Chapitre 1 : Les arbres binaires et tas

Informatique pour tous – MP

Structure arborescente : motivation

- arbres généalogiques
- tournois sportifs
- probabilités
- stockage d'expressions algébriques
- théorie des jeux...

Table des matières

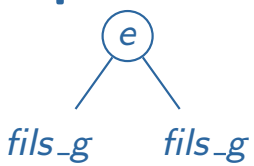
Définition récursive des arbres binaires

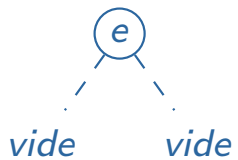
Définition : un arbre binaire étiqueté est

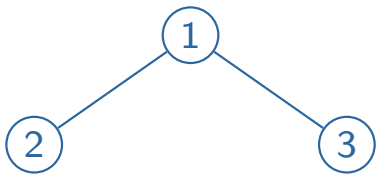
- soit
- soit de la forme
  - $e \in E$:
  - *fils_g*, *fils_d* :

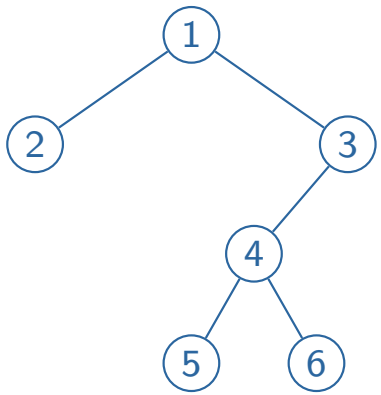
Rq : on peut définir des arbres avec plus de fils (liste).

Représentation graphique :

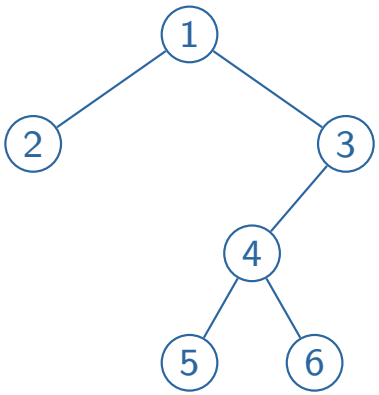




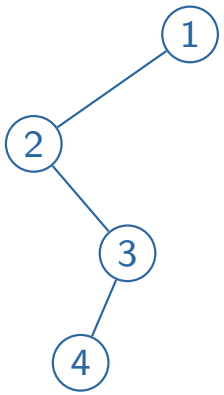




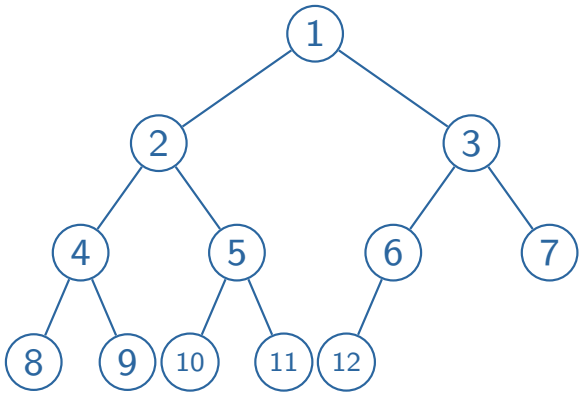
Définition récursive des arbres binaires



Exemple 1



Exemple 2



Exemple 3

Vocabulaire :

- *fils_g* et *fils_d* sont
- *a* est
- les triplets $[e, \mathit{fils\_g}, \mathit{fils\_d}]$
- le seul nœud qui n'a pas de père
- les nœuds avec fils vides

Notion de hauteur

Definition (Hauteur)

- La hauteur (ou profondeur, ou niveau) d'un nœud est définie par :

$$h(x) = \begin{cases} & \text{si } x \text{ est la racine} \\ & \text{si } y \text{ est le père de } x \end{cases}$$

- La hauteur d'un arbre est

Propriété : la hauteur h d'un arbre à n nœuds vérifie

Notion de hauteur

Propriété : la hauteur h d'un arbre à n nœuds vérifie

$$\lfloor \log_2 n \rfloor \leq h \leq n - 1$$

Preuve

- majoration de h :
  -
  -
  -
  -
- minoration de h :
  -

    (I)
    (H)

    (C) $(\mathcal{P}_k)$ est vrai pour tout $k \in \mathbb{N}$
  -

    Ainsi

Arbres parfaits

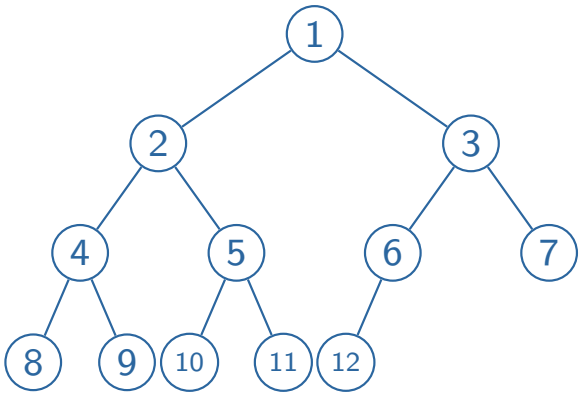
Définition : un arbre est **parfait** :

- si tous ses niveaux
- sauf éventuellement
- le dernier niveau est

Propriété :

la hauteur d'un arbre parfait à n nœuds est

Exemple 3 :



Primitives du type abstrait `arbre binaire`

- `est_vide(a)` :
- `arbre_vide()` :
- `cons(e,g,d)` :

- `contenu(a)` :
- `f_g(a)` (*resp.* `f_d(a)`) :

Implémentation par une liste

En Python, on peut simplement utiliser la structure de liste :

$$\texttt{cons}(\texttt{e}, \texttt{g}, \texttt{d}) =$$

- **Exemple 1 :**
- **Exemple 2 :**
- **Exemple 3 :**

Représentation par un tableau

ID : stocker dans un tableau `t` les étiquettes

Mise en œuvre

- on suppose les nœuds numérotés :
  - de haut en bas
  - de gauche à droite
  - à partir de 1
- dans la case `t[k]`,

Conséquences

- niveau `i` :
- fils de `t[k]` :
- père de `t[k]` :

Représentation par un tableau : exemple

- **Exemple 1 :**
- **Exemple 2 :**
- **Exemple 3 :**

Conclusion

- Adapté aux arbres parfaits (donc aux tas...)
- sinon :

Définition des tas

Definition (Tas)

Un **tas** est un arbre binaire :

- ▶
- ▶

Conséquence :

Exemple : $[-1, 3, 5, 9, 6, 8, 11, 10, 12, 18, 14]$ est un tas

Plus clair comme ça :

Point CG : introduit par J.W.J. Williams en 1964 pour expliciter son algorithme de tri par tas...(sujet du TP)

Affichage des arbres

Objectif

Indications :

- `' '*n` construit une chaîne de `n` espaces
- `'{: 3}'.format(e)` crée une chaîne de 3 caractères où `e` est centré