TP d'algorithmique 2
Arbre Binaire de Recherche

On travaille sur des arbres binaires de recherche étiquettés par des entiers. Les arbres pourront contenir des doublons.

1. Fonctions de manipulation
   Ecrire les fonctions :

   - `int Ajout(Arbre *A,int x)` permettant d'ajouter l'entier `x` à l'arbre `*A`.
   - `Arbre Recherche (Arbre A,int x)` effectuant la recherche de l'entier `x` dans l'arbre `A`.
   - `int RechercheOccurence (Arbre A,int x)` qui renvoie le nombre d'occurence de l'entier `x` dans l'arbre `A`.
   - `Arbre ExtraireMax(Arbre *A)` qui effectue l'extraction du nœud contenant la plus grande valeur et renvoie un pointeur sur ce nœud
   - `Arbre ExtraireMin(Arbre *A)` qui effectue l'extraction du nœud contenant la plus grande valeur et renvoie un pointeur sur ce nœud
   - `int Supprime(Arbre *A,int x)` qui supprime de l'arbre la première occurence de l'entier `x`

2. On veut maintenant mémoriser la hauteur de l'arbre.
   Chaque nœud contient pour cela un champ indiquant la hauteur du sous arbre enraciné en ce nœud. On utilise la structure :

   ```
   typedef  struc neu{
   int valeur, int hauteur;
   struc neu *fg,*fd;
   }Noeud, *Arbre;
   ```

   Reprendre les fonctions précédentes afin de maintenir et mettre à jour l'information `hauteur`.
   Pour la suppression on choisira de supprimer l'extremum du sous arbre de plus grande hauteur.
   Remarque cela ne suffit pas pour garantir des opérations en $\mathcal{O}$(lg nombre de nœud).Il faut des opérations qui changent la structure de l'arbre (voir arbres AVL)