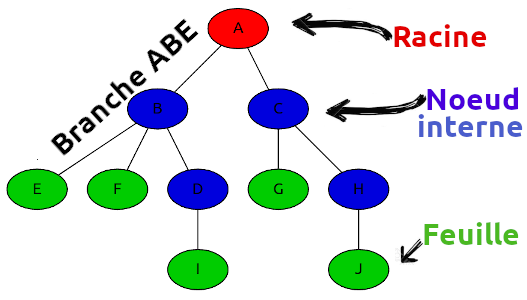
**Arbres: Définitions**

Un graphe n'ayant **\_\_\_\_\_\_\_\_\_\_\_\_** s'appelle un arbre.

Un arbre est une **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** composée de nœuds.

● Les nœuds peuvent avoir des **\_\_\_\_\_\_\_\_\_\_\_** (ou nœuds enfants) situés au niveau inférieur

● Le nœud situé au niveau supérieur est appelé le nœud **\_\_\_\_\_\_\_\_\_\_\_\_\_** (ou nœud parent)

● Le nœud qui n'a pas de père, généralement placé tout en haut,   
est la **\_\_\_\_\_\_\_\_\_**de l'arbre

● Les nœuds qui n'ont pas d'enfants s'appellent les **\_\_\_\_\_\_\_\_\_\_\_**de l'arbre

La **\_\_\_\_\_\_\_\_\_**de l'arbre correspond au nombre de nœuds.

● On appelle **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**d'un nœud le nombre d'arêtes qu'il faut  
parcourir depuis la racine pour atteindre ce nœud.

● La **\_\_\_\_\_\_\_\_\_**de l'arbre correspond à la profondeur de la feuille la plus éloignée de la racine

Remarque: Il existe une définition alternative de la hauteur

● **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**correspond au nombre maximum d'enfants qu'un nœud peut avoir.

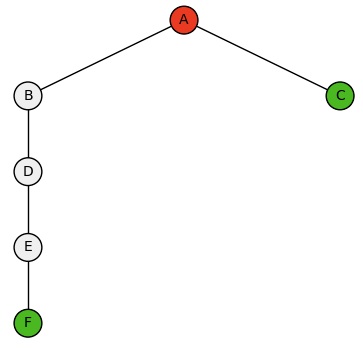
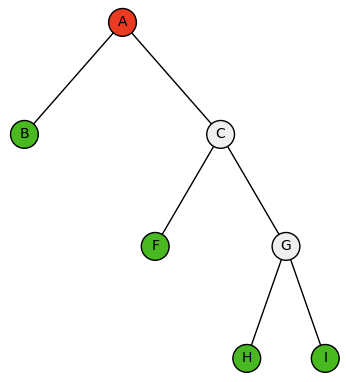
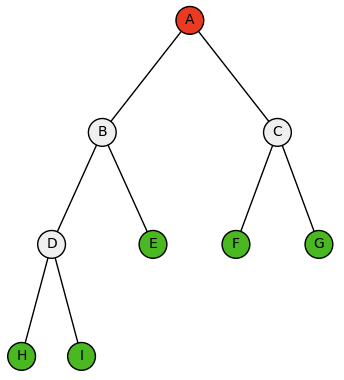
● Une **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**est une suite finie de sommets consécutifs de la racine vers une feuille

**Arbres binaires: Définitions**

Un arbre binaire est un arbre **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Dans un arbre binaire **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**ou localement complet, tous les nœuds n'ont que 0 ou 2 enfants. (Jamais 1).

Ne pas confondre avec un arbre binaire complet, où tous les niveaux sont remplis, à l'exception éventuelle du dernier, dans lequel les feuilles sont alignées à gauche.

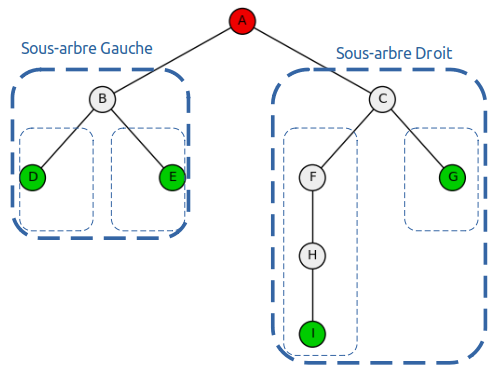
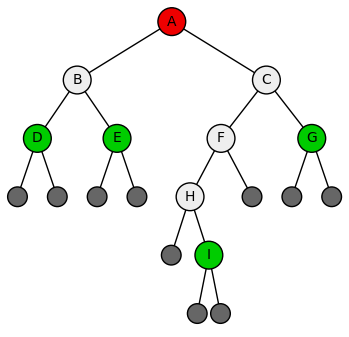
  

**Arbres binaires: Implémentation avec une liste récursive**

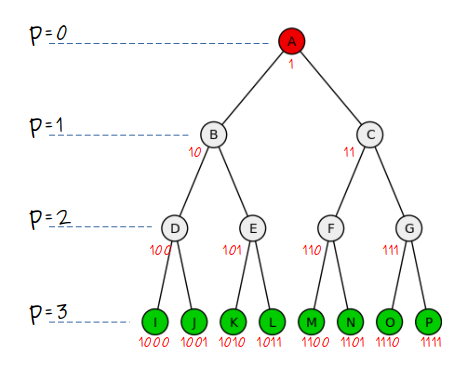
On peut définir un arbre comme une racine munie de deux sous-arbres :

ArbreA = [ , , ]



**Arbres binaires: Rapport taille/hauteur**

Le premier nœud de la pième ligne porte le numéro :

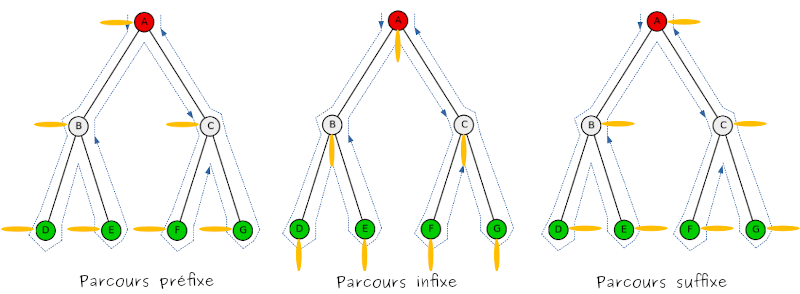
**N =**

On prouve ainsi que :

**p =**

**Arbres binaires: Parcours**

On définit classiquement plusieurs parcours dans les arbres :

****Le parcours en largeur et les parcours**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Arbres Binaires de Recherche: Définitions**

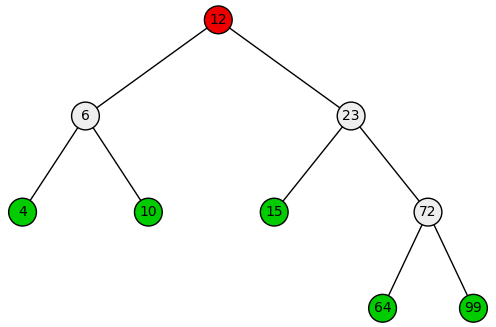
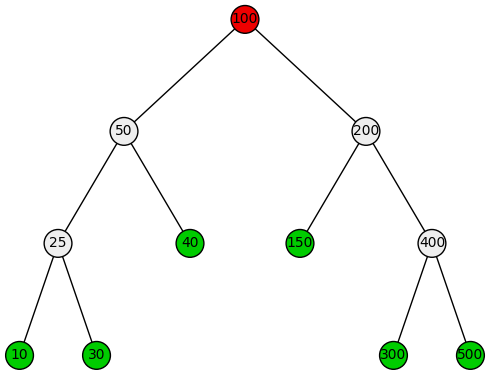
Un arbre binaire de recherche est un arbre binaire dont tous les nœuds portent des **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**selon les règles suivantes.

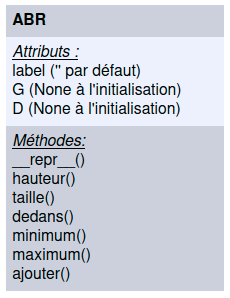
L'étiquette portée par un nœud est:

- plus **\_\_\_\_\_\_\_\_\_\_\_\_**que toutes les étiquettes de son sous-arbre gauche.

- plus **\_\_\_\_\_\_\_\_\_\_\_\_**que toutes les étiquettes de son sous-arbre droit.

Remarque: Un ABR se nomme BST en anglais pour binary search tree

**ABR : Implémentation en POO**

**\_\_repr\_()** : Méthode spéciale de l’objet qui permet de préciser   
sa représentation

**dedans()** : Renvoie un booléen en fonction de la présence ou non  
 d’une valeur recherchée

**minimum()** : Cherche le dernier fils gauche de la structure

**maximum()** : Cherche le dernier fils droit de la structure

**taille()** =

**Hauteur()** =

**ABR versus listes : Coût d’une recherche**

Lorsque la taille d’une liste augmente, le temps de recherche d’un élément dans cette liste augmente **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**.

Le nombre maximum de comparaisons correspond au nombre d’éléments de la liste.

Alors que dans un ABR, le nombre maximum de comparaisons correspond à la **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**de l’ABR.