Fiche de TD/TP/listes

Syntaxe :

```
Type maillon= structure   {valeur : type ;
                           Lien : pointeur sur maillon ;} ;
 L : pointeur sur maillon;
```

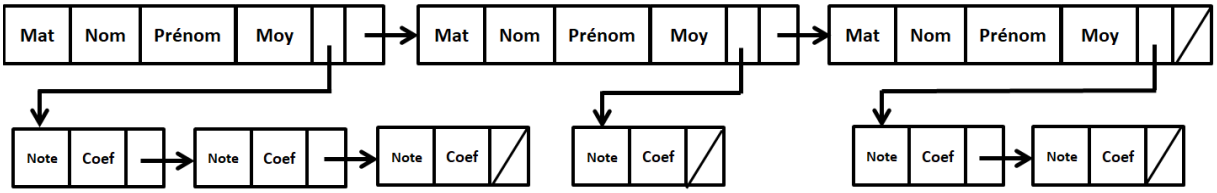
Pour éviter de réécrire le type : pointeur sur maillon on peut déclarer la liste comme suit :

```
 Type ptr : pointeur sur maillon ;
Type maillon= structure   {valeur : type ;
                           Lien : ptr;} ;   L : ptr;
```

Exercice 1

Le département d'informatique souhaite gérer l'ensemble de ses étudiants à travers une liste chainée. Chaque étudiant est représenté par son Matricule alphanumérique, son Nom, son prénom et sa moyenne générale.
Chaque élément de la liste représente un étudiant qui pointe vers :

1. L'étudiant suivant
2. L'ensemble des notes obtenues par l'étudiant avec coefficient.



1-)Donner les déclarations d'une telle liste.
2-) Ecrire une procédure qui calcule la moyenne de l'ensemble des étudiants sur une telle liste.

Exercice 2

Soit L une liste simplement chainée d'entiers, accessible par pointeur.

1- Donner la déclaration de L

2- Ecrire une action qui à partir de L crée une liste L', où chaque élément est la factoriel de l'élément correspondant de L.

EX : L= (4, 2, 0, 1, 3) L'= (24, 2, 1, 1, 6)

3- Ecrire un algorithme qui construit L' et affiche les éléments de L'

Exercice 3

Soit L une liste chaînée d'entiers.

a. Recherche itérative d'un élément e dans la lise L : en sortie, on souhaite obtenir un pointeur sur L'élément recherché s'il existe, Nil sinon.
b. Recherche récursive d'un élément e dans la lise L : en sortie, on souhaite obtenir un pointeur sur L'élément recherché s'il existe, Nil sinon.

Exercice 4

Ecrire un sous-algorithme qui inverse l'ordre des éléments d'une liste chaînée, et qui retourne un pointeur sur le premier élément de la liste nouvellement formée.