Programmation-Algorithmique

Deuxième Projet Gestion d'un fichier client

Ce deuxième projet devra se faire avec le même binôme que le précédent. Le rapport du projet contenant les sources commentés ainsi que l'explication des différents algorithmes utilisés, une brève explication de l'implémentation, et toutes autres informations utiles devra être rendu pour le 18/01/2002. Ce projet fera avec le précédent l'objet d'une soutenance commune fin janvier. Il devra impérativement fonctionner sur les machines de l'UFR.

1 Présentation générale

Le but de ce deuxième projet est de proposer deux autres implémentations du projet précédent. L'une à l'aide des tables de hachage et l'autres avec des arbres 2.3.4. On restreindra les commandes à implémenter de la manière suivante : les recherches et les suppressions se feront uniquement à partir du numéro de client.

2 Tables de hachage

On utilisera la classe Hashtable de Java.

3 Arbres 2.3.4

Pour cette implémentation, on donne ci-dessous les définitions nécessaire ainsi qu'une partie des algorithmes. Les algorithmes manquant devront être expliqués dans le rapport, et la complexité de **tous** les algorithmes devra être étudiée.

3.1 Définitions

Un i-nœud est un nœud qui contient $i-1$ éléments distincts et ordonnés $< x_1, x_2, ..., x_{i-1} >$ et i sous-arbres tels que :

- tous les éléments du premier sous-arbre sont inférieurs ou égaux à x_1 ;
- tous les éléments de $k^{\text{ème}}$ sous-arbre $(2 \leq k \leq i-1)$ sont supérieurs à x_{k-1} et inférieurs à x_k ;
- tous les éléments du $i^{\text{ème}}$ sous-arbre sont strictement supérieurs à x_{i-1}.

Un arbre-2.3.4 est un arbre général étiqueté dont chaque nœud contient un 2-nœud, un 3-nœud ou un 4-nœud et dont toutes les feuilles sont situées au même niveau.

Une feuille de cet arbre est un 2-nœud, un 3-nœud ou un 4-nœud dont tous les sous-arbres sont vides.

Donner un encadrement de la hauteur d'un tel arbre. Quelle caractéristique a la liste obtenue par un parcours en profondeur d'un arbre-2-3-4 ?

3.2 Recherche, adjonction, suppression

La recherche dans un arbre-2.3.4 suit le même principe que celle dans un arbre binaire de recherche.

Pour ajouter un élément x dans un arbre-2.3.4 :

1. Déterminer la feuille f où doit s'insérer x ;
2. Si f est un 2-nœud ou un 3-nœud, insérer x dans le nœud ;
3. Si f est un 4-nœud, éclater f en deux 2-nœuds contenants les éléments de f et x moins l'élément médian qui est ajouté au père de f ;
4. Si le père de f est un 2-noeud ou un 3-nœud, on peut faire l'adjonction directement sinon on éclate à nouveau ;
5. Si cet éclatement se propage jusqu'à la racine de l'arbre, la racine est elle-même éclatée en 2 nœuds et un nouveau 2-nœud est créé, ce nouveau nœud racine contient l'élément médian.

Vous devez trouver vous-même l'algorithme de suppression.