Centre Universitaire de Mila

Institut des sciences et de la technologie

2eme année LMD informatique

Module A.S.D

**Série de TP N 02**

**Listes chainées**

Une liste linéaire chaînée (LLC) est un ensemble de maillons, alloués dynamiquement, chaînés entre eux.

* Un maillon est un enregistrement avec deux champs : champ Elément contenant l’information. Champ Suivant donnant l’adresse du prochain maillon.
* Une liste chainée est représentée par un pointeur contenant l’adresse du premier élément (Maillon) de la liste.
* La liste vide, sans éléments, a pour référence NULL.

**Définition du type Liste en C++**

struct maillon {

 type\_des\_élements ele; // déclaration d’un maillon en C++

 maillon \*suivant;

};

typedef maillon \*liste; // définition de type liste

**Exemple :** Déclaration d’une liste chainée d’entier

# include <iostream>

struct maillon {

 int ele; // typeqq

 maillon \*suivant;

}

typedef maillon \*liste;

int main ( )

{ Liste L=NULL, L1 ;

. . .

}

**Les opérations primitives sur les listes chainées sont :**

void ajouter (Liste &L, int x) ; // ajoute un élément en tete de la liste L

Liste ajouter (Liste L, int x) ; // ajouter comme fonction

int premier (Liste L) ; //retourne le premier element de la liste L

bool est\_vide (Liste L) ; // verifie si L est vide ou non

Liste reste (Liste L) ; // retourne l’adresse du deuxième élément

int longueur (Liste L)  // retourne le nombre d’élément de la liste L.

**Exercice 1** (Template du programme est dans le site e-learning)

Nous considérons une liste chainnée d’entiers dont la déclaration est sous-dessous on vous demander de :

1. Compléter le programme sous-dessous par la définition des fonctions: Ajouter (qui permet d’ajouter un élément dans la liste d’entier L), afficher (qui permet d’afficher tous les éléments de la liste L) et somme qui retourne la somme des éléments de la liste L.
2. En utilisant des fonctions itératives (boucles) améliorer le programme pour qu'il permette aussi:
3. D’afficher le maximum de liste L.
4. De rechercher un élément dans la liste.
5. D’afficher le dernier élément de la liste L.
6. De supprimer le dernier élément de la liste L.
7. De vérifier si la liste eL est triée ou non.

**/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* TP2.cpp \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**

# include <iostream>

using namespace std;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Définition du type liste\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

struct maillon{

 int ele;

 maillon \*suivant;

};

typedef maillon \*liste;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*déclaration fonctions\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void afficher\_menu();

void ajouter (liste &L, int x); // ajoute un element dans la liste L

int premier (liste L); // retourne le premier élement de la liste L

void afficher (liste L); // affiche tous les élements de la liste L

int somme (liste L); // reourne la somme des éléments de la liste L

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Le programme principale \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int main()

{

liste L=NULL; int choix;

do

{ afficher\_menu();

 cin>>choix;

 switch (choix)

 {

 case 1: int x; cout<<"Entrez un entier X: "; cin>>x; ajouter(L, x); break;

 case 2: cout<<"les elements de la liste sont: "<<endl; afficher(L); break;

 case 3: cout<<"la somme des elements de la liste est :"<< somme(L); break;

 }

}while (choix !=0);

 system("pause") ;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*définition des fonctions\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void afficher\_menu()

{

 system("cls");

 cout<<"\n\n- Choisir une option dans le menu suivant:";

 cout<<"\n\t + Ajouter un element dans la liste <1>:";

 cout<<"\n\t + Afficher les elements de la liste <2>:";

 cout<<"\n\t + Afficher la sommes des elements de la liste <3>:";

 cout<<"\n\t + pour quiter <0>:";

 cout<<"\n\n- Entrez votre choix:";

}

.

.

.

**Exemple du résultat final d’exécution :**

****

**Exercice 2:** Modefier le programme l’exercice 1 par la définition des versions recursives (au lieu d’itératives) pour les fonctions itératives utilisées.