

**République Algérienne Démocratique et Populaire**  
**Université des Sciences et de la Technologie Houari Boumediene**

**Faculté d'Electronique et d'Informatique**

**Département Informatique**



# **Les Listes Chaînées-2**

**Cours Algorithmique de 1ere Année MI**

**Présenté par : Dr. B. BESSAA**

## Manipulation Listes Chaînées

Le type liste n'est **pas un type prédéfini**, c'est l'utilisateur qui le définit de sa manière, il faut juste respecter la syntaxe de déclaration.

La manipulation, dépend donc du problème à résoudre. Cependant il y a des opérations **standards** qu'on peut retrouver dans différents problèmes.

Nous allons voir principalement, la **création** des listes, **l'insertion** et la **suppression** et quelques **applications**.

**Une remarque importante** lorsqu'on manipule une liste, il faut **retenir** que le **premier** élément de la liste (**Tête de liste**) est le **seul point d'accès** à la liste, et il ne faut surtout **pas le perdre**, car si on le perd, tous les autres éléments **seront perdus**.

**Si on perd la Tête on devient Fou**

Pour tout ce qui suit, on va utiliser une liste **d'entiers**, mais le principe des algorithmes reste le même pour tout autre type d'élément de la liste.

## Déclaration

**Type**    Pliste =  $\wedge$  Eliste;  
          Eliste = **Enregistrement**  
                  Info : entier;  
                  Suiv : Pliste;  
          **Fin;**

### 1- Création des listes chaînées

Il existe deux modes de création d'une liste chaînée:

**a- LIFO (Last In First Out) :** dans ce cas chaque élément ajouté est placé en tête de liste, et donc le **dernier** élément ajouté sera le **premier** de la liste.

**b- FIFO (First In First Out) :** dans ce cas chaque élément ajouté est placé à la fin de la liste, et donc le **premier** élément ajouté sera le **premier** de la liste.

## Création LIFO (Last In First Out)

Pour créer une liste en mode **LIFO**, on utilise **deux** Pointeurs:

- Un pointeur pour la **Tête** de liste.
- Un pointeur **intermédiaire** pour créer les éléments.

L'opération passe par les étapes suivantes:

Soient **T** et **P** les deux pointeurs à utiliser.

- 1- Initialiser la tête **T** à Nil.
- 2- Créer un maillon avec **P**.
- 3- Remplir la partie information.
- 4- Enchaîner avec la tête **T**.
- 5- **Transmettre** l'adresse de **P** à **T**.
- 6- Aller à 2 pour créer l'élément suivant (**boucle de création**)

```
T ← Nil ;  
Allouer(P);  
Lire(P^.Info);  
P^.Suiv ← T;  
T ← P;
```

### Exemple

Soit à créer une liste composée des éléments : **5 - 3 - 6**

Donc le premier élément de la liste sera **6**.

0- **Var** T,P : Pliste;

1- **T**  $\leftarrow$  Nil ;

2- **Allouer**(P);

3- **Lire**(P^.Info);

4- P^.Suiv  $\leftarrow$  T;

5- **T**  $\leftarrow$  P;

**Fin** de la première itération:

**T** (Tête) pointe vers le **premier** élément **@1** (5)

Le **suivant** de **5** est **Nil** (donc c'est aussi le dernier).

**P** pointe **aussi** vers **5** ( ne dérange pas).

6- Aller à 2 pour créer l'élément suivant (**boucle de création**).

On va donc suivre ces étapes pour créer la liste : **5 - 3 - 6**

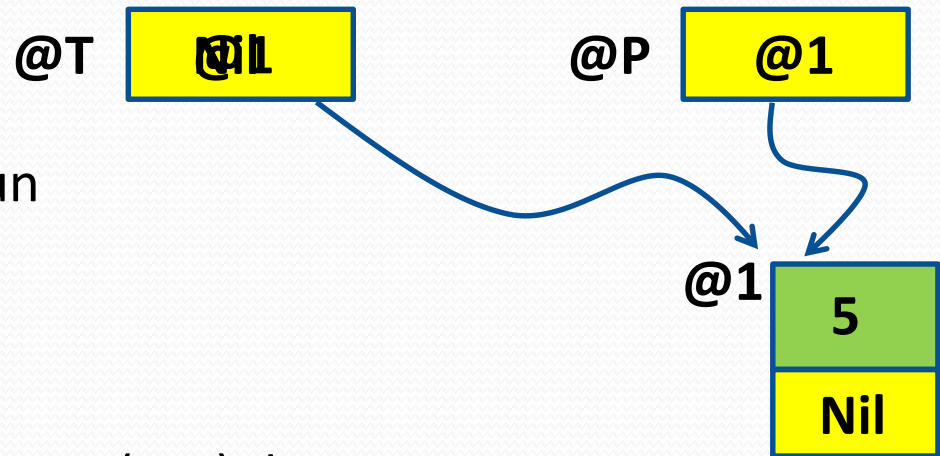
Créer **2** espaces **statiques** ayant pour adresses **@T** et **@P**

Créer un espace **dynamique** pour un maillon (2 champs). Soit à l'adresse **@1**, puis mettre **@1** dans P (**@P**)

Lecture de **5**

Transmettre **Nil** au champ Suivant de l'adresse **@1**

Mettre P (**@1**) dans T  
T pointe vers **@1**



## Deuxième Itération : Ajouter 3

0- Var T,P : Pliste;

1-  $T \leftarrow Nil$  ;

2- Allouer(P);

3- Lire(P^.Info);

4-  $P^.Suiv \leftarrow T$ ;

5-  $T \leftarrow P$ ;

Fin de la deuxième itération:

T (Tête) pointe vers le **deuxième** élément @2 (3)

Le **suivant** de 3 est 5.

P pointe **aussi** vers 3 ( ne déränge pas).

6- Aller à 2 pour créer l'élément suivant (**boucle de création**).

la liste à créer : 5 - 3 - 6

Créer 2eme maillon (@2)

Nouveau lien pour P

Lecture de 3

Transmettre @1 au champ

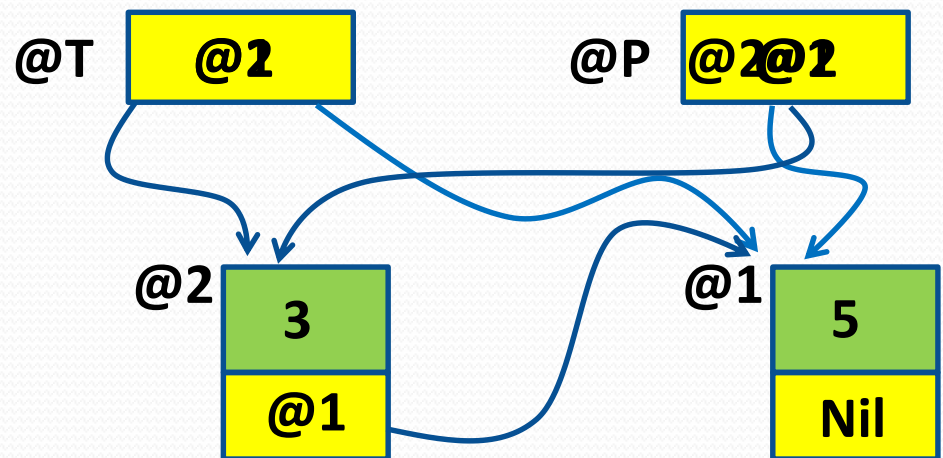
Suivant de l'adresse @2

Chainage de @2 vers @1

Mettre P (@2) dans T

T pointe vers @2

Nouveau chainage pour T



## Deuxième Itération : Ajouter 6

0- Var T,P : Pliste;

1-  $T \leftarrow \text{Nil}$  ;

2- Allouer(P);

3- Lire(P^.Info);

4-  $P^{\wedge}.\text{Suiv} \leftarrow T$ ;

5-  $T \leftarrow P$ ;

Fin de l'opération

T (Tête) pointe vers le **troisième** élément @3 (6)

Le **suivant** de 6 est 3, le **suivant** de 3 est 5.

P pointe **aussi** vers 6 ( ne déränge pas).

6- Aller à 2 pour créer l'élément suivant (**boucle de création**).

la liste à créer : 5 - 3 - 6

Créer 3eme maillon (@3)

Nouveau lien pour P

Lecture de 6

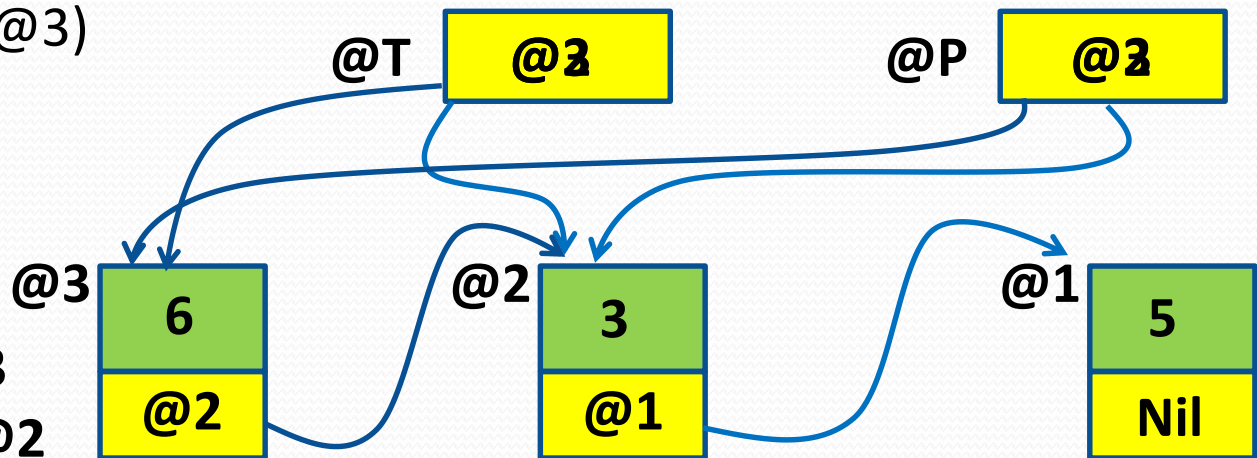
Transmettre @2 au

Suivant de l'adresse @3

Chainage de @3 vers @2

Mettre P (@3) dans T, T pointe vers @3; Nouveau chainage pour T

En ignorant le pointeur intermédiaire P, la liste de tête **T** sera :



# Algorithme de création LIFO

L'algorithme permettant de créer une liste de **N** entiers peut donc s'écrire

**Algorithme** CreerLifo;

**Var** T,P : Pliste; **//0**

I,N : entier;

**Debut**

**Lire**(N); *//nombre d'éléments de la liste*

T ← Nil; **//1** *//initialiser à vide*

**Pour** I ← 1 à N *//boucle de création*

**Faire**

**Allouer**(P); **//2**

**Lire**(P^.Info); **//3**

P^.Suiv ← T; **//4**

T ← P; **//5**

**Fait;**

**Fin.**



# Algorithme de création LIFO

On peut transformer cet algorithme en procédure avec **2 paramètres T et N**, ils seront donc enlevés des variables locales.

**Procédure** CreerLifo(E/S/ T:Pliste ; E/ N : entier);

**Var** P : Pliste;

I : entier;

**Debut**

T  $\leftarrow$  Nil ;

*//initialiser à vide*

**Pour** I  $\leftarrow$  1 à N

*//boucle de création*

**Faire**

**Allouer**(P);

**Lire**(P^.Info);

    P^.Suiv  $\leftarrow$  T;

    T  $\leftarrow$  P;

**Fait;**

**Fin;**

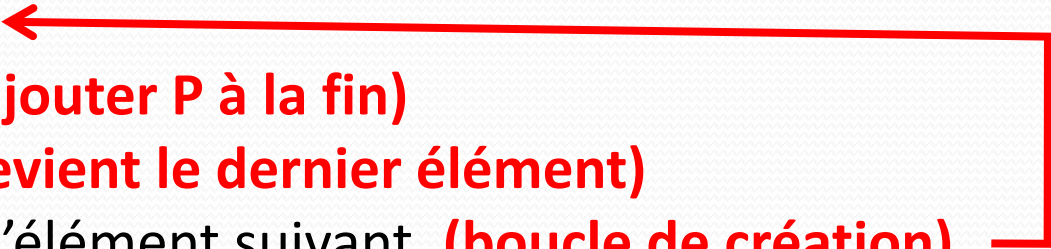
## Création FIFO (First In First Out)

Pour créer une liste en FIFO, **il faut d'abord créer la tête**, ensuite créer les autres éléments puis les enchaîner à la tête, pour cela on utilise **trois** Pointeurs:

- Un pointeur pour la Tête de liste (**T**).
- Un pointeur intermédiaire (**P**) pour créer les éléments.
- Un pointeur pour sauvegarder le dernier élément créé (**Q**).

L'opération passe par les étapes suivantes:

Soient **T**, **P** et **Q** les trois pointeurs à utiliser.

- 1- Initialiser la tête **T** à Nil.
- 2- Créer la tête **T**.
- 3- Initialiser **Q** avec **T**. (**Q représente dernier élément créé : Queue**)
- 4- Créer un maillon **P**. 
- 5- Enchaîner avec **Q** (**ajouter P à la fin**)
- 6- Affecter **P** à **Q**. (**P devient le dernier élément**)
- 7- Aller à 4 pour créer l'élément suivant. (**boucle de création**)
- 8- Affecter **Nil** au suivant de **Q**. (**mettre fin à la liste**)

## Exemple

Soit à créer une liste composée des éléments : **5 - 3 - 6**

Donc le premier élément de la liste sera **5**.

### Création du premier élément (Tête)

0- Var T,P,Q : Pliste;

1- T ← Nil ;

2.1- Allouer(T);

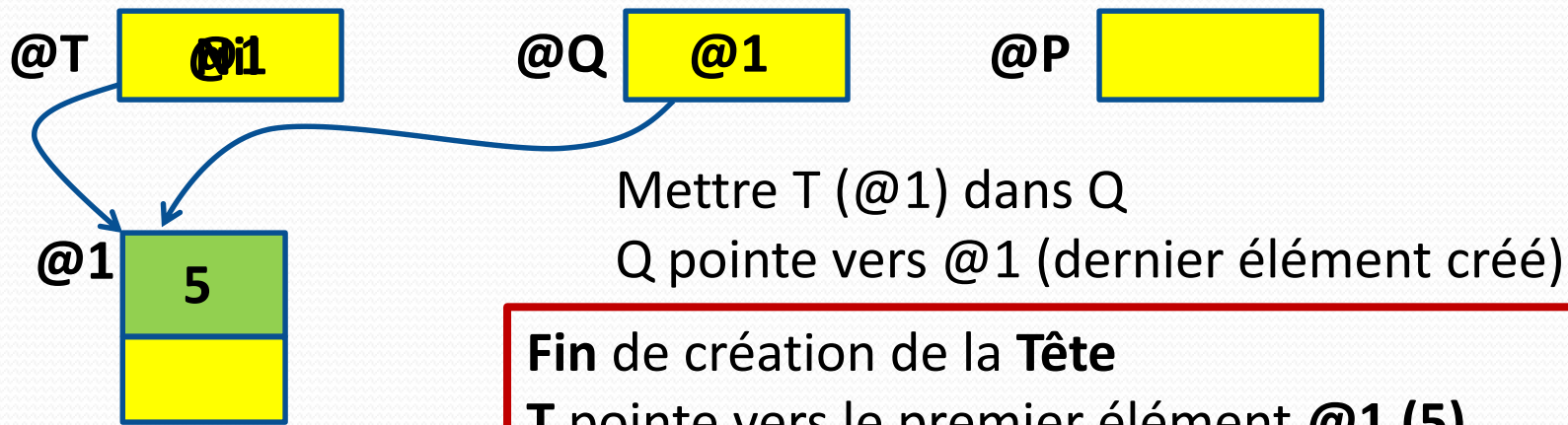
2.2- Lire(T^.Info);

3- Q ← T;

Créer 3 espaces statiques ayant pour adresses @T , @Q et @P

Créer un espace dynamique pour un maillon (2 champs). Soit à l'adresse @1, puis mettre @1 dans T (@T)

Lecture de 5



**Fin de création de la Tête**

T pointe vers le premier élément **@1 (5)**.

Q pointe vers le **même** élément.

## Créer les autres éléments (3 – 6)

### Première itération : Ajouter 3

4.1- Allouer(P);

4.2- Lire(P^.Info);

5- Q^.Suiv ← P;

6- Q ← P;

7- Boucle

8- Q^.Suiv ← Nil;

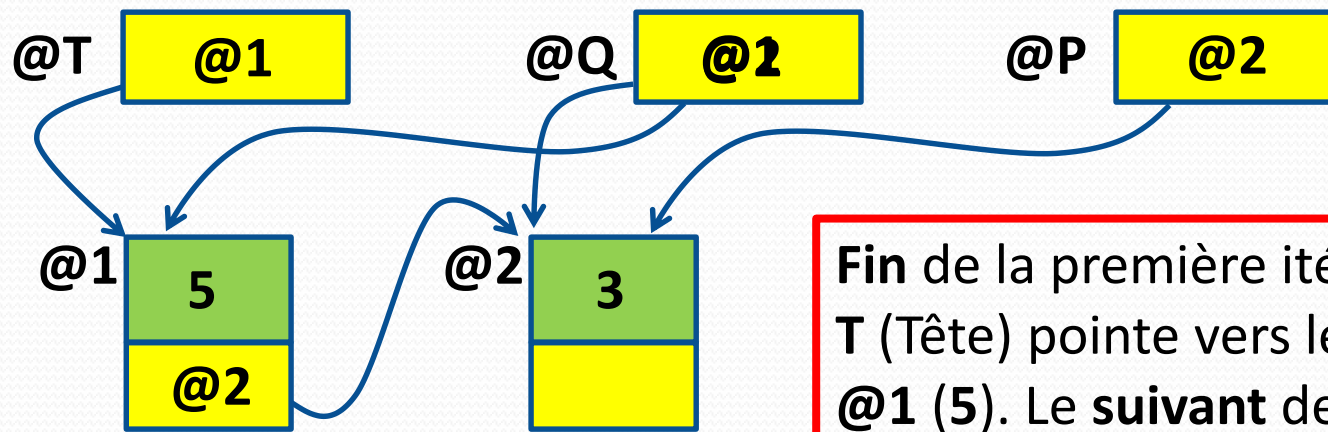
Créer un espace dynamique pour un maillon (2 champs). Soit à l'adresse @2, puis mettre @2 dans P (@P)

Lecture de 3

Transmettre @2 au Suivant de Q (@1)

**Chainage de @1 vers @2**

Mettre P (@2) dans Q, Q pointe vers @2; **Nouveau chainage pour Q**



**Fin de la première itération:**  
T (Tête) pointe vers le **premier** élément @1 (5). Le **suivant** de 5 est 3.  
P et Q pointent vers 3 ( ne déränge pas).

## Créer les autres éléments (3 – 6)

### Deuxième itération : Ajouter 6

4.1- Allouer(P);

4.2- Lire(P^.Info);

5-  $Q^{\text{Suiv}} \leftarrow P$ ;

6-  $Q \leftarrow P$ ;

7- Boucle

8-  $Q^{\text{Suiv}} \leftarrow \text{Nil}$ ;

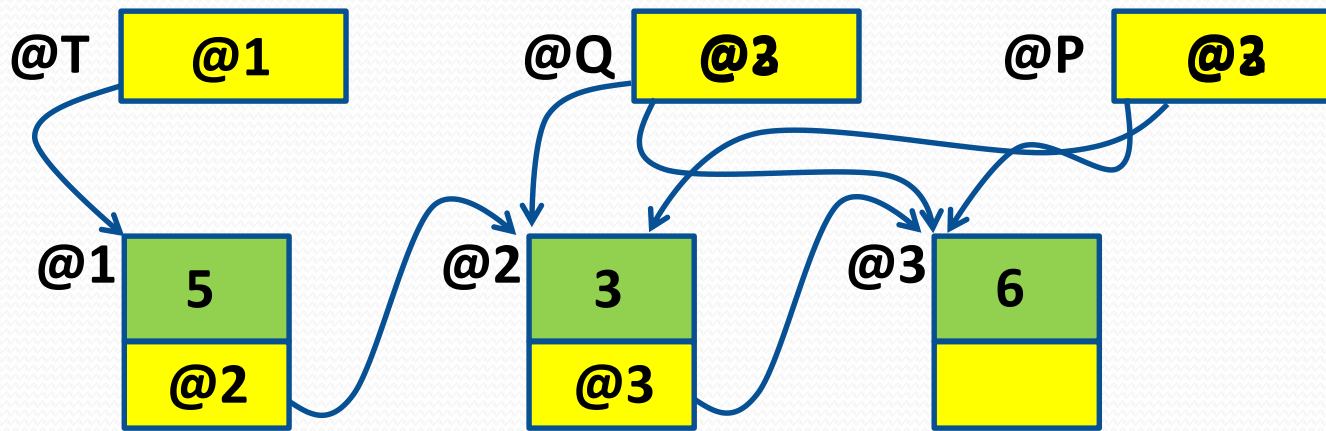
Créer un espace dynamique pour un maillon (2 champs). Soit à l'adresse @3, puis mettre @3 dans P (@P). **Nouveau chainage pour P**

Lecture de 6

Transmettre @3 au Suivant de Q (@2)

**Chainage de @2 vers @3**

Mettre P (@3) dans Q, Q pointe vers @3; **Nouveau chainage pour Q**



## Créer les autres éléments (3 – 6)

### Deuxième itération : Ajouter 6

4.1- Allouer(P);  
4.2- Lire(P^.Info);  
5- Q^.Suiv ← P;  
6- Q ← P;  
7- Boucle  
8- Q^.Suiv ← Nil;

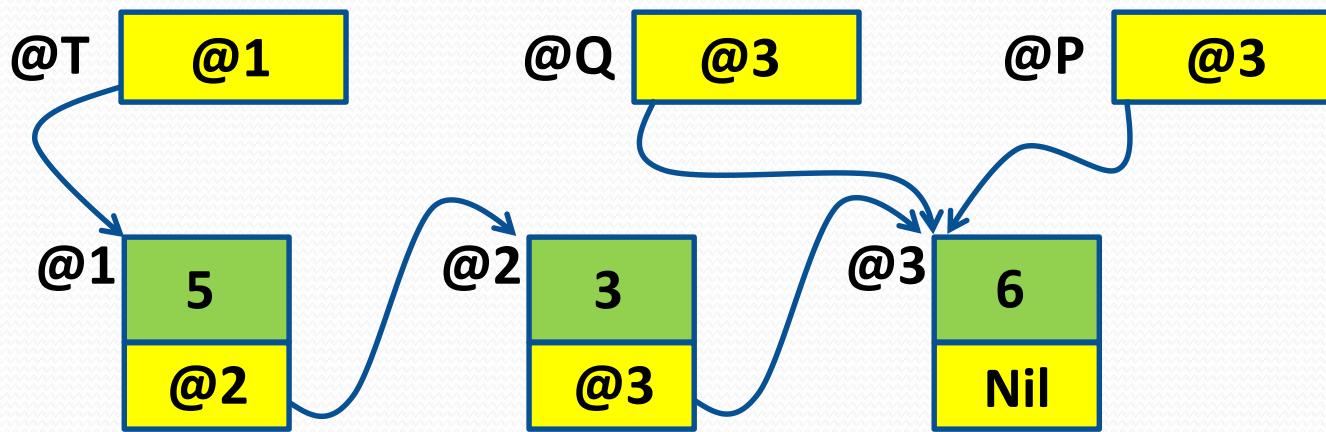
Fin de la boucle, mettre le suivant de Q (@3) à Nil

Fin de l'opération:

T (Tête) pointe vers le **premier** élément @1 (5).

Le **suivant** de 5 est 3, Le **suivant** de 3 est 6.

P et Q pointent vers 6 ( ne déränge pas).



En ignorant les pointeurs intermédiaires P et Q, la liste de tête **T** sera :

# Algorithme de création FIFO

L'algorithme permettant de créer une liste de **N** entiers peut donc s'écrire

**Algorithme** CreerFifo;

**Var** T,P,Q : Pliste;

I,N : entier;

**Debut**

**Lire**(N);     *//nombre d'éléments de la liste*

T ← Nil;     *//initialiser à vide*

**Si** N ≠ 0 **Alors**

**Allouer**(T); **Lire**(T^.Info);     *// créer la tête*

    Q ← T;     *//dernier élément créé*

**Pour** I ← 2 à N                     *//boucle de création*

**Faire**

**Allouer**(P); **Lire**(P^.Info);

        Q^.Suiv ← P;

        Q ← P;

**Fait**;

    Q^.Suiv ← Nil;     *//mettre fin à la liste*

**Fsi**;

**Fin.**

# Algorithme de création FIFO

La même chose ici, on peut transformer cet algorithme en procédure

**Procédure** CreerFifo(E/S/ T : Pliste ; E/ N:entier);

**Var** P,Q : Pliste;

I : entier;

**Debut**

T ← Nil ; //initialiser à vide

**Si** N ≠ 0 **Alors**

**Allouer**(T); **Lire**(T^.Info); // créer la tête

    Q ← T; //dernier élément créé

**Pour** I ← 2 à N //boucle de création

**Faire**

**Allouer**(P); **Lire**(P^.Info);

        Q^.Suiv ← P;

        Q ← P;

**Fait**;

    Q^.Suiv ← Nil; //mettre fin à la liste

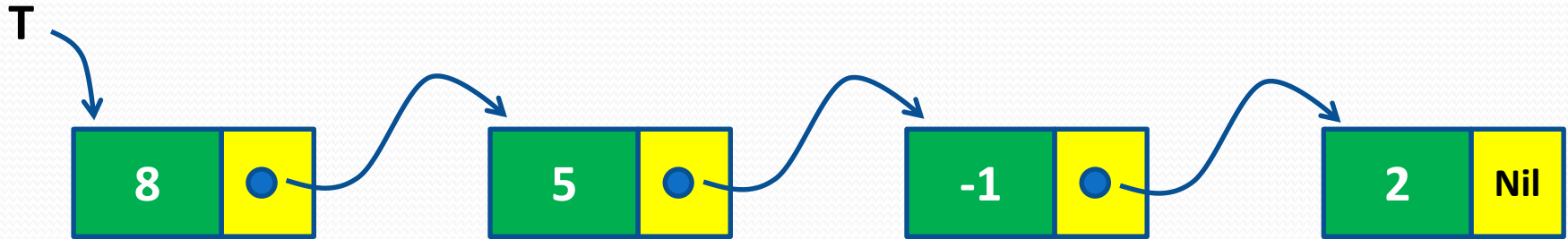
**Fsi**;

**Fin.**



## Remarque

Pour simplifier le schéma représentant une liste, on peut négliger les adresses, par exemple, la liste  $\{8,5,-1,2\}$  sera représentée par le schéma:



**Mais comment on peut savoir si la liste est FIFO ou LIFO ?**

**Bon**, **FIFO** ou **LIFO** sont des **modes** de création, et non **pas** une **caractéristique** de la liste. Donc, une fois créée, on ne cherche pas à savoir comment elle a été créée. L'essentiel, on a une suite d'éléments avec une **Tête** comme point d'accès.

**Alors**, on passe maintenant à l'insertion dans une liste. On suppose qu'on a déjà une liste à laquelle on veut ajouter un élément.

## 2- Insertion dans une liste chaînée

Soit **T** la tête d'une liste **donnée**, et **X** un élément (partie **info**) à insérer dans cette liste.

Trois cas se présentent pour l'insertion:

1- Insérer au **début**.

2- Insérer à la **fin**.

3- Insérer au **milieu**.

**Insérer au début ou à la fin, c'est clair, mais le milieu ?! Comment on va le définir ?**

**Bien**, quand on dit **milieu**, ce n'est pas dans le sens **moitié**, mais c'est **quelque part** dans la liste, **autre** que le **début** et la **fin**.

**Comment le définir ?** Là c'est le cas d'utilisation qui le définit, ça peut être une position (Ex: insérer à la 3 position), comme ça peut être avant ou après une valeur donnée, ou encore avant ou après une adresse donnée,...

Ce qui nous intéresse c'est le principe d'insertion au **milieu**, indépendamment du cas de figure.

**A Suivre...**

*Merci!*



**[brbessaa@gmail.com](mailto:brbessaa@gmail.com)**