TP : Arbre binaire et Arbre binaire de recherche

1.1 Travail demandé :

On souhaite pouvoir créer des arbres binaires et des arbres binaires de recherche.
Pour cela, la documentation javadoc des classes génériques Node, BinTree et SearchBinTree sont disponibles, ainsi qu'une énumération permettant de choisir le type de parcours de l'arborescence.

1.2 Classe Node

Un arbre binaire a comme attribut la racine de l'arborescence qui stocke les données. Cette racine est une instance de la classe **Node** qui possède trois attributs :

- Un élément de type **T**
- La référence du fils gauche et celle du fils droit

Les méthodes de la classe sont décrites dans la javadoc.

1.3 BinTree

Un arbre binaire est défini par la référence de sa racine instance de **Node.**
Quand la racine est nulle, l'arbre est vide.
L'ajout de valeurs dans l'arbre se fait de façon à conserver un arbre équilibré, dont les profondeurs des fils, gauche et droit, ne diffèrent que de 0 ou 1.
Quand l'arbre est vide, la valeur est ajoutée à la racine.
Ensuite, plusieurs techniques sont possibles. Par exemple, quand la racine a un fils null, la valeur est ajoutée à ce fils.
Quand les deux fils existent, on crée un nouveau nœud dont le fils gauche référence l'ancienne racine et le nouveau nœud devient la racine de l'arbre. Son fils droit est donc null.

Trois inner-class sont créées afin de permettre les parcours de l'arbre : infixe, prefixe et postfixe.
Ces classes implémentent l'interface Iterator. La classe BinTree implemente l'interface Iterable et permet au client d'obtenir un iterateur. Selon son choix de parcours, le client peut ainsi parcourir les données selon trois parcours différents.

Les méthodes de la classe sont décrites dans la javadoc.

1.4 SearchBinTree

Un arbre binaire de recherche **est un** arbre binaire, dont les valeurs sont insérées de façon imposée.
Quand l'arbre est vide, la valeur est ajoutée à la racine.
Dans les autres cas, on cherche récursivement l'emplacement de la nouvelle valeur selon l'algorithme suivant :

- si la valeur est ≤ à celle du nœud courant, et qu'il existe un fils gauche alors le nœud courant est positionné au fils gauche (on descend dans l'arbre), et on recommence,
- s'il n'existe pas de fils gauche, alors un nouveau nœud est créé dont la référence est assignée à ce fils gauche,
- si la valeur est > à celle du nœud courant, et qu'il existe un fils droit alors le nœud courant est positionné au le fils droit (on descend dans l'arbre), et on recommence,
- s'il n'existe pas de fils droit, alors un nouveau nœud est créé dont la référence est assignée à ce fils droit.

Avec un arbre binaire de recherche, le parcours infixe permet de récupérer les valeurs triées.

Les méthodes de la classe sont décrites dans la javadoc.