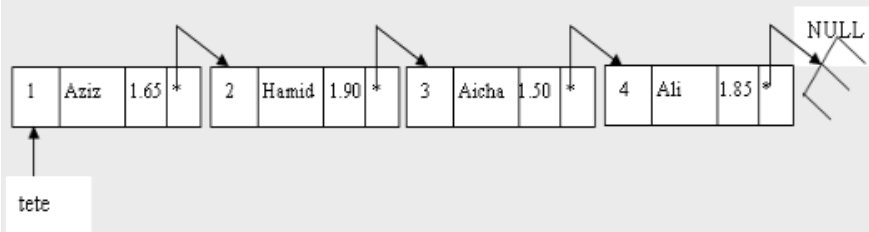


TP Listes Chaînées

EXERCICE 1

Soit la liste chaînée, représentée par le schéma ci-dessous, représentant des données d'individus.

Pour chacun on mémorise les données suivantes :[son matricule, son nom et sa taille].



Question 1 :

Donner les déclarations permettant de créer cette liste.

Proposer deux façons différentes

- la structure de chaque maillon comprend quatre champs (trois champs de données et un pointeur)
- la structure d'un élément de la liste comprend deux champs (un champ de données sous forme de structure à trois champs et un pointeur).

Question 2:

Ecrire une fonction `existe()` qui reçoit la tête de la liste et le matricule d'un individu et renvoie vrai si l'individu se trouve dans la liste et faux sinon..

Question 3 :

Ecrire une fonction `C` qui permet d'ajouter un individu en tête de la liste s'il n'y existe pas. Cette fonction reçoit la tête de la liste et les données de l'individu et renvoie la tête de la liste.

Question 4 :

Ecrire une fonction qui retourne le nombre d'individus dans la liste dont la tête est passée par paramètre.

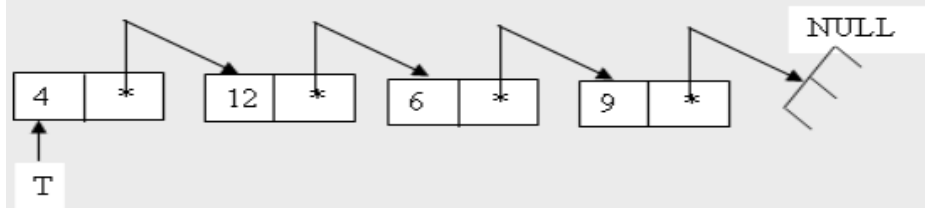
Question 5 :

Ecrire une fonction qui reçoit la tête de la liste et renvoie la moyenne des tailles des individus de la liste.

EXERCICE 2

Soit une liste chaînée d'entiers.

Schéma :



Question 1 :

Déclarer une liste chaînée de tête T globale.

Dans les questions qui suivent on considère la liste chaînée de tête T une liste non vide (contient des éléments)

Question 2 :

Ecrivez une fonction qui affiche les données de la liste T.

Question 3 :

Ecrivez une fonction qui étant donné un entier x passé comme paramètre, recherche dans la liste T, et renvoie l'adresse du premier maillon portant la valeur x, ou NULL si un tel maillon n'existe pas.

Question 4 :

Ecrivez deux fonctions qui permettent d'ajouter un entier x passé comme paramètre dans la liste T respectivement :

- en tête de liste ;
- en queue de liste;

Question 5 :

Ecrivez une fonction qui recherche et supprime le premier maillon portant la valeur x passé comme paramètre. Si un tel maillon n'existe pas la liste reste inchangée.

EXERCICE 3 :

La partie B du DS1 MP CPGE Ibtahir:

Dans cette partie, on se propose de représenter des polynômes à coefficient réels par des listes chaînées définies en langage comme suit :

```
struct monome
{ float coefficient;      /* le coefficient du monôme
  int exposant;          /* le degré du monôme
  struct monome *suiv;    /* l'adresse de l'élément suivant
};
typedef struct monome monome;
typedef monome* polynome;
```

Appellation :

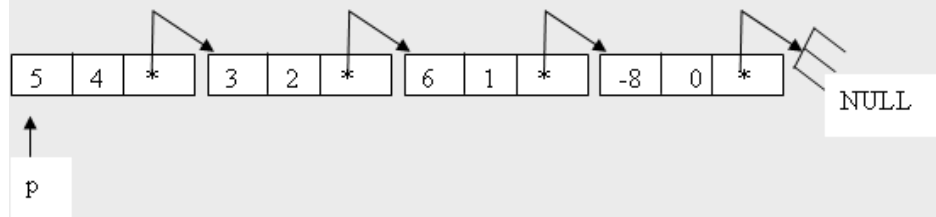
On appellera polynôme liste p , une liste chaînée d'éléments de type *monome* (définie plus haut) et possédant les propriétés suivantes :

- Le premier élément a l'adresse p .
- Le dernier élément a dans son champ *suiv* la valeur *NULL*.

Exemple :

Le polynôme $P=5x^4+3x^2+6x-8$ sera représenté par le polynôme liste p comme suit :

Schéma :



Question 1 : saisie de polynômes

Soit la déclaration globale suivante : polynome p;

+ Écrire une fonction d'entête : void saisir() qui permet de saisir le polynôme liste p. Tapez un exposant négatif pour mettre fin à la saisie.

Rappel : L'appel malloc(t) (t étant un entier positif), permet d'allouer t octets dans la mémoire dynamique et retourne l'adresse mémoire du block alloué. La fonction malloc() est définie dans le fichier de la bibliothèque stdlib.h

Question 2 : calcul de la valeur d'un polynôme

+ Ecrire une fonction qui renvoie la valeur du polynôme liste p pour une valeur donnée x passée comme paramètre à la fonction.

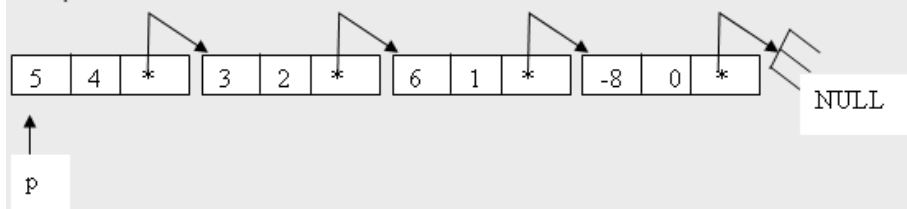
Question 3 : multiplication d'un polynôme par un monôme.

+ Écrire une fonction C qui renvoie le polynôme liste résultat de la multiplication d'un polynôme liste passés comme paramètre par un monôme de la forme ax^n ou a et n sont respectivement un réel et un entier passés aussi comme paramètres à la fonction.

Question 4 : Affichage du polynôme liste.

+ Écrire une fonction en langage C qui affiche le polynôme liste p sous la forme d'affichage conventionnel des polynômes.

Exemple :



Pour le polynôme liste p représenté dans le schéma ci-dessus, la fonction affiche :

$A=5x^4+3x^2+6x^1-8$

EXERCICE 4:

Gestion des données des élèves d'une classe.

Soient les déclarations suivantes :

```
typedef struct fiche
{int mat ;
char nom[25] ;
float note_math ;
float note_pc ;
float moyenne ;
} fiche;
struct liste
{fiche donnee ;
struct liste *next ;
};
typedef struct liste liste ;
typedef liste * llist;
llist tete=NULL;
```

Question 1 : Ecrire une fonction C d'entête : void saisirfiche(fiche *f) qui permet de saisir les données d'une fiche.

Question 2 : Ecrire une fonction C qui permet de saisir les données des élèves d'une classe dans une liste chaînée globale de tête tete. Le nombre des élèves n'étant pas connu à l'avance, on termine la saisie des élèves en introduisant un matricule négatif.

Question 3 : Ecrire une fonction qui retourne le nombre des élèves dans la liste tete.

Question 4 : Ecrire une fonction qui cherche un élève dans la liste sachant son matricule, cette fonction retourne le pointeur sur le maillon représentant cet élève et NULL si elle ne le trouve pas.

Question 5 : Ecrire une fonction qui supprime un élève de la liste sachant son matricule. Exploiter la fonction précédente pour le chercher.

Question 6 : Ecrire une fonction d'entête : void maj() , qui met à jour le champ moyenne sachant que le coefficient des math est 14 et celui de la physique chimie est 13.

Question 7 : Ecrire une fonction qui retourne la moyenne de la classe.

Question 8 : Ecrire une fonction C qui affiche les élèves de cette liste qui sont admis c à dire qui ont une moyenne supérieure ou égale à la moyenne de la classe.

EXERCICE 6

Tri d'une liste chaînée En utilisant un tri par insertion.

Soit une liste chaînée dont les chainons ont la structure suivante:

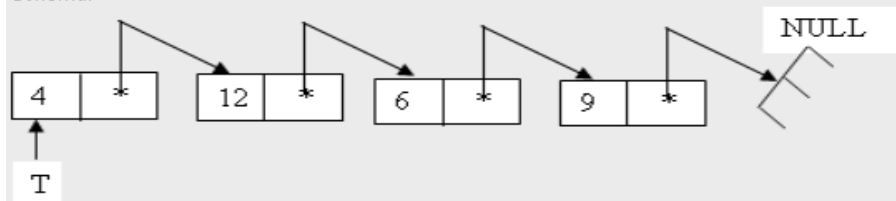
```
struct suite
{int val;
struct suite *net;
};
typedef struct suite suite;
typedef suite* llist;
```

- Ecrire une fonction qui permet d'insérer un chainon passé comme paramètre dans une liste chaînée passée comme paramètre elle aussi.
- Ecrire une fonction qui permet de trier une liste passée comme paramètre en exploitant la fonction précédente.
- Ecrire deux fonctions ajout et affichage qui permettent successivement d'ajouter un élément a une liste et qui affiche tous les elements d'une la liste.
- Donner la fonction main() pour tester les fonctions ci dessus.

EXERCICE 7

- Donner les déclarations nécessaires qui permettent de créer des listes chaînées de valeurs entières avec val comme champ de donnée et next comme pointeur sur le chaînon suivant.

Schéma:



- Ecrire une fonction qui permet d'ajouter un élément à une liste passé comme paramètre.
- Ecrire une fonction qui affiche les éléments d'une liste passée comme paramètre
- Ecrire une fonction qui retourne la somme des valeurs d'une liste passée comme paramètre
- Ecrire une fonction qui cherche une valeur dans une liste et retourne un pointeur sur la première occurrence de cette valeur
- Ecrire une fonction qui supprime une valeur de la liste, la liste et la valeur sont passées comme paramètres
- Ecrire une fonction qui insère un chaînon dans la liste, le pointeur sur le chaînon et la liste sont passés comme paramètres.
- Ecrire une fonction qui trie une liste en utilisant un tri par insertion
- Ecrire une fonction qui reçoit une liste et l'éclate en deux sous listes, une renferme les valeurs positives et l'autre les valeurs négatives.
- Dans la fonction main() présenter un menu qui présente les traitements à réaliser sous forme d'options à choisir.

MENU PRINCIPAL

- 1 - Ajout.....
- 2 - Afficher la liste.....
- 3 - Supprimer
- 4 - Insérer
- 5 - Trier la liste.....
- 6 - Somme des éléments.....
- 7 - Eclater la liste.....
- 8 - Quitter