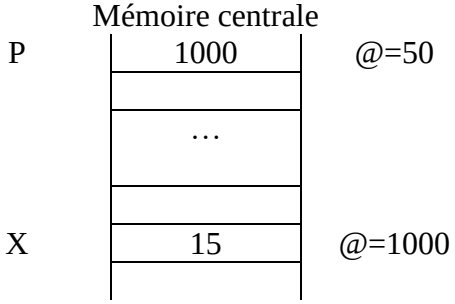
Listes chainées

1. Le type pointeur

1.1.Définition :

- En programmation, Un **pointeur** est une variable contenant une adresse mémoire.

Exemple :



x= 15

P un pointeur contient l'adresse de x dans la mémoire

P= 1000

1.2.Déclaration d'un pointeur :

En algorithmique	En langage C++
<Nom de la variable> : * Type de la variable ;	***Type de la variable * <Nom de la variable> ;***
Exemples :	**Exemples :**
p : *****entier;	int * P ;
Q :*réel ;	float * Q ;

Exemple 1:

```
#include<iostream>
int main ( )
{
    int x=4 ;        //déclaration d'une variable entier x
    int *p ;                 //déclaration d'une variable pointeur p
    p=&x ;                   //p pointe x ( p contient l'adresse de x)
    cout<< *p ;              // imprime le contenu de x
    getchar();
    return 0;
}
```

Résultat : 4

Example 2:

```
#include< iostream >
Int main ( )
 {
    int x=4 ;              //déclaration d'une variable x
    int *p ;               //déclaration d'un pointeur p
    P=&x ;                 //p pointe x ( p contient l'adresse de x)
   *p=15 ;                 // la valeur 4 de x est remplacée par 15
    cout<< x  ;            // imprime le contenu de x
    Return 0 ;
  }
```

Résultat : 15

Exemple 3:

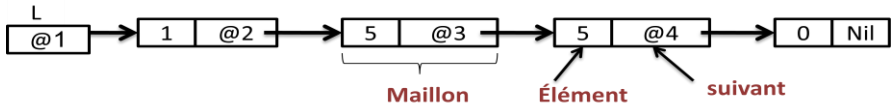
```
#include< iostream >
int main ()
 {
    int *p ;          //déclaration d'un pointeur p
    P=new int ;       //Allocation dynamique de l'espace mémoire
    *p= 10 ;          // Affecter la valeur 10 à l'espace méoire pointé par p
     cout<< *p;       // imprime le contenu de x
     Return 0 ;
  }
```

2. Structure de données « Liste »

2.1. Définition

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.



2.2. 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 ;
    . . .
}
```

2.3. 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.

Exemple :

Nous considérons une liste chainnée d'entiers dont la déclaration est sous-dessous. On vous demander de 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.

```
include<iostream>
using namespace std;
/***************definition de type liste*****************************/
  struct maillon{
        int ele;
        maillon *suivant;
    };
  typedef maillon *liste;
/***************declaration des fonctions*****************************/
void ajouter (liste &L, int x)// ajouter un element dans la liste L
 {
    liste m;
    m=new maillon;
    m->ele=x;
    m->suivant=L;
    L=m;
 }
void afficher(liste l)
{
    liste courant=l;
    while(courant!=NULL)
    {
          cout<<courant->ele<<endl;
          courant=courant->suivant;
    }
}
```

```
int somme(liste l)
{
    liste courant=l; int s=0;
    while(courant!=NULL)
    {
          s=s+courant->ele;
          courant=courant->suivant;
    }
    return s;
}
/***************Le programme principale*****************/
int main()
{
liste L=NULL;
int choix;

do
{
    cout<<"liste de choix :"<<endl;
    cout<<"1- Ajouter un element"<<endl;
    cout<<"2- Afficher la liste des elements"<<endl;
    cout<<"3- Affichher la somme des elements"<<endl<<endl;
    cout<<"0- Pour quiter"<<endl<<endl;
    cout<<"Entrer votre choix SVP :";
   cin>>choix;
   switch (choix)
   {
      case 1:       system("cls");
                    int x;
                     cout<<"Entrez un entier X:  "; cin>>x;
                    ajouter(L, x);
                     break;
     case 2:                system("cls");
                            cout<<"les elements de la liste sont: "<<endl;
                            afficher(L);
                            system("pause");
                            break;
     case 3:                system("cls");
                             cout<<"la somme des elements de la liste est :"<<somme(L);
                            system("pause");
                             break;
   }
   system("cls");
}while (choix !=0);
return 0;
}
```

Exercice 1 :

1. En utilisant des fonctions itératives (boucles) améliorer le programme pour qu'il permette aussi:
    1. D’afficher les éléments pairs de liste L.
    2. D’afficher le maximum de liste L.
    3. De rechercher un élément dans la liste.
    4. D’afficher le dernier élément de la liste L.