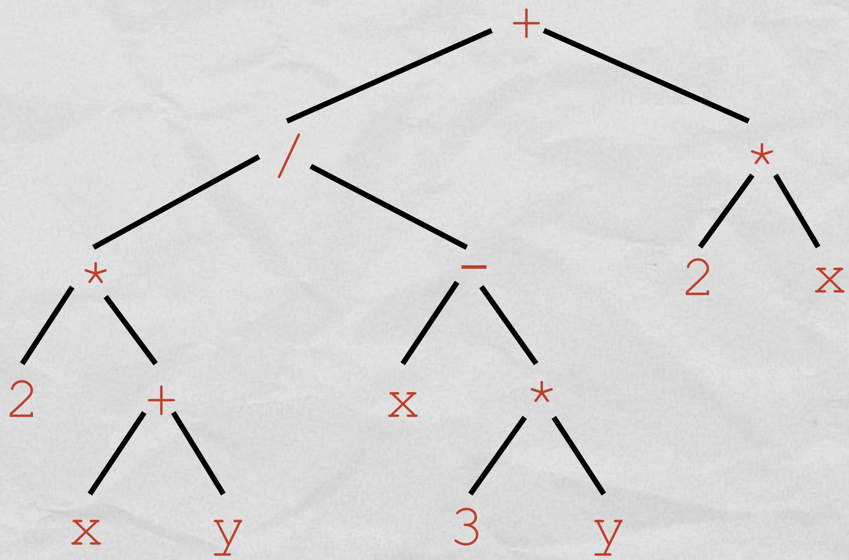
INTRODUCTION À L'ALGORITHMIQUE

LES ARBRES

Chargée de cours: Lélia Blin
Transparents:http://www-npa.lip6.fr/~blin/Enseignements.html
Email: lelia.blin@lip6.fr

LES ARBRES

- Structures les plus importantes et les plus utilisées en informatique
- Liste = cas dégénéré d'arbre
- Exemples:
  - Arbres généalogiques
  - Arbres de classification
  - Arbres d'expression



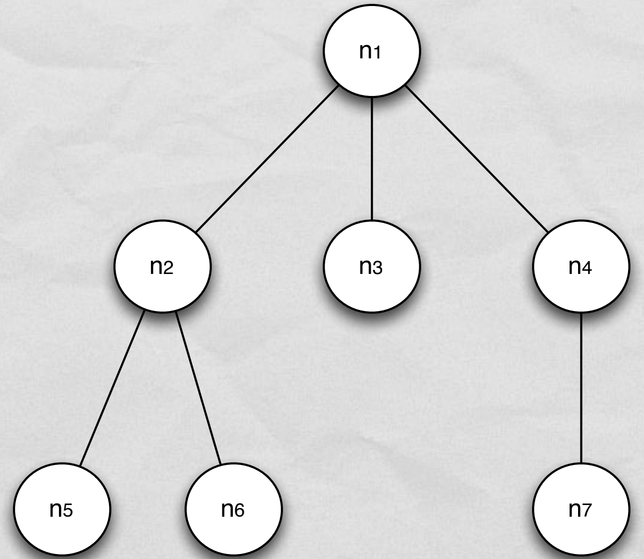
Traduction de l'expression

(2 * (x + y)) / (x – 3 * y) + 2 * x

TERMINOLOGIE

- Un *arbre* :
  - Ensemble de nœuds reliés entre eux par des arcs
- 3 propriétés pour les arbres **enracinés**:
  - Nœud particulier nommé *racine*
  - Tout nœud c autre que la racine est relié par un arc à un nœud p appelé *parent de* c
  - Un arbre est *connexe*

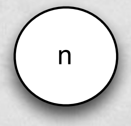
TERMINOLOGIE



- Un nœud peut avoir 0 ou plusieurs enfants
- Un nœud (sauf la racine) a exactement 1 parent

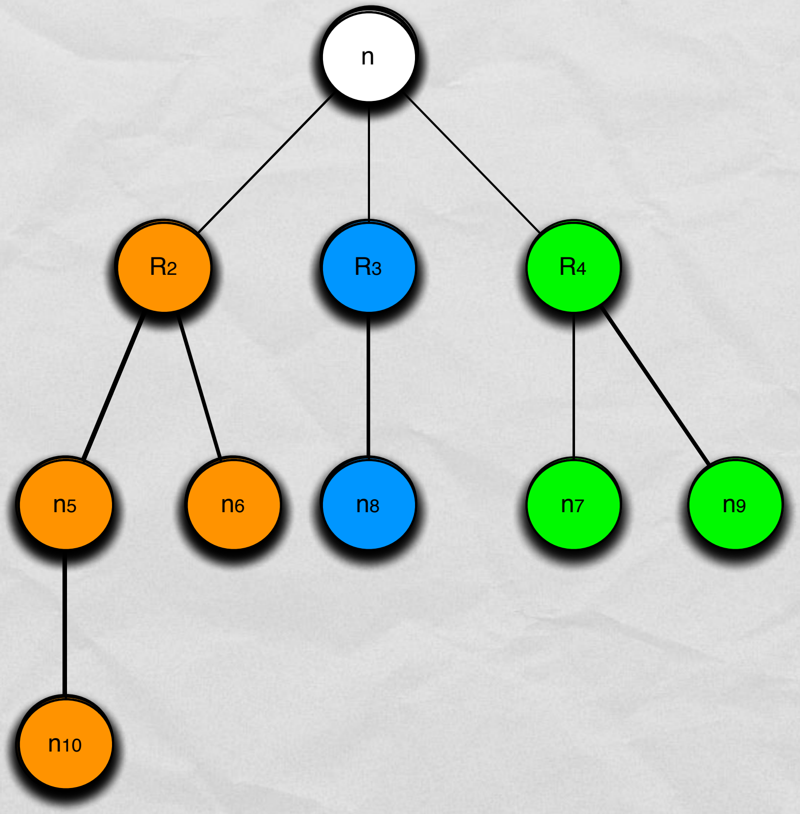
DÉFINITION RÉCURSIVE

- *Base :*
  - un nœud unique n est un arbre
  - n est la racine de l'arbre



- *Récurrence :*
  - Soit r un nouveau nœud
  - T_1, T_2, …, T_k sont des arbres ayant pour racine r_1, r_2, …, r_k.
  - Nouvel arbre a pour racine r et on ajoute un arc entre r et r_1, r et r_2, .., r et r_k.

DÉFINITION RÉCURSIVE : EXEMPLE



CHEMINS, ANCÊTRES, DESCENDANTS ...

- Les *ancêtres* d'un nœud :
  - Nœuds trouvés sur le *chemin unique* entre ce nœud et la racine
- Le nœud d est un *descendant* de a si et seulement si a est un ancêtre de d.
- Soit m_1, m_2, m_3, ..., m_k une séquence de nœuds :
  - *Longueur du chemin* = nombre d'arcs parcourus (k-1)

GÉNÉALOGIE

- La racine est un ancêtre de tous les nœuds
- Chaque nœud est un descendant de la racine
- Les nœuds ayant le même parent = *frères*
- Un nœud n et tous ses descendants = *sous-arbre*

FEUILLES ET NŒUDS INTÉRIEURS

- Une *feuille* est nœud qui n'a pas de enfants
- Un *nœud intérieur* est un nœud qui a au moins 1 enfant
- Tout nœud de l'arbre est :
  - Soit une feuille
  - Soit un nœud intérieur

HAUTEUR (PROFONDEUR)

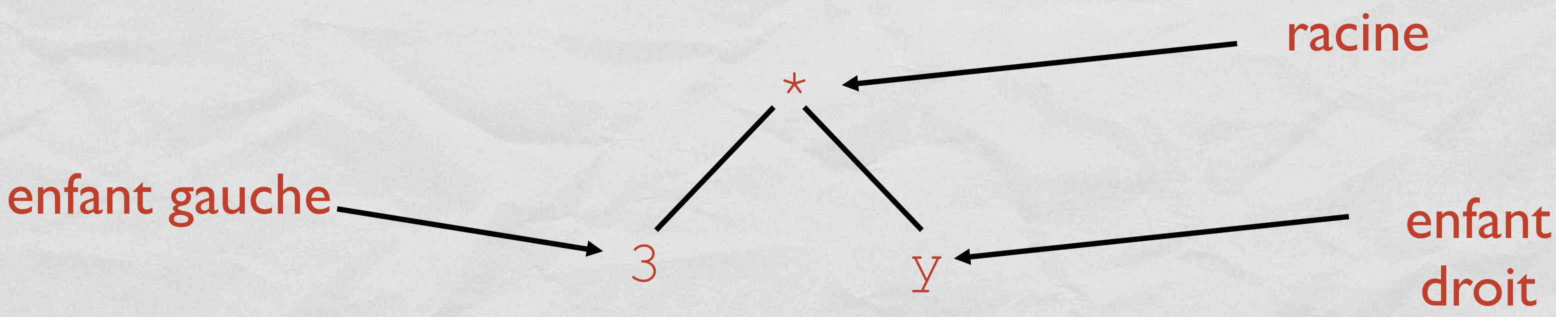
- La *hauteur d'un nœud* n est la longueur du plus long chemin depuis la racine jusqu'à n.
- La *hauteur d'un arbre* :

$$\max \{h(x),\ x \text{ nœud de l'arbre}\}$$

LES ARBRES BINAIRES

- Etude d'une classe particulière
- Propriétés
- Algorithmes

LES ARBRES BINAIRES



Soit un arbre $A = \langle o, A_1, A_2 \rangle$

- *racine* de A, le nœud o
- *sous-arbre gauche* de A (ou o), l'arbre A_1
- *sous-arbre droit* de A (ou o), l'arbre A_2

DÉFINITIONS

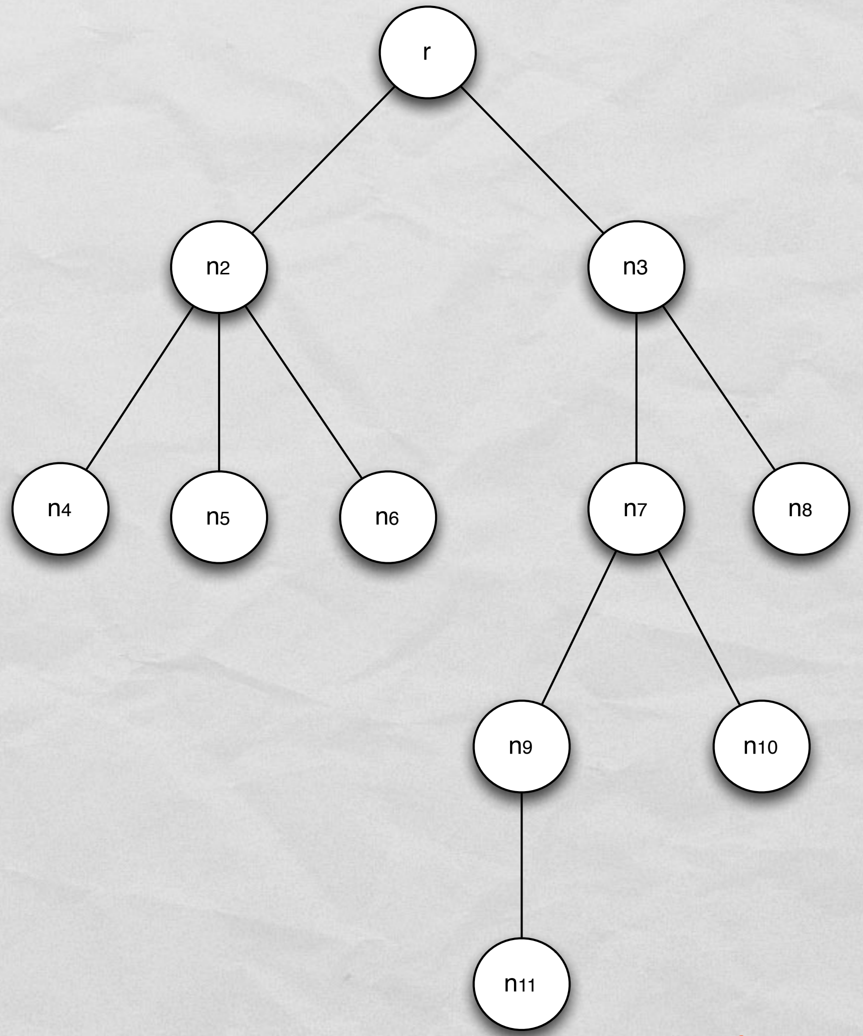
- ***enfant gauche*** de n = racine de son sous-arbre gauche
- ***enfant droit*** de n = racine de son sous-arbre droit
- ***bord gauche*** de l'arbre = le plus long chemin depuis la racine en ne suivant que les enfants gauches
- ***bord droit*** de l'arbre = le plus long chemin depuis la racine en ne suivant que les enfants droits

MESURES SUR LES ARBRES

- ***taille*** de l'arbre = le nombre de ses nœuds, noté ***taille(A)***
- ***nombre de feuilles***, noté ***nf(A)***

EXEMPLE

- ***Taille de l'arbre:***
  - Taille(A) = 11
- ***Nombre de feuille:***
  - nf(A)=6
- ***Hauteur de l'arbre***
  - h(A)=4
- ***Hauteur de noeud:***
  - $h(n_9)=3$



MESURES SUR LES ARBRES (2)

- ***longueur de cheminement*** de l'arbre = somme des longueurs de tous les chemins issus de la racine

  LC(A) = ∑ {h(x) | x nœud de A}

- ***longueur de cheminement externe*** de l'arbre = somme des longueurs de tous les chemins aboutissant à une feuille issus de la racine

  LCE(A) = ∑ {h(x) | x feuille de A}

MESURES SUR LES ARBRES (3)

- ***profondeur moyenne*** (d'un nœud) de l'arbre = moyenne des hauteurs de tous les nœuds

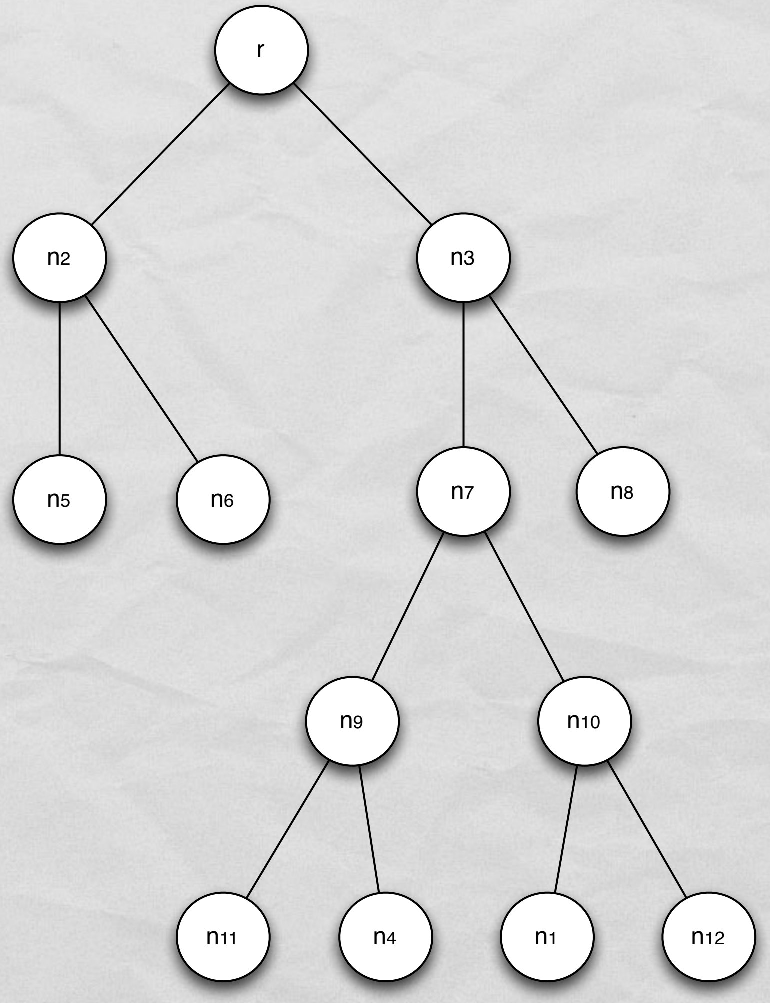
  PC(A) = LC(A) / taille(A)

- ***profondeur moyenne externe*** (d'une feuille) de l'arbre = moyenne des longueurs de toutes les branches

  PCE(A) = LCE(A) / nf(A)

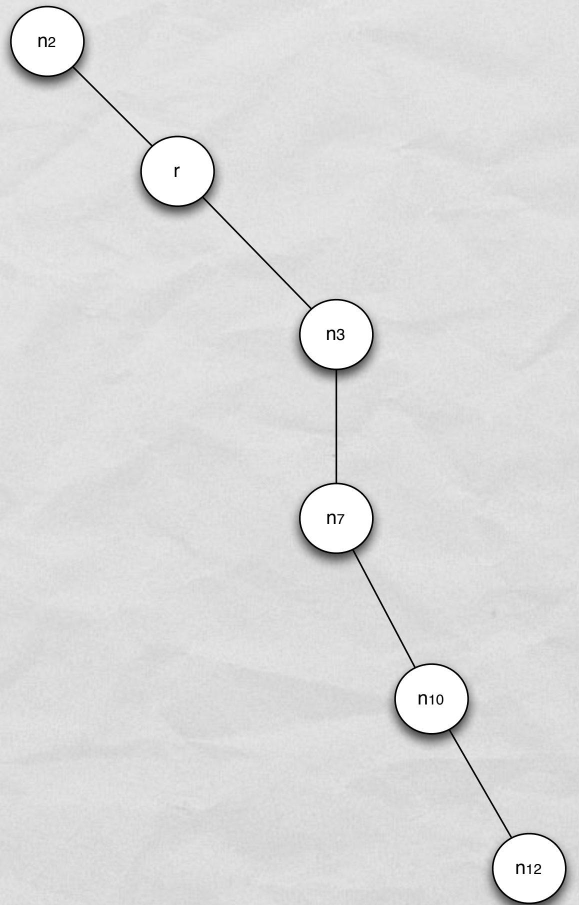
EXEMPLE : ARBRE BINAIRE

- ***Taille de l'arbre:***
  - Taille(A) = 13
- ***Nombre de feuille:***
  - nf(A)=7
- ***Longueur de cheminement:***
  - 32
- ***Profondeur moyenne:***
  - *2,46*
- ***Longueur de cheminement externe:***
  - *22*
- ***Longueur de cheminement externe:***
  - *3,14*



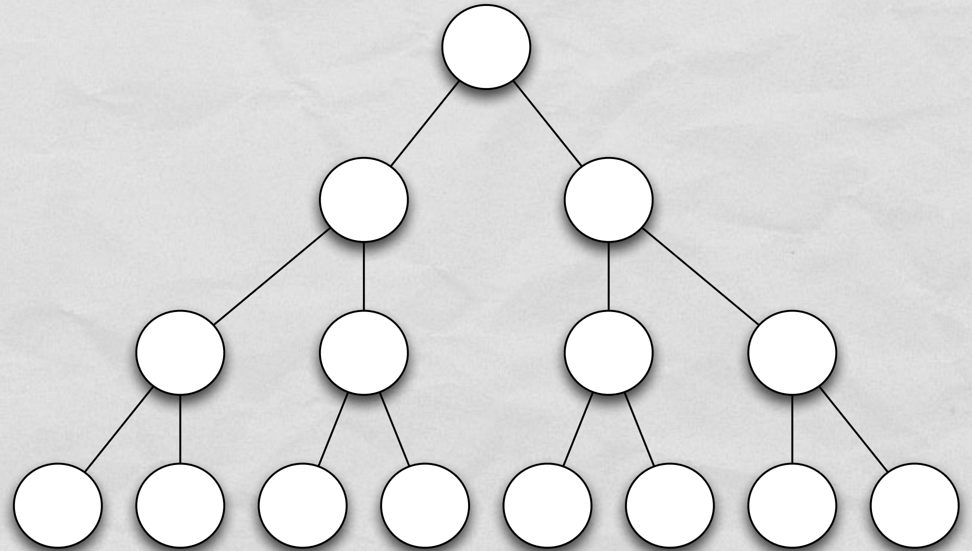
QUELQUES ARBRES BINAIRES PARTICULIERS

- Arbre binaire dégénéré ou filiforme



QUELQUES ARBRES BINAIRES PARTICULIERS

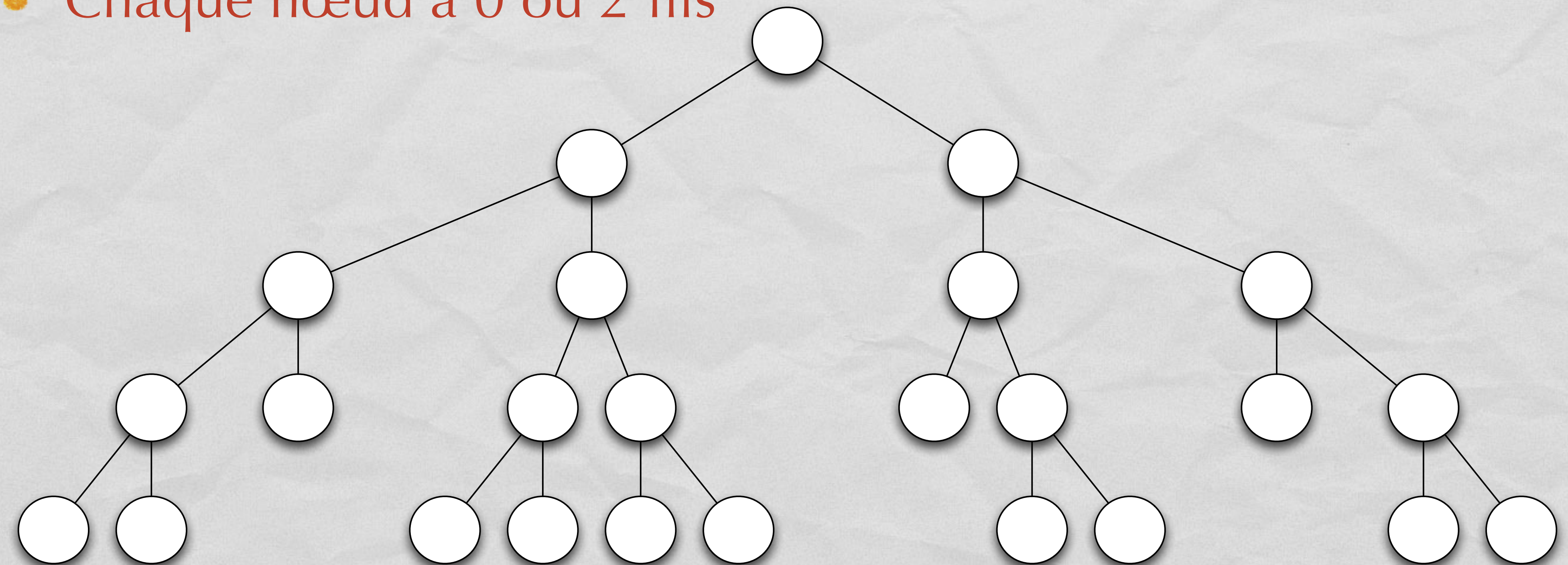
- Arbre binaire complet :
  - 1 nœud à la hauteur 0
  - 2 nœuds à la hauteur 1
  - 4 nœuds à la hauteur 2
  - …
  - Nombre total de nœuds :



$$1 + 2 + 2^2 + \ldots + 2^h = 2^{h+1} - 1$$

QUELQUES ARBRES BINAIRES PARTICULIERS

- Un arbre binaire **localement complet** :
  - Chaque nœud a 0 ou 2 fils



PROPRIÉTÉS SUR LES ARBRES (1)

Lemme : $h(A) \leq$ taille$(A) - 1$

- égalité pour un arbre dégénéré

Lemme : Un arbre binaire ayant n nœuds au total et de hauteur h :

$$\lfloor \mathrm{Log}_2\, n \rfloor \leq h \leq n-1$$

- Arbre filiforme : arbre ayant la plus grande hauteur
- $n = h + 1$ (seconde inégalité)
- Arbre complet : arbre ayant la plus petite hauteur possible : $n = 2^{h+1} - 1$ (première inégalité)

PROPRIÉTÉS

Corollaire : tout arbre binaire non vide B ayant f feuilles a une hauteur h(B) supérieure ou égale à $\lceil \log_2 f \rceil$.

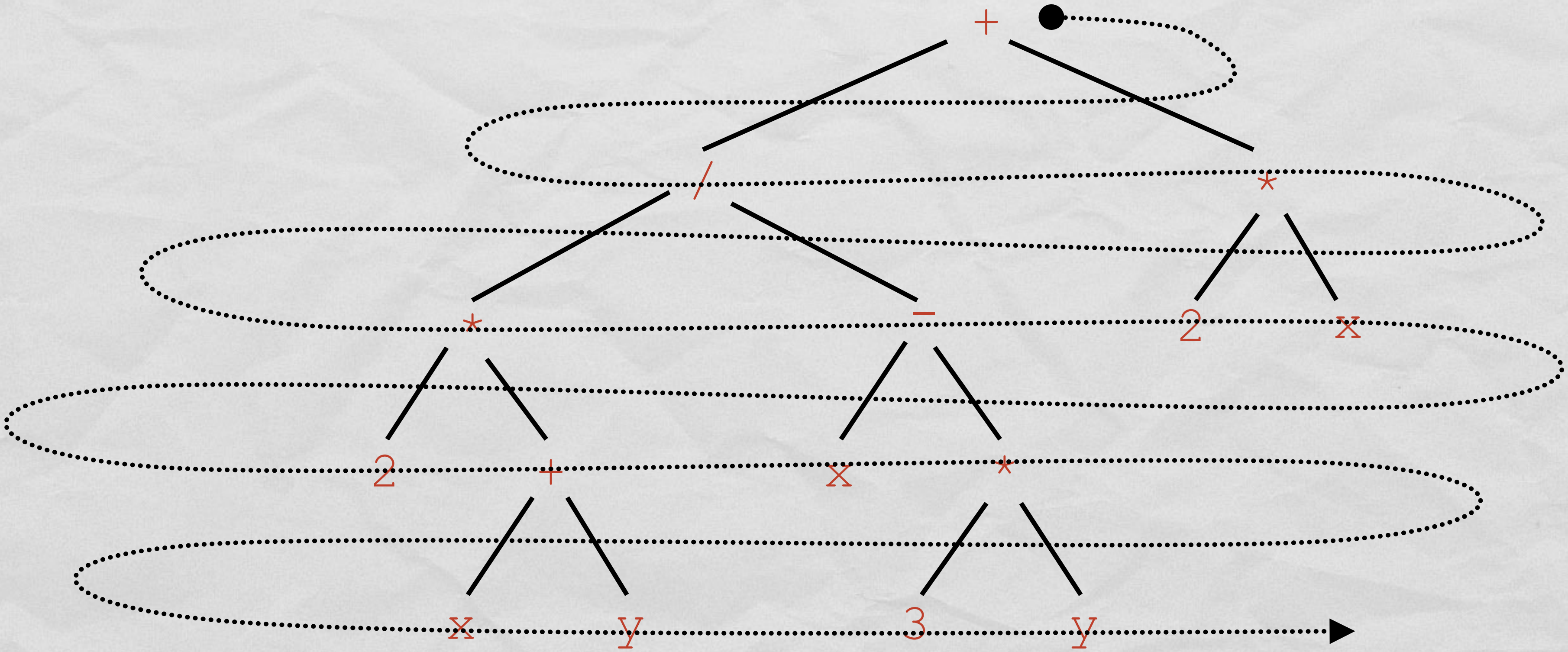
PROPRIÉTÉS

Lemme : un arbre binaire localement complet ayant n nœuds internes a (n+1) feuilles

EXPLORATION

- pas aussi simple que dans le cas des listes
- pas d'ordre « naturel »
- Deux types de parcours
    - En largeur d'abord
    - En profondeur d'abord
        - trois types principaux de traitement
            - préfixé
            - infixé
            - postfixé

PARCOURS EN LARGEUR D'ABORD (1)

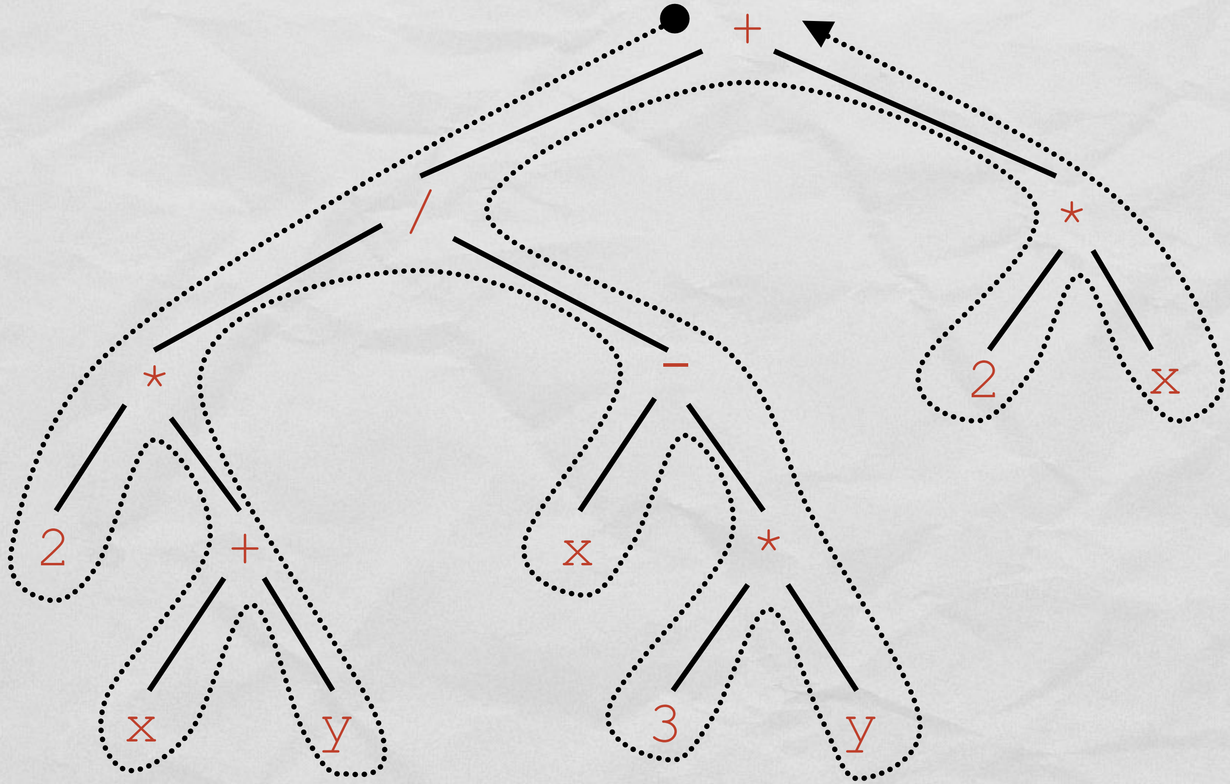


- Ordre d'évaluation des noeuds :
  - + / * * - 2 x 2 + x * x y 3 y

PARCOURS EN LARGEUR D'ABORD (2)

```
ParcourirEnLargeur(a : Noeud)
début
  ce_niveau = {a}
  tantque ce_niveau est non vide {
    niveau_inférieur = {}
    pour chaque nœud o de ce_niveau faire
      traiter o
      niveau_inférieur =
        niveau_inférieur ∪ les enfants de o
    fin pour
    ce_niveau = niveau_inférieur
  fintq
fin
```

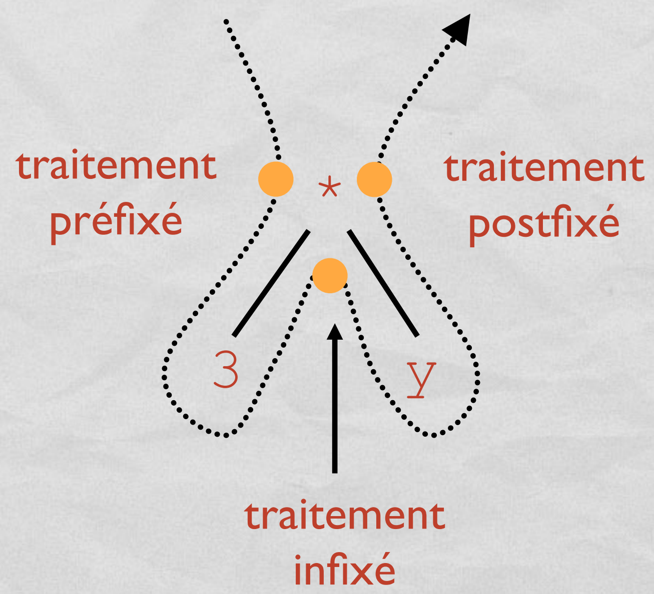
PARCOURS EN PROFONDEUR D'ABORD



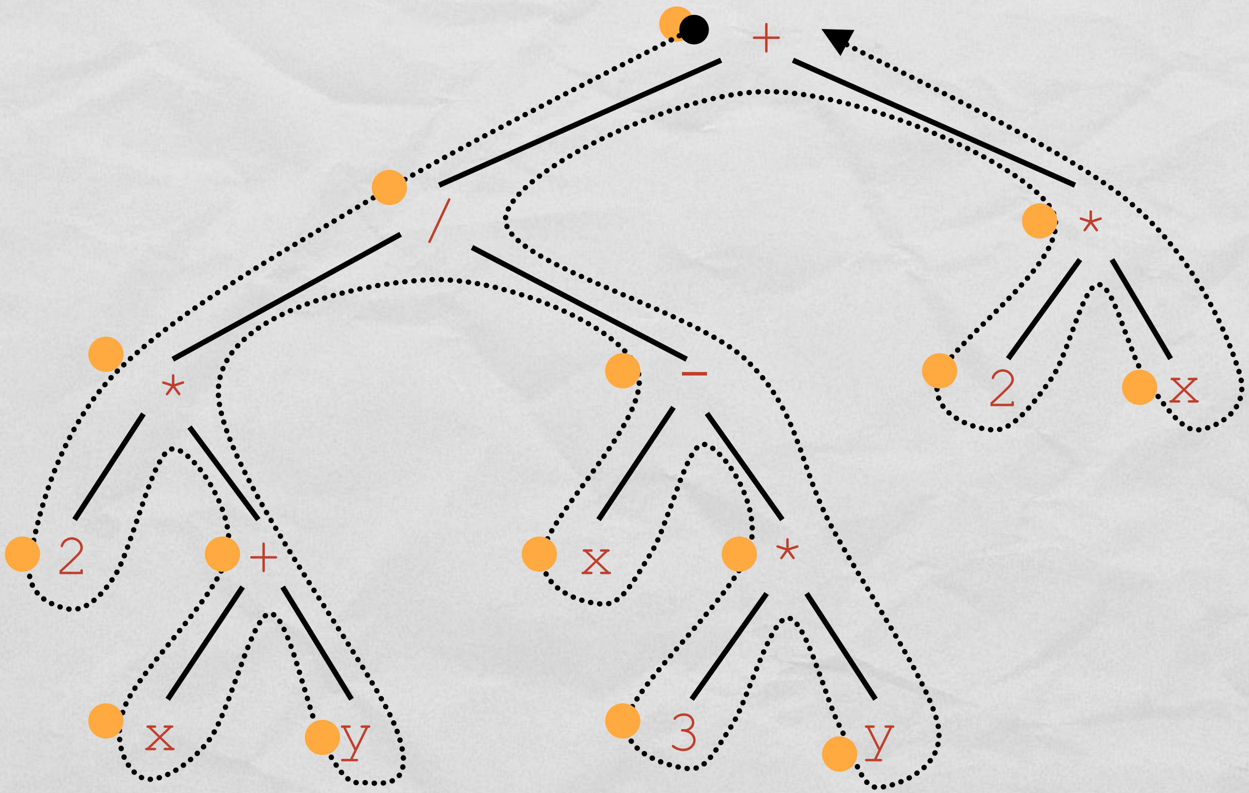
- Ordre d'évaluation des noeuds :
  - Ça dépend...

PARCOURS EN PROFONDEUR : ALGO

```
procédure explorer (A : arbre)
début
    si A = ∅ alors
    trait_arbre_vide
    sinon
    trait_préfixé(racine(A))
    explorer(g(A))
    trait_infixé(racine(A))
    explorer(d(A))
    trait_postfixé(racine(A))
    finsi
fin
```

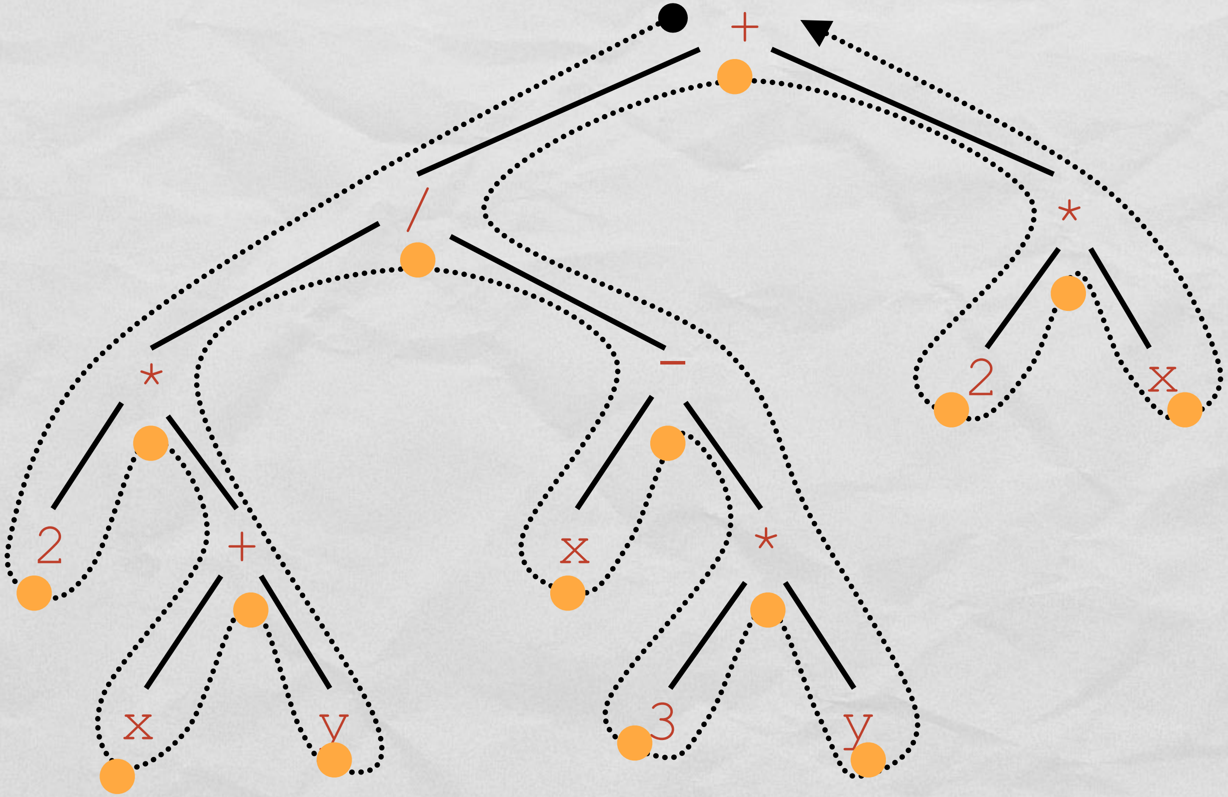


PARCOURS EN PROFONDEUR D'ABORD



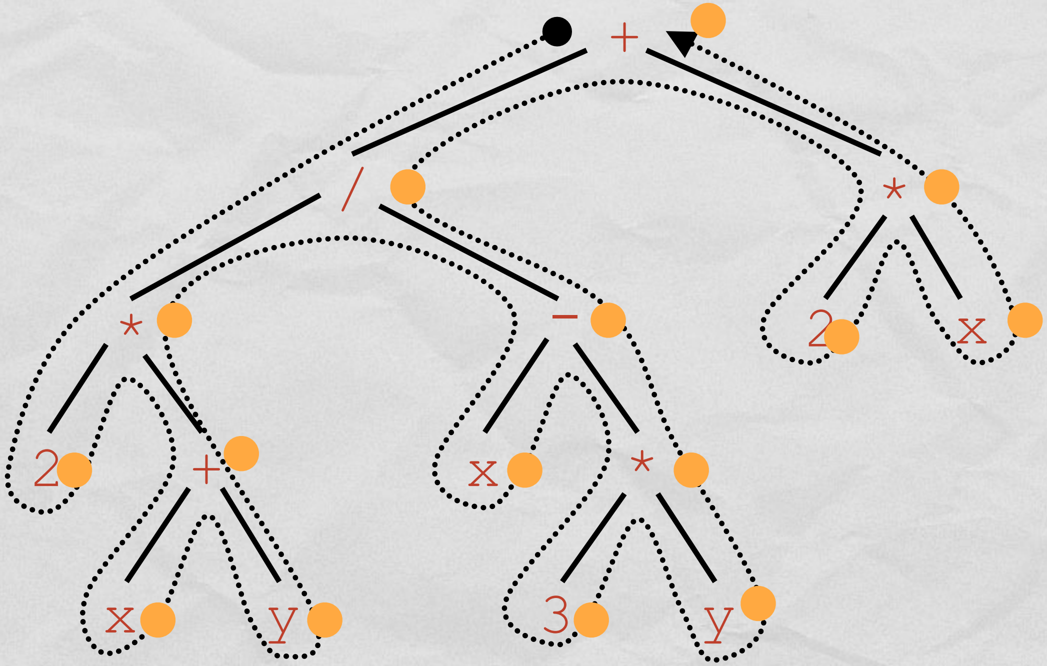
- Parcours préfixé :
    - + / * 2 + x y – x * 3 y * 2 x

PARCOURS EN PROFONDEUR D'ABORD



- Parcours infixé
  - ❑ 2 * x + y / x – 3 * y + 2 * x

PARCOURS EN PROFONDEUR : EX.

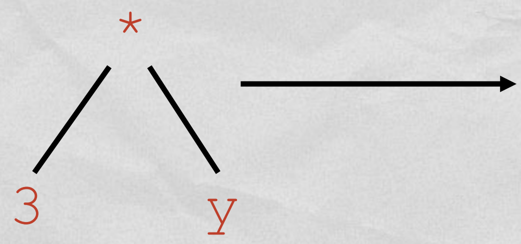


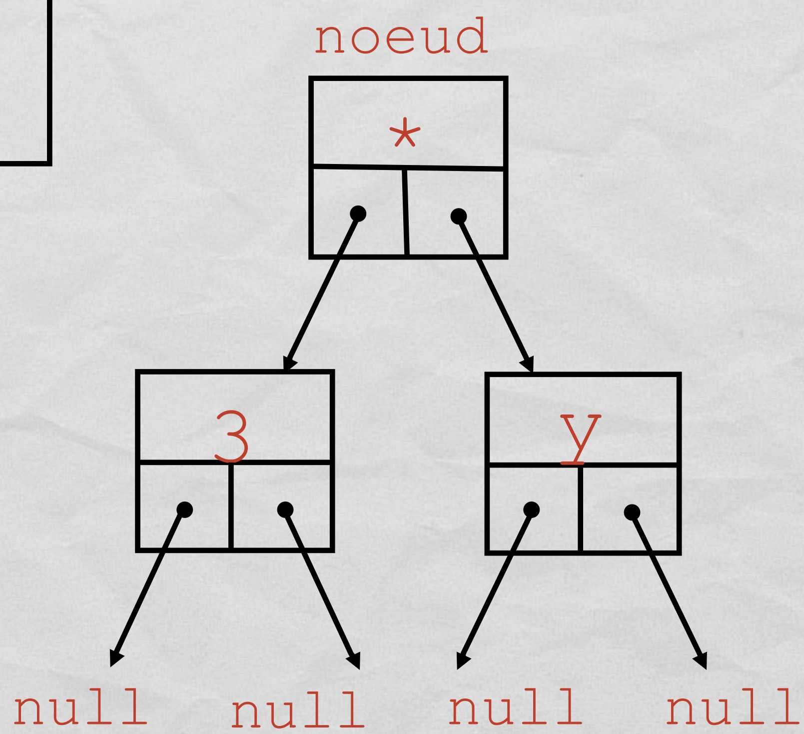
- Parcours postfixé
  - 2 x y + * x 3 y * - / 2 x * +

IMPLANTATION CHAÎNÉE (1)

```
Enregistrement Nœud {
    contenu : Entier;
     gauche : Noeud;
      droit : Noeud;
          }
```

- La plus naturelle



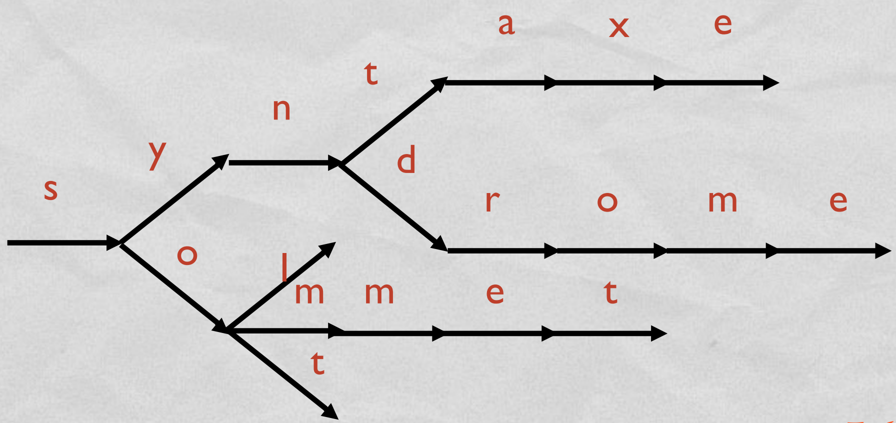


IMPLANTATION CHAÎNÉE (2)

- algorithmes efficaces pour la recherche et l'exploration
- Ajout/suppression de nœuds
  - modification d'une référence
- déplacement d'un sous arbre
  - Modification de 2 références

ARBRES GÉNÉRAUX ET FORÊTS

- Arbre général
    - arbre où les nœuds peuvent avoir nombre quelconque de fils
- Forêt
    - collection d'arbres en nombre quelconque
- Ex:
    - recherche des coups à jouer aux échecs
    - arbre à lettres



ARBRES GÉNÉRAUX - TERMINOLOGIE

- on parle plutôt de :
  - premier enfant, deuxième enfant, etc.
  - enfant aîné et enfant droit
- bijection avec les arbres binaires en posant
  - enfant_gauche(x) = aîné (x)
  - enfant_droit(x) = frère_droit(x)

ARBRES GÉNÉRAUX - PARCOURS

```
procédure explor_gen(A : arbregen)
var i : entier ; f : forêt ; n : nœud
début
  f := sousarbres(A)
  n := racine(A)
  si f = ∅ alors trait_feuille(n)
  sinon
    trait_début(n)
    pour i := 1 jusqu'à longueur(f) faire
      traitement(n, i)
      explore_gen(ième (f, i))
    fait
    trait_fin(n)
  finsi
fin
```

ARBRES GÉNÉRAUX - IMPLANTATION

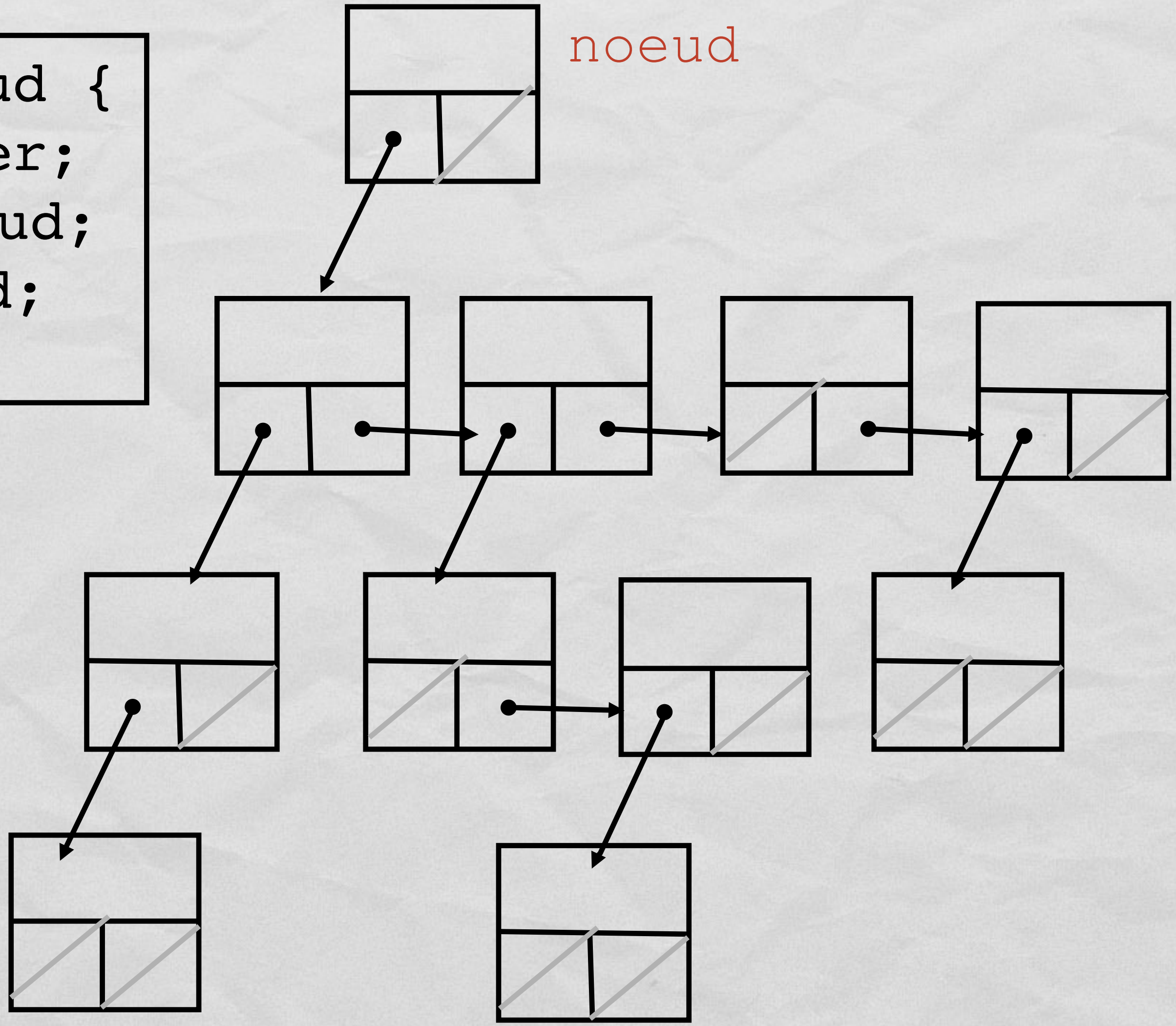
- Représentation chaînée avec un pointeur par fils
  - nécessaire de borner le nb de fils
  - adapté pour arbres p-aires
  - perte de place (p pointeurs fils même pour une feuille)

ARBRES GÉNÉRAUX - IMPLANTATION

- Représentation chaînée avec aîné et frère droit
  - deux pointeurs par nœud quelque soit le nb de fils
  - variante : booléen frère_null → indique si frère droit ou pas (frère droit remplacé par père si inexistant)

ARBRES GÉNÉRAUX - IMPLANTATION

```
Enregistrement Nœud {
    contenu : Entier;
   fils droit: Noeud;
    frêres : Noeud;
          }
```



AFFICHAGE PRÉFIXÉ

```
Prefixe(Noeud a)
Début
Nœud t
    afficher contenu du nœud
    A
    t ← A.filsAine;
    tant que ( t <> NIL)
    Prefixe(t)
    t ← t.frereDroit;
    fintantque
Fin
```

AFFICHAGE POSTFIXÉ

```
Postfixe(Noeud a)
Début
Nœud t
    t ← A.filsaine;
    tant que ( t <> NIL)
    Postfixe(t)
    t ← t.freredroit;
    fintantque
    afficher contenu du nœud A
Fin
```

CALCUL DE LA PROFONDEUR D'UN ARBRE

- *Base :*
  - Profondeur d'une feuille : 0
- *Récurrence :*
  - Profondeur d'un nœud intérieur = 1 + max Profondeur de ses fils.
- *Structure de données :*

```
Enregistrement Nœud {
  contenu : Entier;
 fils droit: Noeud;
  frêres : Noeud;
       }
```

ALGORITHME

```
CalculProfondeur(Nœud A)
Début
T : Nœud;
    A.profondeur ← 0;
    T ← A.filsaine;
    Tant que (T <> NIL)
        CalculProfondeur(T);
        Si (T.profondeur ≥ A.profondeur)
            A.profondeur ← T.profondeur + 1;
        Fin Si
        T ← T.freredroit;
    Fin Tant Que
Fin
```