Université Mohamed Khider-Biskra
Département d'informatique

$2^{ème}$ année LMD
Série d'exercices N°2.2

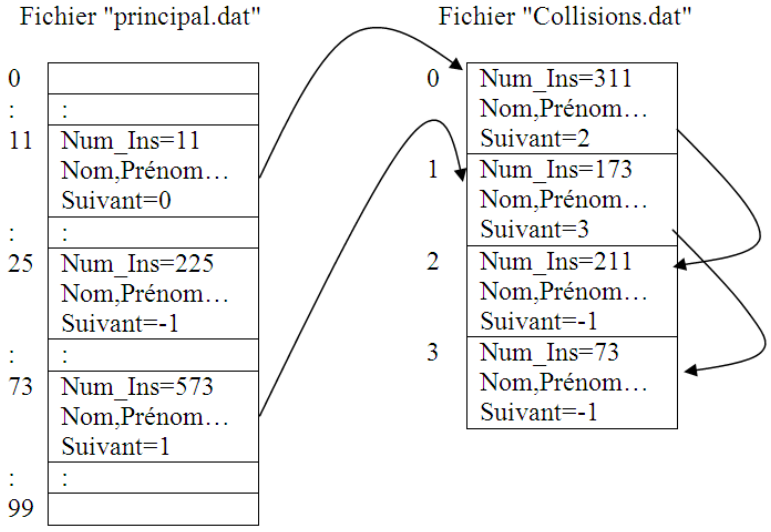
TD Algorithmique
Nomvembre 2023

Fichiers

Exercice 1 On souhaite développer un programme pour le stockage des informations des étudiants. Chaque étudiant est défini par les informations suivantes :

- Numéro d'inscription (clé) : 12336625
- Nom,
- Prénom,
- Date de naissance,
- Année d'étude : (2LMD, 3LMD, M1, M2, ...)
- Groupe : (G1, G2, ...)

On vous propose d'utiliser une méthode de Hash Coding sur les fichiers, utilisant les LLCs pour la gestion des collisions. La fonction de hachage est $H(x) = x \ mod \ 100$.
On vous propose d'utiliser deux fichiers : un fichier principal de 100 enregistrements représentant la table principale et contenant le premier enregistrement inséré pour chaque clé, et un deuxième fichier pour contenir les enregistrements provocant d'éventuelles collisions. A chaque fois qu'une collision se produise, on insère l'enregistrement correspondant dans le fichier des collisions et on le chaine aux enregistrements précédents par le numéro d'enregistrement dans le fichier collisions (voir la figure)



Ecrire en C :

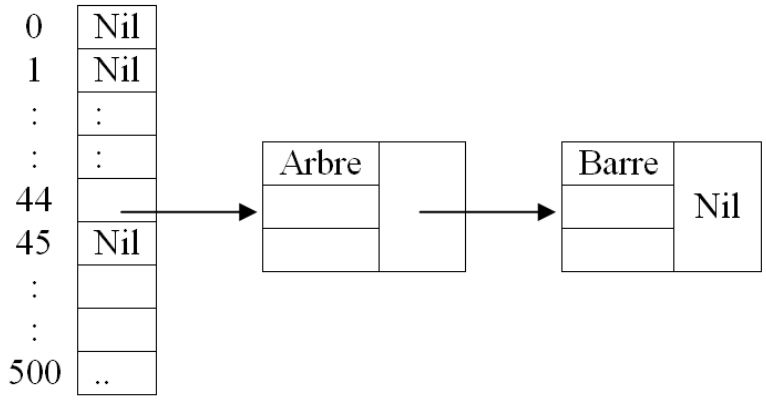
1. Les déclarations permettant de représenter de telles structures.
2. La procédure **Initialisation** permettant la création des deux fichiers et leur initialisation.
3. La procédure **Afficher(Num_Insc :integer)** permettant de rechercher l'étudiant dont le numéro d'inscription est égal à Num_Insc et d'afficher son nom et prénom s'il est trouvé.

4. La procédure **Ajouter(E : TEtudiant)** permettant d'jouter l'étudiant dont toutes les informations nécessaires se trouvent dans E.

Exercice 2 On souhaite implémenter un programme dictionnaire Français - Français - Anglais qui permette d'enregistrer les mots en français avec leur signification et leur traduction en anglais. Pour cela, on utilise un fichier "Dictio.dat" composé des champs suivants : Mot, Signification, Traduction.
Pour accélérer la recherche des mots, le fichier est transféré au début de l'exécution du programme dans une table de hachage composé de 500 enregistrements qui utilise la fonction de hachage suivante :

H(mot) = (la somme des positions des lettres du mot dans l'alphabet) mod 500

Exemple : $H('arbre') = (1 + 18 + 2 + 18 + 5)\ mod\ 500 = 44$
Pour résoudre les éventuelles collisions, on utilise la méthode des listes linéaires chaînées suivantes :



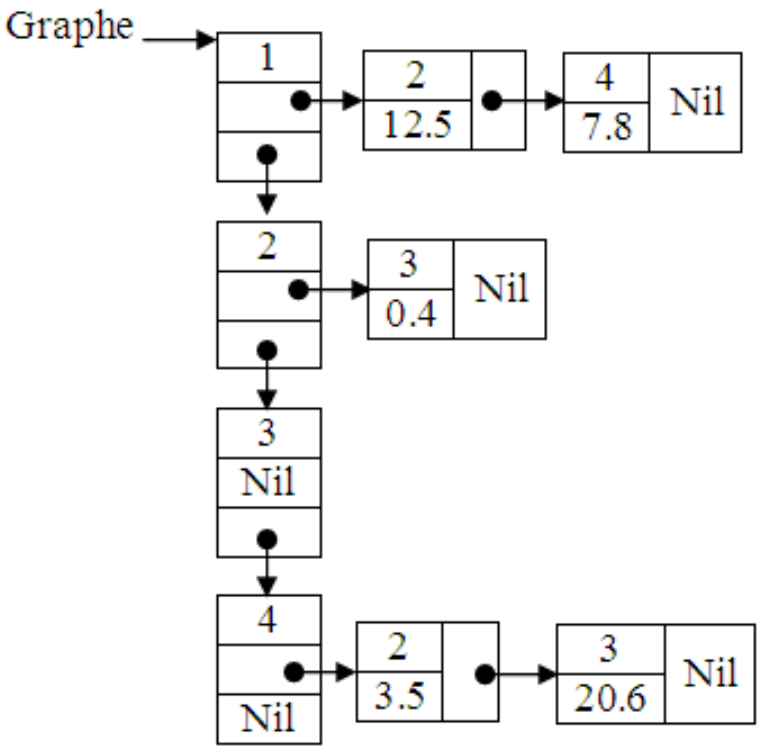
Après l'utilisation du dictionnaire et avant de quitter le programme, la table est transférée dans le fichier "Dictio.dat" qui doit contenir exactement les mots existant dans la table (aucun enregistrement vide) pour économiser l'espace de stockage.

Questions

1. Donner les déclarations nécessaires à la représentation de ces structures en mémoire.
2. Ecrire la procédure **Rechercher(mot, Pos , Exist)** qui recherche le mot m dans la table et retourne Exist à vrai et sa position dans Pos s'il existe, et Exist à faux sinon.
3. Ecrire la procédure **Ajouter_Mot(m)** qui permette d'ajuter le mot m au dictionnaire.
4. Ecrirre la procédure **Init_Dict** qui permette de charger le fichier dans la table.
5. Ecrire la procédure **Supprimer(mot)** qui enlève mot de la table.
6. Ecrire la procédure **Sauv-Dict** qui enregistre la table dans le fichier.

NB
– Toutes les procédures doivent être écrites en Pascal.
– La fonction Ord (car) retourne le code ASCII d'un caractère.
– Le code ASCII de 'A' = 65
– Le code ASCII de 'a' = 97

Exercice 3 On souhaite représenter un graphe orienté étiqueté en mémoire sous forme de listes d'adjacence comme suit :



Ecrire en C :

1. La déclaration des structures permettant de représenter un tel graphe.
2. La procédure **AjouterSommet(x : integer)** permettant d'ajouter un sommet x au graphe.
3. La procédure **AjouterArc(x,y :integer, d : real)** permettant d'ajouter un l'arc (x,y) de distance d au graphe.

On veut enregistrer ce graphe sur disque sous forme de deux fichiers : un fichier "Sommets.dat" contenant la liste des sommets et un fichier "Arcs.dat" contenant la lise des arcs comme suit :

Fichier « Sommets.dat »

1
2
3
4

Fichier « Arcs.dat »

Origine	Extrémité	Distance
1	2	12.5
1	4	7.8
2	3	0.4
4	3	3.5
4	2	20.6

Ecrire en C :

1. La procédure **CreerFichierSommets** permettant de créer le fichier "Sommets.dat" à partir du graphe de listes en mémoire.
2. La procédure **CreerFichierArcs** permettant de créer le fichier "Arcs.dat" à partir du graphe de listes en mémoire.
3. La procédure **CreerGraphe** permettant de créer le graphe de listes en mémoire à partir des deux fichiers "Sommets.dat" et "Arcs.dat" .