# Travaux Dirigés d'algorithmique n°7

**Cours d'Informatique de Deuxième Année**

*—DEUG Sciences M3-MI—*

---

## Arbres binaires, exemple d'application

L'objet de cette feuille d'exercices est la manipulation d'une nouvelle structure de données récursive : la structure d'*arbre binaire*. L'algorithmique que l'on utilise sur ces structures est voisine de celle que l'on utilise sur les listes chaînées à ceci près que l'on ne parle plus du "suivant" d'une cellule mais plutôt du "fils gauche" et "fils droit" d'un nœud. De même, on ne parle plus du "début" de la liste mais plutôt de la "racine" de l'arbre.
Nous allons considérer :

- la déclaration de la structure et les algorithmes de manipulation ;
- le codage de Huffman.

---

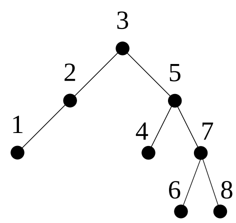
## Structure d'arbre binaire

On considère les déclarations suivantes :

```
typedef struct noeud
{ float Val;                  /* valeur dans le noeud                */
  struct noeud * Gauche;      /* pointeur sur le sous-arbre gauche   */
  struct noeud * Droit;       /* pointeur sur le sous-arbre droit    */
} StrNoeud;                   /* nouveau type : StrNoeud             */

typedef StrNoeud * PtrNoeud;  /* pointeur sur un noeud               */
```

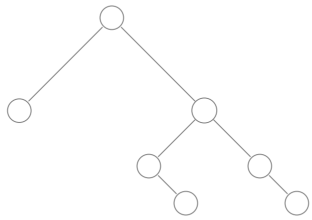
Un arbre binaire est assimilé avec un pointeur sur sa racine. Exemple :




▸ **Exercice 1. (Parcours)** *Donner les fonctions de parcours préfixe, suffixe et infixe, d'un arbre binaire de recherche.*

▸ **Exercice 2.** *Nous nous intéressons au parcours suffixé d'un arbre binaire :*

- *quel est toujours le dernier sommet (nœud ou feuille) affiché lors du parcours suffixé d'un arbre binaire ?*
- *compléter les 7 nœuds et feuilles de l'arbre binaire suivant, afin que son parcours suffixé soit la suite croissante 1 2 3 4 5 6 7.*

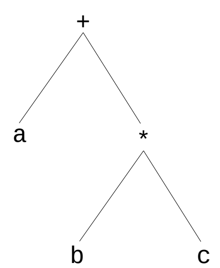


▸ **Exercice 3. (Codage de Huffman)** *On considère les caractères de* `a` *à* `f` *suivants et leur probabilité d'apparition dans un fichier donné (qui ne contient que des caractères) :*

`a` : *0,20* , *b : 0,15*
`c` : *0,25* , *d : 0,25*
`e` : *0,05* , *f : 0,10*

1. *Expliciter à partir de cet exemple la génération d'un arbre de Huffman et le code de longueur variable associé à chacun des caractères.*
2. *On suppose que le fichier considéré comprend 100.000 caractères :*
   (a) *quelle serait la taille de ce fichier, en codant les caractères sans tenir compte de leur probabilité d'apparition ?*
   (b) *Quelle est la taille moyenne en retenant le code trouvé en 1 ?*

▸ **Exercice 4.** *Une expression arithmétique peut être représentée sous forme d'arbre binaire, où les feuilles sont les opérandes, et les noeuds sont les opérateurs. Par exemple :* $a + (b * c)$ *:*




1. *Dessiner l'arbre correspondant à l'expression* $(a - b) + ((c * d) - a)$.
2. *En déduire la notation en polonaise inverse de l'expression précédente.*
3. *En déduire un algorithme pour évaluer une expression arithmétique.*