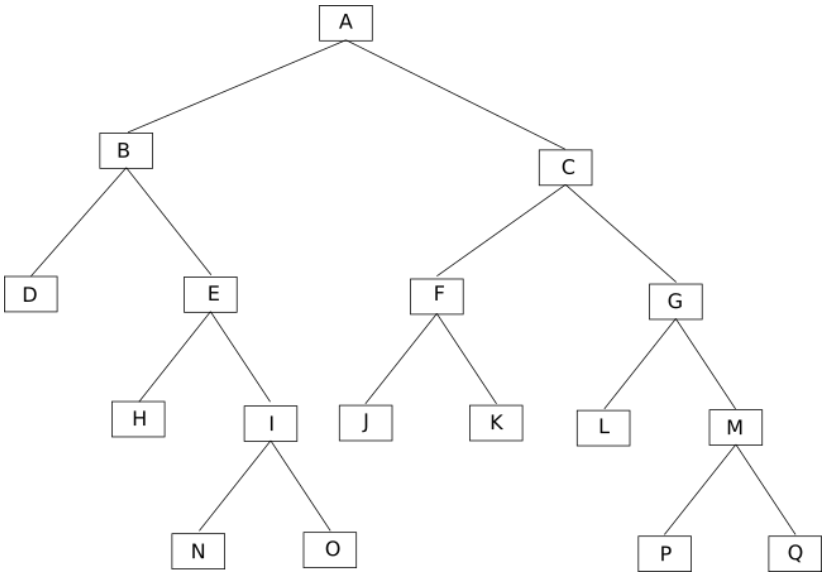
Les arbres

Les arbres sont des types abstraits très utilisés en informatique. On les utilise notamment quand on a besoin d'une structure hiérarchique des données.



Vocabulaire

Chaque élément de l'arbre est appelé **noeud** (par exemple : A, B, C, D,...,P et Q sont des noeuds)
Le noeud initial (A) est appelé noeud racine ou plus simplement **racine**
On dira que le noeud E et le noeud D sont les **fils** du noeud B. On dira que le noeud B est le **père** des noeuds E et D
Dans un **arbre binaire**, un noeud possède au plus 2 fils
Un noeud n'ayant aucun fils est appelé **feuille** (exemples : N et K sont des feuilles)
À chaque noeud d'un arbre binaire, on associe une clé ("valeur" associée au noeud on peut aussi utiliser le terme "valeur" à la place de clé), un "sous-arbre gauche" et un "sous-arbre droit"
On appelle **arête** le segment qui relie 2 noeuds.
On appelle **taille** d'un arbre le nombre de noeuds présents dans cet arbre
On appelle **profondeur d'un nœud** ou d'une feuille dans un arbre binaire le nombre de nœuds du chemin qui va de la racine à ce nœud. La racine d'un arbre est à une profondeur 1. Faites attention, suivant l'exercice la profondeur de la racine est parfois à 0. Les 2 définitions sont valables, il faut juste préciser si vous considérez que la profondeur de la racine est de 1 ou de 0.
On appelle **hauteur** d'un arbre la profondeur maximale des nœuds de l'arbre.

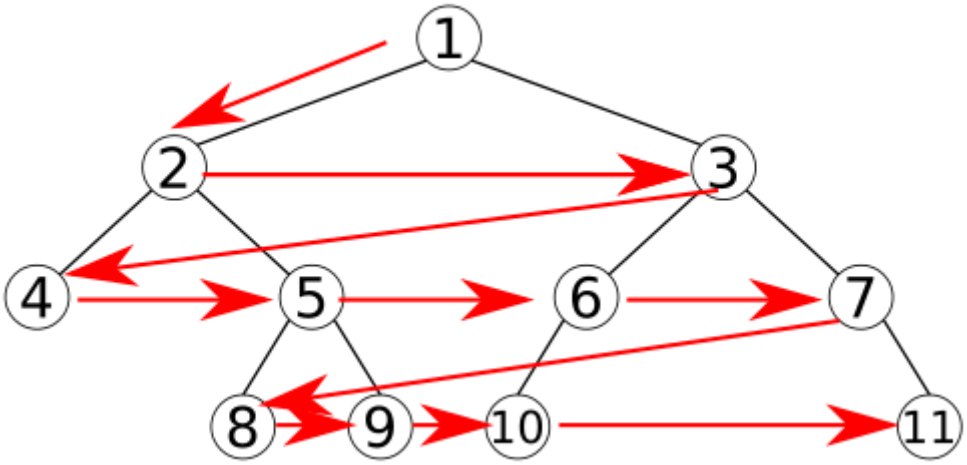
Un **arbre binaire de recherche** est un cas particulier d'arbre binaire. Pour avoir un arbre binaire de recherche :

- il faut avoir un arbre binaire
- il faut que chaque nœud vérifie : $valeur\ fils\ gauche < valeur\ noeud < valeur\ fils\ droit$

Parcours d'arbres

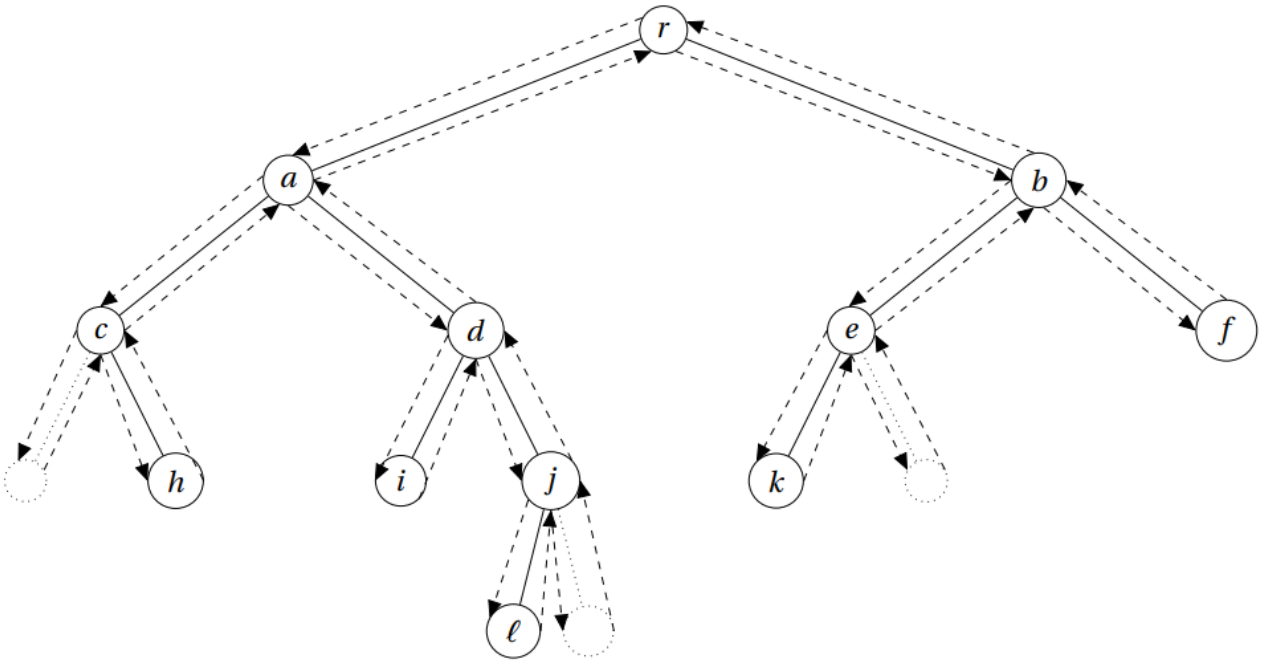
Parcours en largeur d'abord

On parcours « étage par étage » :



Parcours en profondeur

On ajoute les fils manquants et on parcours un arbre suivant la ballade suivante :



Parcours préfixe

On liste le nœud quand on passe à gauche du nœud (première fois que l'on rencontre le nœud)

```
def parcours_prefixe(arbre):
    print(racine(arbre))
    parcours_prefixe(sag(arbre))
    parcours_prefixe(sad(arbre))
```

Parcours infixe

On liste le nœud quand on passe en dessous du nœud (deuxième fois que l'on rencontre le nœud)

```
def parcours_infixe(arbre):
    parcours_infixe(sag(arbre))
    print(racine(arbre))
    parcours_infixe(sad(arbre))
```

Parcours suffixe

On liste le nœud quand on passe à droite du nœud (troisième fois que l'on rencontre le nœud)

```
def parcours_suffixe(arbre):
    parcours_suffixe(sag(arbre))
    parcours_suffixe(sad(arbre))
    print(racine(arbre))
```