Etablissement : ISET-Charguia	**Département :** Technologies de l'Informatique
Matière : Algorithmique et Programmation II	**Année Universitaire :** 2022- 2023
Niveau : 1ère année Tronc Commun	

TD n°2 : Les listes chaînées

Listes simplement chaînées

Exercice 1 :

Soit L une liste à chaînage simple, écrire les procédures et les fonctions permettant de:

1. Calculer le nombre d'occurrence d'un élément E donné dans la liste chaînée :

   Fonction NbOccurences (L : Liste, E : Entier) : Entier

2. Retourner un pointeur sur le k^{ème} élément de la liste si k existe, NULL sinon :

   Fonction *Pointeur* (var L : Liste, k : Entier) : Liste

3. Compter le nombre de cellules dans une liste chaînée :

   Fonction Taille (L : Liste) : Entier

4. Rechercher un élément E dans une liste en renvoyant vrai si l'élément existe, faux sinon.

   Fonction Recherche (L : Liste, E : Entier) : Booléen

Exercice 2 :

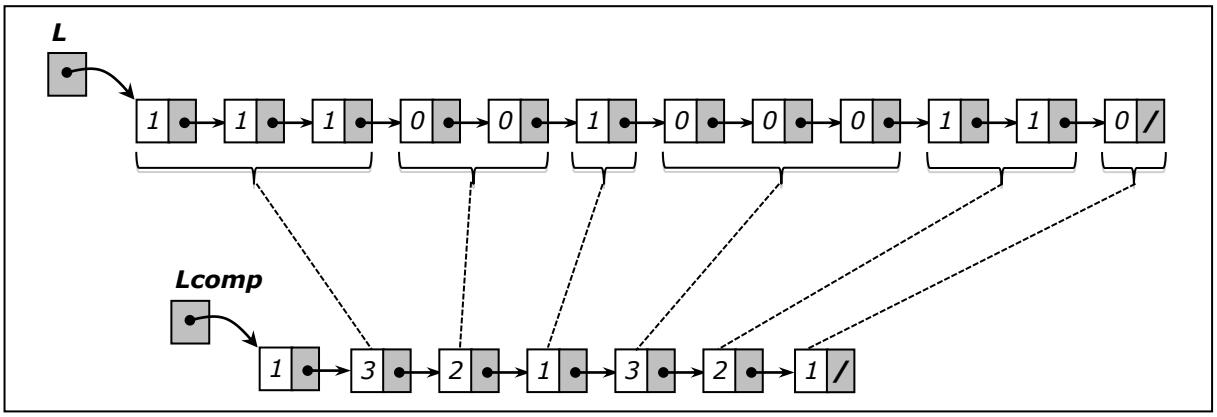
L'objectif de cet exercice est de compresser une suite binaire selon le principe suivant : Une suite du même chiffre (0 ou 1) est remplacée par son nombre d'occurrences consécutives :

1 1 1 1 1 → 5 ; 0 0 0 → 3

On représente la suite binaire à travers une **liste simplement chaînée** de type **Liste**.

Après compression, on obtient une autre liste dont la première cellule indique le bit avec lequel commence la suite binaire suivie des cellules contenant chacune le nombre d'occurrences consécutives du bit en question.

Exemple :



1) Ecrire une procédure **COMPRESSER** qui, étant donnée une liste chaînée **L** de chiffres binaires, crée une autre liste **Lcomp** compressée. La liste **L** et **Lcomp** sont de type **Liste**.
2) Ecrire une procédure **DECOMPRESSER** qui, étant donnée une liste chaînée **Lcomp** d'une suite binaires compressée, crée la liste **L** décompressée.

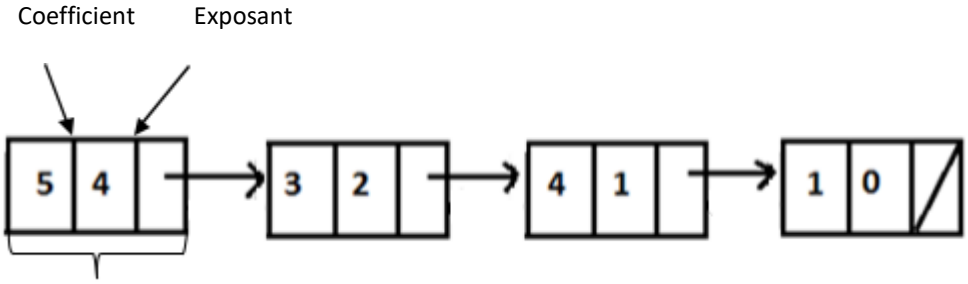
Exercice 3 :

On se propose d'écrire un ensemble de traitement sur les polynômes, pour ce faire un polynôme est représenté comme étant une **liste simplement chaînée**. Chaque cellule représente un monôme du polynôme. Une cellule possède un **coefficient**, un **exposant** et l'adresse du monôme suivant.

Exemple :

Le polynôme $P(x)=5X^4 + 3X^2 + 4X + 1$:

- Ce polynôme contient les monômes $5X^4$, $3X^2$, $4X^1$ et $1X^0$
- Le monôme $3X^2$ a pour coefficient 3 et pour exposant 2



Remarque :

- On suppose que les monômes d'un polynôme sont rangés dans la liste chaînée par **ordre décroissant de leurs exposants**.
- Un monôme à **coefficient nul n'est pas représenté** dans une cellule de la liste chaînée.
- Le **degré** d'un polynôme représente le plus grand exposant de ses monômes : dans le cas du polynôme P(x) le degré est 4.

Travail à faire

1. Définir le nouveau type **Cellule**. (2 pts)
2. Définir le nouveau type **ListePoly** qui représente le nouveau type de liste chaînée. (0.5 pt)
3. Ecrivez un sous-algorithme **SaisiePoly** qui :
   - Demande à l'utilisateur le **degré** du polynôme à saisir,
   - Lit les champs des monômes (coefficient et exposant)
   - Range les monômes dans la liste concernée par **ordre décroissant** de leurs exposants (voir figure ci- dessus). (3 pts)
4. Ecrire une procédure récursive **AfficherPoly** qui permet d'afficher les monômes d'un polynôme par ordre croissant de leurs exposants. (2 pts)

5. Ecrire une procédure **Som2Poly** qui, étant donnés **deux polynômes** représentés sous forme de listes chaînées, permet de déterminer leurs sommes. Le polynôme somme sera représenté aussi sous forme de liste chaînée. (4.5 pts)

Exemple :

- ✓ $P(x)=X^4 + 3X^2 + 4X^1 + 1\ X^0$
- ✓ $Q(x)=2X^5 - X^2 + X^1 + 3\ X^0$
- ✓ $S(X) = P(X) + Q(X) = 2X^5 + X^4 + 2X^2 + 5\ X^1 + 4\ X^0$

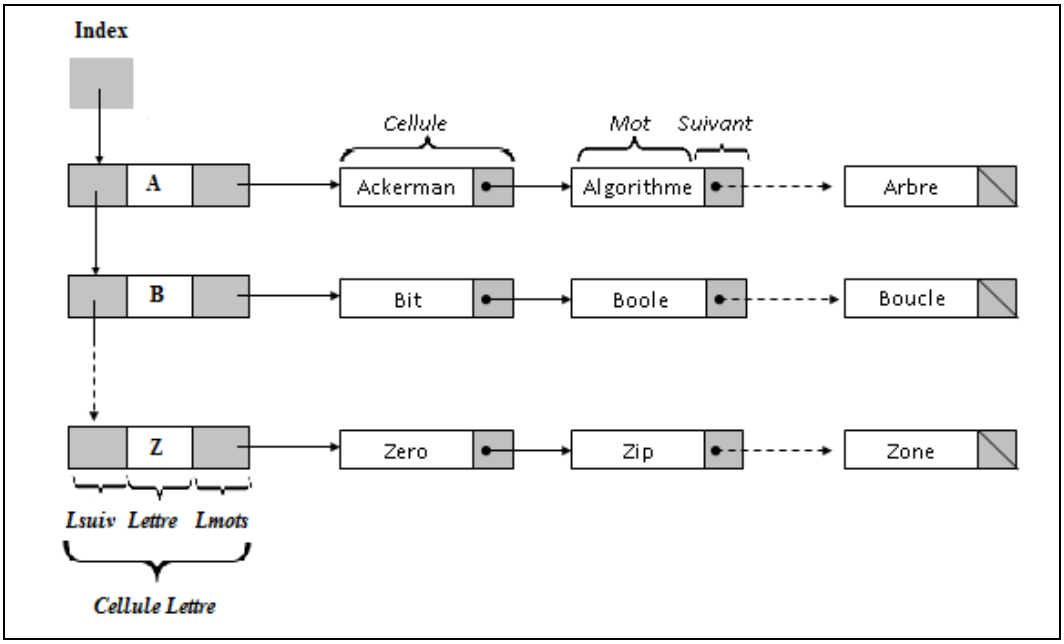
Exercice 4 :

Nous voulons créer les structures de données dynamiques nécessaires pour la mise en place et la gestion d'un index pour un ouvrage ce qui consiste en une liste alphabétique des mots clés apparaissant dans le livre.

La figure ci-dessous donne une représentation graphique des structures utilisées.

L'index est composé d'une liste triée des lettres de l'alphabet (A … Z), chaque lettre est dans une cellule appelée ***CelluleLettre*** ayant trois champs (***Lsuiv***, ***Lettre***, ***Lmots***).

Chaque lettre lui est rattachée une liste triée de mots composée de *Cellules* ou chacune contient un *mot* et un champ *suivant*.



Travail demandé

1- Définir les structures de données nécessaires pour la création de cet Index.

2- Ecrire un sous-algorithme **CréerIndex**, qui permet de créer uniquement la liste verticale des lettres ordonnées alphabétiquement de 'A' à 'Z'.

3- Nous supposons que l'index est déjà créé et trié selon la figure. Ecrire une fonction ***RechercheMot*** qui permet de rechercher, un mot donné en paramètre, dans l'index et de retourner vrai si elle le trouve, faux sinon.

4- Ecrire un sous-algorithme **InsérerLivre,** qui étant donné l'index et un nouvel livre, permet de l'insérer dans l'index s'il n'existe pas.

Exemples :

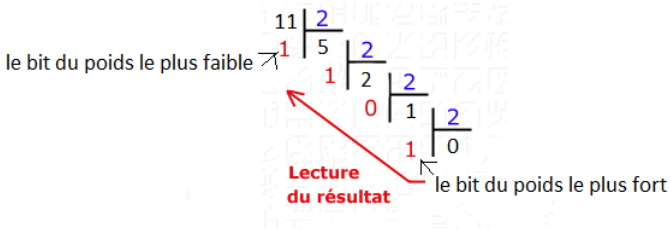
- Si le mot cherché est "**Zip**", la méthode renvoie **Vrai**
- Si le mot cherché n'existe pas, la méthode renvoie **Faux**

Listes doublement chaînées

Exercice 5:

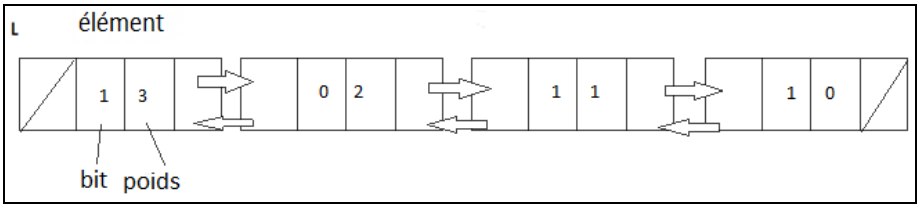
On s'intéresse à l'opération consistant à convertir un entier naturel n en sa représentation binaire, tel que : $n = b_0.2^0 + b_1.2^1 + \cdots + b_k.2^k$

Le principe de la conversion binaire étant de réaliser une série de divisions par 2. Le reste de la première division représente le bit de poids faible alors que le reste de la dernière division représente le bit de poids fort.



Le but de l'exercice est d'insérer les bits de la conversion binaire dans une liste doublement chaînée.

Par exemple, pour n = 11 la liste renvoyée doit être :



1. Donnez une définition du type nouveau type élément, puis celle du type liste.
2. Ecrivez un sous-algorithme **nToBin(…)** qui étant un entier naturel **n** renvoie la **liste** b_0; b_1; b_2; … b_k de sa conversion binaire dans une liste doublement chaînée.
3. Ecrivez un sous-algorithme **binTOn(…)** qui, à partir d'une liste L, renvoie l'entier correspondant.
4. Ecrivez un sous-algorithme **AddBin(…)** qui permet d'additionner 2 nombres binaires représentés par 2 listes chaînées. Le résultat est lui-même inséré dans une liste chaînée.

Remarque : les quatre opérations principales dédiées à l'addition binaire à savoir :

Exemple

0 + 0 = 0.
1 + 1 = 0 (retenue = 1).
1 + 0 = 1
0 + 1 = 1

$$\begin{array}{r} 110 \\ + \\ 111 \\ \hline = 1101 \end{array}$$

5. Ecrivez un sous-algorithme **AfficherBin(…)** qui affiche le nombre binaire à partir d'une liste L.
6. Ecrivez l'algorithme principal qui permet de :
   - Saisir deux entiers positifs n1 et n2,
   - Les convertir en nombres binaires dans deux listes chaînées L1 et L2,
   - Afficher leur somme.