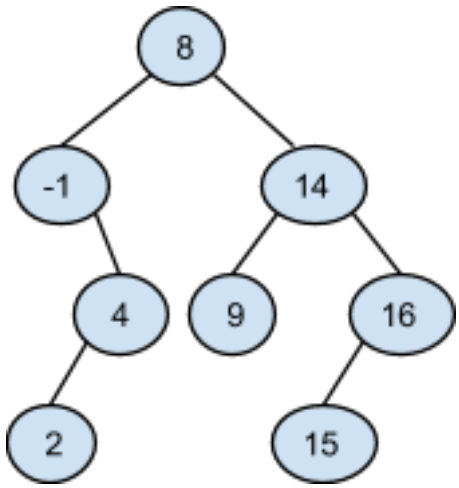
TP 8 SDA bis

Arbres binaires de recherche

Un arbre binaire de recherche est un arbre binaire dans lequel les éléments sont triés : le sous arbre gauche d'un noeud *n* contient uniquement des éléments inférieurs ou égaux à *n*, le sous arbre droit contient des éléments supérieurs à *n.*



En partant des arbres binaires du TP précédent, ajoutez les fonctions suivantes :

```
void insert(T elt, Arbre* a);       // ajoute elt a sa place dans l'arbre

Pour les 3 suivantes faire une version récursive et une version itérative:
T plusPetitElt(Arbre* a);           // rend l'élément le plus petit
T plusGrandElt(Arbre* a);           // rend l'élément le plus grand
Bool recherche(T elt, Arbre* a);    // elt est-il dans l'arbre
```

Pour éviter des problème d'affichage avec dot, on n'insérera pas un nombre qui est déjà dans l'arbre.

Insérer 100 nombres aléatoires [0,1000] dans un arbre binaire de recherche et afficher le. Afficher les résultats des fonctions plusPetit, plusGrand et recherche. Vérifier avec le graphe affiché avec dot.

Faire une fonction qui insère *n* nombres aléatoires [0,1000] dans un arbre binaire de recherche vide (NULL), qui calcule sa hauteur et le nombre de noeuds. Afficher et comparer avec les hauteurs min et max possibles.

Expérimenter en faisant varier n de 100 à 100000 (compiler avec optimisation -O3)
Que peut-on en conclure sur la complexité des fonctions plusPetit, plusGrand et recherche.