ISTITUT DE MATHEMATIQUES APPLIQUEES 20 Décembre 2017
LICENCE 2

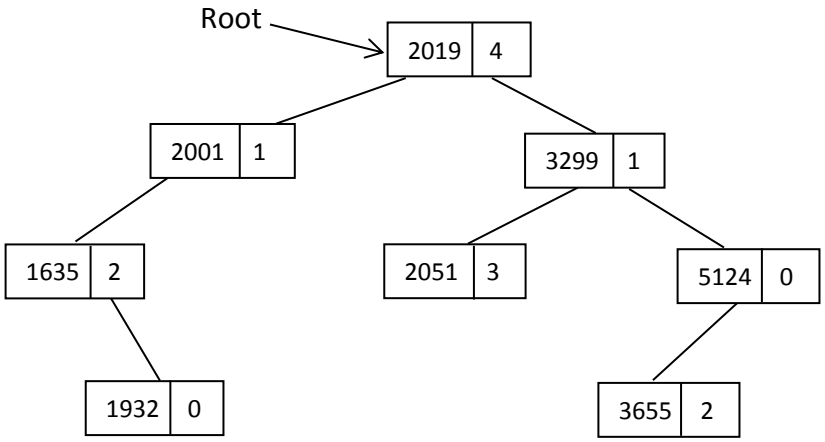
Algorithmique et Structuration des Données

Contrôle continu n°2

Durée : 1h30
Documents autorisés : NON
Calculatrice : NON

Exercice 1

Pour faciliter le prêt des livres d'une bibliothèque, on propose d'utiliser un arbre binaire de recherche contenant dans chaque nœud le numéro (un entier) et le nombre d'exemplaires d'un livre. L'arbre est trié selon le numéro des livres.



- Donnez la définition en Java de la classe `TNode` représentant un nœud de l'arbre. Si vous avez des *getteurs* et des *setteurs*, il suffit de donner seulement leur nom.

- Définissez ensuite la classe `BookTree` qui a un seul champ `root` donnant accès à la racine de l'arbre et donnez le code des méthodes suivantes :

- un constructeur qui initialise le champ `root` à `null`

- `public static void available()` qui affiche à l'écran le numéro des livres dont le nombre d'exemplaires est positif

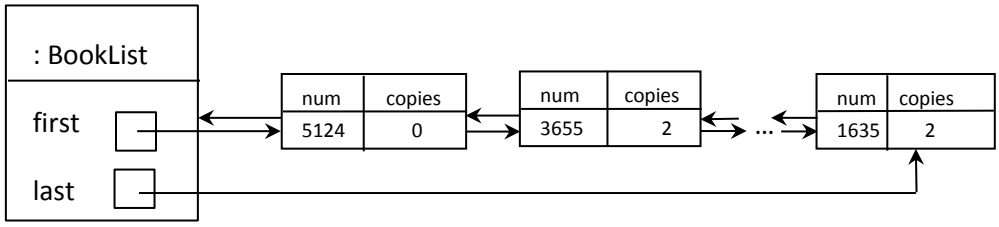
- `public int amount()` qui renvoie le nombre total d'exemplaires de tous les livres

- `private Node searchBook(int num)` qui renvoie le nœud contenant le livre numéro `num` et qui déclenche une `NoSuchElementException` si le nœud n'est pas trouvé

- `public void lend(int num)` qui décrémente de 1 le nombre d'exemplaires du livre numéro `num` si ce nombre est positif, affiche un message approprié dans le cas contraire et qui déclenche une `NoSuchElementException` si l'arbre n'a pas un nœud avec ce numéro

- Après l'inventaire certains livres peuvent être retirés des prêts. Quel est l'arbre obtenu si on supprime d'abord le livre numéro 2019, puis le livre numéro 3299 en utilisant l'algorithme donné en cours ?

Pour une recherche rapide on crée d'abord un arbre équilibré ayant le nombre de nœuds égal au nombre de livres puis remplir les nœuds des données des livres se trouvant dans une liste. La liste est doublement chaînée avec accès direct au premier et au dernier nœud de la liste dans laquelle les livres apparaissent en ordre décroissant de leur numéro. Chaque nœud contient également le numéro et le nombre d'exemplaires d'un livre.



- Donnez la définition de la classe `LNode` représentant un nœud de la liste. Si vous avez des *getteurs* et des *setteurs*, il suffit de donner seulement leur nom.

  Définissez la classe `BookList` avec les champ `first` et `last`, le constructeur qui les initialise à `null`, la méthode `getLast()` et les méthodes suivantes :

- `public void removeLast()` qui enlève le dernier nœud de la liste et qui déclenche une exception `NoSuchElementException` si la liste est vide.

- `public int size()` qui renvoie le nombre d'éléments de la liste.

- Définissez maintenant dans la classe `BookTree` la méthode :
  `private static void load(BookList blist, TNode r)` qui fait un parcourt gauche-racine-droit de l'arbre dont la racine est `r` et à chaque nœud de l'arbre :
  - y copie les données du dernier de la liste `blist`
  - et enlève ce nœud de la liste.

  Est-ce que l'arbre obtenu est un arbre binaire de recherche ?

- Soit dans la classe `BookTree` la méthode :
  `private void createTree(int n)` qui crée un arbre binaire équilibré de `n` nœuds dont la racine est `root`.

  Définissez alors le constructeur :
  `public BookTree(BookList blist)` qui crée un arbre équilibré contenant tous les livres se trouvant dans `blist`.

  Est-ce que l'arbre obtenu est trié selon le numéro des livres comme dans l'exemple ?