TP4 AP3 - Liste simplement chaînée
Application à la gestion d'ensembles

But du TP :

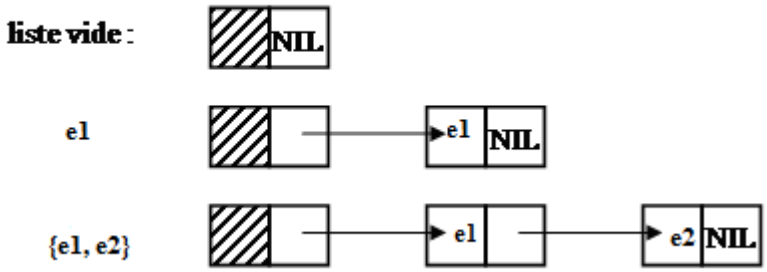
Le but de ce TP est d'implanter une liste simplement chaînée dans le langage C. Vous utiliserez ensuite cette structure de données pour gérer un ensemble d'étudiants.

Structure de données *etudiant* :

Réaliser une structure etudiant contenant un nom, un prénom, un numéro de téléphone et un âge. Les champs nom, prénom, numéro de téléphone sont de type chaine de caractère et l'âge est de type entier.

La structure de données *t_ensemble* :

La structure de données utilisée sera une liste chaînée. Toutes les listes chaînées auront une tête fictive. Une tête fictive est une cellule dont la donnée n'est pas utilisée (zone hachurée) et un pointeur pointant sur la première cellule de la liste. Une liste vide est donc une cellule fictive dont le pointeur est à NIL (NULL). Les cellules de la liste chaînée doivent être créées dynamiquement (à l'aide de l'opérateur malloc).



Comment gérer un ensemble d'étudiants avec une liste chaînée ?

A l'issue du TP, vous aurez implanté une unité de gestion de liste chaînée avec les sous-programmes nécessaires pour ajouter, supprimer et afficher le contenu de la liste. Les fonctions d'ajout et de suppression serviront à gérer l'ensemble. Si un élément est déjà dans la liste chaînée, il ne doit pas être ajouté de nouveau : un élément est présent une et une seule fois dans un ensemble.

Travail à faire

Vous devez organiser votre programme en 3 fichiers différents. Deux fichiers pour l'unité et un fichier pour le programme principal. Les deux fichiers de l'unité sont : un fichier ensemble.h où vous déclarerez les sous-programmes et un fichier ensemble.c où vous définirez les sous-programmes du fichier .h

a Le fichier ensemble.h

Le fichier ensemble.h contient les déclarations des sous-programmes suivants :

1. t_ensemble vide();
   Renvoie un ensemble vide.
2. bool est_vide(t_ensemble E);
   Renvoie vrai si l'ensemble est vide.
3. void inserer (t_ensemble E, etudiant x);
   Insère l'étudiant x passé en paramètre au début de l'ensemble E (même s'il existe déjà dans la liste).
4. void afficher(t_ensemble E);

Affiche les étudiants de l'ensemble E.

5. void free(t_ensemble E);
   Libère la mémoire lorsque l'on n'a plus besoin des cellules.
6. bool est_dans(t_ensemble E, etudiant x) ;
   Renvoie vrai si l'étudiant x est dans l'ensemble E.
7. void ajouter_element(t_ensemble E, etudiant x);
   Ajoute un étudiant dans l'ensemble E sauf si il y est déjà. (Utiliser la fonction est_dans).
8. int cardinal (t_ensemble E);
   Renvoie le nombre d'étudiants de E.
9. t_ensemble union(t_ensemble E; t_ensemble F) ;
   Renvoie un ensemble union des deux précédents.
10. La fonction bool egal (t_ensemble E, t_ensemble F)
    Renvoie true si E est égal à F, false sinon
11. La fonction t_ensemble intersection (t_ensemble E, t_ensemble F)
    Renvoie une liste contenant les étudiants communs à E et F
12. La fonction bool inclusion (t_ensemble E, t_ensemble F)
    Renvoie true si E est inclus dans F, false sinon
13. La fonction t_ensemble difference (t_ensemble E, t_ensemble F)
    Renvoie l'ensemble E \ F
14. Une implémentation de la fonction difference d'une autre façon avec le prototype suivant : void difference (t_ensemble E, t_ensemble F) où le résultat de E \ F sera renvoyé dans E
15. Une fonction de lecture qui créée une liste à partir des données d'un fichier.
16. Une fonction d'écriture qui sauvegarde les données d'une liste dans un fichier.
17. Pour chaque sous-programme ci-dessus, déterminer sa complexité.

b Le fichier ensemble.c

Définir les sous-programmes déclarés dans le fichier ensemble.h.

c Programme principal

Pour tester votre unité ***ensemble (.h et .c)***, vous devrez créer un programme principal (prog.c) qui implante les fonctionnalités suivantes :

- créer deux ensembles vides d'étudiants A et B.
- afficher A et B
- Créer 5 étudiants e1, e2, e3, e4, e5, e6, e7.
- A={e1,e2,e3}
- B={e4,e5}
- afficher A et B
- Utiliser vos fonctions de lecture/écriture
- ajouter e6 à l'ensemble A
- afficher(A)
- ajouter e7 à l'ensemble B
- afficher(B)
- Retirer e3 de l'ensemble A
- afficher(A)
- créer C= AUB

d Facultatif

Ajouter dans la structure etudiant, un champ liste chainée de notes et réécrivez la fonction insérer un étudiant dans une liste avec ses notes, puis l'afficher.