CH14 :
Arbres binaires

T^ale NSI

21 novembre 2023

Généralités

Arbres

La structure d'arbre n'est pas linéaire : les éléments n'y sont pas rangés les uns à la suite des autres.

Arbres

La structure d'arbre n'est pas linéaire : les éléments n'y sont pas rangés les uns à la suite des autres.
C'est une structure hiérarchique : les éléments y sont organisés en *niveaux*.

Arbres

La structure d'arbre n'est pas linéaire : les éléments n'y sont pas rangés les uns à la suite des autres.
C'est une structure hiérarchique : les éléments y sont organisés en *niveaux*.

Dans ce chapitre nous étudions les arbres binaires.

Définition : arbre

Un arbre binaire peut être vide.

Définition : arbre

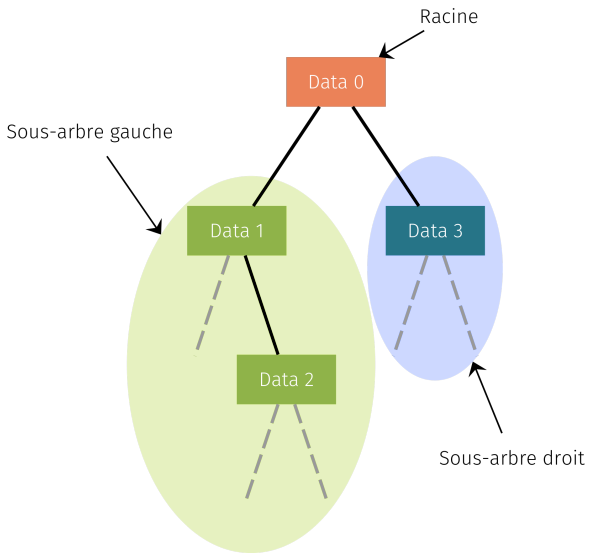
Un arbre binaire peut être vide.
Sinon, il est composé d'au moins un nœud, appelé racine et le reste des nœuds peut être partagé en 2 sous-ensembles :

Définition : arbre

Un arbre binaire peut être vide.
Sinon, il est composé d'au moins un nœud, appelé racine et le reste des nœuds peut être partagé en 2 sous-ensembles : le sous-arbre gauche et le sous-arbre droit.

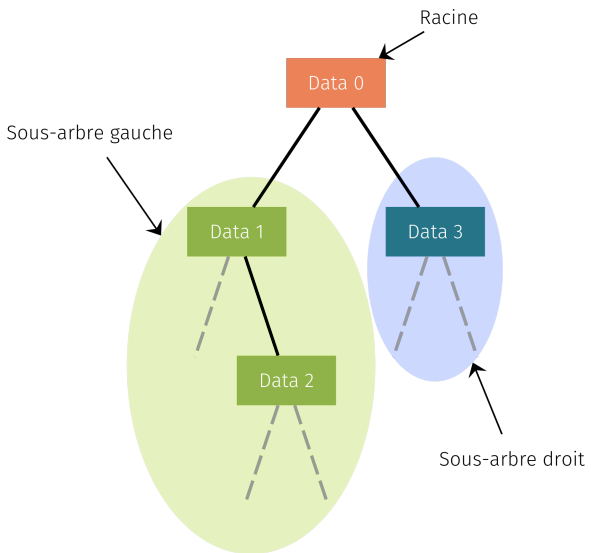
Définition : arbre

Un arbre binaire peut être vide.
Sinon, il est composé d'au moins un nœud, appelé racine et le reste des nœuds peut être partagé en 2 sous-ensembles : le sous-arbre gauche et le sous-arbre droit.



Définition : arbre

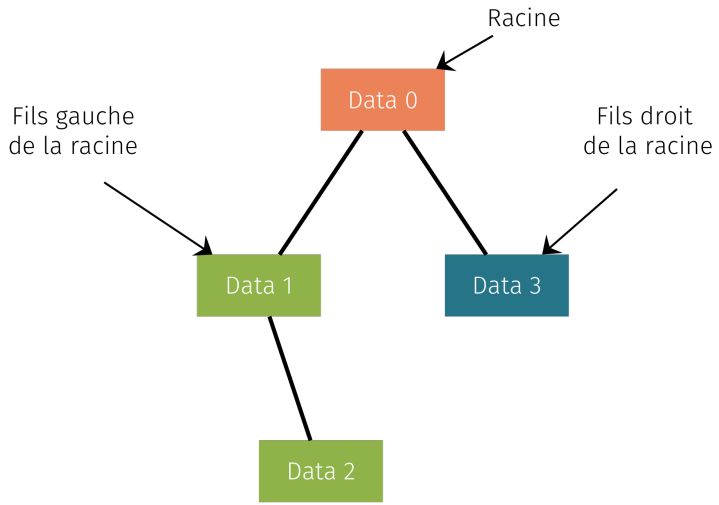
Un arbre binaire peut être vide.
Sinon, il est composé d'au moins un nœud, appelé racine et le reste des nœuds peut être partagé en 2 sous-ensembles : le sous-arbre gauche et le sous-arbre droit.



Un nœud possède toujours un sous-arbre droit et un sous-arbre gauche, même si ceux-ci peuvent être vides (segments gris en pointillés).

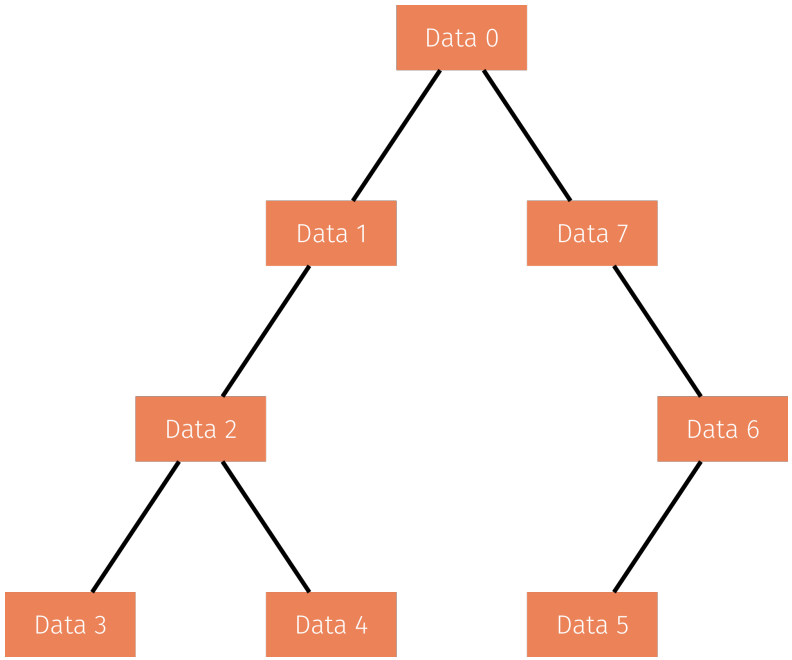
Définition

Quand le sous-arbre gauche d'un nœud est non-vide, sa racine est appelée fils gauche du nœud. On définit de même la notion de fils droit



Taille

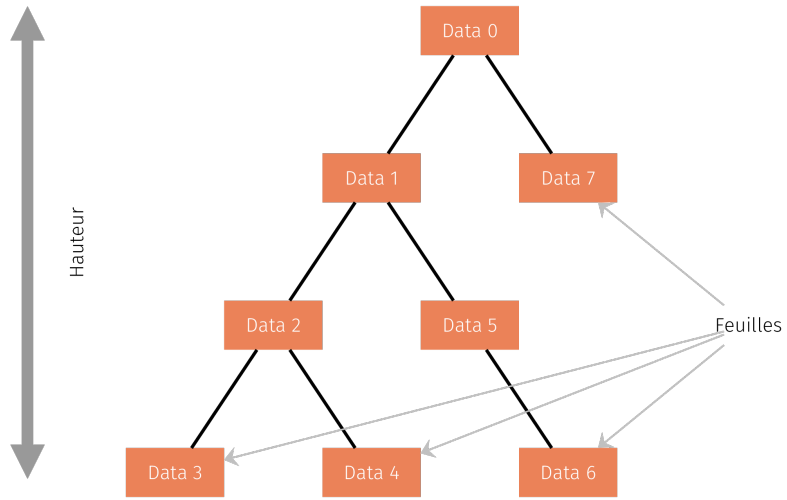
La taille d'un arbre est le nombre de ses nœuds.



Voici un arbre de taille 8.

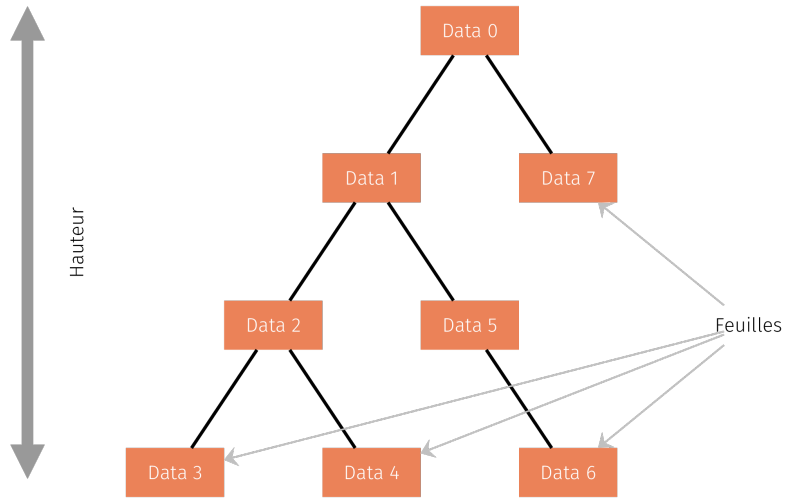
Hauteur et feuilles

On appelle feuille tout nœud qui n'a ni fils droit ni fils gauche.



Hauteur et feuilles

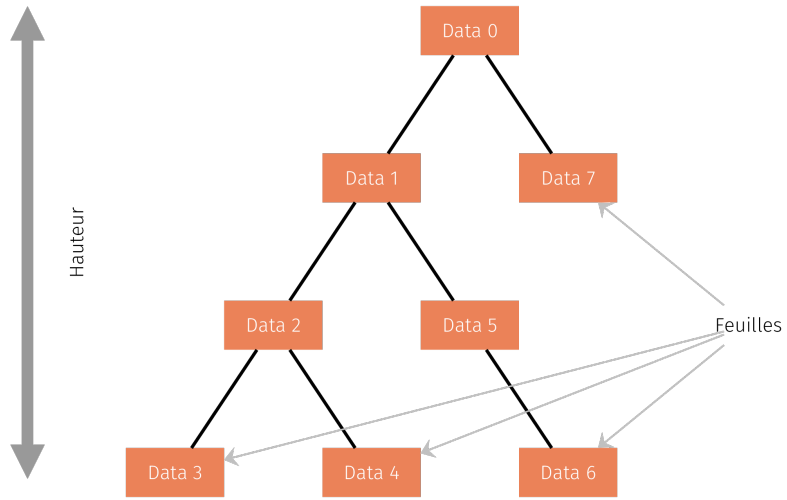
On appelle feuille tout nœud qui n'a ni fils droit ni fils gauche.



La hauteur d'un nœud est le nombre d'arêtes qui mène de la racine à ce nœud).

Hauteur et feuilles

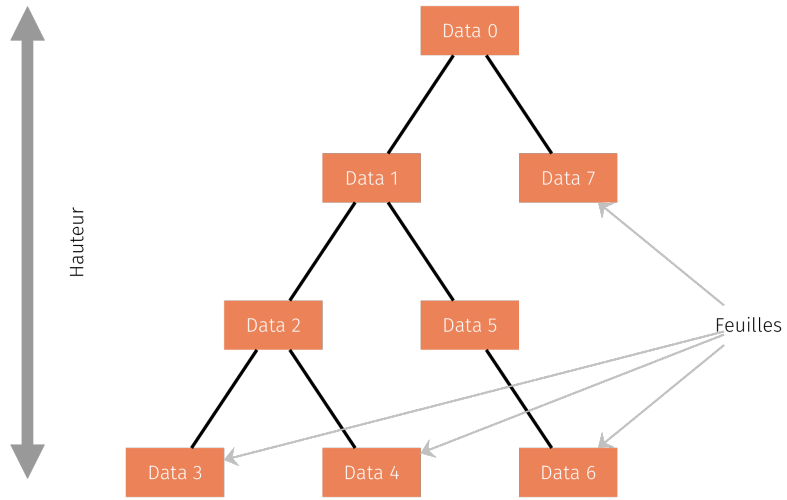
On appelle feuille tout nœud qui n'a ni fils droit ni fils gauche.



La hauteur d'un nœud est le nombre d'arêtes qui mène de la racine à ce nœud).
La hauteur de l'arbre est la plus grande des hauteurs des nœuds qui le composent.

Hauteur et feuilles

On appelle feuille tout nœud qui n'a ni fils droit ni fils gauche.



La hauteur d'un nœud est le nombre d'arêtes qui mène de la racine à ce nœud).
La hauteur de l'arbre est la plus grande des hauteurs des nœuds qui le composent.
Cet arbre est de hauteur 3.

Remarque

La définition de hauteur n'est pas standard. Selon celle-ci la hauteur de l'arbre vide n'est pas définie.

Remarque

La définition de hauteur n'est pas standard. Selon celle-ci la hauteur de l'arbre vide n'est pas définie.

On choisit parfois de définir la hauteur d'un nœud comme le nombre de nœuds pour aller jusqu'à la racine incluse, et on fixe la hauteur de l'arbre vide à zéro.

Cela donne une hauteur qui est supérieure d'une unité par rapport à celle choisie dans ce cours.

Remarque

La définition de hauteur n'est pas standard. Selon celle-ci la hauteur de l'arbre vide n'est pas définie.

On choisit parfois de définir la hauteur d'un nœud comme le nombre de nœuds pour aller jusqu'à la racine incluse, et on fixe la hauteur de l'arbre vide à zéro.

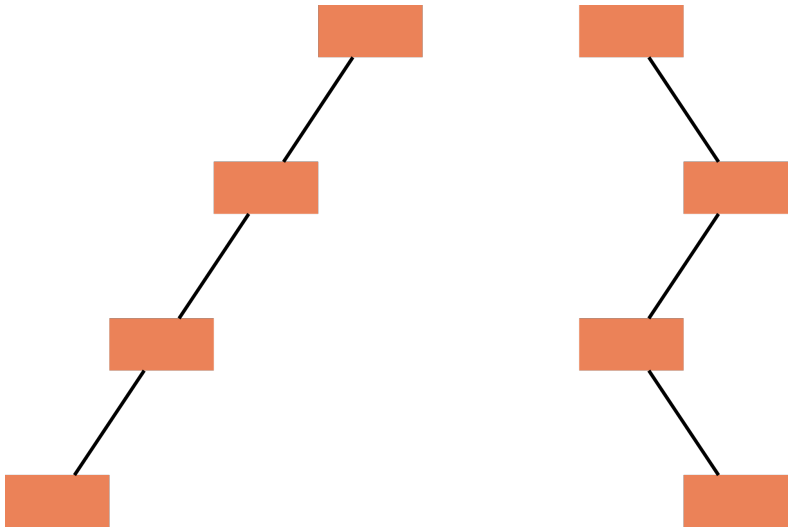
Cela donne une hauteur qui est supérieure d'une unité par rapport à celle choisie dans ce cours.

Nœud interne

Un nœud qui n'est ni la racine ni une feuille est dit interne.

Nœud interne

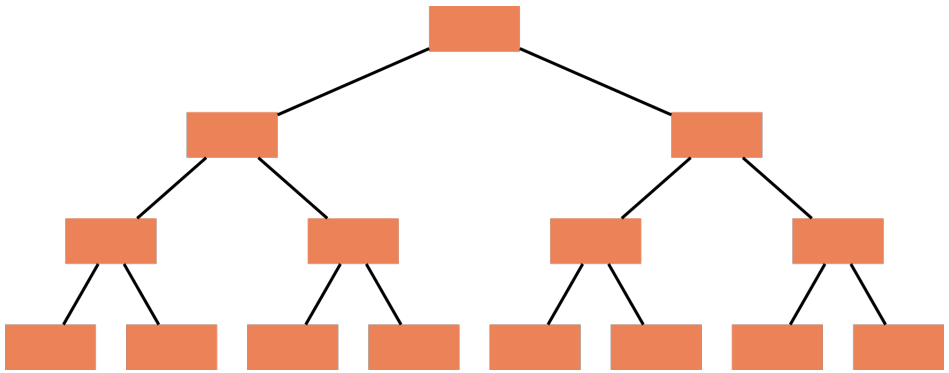
Un nœud qui n'est ni la racine ni une feuille est dit interne.



Si tous les nœuds internes n'ont qu'un seul fils, l'arbre est dit dégénéré.

Arbres parfaits

Si tous les nœuds internes ont 2 fils et que toutes les feuilles sont à la même hauteur, on dit que l'arbre binaire est parfait.



Structure de données

Modèle théorique de l'arbre

- *arbre_vide()* : crée un arbre binaire vide

Modèle théorique de l'arbre

- *arbre_vide()* : crée un arbre binaire vide
- *racine(arbre)* : renvoie le nœud qui est la racine de l'arbre

Modèle théorique de l'arbre

- *arbre_vide()* : crée un arbre binaire vide
- *racine(arbre)* : renvoie le nœud qui est la racine de l'arbre
- *gauche(arbre)* : renvoie le sous-arbre gauche de l'arbre

Modèle théorique de l'arbre

- *arbre_vide()* : crée un arbre binaire vide
- *racine(arbre)* : renvoie le nœud qui est la racine de l'arbre
- *gauche(arbre)* : renvoie le sous-arbre gauche de l'arbre
- *droit(arbre)* : renvoie le sous-arbre droit de l'arbre

Modèle théorique de l'arbre

- *arbre_vide()* : crée un arbre binaire vide
- *racine(arbre)* : renvoie le nœud qui est la racine de l'arbre
- *gauche(arbre)* : renvoie le sous-arbre gauche de l'arbre
- *droit(arbre)* : renvoie le sous-arbre droit de l'arbre
- *contenu(nœud)* : renvoie l'élément stocké dans le nœud

Modèle théorique de l'arbre

- *arbre_vide()* : crée un arbre binaire vide
- *racine(arbre)* : renvoie le nœud qui est la racine de l'arbre
- *gauche(arbre)* : renvoie le sous-arbre gauche de l'arbre
- *droit(arbre)* : renvoie le sous-arbre droit de l'arbre
- *contenu(nœud)* : renvoie l'élément stocké dans le nœud

On ne suivra pas complètement ce modèle théorique.

Implémentation en Python

```python
class Node:
    def __init__(self, value, left=None, right=None):
        self.value = value
        self.left = left  # left child
        self.right = right  # right child
```

Implémentation en Python

```python
class Node:
    def __init__(self, value, left=None, right=None):
        self.value = value
        self.left = left  # left child
        self.right = right  # right child
```

Dans ce modèle on ne définit que la classe `Node`.

Implémentation en Python

```python
class Node:
    def __init__(self, value, left=None, right=None):
        self.value = value
        self.left = left  # left child
        self.right = right  # right child
```

Dans ce modèle on ne définit que la classe `Node`.
`Node(2)` renvoie un nœud sans fils droit ni fils gauche contenant la valeur 2.

Exemple

Ce code :

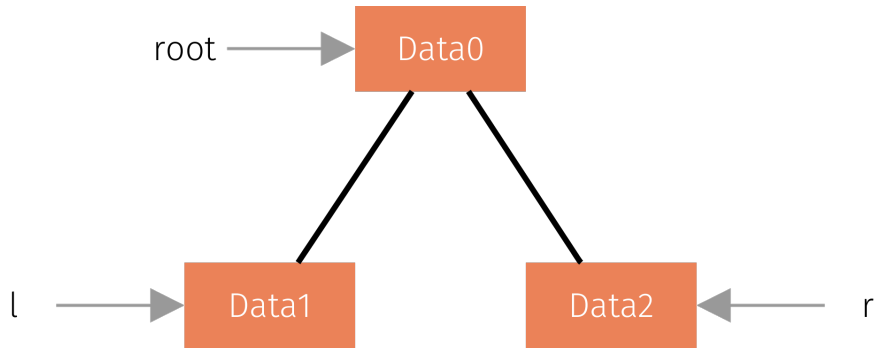
```
>>> l = Node('Data1')
>>> r = Node('Data2')
>>> root = Node('Data0', l, r)
```

Exemple

Ce code :

```
>>> l = Node('Data1')
>>> r = Node('Data2')
>>> root = Node('Data0', l, r)
```

Produit ceci :



Remarque

Une instance de la classe `Node` contient

- une valeur ;
- 2 références à 2 autres instances de la classe `Node` (ou bien `None`).

Remarque

Une instance de la classe `Node` contient

- une valeur ;
- 2 références à 2 autres instances de la classe `Node` (ou bien `None`).

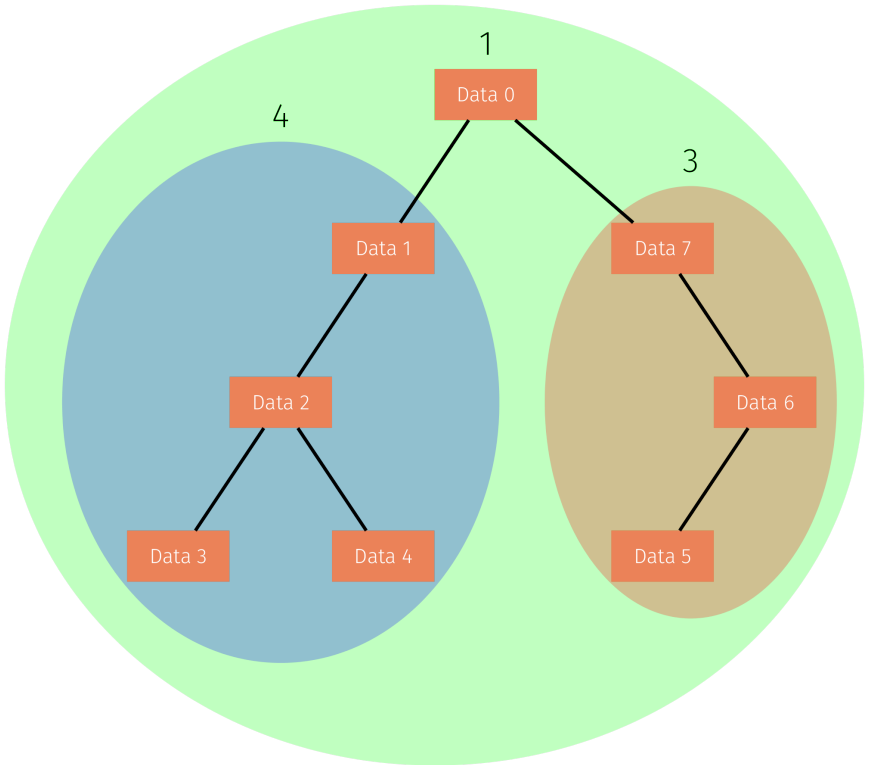
Tout comme les listes chaînées, les arbres sont des structures qui se prêtent bien à la récursivité.

Exemple

La taille de l'arbre vide est 0, sinon c'est « 1 plus la taille du sous-arbre gauche plus la taille du sous-arbre droit».

Exemple

La taille de l'arbre vide est 0, sinon c'est « 1 plus la taille du sous-arbre gauche plus la taille du sous-arbre droit».



La taille de cet arbre est 1 + 4 + 3 = 8.

Parcours

Différents parcours

Pour parcourir un arbre binaire, on peut utiliser diverses méthodes récursives.
Parmi celles-ci, il en existe qui consistent à

- traiter le nœud courant ;
- parcourir le sous arbre gauche s'il est non vide ;
- parcourir le sous-arbre droit s'il est non vide.

Et ceci dans un ordre donné.
Si on choisit de toujours parcourir le sous-arbre gauche avant le droit, cela nous donne 3 méthodes.

préfixe, infixe, postfixe

- **Parcours préfixe :** Traiter d'abord le nœud courant puis ensuite parcourir le sous-arbre gauche, puis le droit.

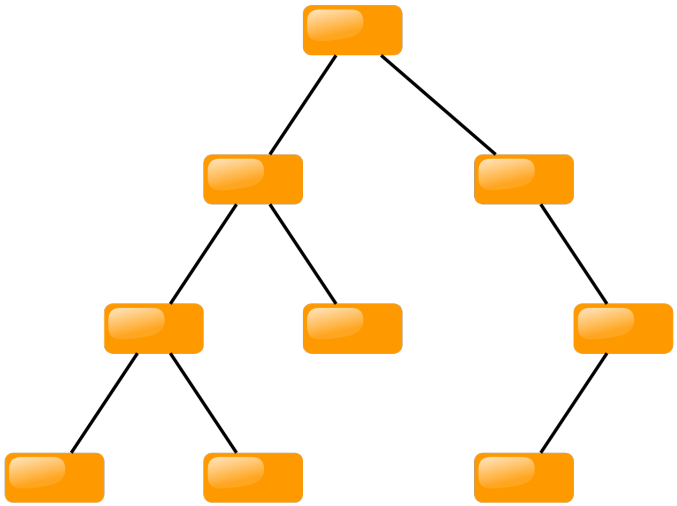
préfixe, infixe, postfixe

- **Parcours préfixe :** Traiter d'abord le nœud courant puis ensuite parcourir le sous-arbre gauche, puis le droit.
- **Parcours infixe :** Parcourir d'abord le sous-arbre gauche, puis traiter le nœud courant et ensuite parcourir le sous-arbre le droit.

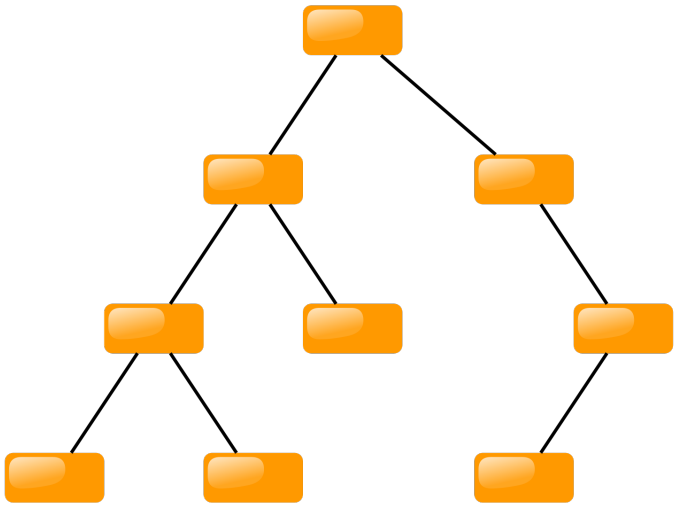
préfixe, infixe, postfixe

- **Parcours préfixe :** Traiter d'abord le nœud courant puis ensuite parcourir le sous-arbre gauche, puis le droit.
- **Parcours infixe :** Parcourir d'abord le sous-arbre gauche, puis traiter le nœud courant et ensuite parcourir le sous-arbre le droit.
- **Parcours postfixe :** Parcourir d'abord les sous-arbres gauche et droit puis traiter le nœud courant.

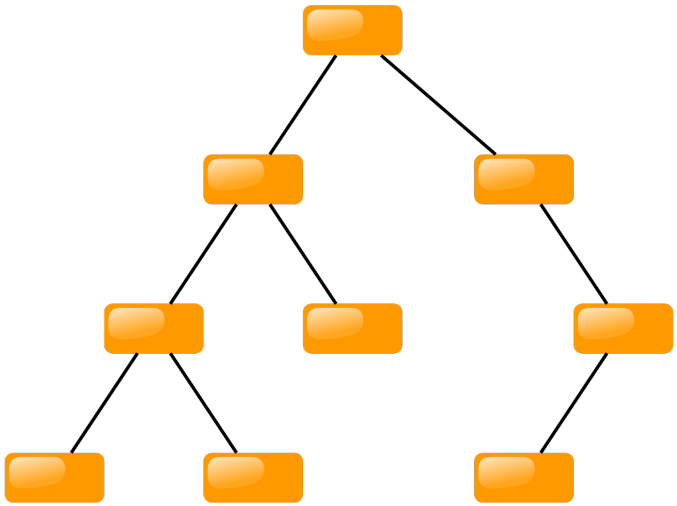
Parcours préfixe



Parcours infixe



Parcours postfixe



Temporary page!

LaTeX was unable to guess the total number of pages correctly. As there was some unprocessed data that should have been added t the final page this extra page has been added to receive it.

If you rerun the document (without altering it) this surplus page w go away, because LaTeX now knows how many pages to expect for th document.