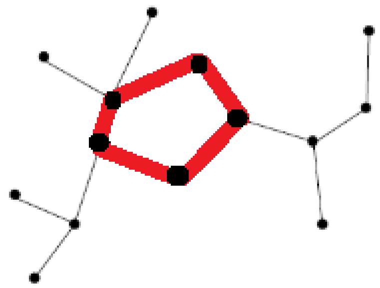


FIGURE 5 – Le chemin rouge est un cycle

Définition 6 (Graphe connexe)

Un graphe connexe est un graphe dont tous les sommets peuvent être relié par un chemin.

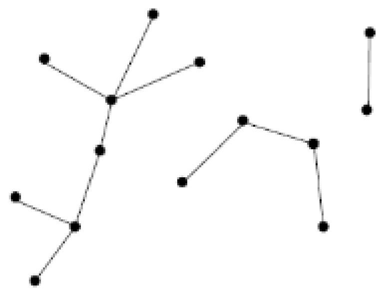


FIGURE 6 – graphe non connexe

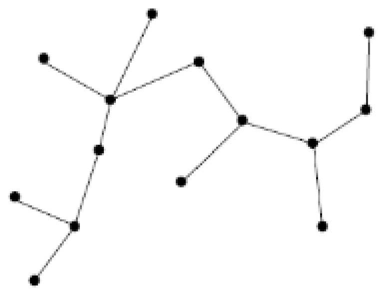


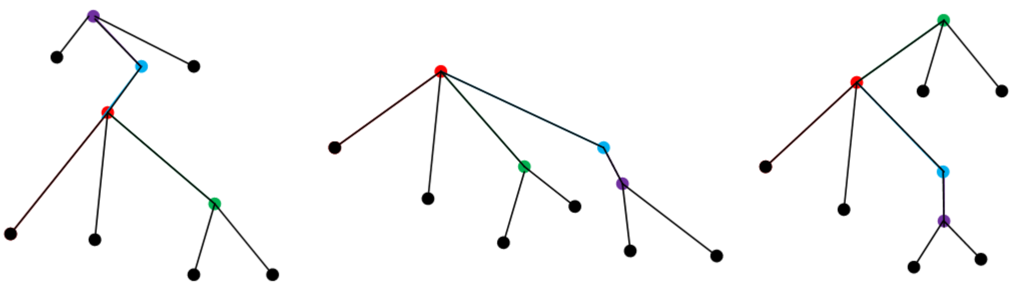
FIGURE 7 – graphe connexe

2.2 Les arbres

Définition 7

Un arbre est un graphe connexe acyclique (qui ne contient pas de cycle)
La figure 5 n'est pas un arbre car elle contient un cycle.
La figure 6 n'est pas un arbre car elle n'est pas connexe.
La figure 7 est un arbre. Mais il faut déterminer sa racine.

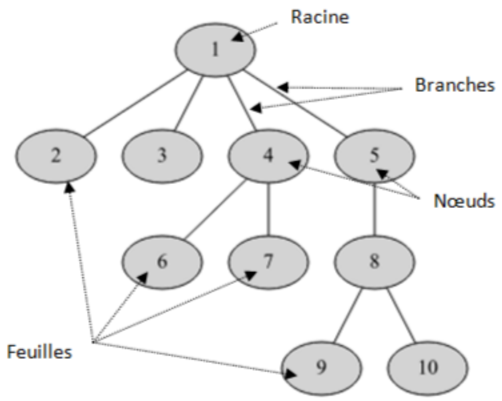
Lorsqu'un sommet se distingue des autres, on le nomme **racine** de l'arbre et celui-ci devient alors une arborescence (par la suite on utilisera le mot **arbre** pour une arborescence).
Les trois graphes suivants représentent le même graphe mais pas le même arbre car la racine n'est pas même. On remarque, qu'en générale, on représente la **racine** en haut et les **branches** qui descendent vers le bas.



2.3 Vocabulaire

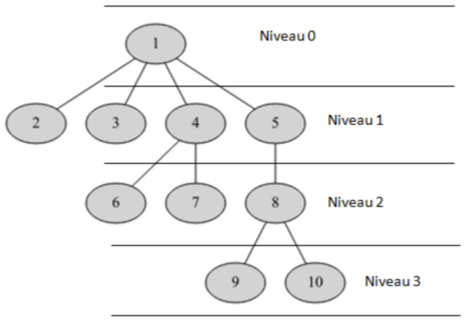
Définition 8 (Arbre, noeuds, pére, fils, feuille et branches)

Un **arbre** est un ensemble organisé de **noeuds** dans lequel chaque noeud a un **père**, sauf un **noeud** que l'on appelle la **racine**. Si le **noeud** n'a pas de **fils**, on dit que c'est une **feuille**. Les **noeuds** sont reliés par des **branches**.



Définition 9 (Hauteur ou profondeur d'un noeud , **première définition**)

La **hauteur** (ou profondeur ou niveau) d'un noeud X est égale au nombre d'arêtes qu'il faut parcourir à partir de la racine pour aller jusqu'au noeud X.



Définition 10 (Hauteur d'un arbre, **première définition**)

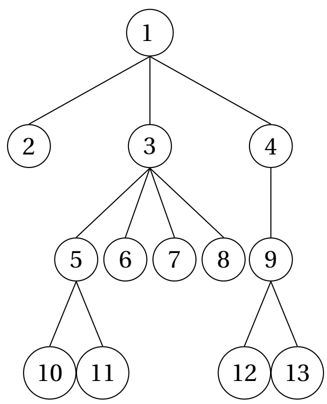
La **hauteur** (ou **profondeur**) d'un **arbre** est égale à la **profondeur** du noeud le plus profond.

- La hauteur d'un arbre réduit à un nœud, c'est-à-dire la racine, est 0.
- La hauteur d'un arbre vide est -1 (par convention).

Dans l'exemple précédent, un des chemins le plus long est $(1,3,5,10)$ donc la hauteur est 3. le noeud le plus profond est de **profondeur** 3, donc l'arbre est de **profondeur** 3.

Définition 11 (Hauteur ou profondeur d'un noeud, **deuxième définition**)

La hauteur d'un noeud N est le nombre de nœuds du chemin qui joint le nœud racine à ce noeud N.



- La hauteur du noeud racine est 1.
- La hauteur du noeud 4 est 2 .
- La hauteur du noeud 5 est 3 .
- La hauteur du noeud 12 est 4 .

Définition 12 (Hauteur d'un arbre, **deuxième définition**)

La **hauteur** (ou **profondeur**) d'un **arbre** est égale à la **profondeur** du noeud le plus profond.

- La hauteur d'un arbre réduit à un nœud, c'est-à-dire la racine, est 1.
- La hauteur d'un arbre vide est 0.

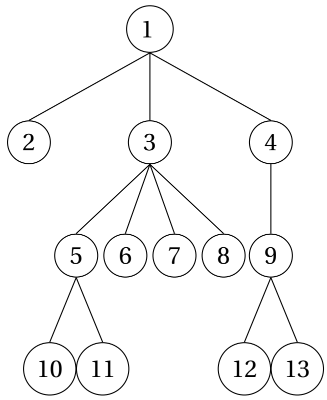
Dans l'exemple précédent, un des chemins le plus long est $(1,3,5,10)$ donc la hauteur est 4.

Définition 13 (Taille d'un arbre)

La **taille** d'un arbre est égale au nombre de noeuds de l'arbre.

Définition 14 (Degré d'un noeud et degré d'un arbre)

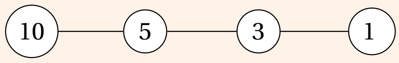
Le **degré** d'un noeud est égal au nombre de ses fils.
Le **degré** d'un arbre est égal au plus grand des degrés de ses noeud.
Le **degré** d'un arbre vide est égal à 0.



- Le noeud 1 est de degré 3.
- Le noeud 2 est de degré 0.
- Le noeud 3 de degré 4.
- Le noeud 4 de degré 1.
- ...
- Le noeud 3 est celui de plus grand degré donc l'arbre est de degré 4.

Remarque

Un arbre dont tous les noeuds n'ont qu'un seul fils est en fait une liste.



2.4 Exercices

Exercice 3

Déterminer les racines, profondeurs, hauteurs, tailles ,degrés et quelques feuilles, des arbres de l'introduction.

Racine	Profondeur	Hauteur	taille	Degré
Feuilles				

TABLE 1 – Arbre phylogénétique

Racine	Profondeur	Hauteur	taille	Degré
Feuilles				

TABLE 2 – Arbre Généalogique

Racine	Profondeur	Hauteur	taille	Degré
Feuilles				

TABLE 3 – Arbre Léxicographique

Racine	Profondeur	Hauteur	taille	Degré
Feuilles				

TABLE 4 – Arbre d'expression mathématique

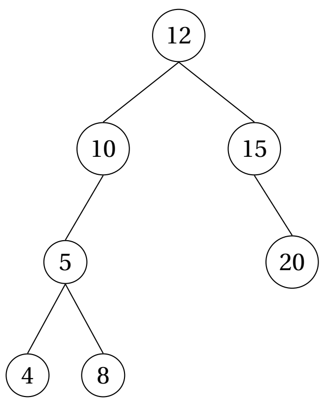
Exercice 4 , Arbre binaire sujet bac candidat libre 2 2021

Cet exercice porte sur les arbres binaires et la programmation orientée objet.

Un arbre binaire est composé de nœuds, chacun des nœuds possédant éventuellement un sous-arbre gauche et éventuellement un sous-arbre droit. Un nœud sans sous-arbre est appelé feuille. La taille d'un arbre est le nombre de nœuds qu'il contient ; sa hauteur est le nombre de nœuds du plus long chemin qui joint le nœud racine à l'une des feuilles. Ainsi la hauteur d'un arbre réduit à un nœud, c'est-à-dire la racine, est 1.
Dans un arbre binaire de recherche, chaque nœud contient une clé, ici un nombre entier, qui est :

- strictement supérieure à toutes les clés des nœuds du sous-arbre gauche ;
- strictement inférieure à toutes les clés des nœuds du sous-arbre droit.

Ainsi les clés de cet arbre sont toutes distinctes.
Un arbre binaire de recherche est dit « bien construit » s'il n'existe pas d'arbre de hauteur inférieure qui pourrait contenir tous ses nœuds
On considère l'arbre binaire de recherche ci-dessous.



1. (a) Quelle est la taille de l'arbre ci-dessus ?
   (b) Quelle est la hauteur de l'arbre ci-dessus ?
   (c) Représenter les sous arbres gauches et droits de chaque noeud qui en possèdent.
2. Cet arbre binaire de recherche n'est pas « bien construit ». Proposer un arbre binaire de recherche contenant les mêmes clés et dont la hauteur est plus petite que celle de l'arbre initial.
3. Les classes *Noeud* et *Arbre* ci-dessous permettent de mettre en œuvre en *Python* la structure d'arbre binaire de recherche. La méthode *insere* permet d'insérer récursivement une nouvelle clé.

```
class Noeud :
    def __init__ (self, cle):
        self.cle = cle
        self.gauche = None
        self.droit = None
    def insere(self, cle):
        if cle < self.cle :
            if self.gauche == None :
                self.gauche = Noeud(cle)
            else :
                self.gauche.insere(cle)
        elif cle > self.cle :
            if self.droit == None :
                self.droit = Noeud(cle)
            else :
                self.droit.insere(cle)
class Arbre :
    def __init__ (self, cle):
        self.racine = Noeud(cle)
    def insere(self, cle):
        self.racine.insere(cle)
```

Donner la représentation de l'arbre codé par les instructions ci-dessous.

```
a = Arbre(10)
a.insere(20)
a.insere(15)
a.insere(12)
a.insere(8)
a.insere(4)
a.insere(5)
```

4. Pour calculer la hauteur d'un arbre non vide, on a écrit la méthode ci-dessous dans la classe Noeud.

```
def hauteur(self):
    if self.gauche == None and self.droit == None:
        return 1
    if self.gauche == None:
        return 1+self.droit.hauteur()
    elif self.droit == None:
        return 1+self.gauche.hauteur()
    else:
        hg = self.gauche.hauteur()
        hd = self.droit.hauteur()
        if hg > hd:
            return hg+1
        else:
            return hd+1
```

   (a) Quelle est la méthode de programmation utilisée ? Quelle est la condition d'arrêt ?
   (b) Écrire la méthode *hauteur* de la classe *Arbre* qui renvoie la hauteur de l'arbre.

5. Écrire les méthodes *taille* des classes *Noeud* et *Arbre* permettant de calculer la taille d'un arbre.
6. On souhaite écrire une méthode *bien_construit* de la classe Arbre qui renvoie la valeur *True* si l'arbre est « bien construit » et *False* sinon.
   (a) Montrer que la taille maximale d'un arbre binaire de recherche de hauteur h est $2^h - 1$.
   (b) Quelle est la taille minimale, notée t_{min} d'un arbre binaire de recherche « bien construit » de hauteur h ?
   (c) Écrire la méthode *bien_construit* demandée.