Rapport de TP

Listes chaînées

Oumaima Talouka – GI01

Guillaume Olympe – GB04

Daniel Suárez Escudero – GB04

Encadrant : M. Akheraz

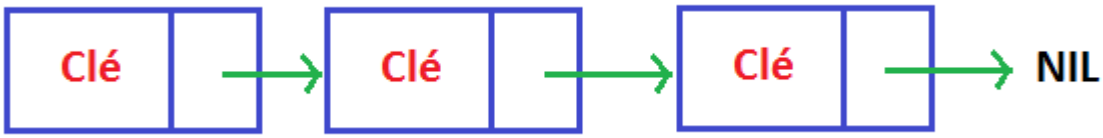
P15

utc
Université de Technologie
Compiègne

Introduction

Le but de ce TP est de programmer, en se servant de listes simplement chaînées, la gestion d'un magasin et des rayons qui le constituent. Bien sûr, le magasin peut contenir un nombre indéfini de rayons, qui peuvent à leur tour contenir un nombre indéfini de produits.

Les listes simplement chaînées sont une structure qui permet d'organiser facilement une suite d'éléments. Elles sont composées de cases, et chacune des cases contient une clé et un pointeur vers la case suivante. La dernière case pointe vers NIL.



Il sera ainsi affaire de créer une liste chaînée pour organiser une liste de magasins, une liste chaînée pour organiser une liste de rayons et une liste chaînée pour organiser une suite de produits.

A noter que magasins, rayons et produits sont définis grâce à des structures en C.

Forme du programme

Le programme se compose de diverses fonctions qui permettent de créer des listes (de magasins, de rayons ou de produits), d'ajouter des éléments dans les listes (ceci en respectant l'ordre alphabétique), de supprimer des éléments, d'afficher les éléments d'une liste et de rechercher un élément particulier dans une liste.

Le programme propose dans un menu principal les choix suivants :

1. Créer un magasin
2. Ajouter un rayon au magasin
3. Ajouter un produit dans un rayon
4. Afficher les rayons du magasin
5. Afficher les produits d'un rayon
6. Retirer un produit
7. Supprimer un rayon
8. Rechercher un produit par prix
9. Quitter

Fonctions

Liste des fonctions demandées :

Ajouter_Produit : Cette fonction permet d'ajouter un produit dans un rayon, en respectant l'ordre alphabétique et en ne permettant pas d'ajouter deux fois le même produit. Elle renvoie 1 si l'ajout s'est bien passé et 0 sinon.

Retirer_Produit : Cette fonction permet de retirer un produit d'un rayon. Elle « coupe » le lien du produit à éliminer dans la liste chaînée des produits et libère la mémoire qu'il utilise. L'ordre alphabétique est bien sûr respecté lors de la suppression.

Ajouter_Rayon : Cette fonction permet d'ajouter un rayon dans un magasin, en respectant l'ordre alphabétique et en ne permettant pas d'ajouter deux fois le même produit. Elle renvoie 1 si l'ajout s'est bien passé et 0 sinon.

Supprimer_Rayon : Cette fonction permet de retirer un rayon d'un magasin. Elle « coupe » le lien du rayon à éliminer dans la liste chaînée des rayons et libère la mémoire qu'il utilise. L'ordre alphabétique est bien sûr respecté lors de la suppression.

Afficher_Rayon : Cette fonction affiche les produits triés contenus dans un rayon.

Afficher_Magasin : Cette fonction affiche les rayons triés contenus dans un magasin (sans les produits).

Recherche_Produit : Cette fonction renvoie les produits situés dans une fourchette de prix rentrée par l'utilisateur. De cette manière elle parcourt tous les rayons du magasin et tous les produits de chaque rayon, et affiche ceux dont le prix est compris dans l'intervalle demandé. L'utilisation de structures résulte très avantageux, en effet, uniquement en récupérant l'adresse du produit qui convient, on a accès à toutes ses caractéristiques (marque, quantité, qualité).

Fonction bonus :

TraiterListeAchat : Cette fonction permet à l'utilisateur de saisir une liste de produits à acheter (rayon, marque, quantité). Les produits choisis sont ainsi retirés des rayons correspondants. Bien sûr, les champs sont mis à jours. Elle affiche le prix total des produits achetés et la liste des produits qui ne sont pas disponibles (dans le cas où le nombre de produits demandés dépasse la quantité en stock).

Liste des fonctions supplémentaires :

Select_Rayon : Cette fonction renvoie l'adresse d'un rayon voulu via un numéro. Il est plus facile de sélectionner le rayon avec un numéro que de taper le nom entier du rayon. Nous avons mis en place cette fonction uniquement pour les rayons, mais nous aurions pu également le faire pour les produits.

SupprimeProduitsRayon : Cette fonction est appelée dans la fonction Supprimer_Rayon. Elle permet de libérer la mémoire utilisée par les produits contenus dans le rayon à supprimer. Nous avons choisi d'utiliser cette fonction du fait de sa simplicité, mais nous aurions pu également faire appel à la fonction Retirer_Produit.

Modification (correction) du programme suite à la démonstration

Suite à notre démonstration, nous nous sommes rendu compte que lors de la suppression d'un rayon du magasin, nous ne supprimions pas les produits contenus dans ce rayon, donc ces éléments restaient stockés dans la mémoire même si le nom du rayon n'apparaît plus sur la base du magasin et malgré l'utilisation du « free » pour la suppression du rayon qui a bien été employée afin de libérer de la mémoire. En vu d'une correction éventuelle de ce problème, nous avons décidé d'ajouter la fonction SupprimeProduitsRayon et l'insérer dans la fonction Retirer_rayon, pour supprimer tous les produits du rayon tout en libérant la mémoire qu'ils occupaient, pour ensuite supprimer le rayon lui-même et libérer enfin la mémoire allouée pour ce dernier élément.

Complexité

Ajouter_Produit : O(n) où n est le nombre de produits dans le rayon.

Retirer_Produit : O(n) où n est le nombre de produits dans le rayon.

Ajouter_Rayon : O(n) où n est le nombre de rayons dans le magasin.

Supprimer_Rayon : O(n) où n est le nombre de rayons dans le magasin.

Afficher_Rayon: O(n) où n est le nombre de produits dans le rayon. Cependant, bien qu'il y ait n itérations, on pourrait considérer une complexité O(1) du fait qu'il n'y a pas d'affectations dans la fonction.

Afficher_Magasin: O(n) où n est le nombre de rayons dans le magasin. Cependant, bien qu'il y ait n itérations, on pourrait considérer une complexité O(1) du fait qu'il n'y a pas d'affectations dans la fonction.

Recherche_Produit: O(n*m) où n est le nombre de produits dans le rayon et m et le nombre de rayons dans le magasin.

TraiterListeAchat: O(k*n*m) où n est le nombre de produits dans le rayon, m et le nombre de rayons dans le magasin et k et le nombre de produits à traiter.

Select_Rayon: O(n) où n est le nombre de rayons dans le magasin.

SupprimeProduitRayon: O(n) où n est le nombre de produits dans le rayon.

Conclusion

Ce TP nous a permis d'appréhender la manipulation des listes chaînées en C. Nous avons notamment manipulé des pointeurs, des fonctions et des structures, en les mettant en commun pour créer un programme efficace et fonctionnel. Enfin, nous avons appris à réfléchir en matière de d'amélioration de nos algorithmes dans le but de réduire la complexité des fonctions codées.