الجمهـورية الجزائريـة الديمقراطيـة الشعبيـة

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
Université 8 Mai 1945 Guelma
Faculté de mathématiques et de l'informatique et de sciences de la matière

وزارة التـعليم العالي والبحث العلمي
جـــامـعـة 8 مـــاي 1945 قالمـــة
كلية الرياضيات ، الاعلام الآلي و علوم المادة

Module : Structure de fichiers & Structure de données
TP3 : Indexation et Hachage

Note : Les programmes de cette série de travaux pratiques doivent être élaborés en langage C.

Exercice 1 : Indexation primaire

Développer un programme pour la gestion d'une bibliothèque, en utilisant la méthode d'index primaire basée sur les IDs des livres. Le programme doit offrir les fonctionnalités suivantes :

1. Utilisation des structures :
   a. Livre (ID, titre, auteur, année).
   b. Index (ID du livre, position dans le fichier de données).

2. Implémentation des fonctions essentielles :
   a. *ajouterLivre (Livre)*: ajouter un nouveau livre.
   b. *rechercherLivre (ID)*: rechercher un livre, implémenter une recherche dichotomique sur le tableau d'index.
   c. *supprimerLivre (ID)*: supprimer un livre, il faut d'abord tester si le livre existe déjà dans le tableau d'index en appelant la fonction *rechercherLivre (ID)*.
   d. *afficherContenuFichier ()*: afficher la liste des livres depuis le fichier de données.
   e. *afficherContenuIndex ()* : afficher le contenu du tableau d'index.
   f. *chargerIndexEnMemoire()* : charger le fichier d'index dans un tableau automatiquement à chaque ouverture du fichier des livres.
   g. *sauvegarderEtDechargerIndex()* : sauvegarder le tableau d'index sur le fichier index et libérer la mémoire.

3. Contrôle de la saisie pour garantir des IDs uniques à chaque livre. En cas d'ID existant, le programme doit afficher un message d'erreur et insister sur la saisie d'un nouvel ID.

4. Utilisation de fichiers binaires pour stocker les livres (livres.dat) et les index (livres.idx).

5. Affichage d'un menu interactif proposant des options pour ajouter, rechercher, supprimer, afficher la liste des livres et des index, et quitter le programme.

Pour tester le programme, suivez ce scénario :

1. Ajouter un livre (ID : 3, Titre : C, Auteur : C, Année : 2000)
2. Ajouter un livre (ID : 1, Titre : A, Auteur : A, Année : 2001)
3. Ajouter un livre (ID : 2, Titre : B, Auteur : B, Année : 2002)
4. Afficher le contenu du fichier de données
5. Afficher le contenu du fichier index
6. Supprimer le livre avec l'ID = 2
7. Afficher le contenu du fichier de données après la suppression
8. Afficher le contenu du fichier index après la suppression
9. Supprimer le livre avec l'ID = 3
10. Afficher le contenu du fichier de données après la deuxième suppression

11. Afficher le contenu du fichier index après la deuxième suppression
12. Rechercher le livre avec l'ID = 2
13. Rechercher le livre avec l'ID = 1
14. Quitter le programme
15. Relancer le programme
16. Afficher le contenu du fichier de données après le redémarrage
17. Afficher le contenu du fichier index après le redémarrage

Note : d'autres sous-programmes peuvent aussi être ajoutés pour simplifier le code.

Exercice 2 : Indexation par arbre binaire de recherche

Concevoir un programme pour gérer une bibliothèque de livres (Titre, Auteur, Année) avec une indexation par arbre binaire de recherche. Le programme doit permettre d'ajouter des livres à la bibliothèque, chercher des livres par leurs noms, afficher le contenu de la bibliothèque et l'arbre d'index. Utiliser des fichiers binaires pour stocker les données des livres et l'arbre d'index. L'arbre d'index doit être construit en suivant la séquence d'insertion des livres.

Exercice 3 :

Considérant un fichier texte contenant une liste de noms de personnes, chaque nom étant associé à un identifiant unique.

Exemple :

4 Oussama
6 Amina
8 Sarah
13 Hakim
15 Lazhar
16 Fadia

Ecrire un programme qui pourra lire ces données depuis le fichier et les stockera efficacement dans une table de hachage statique. Utiliser une fonction de hachage qui calcule la somme des chiffres de l'identifiant pour mapper les identifiants aux emplacements correspondants de la table de hachage. Pour gérer les éventuelles collisions, adoptez la méthode de résolution par chaînage avec des listes séparées.

Le programme devra suivre les étapes suivantes :

1. Lire le fichier contenant les données.
2. Générer et afficher la table de hachage en illustrant comment les identifiants sont distribués.
3. Offrir à l'utilisateur la possibilité de rechercher une personne en spécifiant son identifiant.

Exercice 4 :

Proposer une version du programme précédent en utilisant l'adressage ouvert avec le hachage linéaire. Cette technique consiste à gérer les collisions en explorant séquentiellement les emplacements suivants dans la table de hachage en cas de conflit. Le programme doit toujours garantir une insertion réussie sans entrer dans une boucle infinie et affiche un message si la table de hachage est pleine.