Chapitre 4

Structures de Donnees arborescentes :

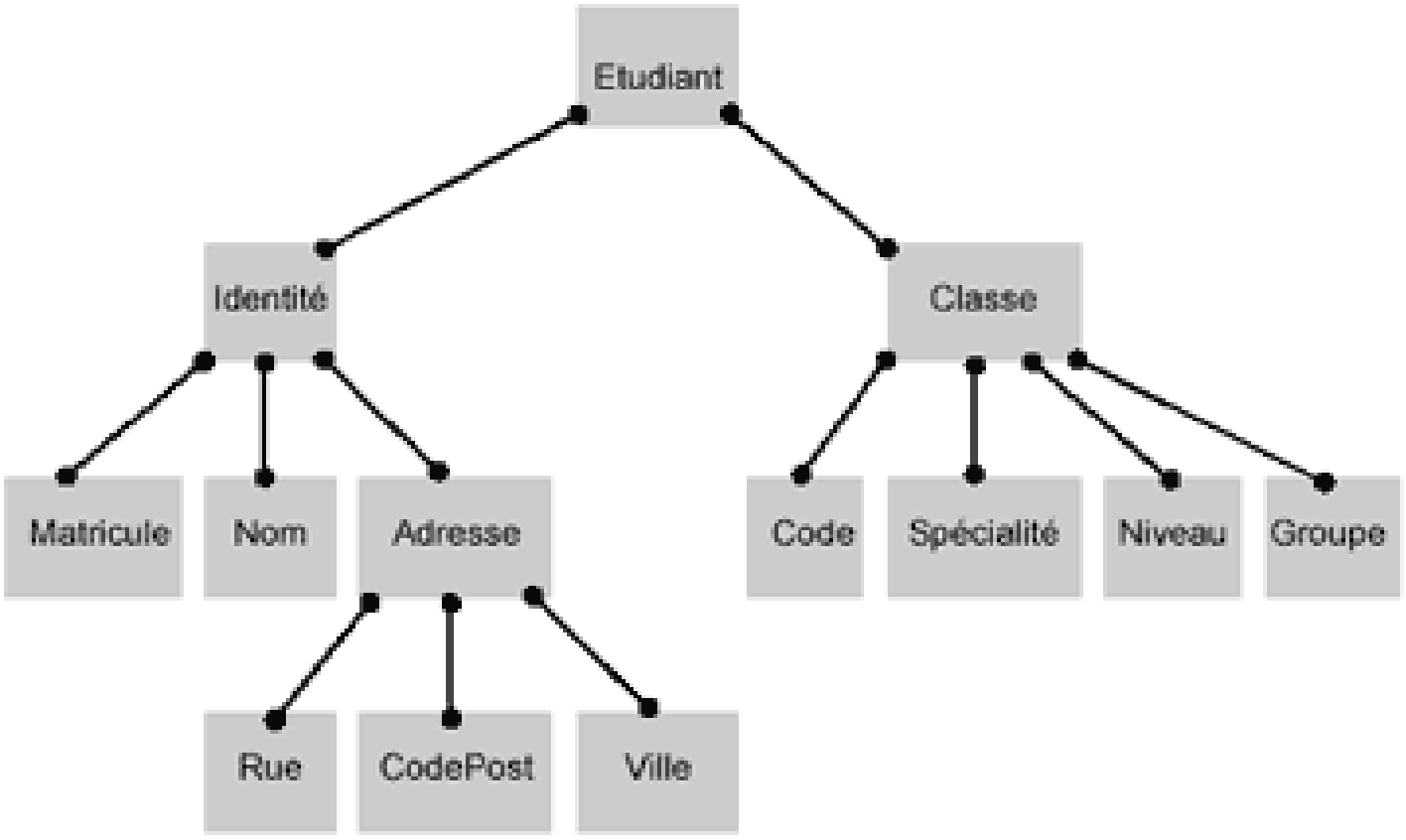
Les Arbres

Plan

- Definition
- Types de parcours
- Primitives des Arbres binaires
- Implementation chaînee
- Implementation contigu¨

Exemples d'utilisation

Une variable structuree peut ˆtre representee sous forme d«un arbre. Par exemple la variable structuree Etudiant (de type TypeEtudiant) suivante peut ˆtre representee par l«arborescence suivante :

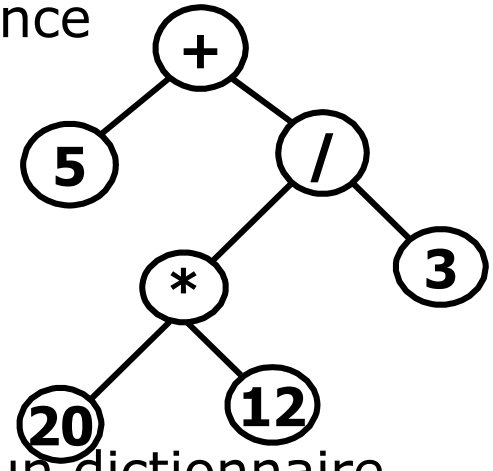


Exemples d'utilisation

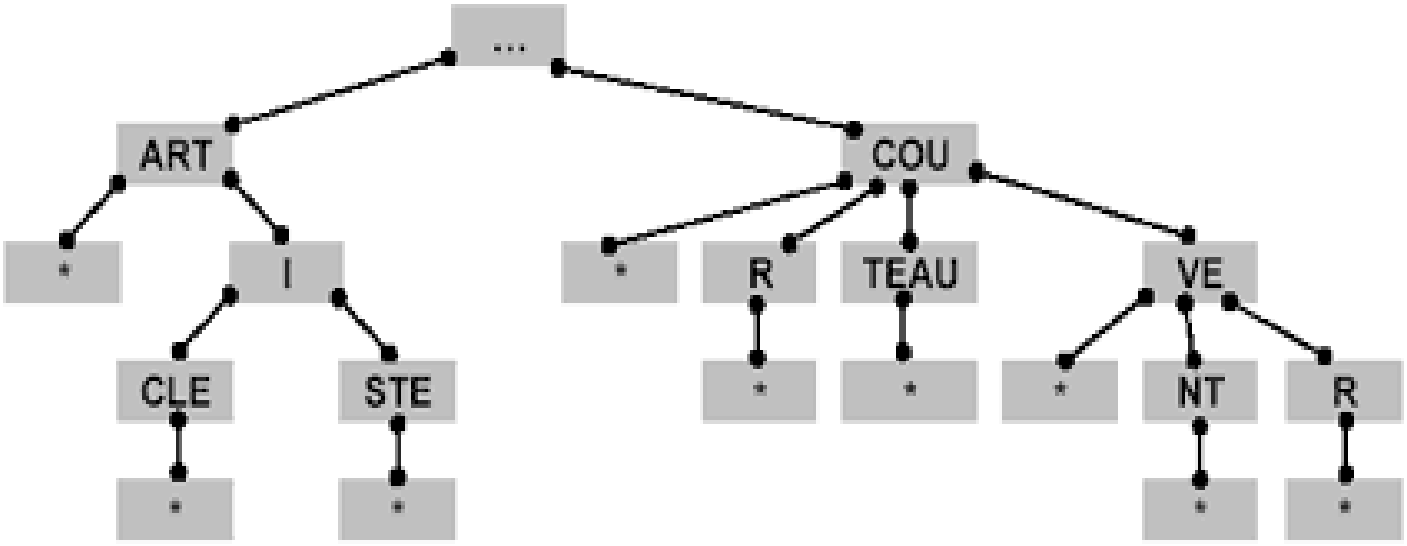
2. Une expression arithmetique peut se representer sous forme d«une arborescence

5+(20 * 12 / 3)

=+ 5 / * 20 12 3



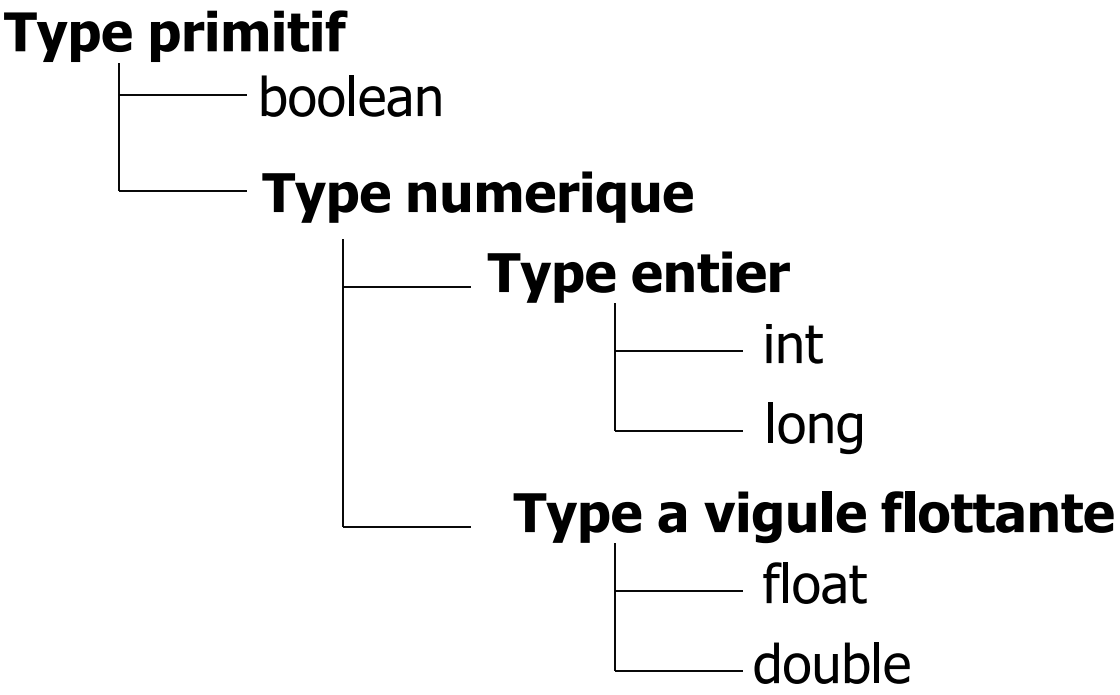
3. On peut construire ainsi un dictionnaire »arborescentà



Rq: * marque la fin d'un mot, l'ordre alphabetique est respecte de gauche a droite):

Exemples d'utilisation

- Quelques types de donnees en Java



Introduction

- Les arbres sont des structures de donnees **non lineaires** : structures **arborescentes**
- Les arbres modelisent une relation **hierarchique** dans laquelle tous les elements peuvent avoir zero ou plusieurs successeurs et un predecesseur unique **sauf** la racine

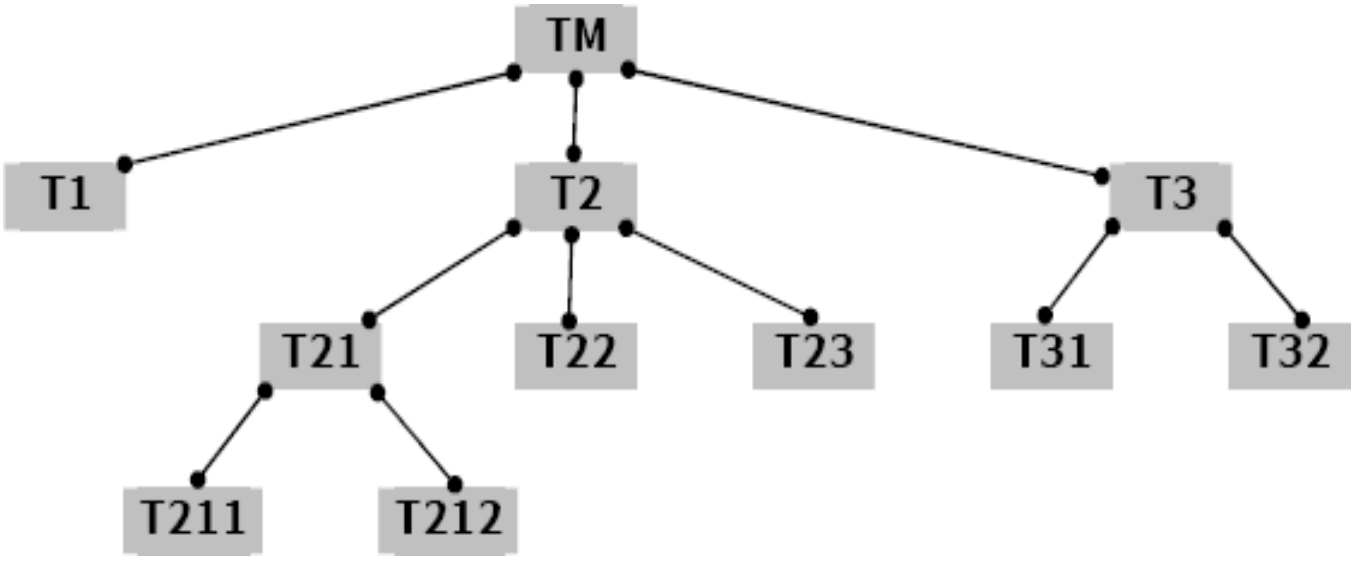
En d…autres termes …

- Un arbre est forme d'un ensemble de **nœuds**, relies par des **arˆtes tq entre deux noeuds il existe au plus un et un seul chemin**
- Un arbre organise de façon hierarchique: c«est un graphe **connexe sans cycle** (acyclique)

Introduction

- Les **schemas recursifs** simplifient beaucoup l«ecriture d«algorithme sur les arbres, ou on peut donner la definition suivante :
  Un arbre est :
  - soit vide,
  - soit constitue d«un element auquel sont chaînes un ou plusieurs arbres.

Exemple:table des matiœres d«un livre



Terminologie (1)

- Un **p`re** (**parent**) est le predecesseur direct d«un n…ud. Exemple TM est le pœre de T1, T2 et T3
- Un **fils** (**enfant**) est un successeur direct d«un n…ud. Exemple T1, T2 et T3 sont les trois fils de TM
- Un **fr`re** d«un n…ud est le fils d«un mˆme pœre. Exemple T2 est un frœre d T1.
- Une **feuille** est un n…ud sans fils. Exemple T1, T211, T212, T31 et T32 sont des feuilles.
- Le **degre d'un nœud** correspond a son nombre de fils. Exemple le degre de TM est 3.
- Le **niveau** de la racine est 0, si un n…ud est au niveau n, son successeur est au niveau n+1. Exemple le niveau de T1 est 1.
- Une **generation** represente les n…uds d'un mˆme niveau. Exemple T21, T22, T23, T31, T32 sont de la mˆme generation
- Deux n…uds sont **adjacents** si l«un est le pœre de l«autre. Exemple TM et T1 sont adjacents.

Terminologie (2)

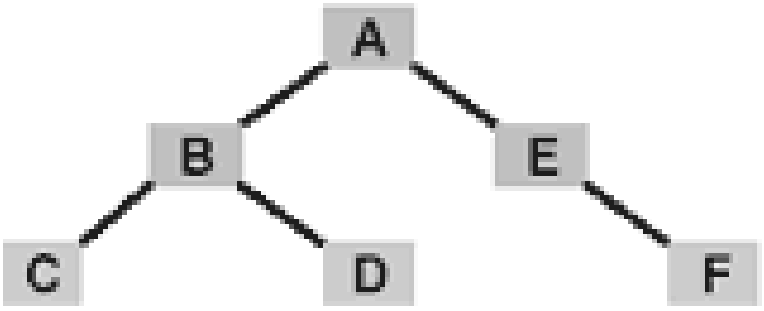
- Si n_1, n_2, –, n_k est une suite de n…uds d«un arbre telle que n_i est le parent de n_{i+1} pour 1 <= i <= k, cette suite est appelee **Chemin** entre le n…ud n_1 et le n…ud n_k. Exemple (T2, T21, T212) est un chemin de T2 a T212
- **La longueur d'un chemin** est egale au nombre de n…uds qu«il contient moins 1. Exemple, le chemin (T2, T21, T212), de T2 a T212 est de longueur 2.
- Une **branche** c'est un chemin qui commence par la racine et qui se termine par une feuille. Exemple (TM, T2, T22).
- S«il existe un chemin entre les n…uds a et b, on dit que a est un **ascendant** ou un ancˆtre de b et reciproquement que b est un **descendant** de a. Par exemple, les ascendants de T21 sont T2 et TM et ses descendants sont T211 et T212.
- Un **sous-arbre** (**SA**) d«un arbre est un n…ud accompagne de toute sa descendance. Par exemple, le sous-arbre de n…ud T2 a trois sous-arbres

Terminologie (3)

- La **taille** d«un arbre est le nombre de n…uds de cet arbre. Exemple la taille de l'arbre est 11.
- La **profondeur d'un arbre (ou hauteur)** est le nombre de n…uds de la branche la plus longue=dernier niveau +1. Exemple la profondeur de l'arbre est 4.
- L«**ordre d'un arbre** est le degre maximum parmi tous ses n…uds. Exemple l'ordre de l'arbre est 3.
- Lorsque l'ordre d'un arbre est de **n** alors l«arbre est dit **n-aire**. Exemple l'ordre de l'arbre est ternaire.
- Si n est egal a 2, l«arbre est dit **binaire**. Un arbre binaire est soit un arbre vide, soit un arbre ou chaque n…ud (a part les feuilles) a un fils gauche, un fils droit ou les deux a la fois. On parlera aussi de **sous-arbre gauche (SAG)** et de **sous-arbre droit (SAD)**.

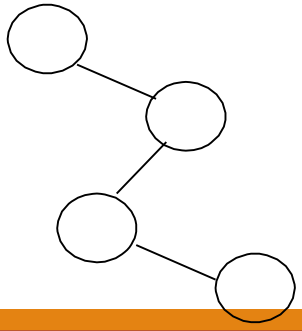
Terminologie (4)

- On peut noter un arbre binaire sous forme graphique ou sous forme parenthesee.

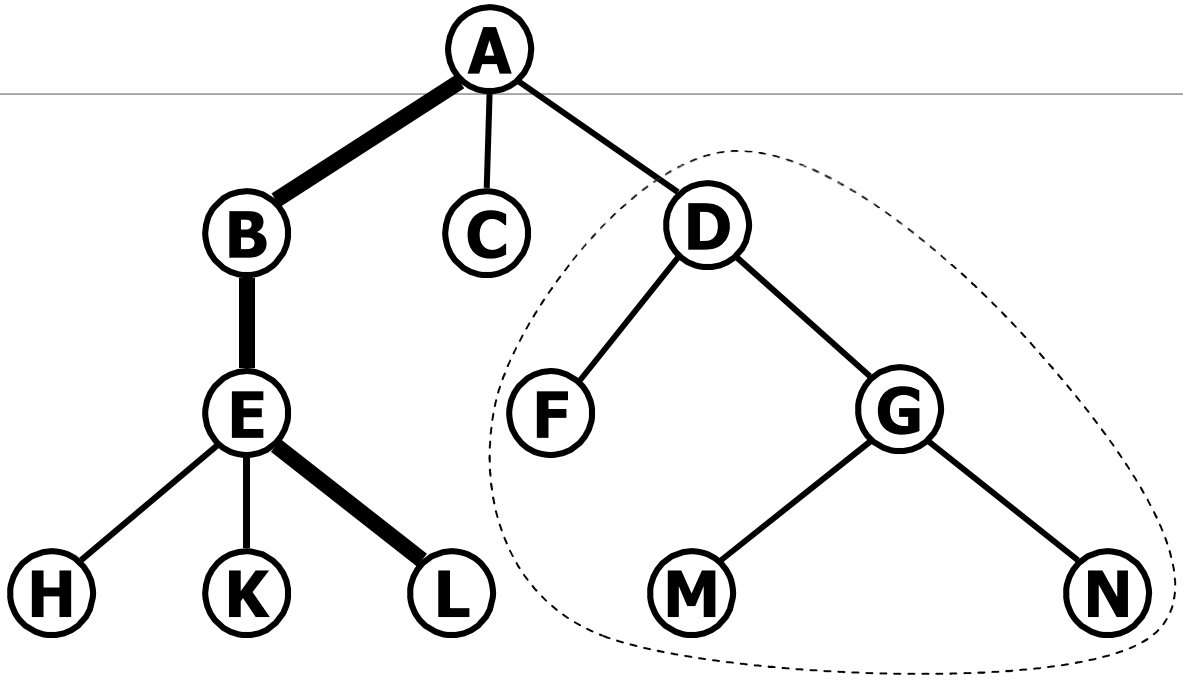


Par exemple cet arbre peut ˆtre aussi noter: A(B(C,D),E(,F)).

- Si chaque n…ud autre qu«une feuille admet deux descendants et si toutes les feuilles sont au mˆme niveau, on dit que **l'arbre binaire est complet**.
- Un arbre binaire est dit **degenere**, si tous les n…uds de cet arbre ont au plus un descendant. Un arbre degenere est equivalent a une liste chaînee.

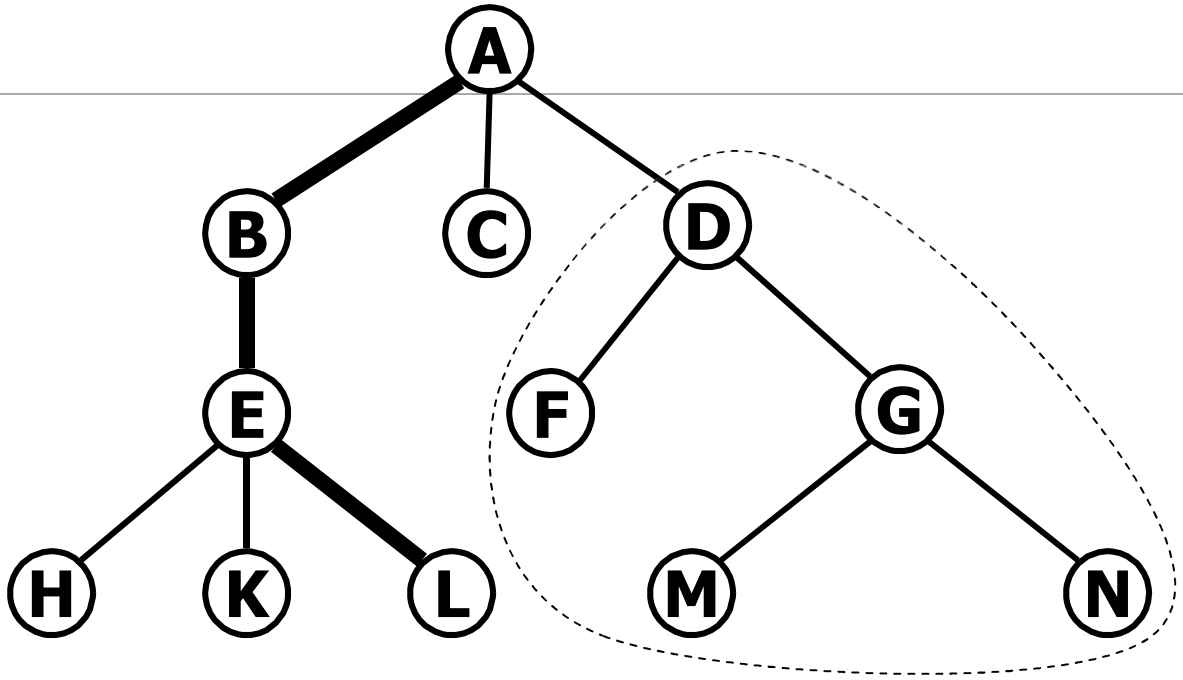


Illustration



- Le n…ud A est la ’ .
- Les n…uds H, K, L, C, F, M et N sont des ’ .
- Le n…ud D est le ’ . des n…uds F et G
- Le n…ud K est ’ . des n…uds H et L
- Lêensemble des n…uds < D, F, G, M, N > forme un ’ de ’ . n…uds et avec D comme ’ .
- La ’ . de cet arbre est 4
- La taille de cet arbre est ’ .
- A-B-E-L est un ’ .
- Les n…uds E, F et G sont de la mˆme ’ , ils appartiennent au niveau ’

Illustration



- Le n…ud A est la **racine**
- Les n…uds H, K, L, C, F, M et N sont des **feuilles**
- Le n…ud D est le **pere** des n…uds F et G
- Le n…ud K est **frere** des n…uds H et L
- Lêensemble des n…uds < D, F, G, M, N > forme un **sous arbre** de **5** n…uds et avec D comme racine
- La **profondeur** de cet arbre est 4
- La **taille** de cet arbre est 12
- A-B-E-L est un **chemin et aussi une branche**
- Les n…uds E, F et G sont de la mˆme **génération**, ils appartiennent au niveau **2**

Algorithmes de parcours dàarbres

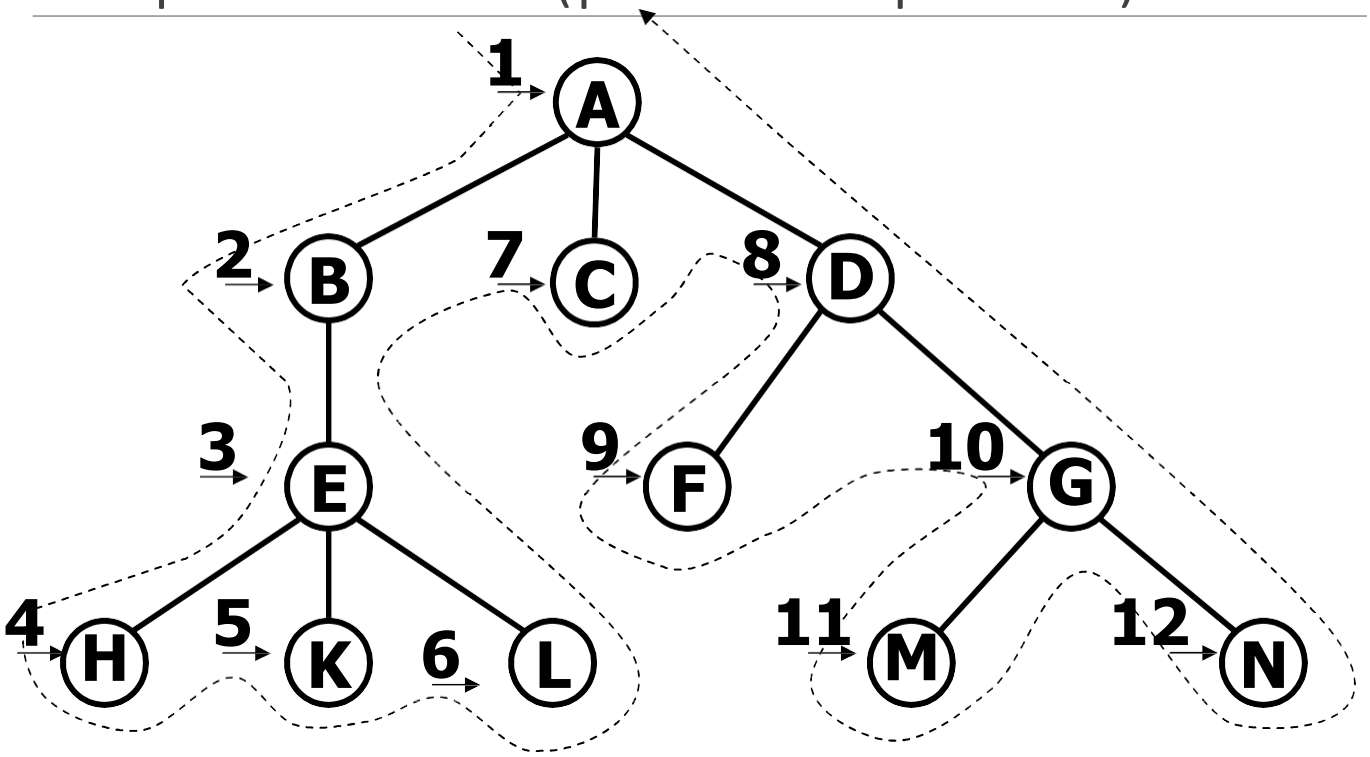
- Un algorithme de parcours est une methode de traitement dêune SD qui applique une operation specifique a chaque element de cette structure

- Il existe plusieurs methodes pour parcourir les n… uds d«un arbre
- Parcours en profondeur:
  - **parcours prefixe** (preorder)
  - **parcours infixe** (inorder)
  - **parcours postfixe** (postorder).
- Parcours en largeur

Parcours en profondeur

- Si un arbre R **est vide**, la liste vide constitue le parcours prefixe, infixe et postfixe de R.
- Si R consiste en **un seul nœud**, la liste composee de ce n…ud constitue le parcours prefixe, infixe et postfixe de R.
- Sinon, supposons que l'arbre a **k** sous-arbres A1, A2, −, Ak (de gauche a droite)
  - **parcours prefixe**
    - Racine
    - parcours prefixe des n…uds de A1, puis A2 et ainsi de suite jusqu«a Ak.
  - **parcours infixe**
    - parcours infixe des n…uds de A1
    - Racine
    - parcours infixe des n…uds de A2, −, Ak
  - **parcours postfixe**
    - parcours postfixe des n…uds de A1 puis de A2 et ainsi de suite jusqu«a Ak
    - Racine

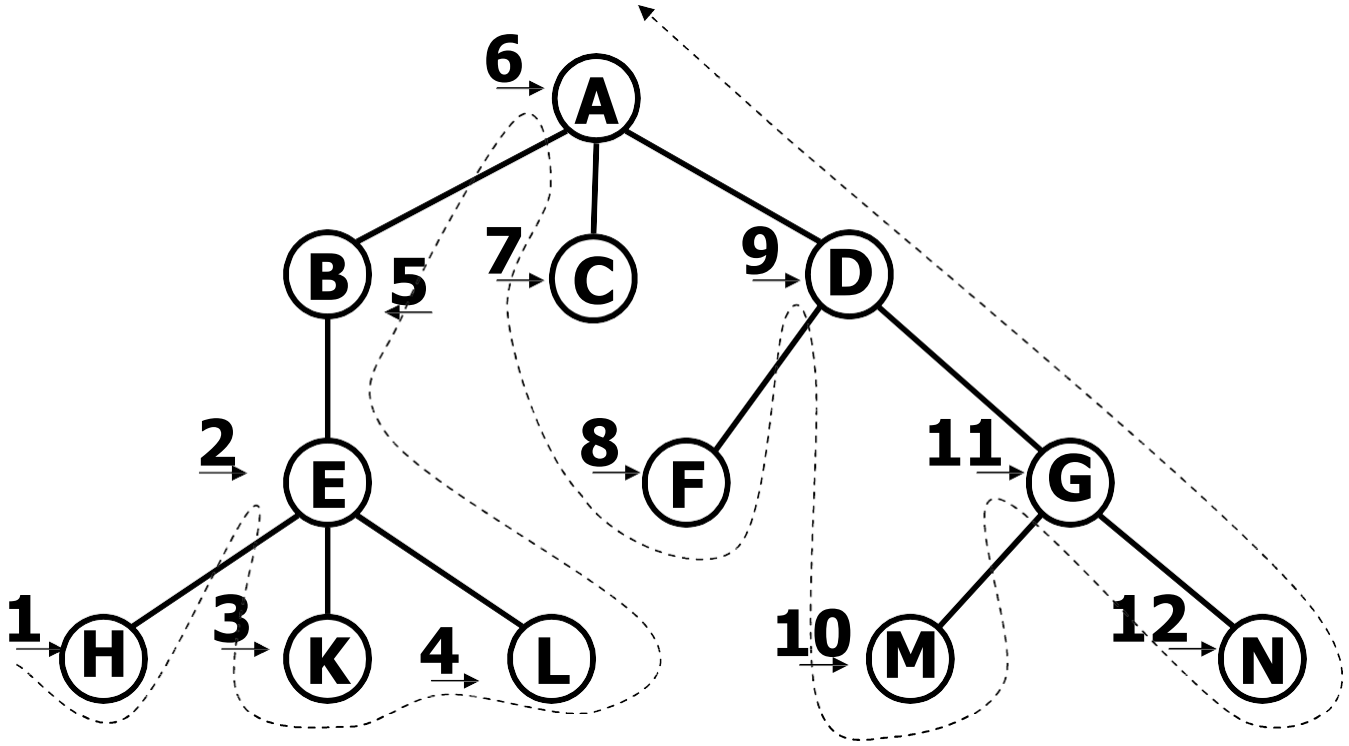
Algorithmes de parcours en profondeur (parcours prefixe)



➔ Parcours prefixe :

A, B, E, H, K, L, C, D, F, G, M, N.

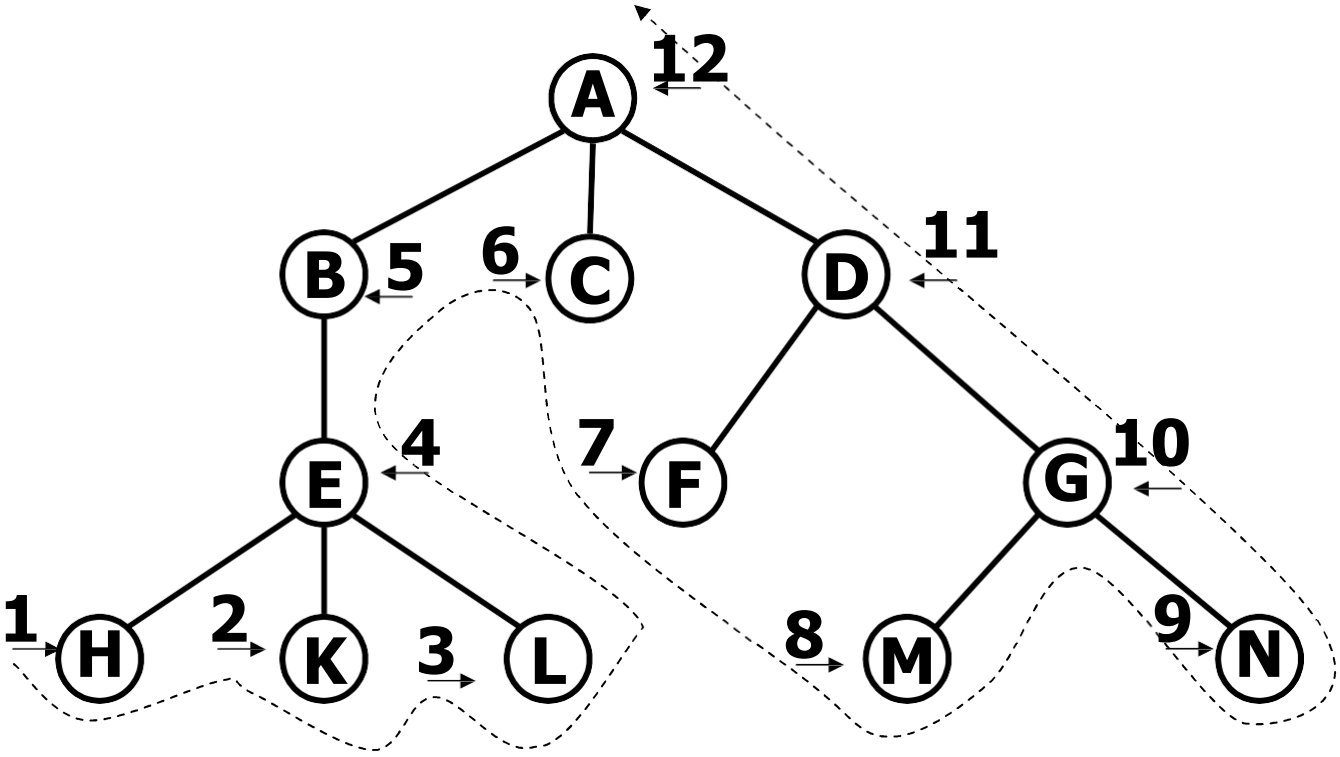
Algorithmes de parcours en profondeur (parcours infixe)



➔ Parcours infixe :

H, E, K, L, B, A, C, F, D, M, G, N.

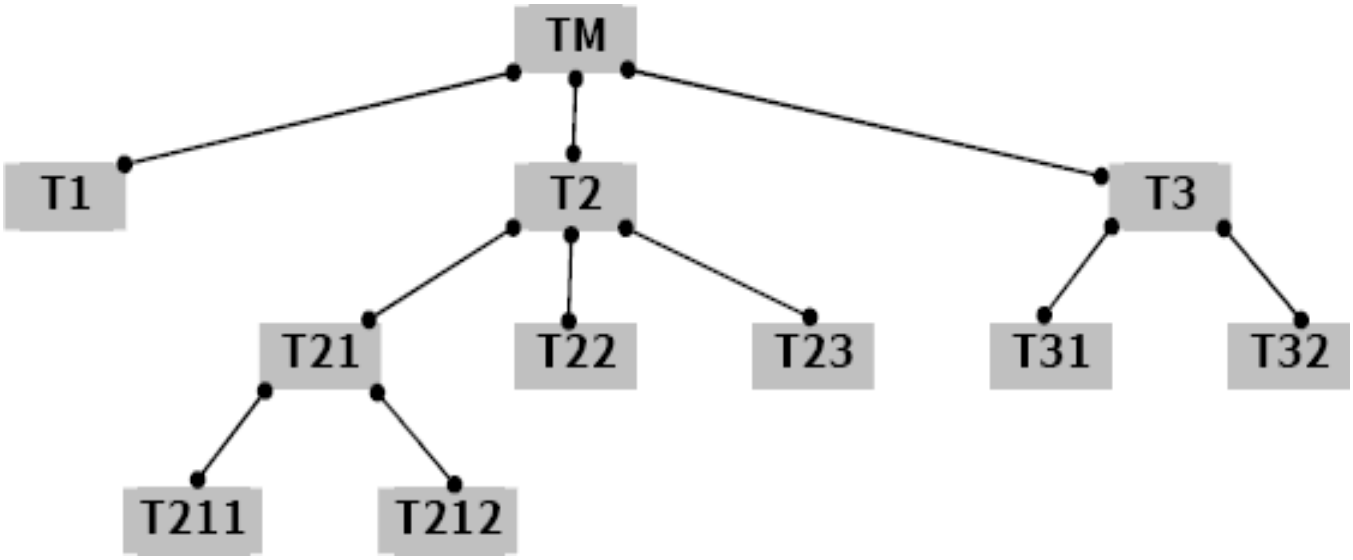
Algorithmes de parcours en profondeur (parcours postfixe)



➔ Parcours postfixe :

H, K, L, E, B, C, F, M, N, G, D, A.

Travail a faire



- **Parcours prefixe** :

TM, T1, T2, T21, T211, T212, T22, T23, T3, T31, T32

- **Parcours Infixe** :

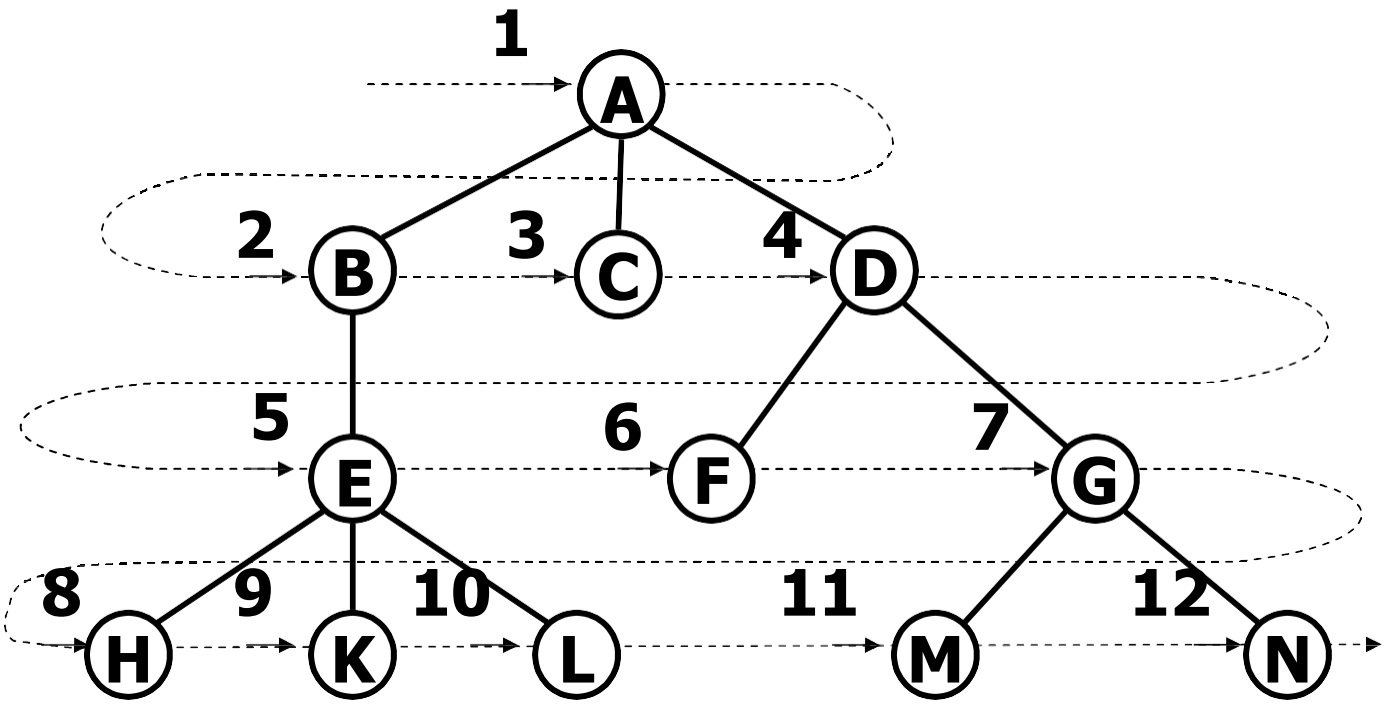
T1, TM, T211, T21, T212, T2, T22, T23, T31, T3, T32

- **Parcours postfixe** :

T1, T211, T212, T21, T22, T23, T2, T31, T32, T3, TM

Algorithme de parcours en largeur

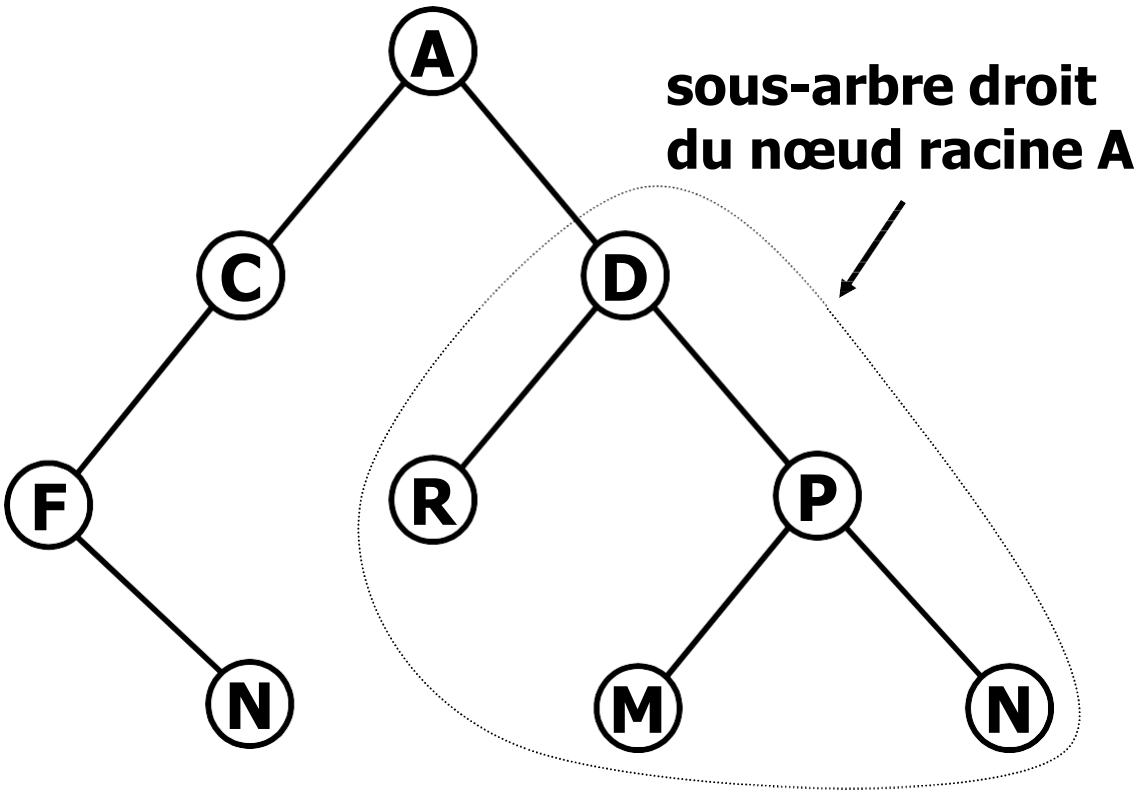
- Lêalgorithme de parcours en largeur visite la racine, puis chaque element du premier niveau, avant de visiter chaque element du deuxiùme niveau, etc., en visitant systematiquement chaque element dêun niveau avant de passer au niveau suivant
- Illustration :



➔Parcours en largeur : A, B, C, D, E, F, G, H, K, L, M, N.

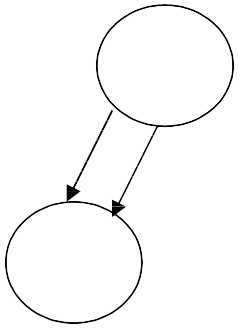
Arbres binaires

- Definitions :
  - Un arbre binaire est soit vide soit il contient une racine avec deux sous-arbres binaires
  - Un arbre binaire est un arbre ou chaque n…ud peut avoir au maximum deux fils : un fils gauche et un fils droit



Specifications des primitives

- Procedure **Construire** (**Don** Racine: typeDesElements, **Don** AG: Arb_Bin, **Don** AD : Arb_Bin) : Arb_Bin



-- **Precond** :
-- Taille(AG) + Taille(AD) +1 < Max-size
-- *Racine* de mˆme type de donnees que
-- les elements de AG et AD
-- AG et AD ne sont pas identiques
-- **Postcond** : AB est un nouveau arbre binaire
-- dont la racine est *Racine*, le sous arbre droit
-- AD est et le sous arbre gauche est AG

- Fonction **Vide** (**Don** AB : Arb_Bin) : Booleen
  -- **Precond** :
  -- **Postcond** : retourne vrai si l«arbre binaire AB
  -- est vide et faux sinon

- Fonction **Racine** (**Don** AB : Arb_Bin) : typeDesElements
  -- **Precond** : Non Vide (AB)
  -- **Postcond** : retourne le contenu de la racine de

Specifications des primitives

- Fonction **SAGauche** (**Don** AB : Arb_Bin) : Arb_Bin
  -- **Precond** : Non Vide (AB)
  -- **Postcond** : retourne le sous-arbre gauche de
  -- AB

- Fonction **SADroit** (**Don** AB : Arb_Bin) : Arb_Bin
  -- **Precond** : Non Vide (AB)
  -- **Postcond** : retourne le sous-arbre droit de AB

- Procedure **Taille**(Don AB : Arb_Bin) : Entier
  -- **Precond** :
  -- **Postcond** : retourne la taille de AB

- Fonction **Hauteur** (Don AB : Arb_Bin) : Entier
  -- **Precond** :
  -- **Postcond** : retourne l'hauteur de AB

Specifications des primitives

- Procedure **Afficher** (Don AB : Arb_Bin, Don Ordre_parcours: Entier)
  -- **Precond** : Ordre_parcours ∈ [1,3];
  -- 1: preordre, 2: inordre, 3: postordre
  -- **Postcond** : affiche les elements de AB selon
  -- l'ordre specifie par Ordre_parcours

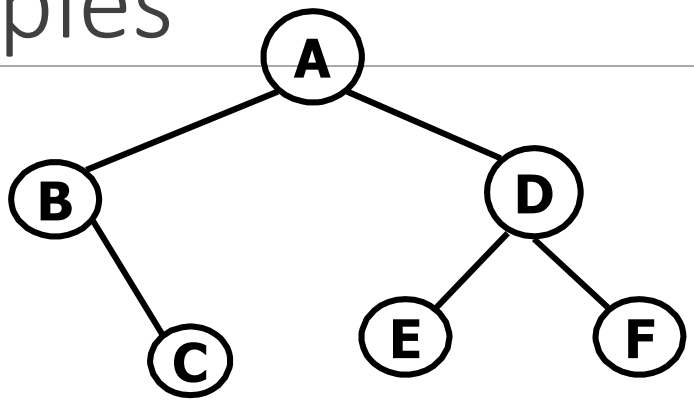
- Fonction **Copier** (Don AB : Arb_Bin) : Arb_Bin
  -- **Precond** :
  -- **Postcond** : permet d'avoir une copie de tous
  –- les elements de AB

- Fonction **Egal**(Don AB1: Arb_Bin, Don AB2: Arb_Bin): Booleen
  -- **Precond** :
  -- **Postcond** : retourne true si AB1 et AB2 ont
  -- les mˆmes elements dans le mˆme ordre
  -- sinon retourne false

Parcours en profondeur

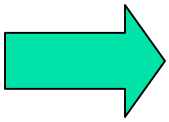
- Si un arbre R **est vide**, la liste vide constitue le parcours prefixe, infixe et postfixe de R.
- Si R consiste en **un seul nœud**, la liste composée de ce n…ud constitue le parcours prefixe, infixe et postfixe de R.
- Sinon:
  - parcours prefixe
    - Racine
    - Parcours prefixe du SAG
    - Parcours prefixe du SAD
  - parcours infixe
    - Parcours infixe du SAG
    - Racine
    - Parcours infixe du SAD
  - parcours postfixe
    - Parcours postfixe du SAG
    - Parcours postfixe du SAD
    - Racine

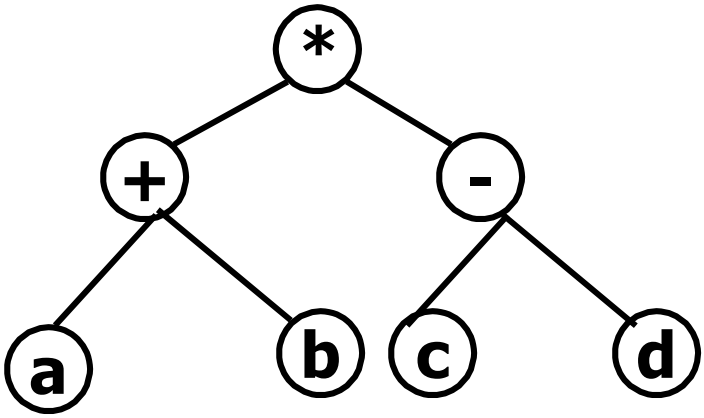
Parcours en profondeur: Exemples



- **Parcours prefixe** : ABCDEF
- **Parcours Infixe** : BCAEDF
- **Parcours postfixe** : CBEFDA

(a+b)*(c-d)





- **Parcours prefixe** : *+ab-cd
- **Parcours Infixe** : a+b*c-d
- **Parcours postfixe** : ab+cd-*

Parcours prefixe dans les arbres binaires

- Algorithme de parcours **prefixe** »RGDà: Ce parcours consiste a effectuer dans l«ordre :
  1. le traitement de la racine,
  2. le parcours du sous-arbre gauche,
  3. le parcours du sous-arbre droit.

```
Procédure Préfixé (Don R : Arbre)
-- PréCond    : R est un arbre binaire
-- PostCond   : parcours préfixé de l'arbre binaire R
-- Schéma récursif
Début
    Si Non Vide (R) Alors
        Afficher (Racine (R))
        Préfixé (SAGauche (R))
        Préfixé (SADroit (R))
    Fin Si
Fin
```

Parcours infixe dans les arbres binaires

- Algorithme de parcours **infixe** »GRDà: Ce parcours consiste a effectuer dans l«ordre :
  1. le parcours du sous-arbre gauche,
  2. le traitement de la racine,
  3. le parcours du sous-arbre droit.

```
Procédure Infixé (Don R : Arbre)
-- PréCond      : R est un arbre binaire
-- PostCond     : parcours infixé de l'arbre binaire R
-- Schéma récursif
Début
    Si Non Vide (R) Alors
        Infixé (SAGauche (R))
        Afficher (Racine (R))
        Infixé (SADroit (R))
    Fin Si
Fin
```

Parcours postfixe dans les arbres binaires

- Algorithme de parcours **postfixe** »GDRà: Ce parcours consiste a effectuer dans l«ordre :
  1. le parcours du sous-arbre gauche,
  2. le parcours du sous-arbre droit,
  3. le traitement de la racine.

```
Procédure Postfixé (Don R : Arbre)
-- PréCond    : R est un arbre binaire
-- PostCond   : parcours postfixé de l'arbre binaire R
-- Schéma récursif
Début
    Si Non Vide (R) Alors
        Postfixé (SAGauche (R))
        Postfixé (SADroit (R))
        Afficher (Racine (R))
    Fin Si
Fin
```