4. Donnez la définition d'une fonction de prototype

   `void killmatrix (matrix_t);`

   qui désalloue une matrice.
5. Donnez la définition d'une fonction de prototype

   `matrix_t addmatrices (matrix_t,matrix_t);`

   qui retourne la matrice résultant de l'addition des matrices passées en paramètres. Si ces matrices sont de dimensions différentes, cette fonction retourne `NULL`. Si chaque coefficient de la somme de ces matrices est zéro, on retourne une matrice zéro.

Rappels : L'allocation mémoire se fait par le biais de la fonction `malloc` et la désallocation par le biais de la fonction `free`.

4 Tri utilisant un tas binaire

L'objectif de l'exercice est d'implanter le tri d'un tableau d'entier en utilisant un *tas binaire*. (On souhaite trier le tableau afin d'avoir le plus petit élément au début et le plus grand à la fin).

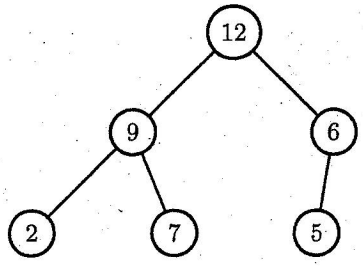
Tas binaire

Définition 1 *Un* tas binaire *est un arbre binaire complet — i.e. les nœuds de l'arbre peuvent être stockés de façon contiguë dans un tableau — ordonné en tas — les nœuds sont ordonnés par leurs clefs et les clefs des fils sont inférieures à celles des pères.*

Représentation d'un tas binaire par un tableau

Comme tout arbre binaire, un tas binaire peut être représenté dans un tableau unidimensionnel indicés à partir de 0 : le père d'un nœud en position i a pour enfants un fils gauche en position $2i + 1$ et un fils droit en position $2(i + 1)$.

Exemple. L'arbre



est codé par le tableau

12	9	6	2	7	5

Les tas binaires sont utilisés pour implanter les files de priorités car ils permettent des insertions en temps logarithmiques et un accès direct au plus grand élément.