DOSSIER ALGORITHMIQUE 1: LECTURES SUPPRESSION DES OCCURRENCE D'UNE LISTE CHAÎNÉE

*** PROBLÈME** : lecture ET suppression de toutes les occurrence d'un élément dans une liste chaînée

***PRINCIPE :** parcourir tout les éléments de la liste et comparer au fur et à mesure chaque élément de la liste tous en suppriment l'occurrence choisir

*** DICTIONNAIRE DES DONNÉES**

NOMS	TYPES	DESCRIPTION
n	entier	Contient le nombre total d'élément que l'utilisateur veut dans la liste
i		Sert d'indice pour la boucle de construction de la liste (de 1à n)
x		Reçoit la valeur saisir par l'utilisateur ; - soit pour construit la liste - soit pour designer l'élément à supprimer
l	liste	Pointeur vers le début de la liste chaînée
p		Pointeur de parcours , utiliser pour parcourir les élément de la liste
t		Pointeur vers l'élément précédent lors du parcours utiles pour relier les maillons après suppression
temp		
cellule	enregistrement	Définition d'un nœud de la liste chaînée , avec deux champs

->val	entier	La valeur stoker dans la cellule
->suiv	Pointeur vers cellule	Pointeur vers la cellule suivante de la liste
liste	Pointeur vers cellule	Type pointeur qui représente l'adresse du premier élément de la liste

***ALGORITHMES: PSEUDO CODE**

```
Algorithme LireSupprimerOccurrences
Var
    n, i, x : entier ;
       cellule : enregistrement
        val : entier
        suiv : ^cellule
   fin enregistrement ;

      type Liste = ^cellule ;
        L, p, t, temp : Liste ;

Début
    L <- Null ;
    p <- Null ;
    t <- Null ;

    Ecrire("Donner le nombre d'éléments : ") ;
    Lire(n) ;

    // Construction de la liste
    Pour i <- 1 à n faire
        Lire(x) ;
        allouer(p) ;
        p^.val <- x
        p^.suiv <- Null
        Si (L = Null) alors
            L <- p
            t <- p
        Sinon
            t^.suiv <- p
            t <- p
        Finsi
    FinPour
```

```
    // Lecture de l'élément à supprimer
    Ecrire("Donner l'élément à supprimer : ")
    Lire(x)

    // Suppression de toutes les occurrences
    p <- L
    t <- Null

    Tantque (p <> Null) faire
        Si (p^.val = x) alors
            Si (t = Null) alors
                // Suppression en tête
                L <- p^.suiv
                temp <- p
                p <- p^.suiv
                libérer(temp)
            Sinon
                // Suppression au milieu ou fin
                t^.suiv <- p^.suiv
                temp <- p
                p <- p^.suiv
                libérer(temp)
            Finsi
        Sinon
            t <- p
            p <- p^.suiv
        Finsi
    FinTantque

    // Affichage de la liste après suppression
    Ecrire("Liste après suppression : ")
    p <- L
    Tantque (p <> Null) faire
        Ecrire(p^.val)
        p <- p^.suiv
    FinTantque
fin
```

*** COMPLEXITÉ**

TEMPORELLE : $O(n^2)$

SPATIALE : $O(n)$