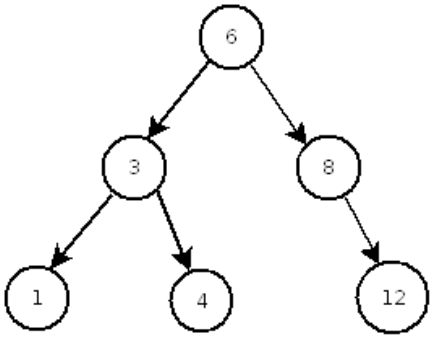
ESIAL 1A – Structures de Données
Implantation d'arbres binaires de recherche

L'objectif de ce TP est de réaliser et tester l'implantation d'un arbre binaire de recherche. Certains fichiers java vous sont fournis dans le répertoire `/home/depot/1A/SD/TP-Arbres`.

Rappel : *un arbre binaire de recherche est un cas particulier d'arbre binaire dans lequel toutes les valeurs du sous-arbre gauche sont inférieures à la valeur de la racine, elle-même inférieure à toutes les valeurs du sous-arbre droit. Les valeurs existent de manière unique dans l'arbre.*



1. Ecrire une classe `MyBinarySearchTree<T>` qui correspond à un arbre binaire de recherche et qui implante l'interface `BinarySearchTree<T>` fournie dans le paquetage. La classe possèdera trois attributs correspondant respectivement à la racine, au sous-arbre gauche et au sous-arbre droit.
   - Vous implanterez les différentes méthodes au fur et à mesure, et réaliserez en parallèle les tests à l'aide d'une classe `TestBinarySearchTree`. Les tests n'auront pas besoin d'être dérivés des axiomes.
   - Vous redéfinirez la méthode `toString()`, héritée de `Object`, pour permettre un affichage de l'arbre sous la forme :

     ```
     6 { 3 { 1 , 4 } , 8 { _ ,12 } }
     ```

   - L'interface `BinarySearchTree<T>` hérite de l'interface `BinaryTree<T>`. Dans notre cas de figure, cela implique qu'il faut à la fois que vous implantiez les méthodes de `BinaryTree<T>` et de `BinarySearchTree<T>`.
2. **Pour ceux qui avancent vite**, reprendre la classe `TestBinarySearchTree` en implantant les tests par dérivation des axiomes identifiés lors de la séance de TD.
3. **Pour ceux qui avancent vite**, proposer une seconde implantation `MySecondBinarySearchTree<T>` (tirant profit de la liaison dynamique).