Structure de données abstraites (SDA)	Structures séquentielles	**Les listes – les listes chainées**

Définition d'une liste :

Une liste est une structure abstraite de données permettant de regrouper des données et dont l'accès est séquentiel.

Elle correspond à une suite finie d'éléments repérés par leur rang (index).
Les éléments sont ordonnés et leur place a une grande importance.
Une liste est évolutive, on peut ajouter ou supprimer n'importe quel élément.
Voici ci-contre les opérations que l'on peut faire avec un objet de type Liste :

Type abstrait : Liste d'éléments de type T

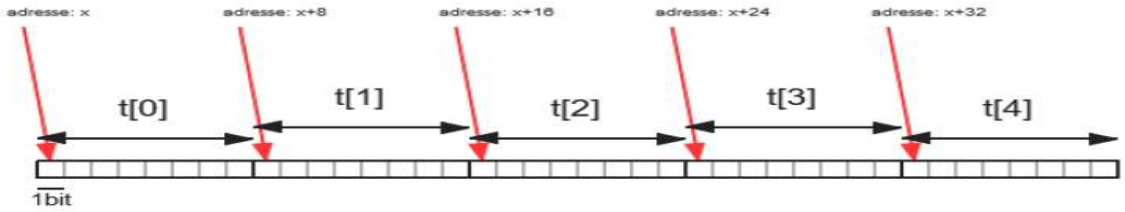
- CREER_LISTE_VIDE() qui retourne un objet de type liste
  La liste est vide et elle est vide
- INSERER(L,e,i)
  L'élément e est inséré à la position i dans la liste L
- SUPPRIMER (L,i)
  L'élément situé à la position i est supprimé de la liste
- LIRE(L,i) qui retourne un objet de type T
  L'élément situé à la position i dans la liste L est retourné
- MODIFIER (L,i,e)
  L'élément situé à la position i dans la liste i est écrasé par le nouvel élément e
- LONGUEUR(L) qui retourne un objet de type entier
  Le nombre d'éléments présents dans la liste L est retourné

IMPLEMENTATION d'une liste :

A) Implémentation par des tableaux dynamique (c'est ce que fait Python avec son type list)

En effet, en **python**, ce que l'on désigne par le terme liste est en réalité ce que l'on appelle en algorithmique un tableau, c'est à dire une suite d'éléments contigus et ordonnés en mémoire. Ces éléments sont stockés en mémoire les uns à la suite des autres.

Prenons l'exemple d'un tableau de 5 entiers codé sur 8 bits :



*Les tableaux permettent **efficacement** d'insérer ou de supprimer un élément à la fin avec les méthodes append() et pop(). Expliquez comment cela peut-il se passer du côté de la mémoire?*

Pour insérer à la fin : ……

……

Pour supprimer à la fin : ……

……

En revanche, ils le sont beaucoup moins lorsqu'il s'agit d'insérer ou de supprimer un élément à une autre position que la dernière. Expliquez comment insérer un élément en première position dans le tableau sera géré du côté de la mémoire.

Pour insérer v au début de la liste

1	2	3	4	5	

……

……

Considérons maintenant un tableau qui comporte 1 millions d'éléments.

1. Combien de déplacements seront nécessaires pour insérer une valeur en première position ?.....
2. Pour supprimer le premier élément ? ……

Comme un tableau possède une taille fixe, leur utilisation ne permet d'implémenter que des listes dont la taille maximale est définie à l'avance. Pour pallier à cette limitation, il existe ce qu'on appelle des **tableaux dynamiques** qui sont des tableaux dont la taille peut varier en fonction des besoins. Une taille est définie au départ, et si le nombre d'éléments de la liste vient à dépasser celle-ci, il faut alors créer un tableau plus grand (le double en général) et y recopier tous les éléments du premier ainsi que la donnée supplémentaire au bon endroit.

Le type list de Python correspond en fait à un tableau dynamique dans la majorité des implémentations Python.
Malgré ce nom ambigu, il s'agit donc d'un type abstrait beaucoup plus complet que le type Liste.

B) Implémentation par des listes chaînées

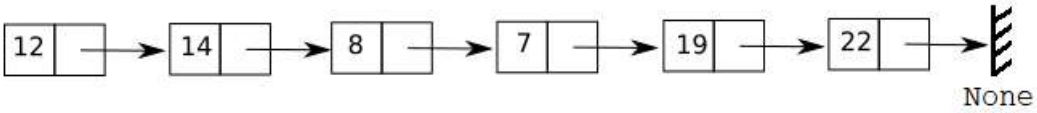
L'élément de base d'une liste simplement chaînée est une **cellule (ou maillon)** constituée de **deux parties.**

- **La première contient une donnée** (par exemple un entier pour une liste d'entiers)
- **la seconde contient un pointeur** (c'est à dire une adresse mémoire) **vers une autre cellule** (ou un pointeur nul).

Une liste est alors une succession de maillons, chacun pointant sur la suivante, et le dernier ayant un pointeur nul

En pratique, la variable «contenant» la liste est simplement un pointeur vers la premier maillon.

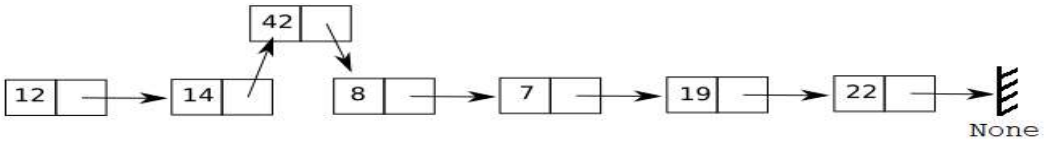
Voici par exemple une représentation d'une liste contenant les entiers 12, 14, 8 , 7 19 et 22.



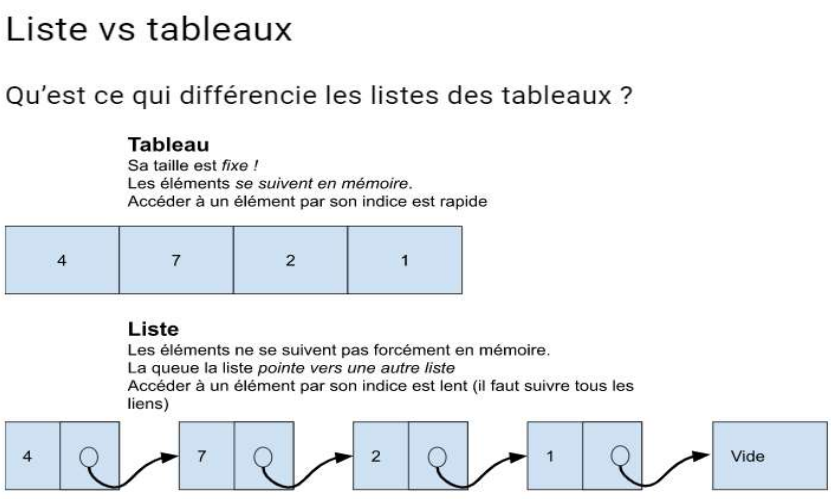
A noter que les cellules **n'ont pas à être placées** de façon contigüe en mémoire - ce qui va donner à cette structure plus de souplesse qu'aux tableaux. Voici les opérations qui peuvent être effectuées sur une liste :

- créer une liste vide
- tester si une liste est vide
- ajouter un élément
- supprimer un élément x d'une liste L
- compter le nombre d'éléments présents dans une liste

L'insertion dans une liste chaînée est plus efficace que dans une liste "classique". Il est inutile de décaler des éléments comme on le ferait pour un tableau .

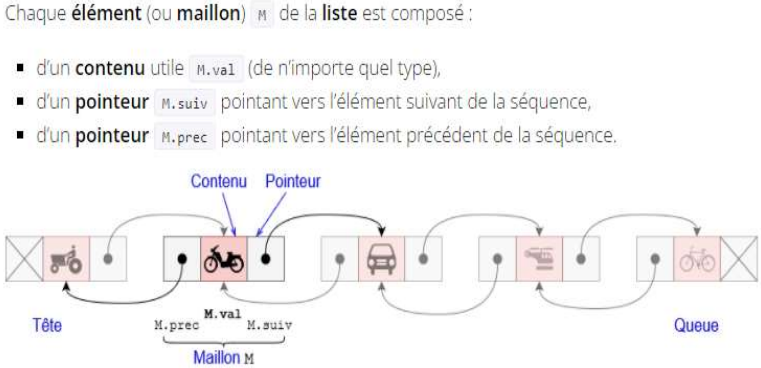


ici on crée la cellule 42 qui pointe vers 8 et on modifie la cellule 14 qui doit pointer vers 42



Opération	Coût pour un tableau	Coût pour Listes chainées
Ajout à la fin		
Ajout au début		
Accès à un élément		
Suppression d'un élément		
Modification d'un élément		
Insertion d'un élément		
Recherche d'un élément		

Remarque : il existe aussi des listes doublement chainées



Et même des listes circulaires doublement chainées !!