Corrigé TD3 Listes

Ex1

```
Nb_occurence_it(l: liste, e : entier): entier
données : l : liste a parcourir
                e : element recherché
var loc : nbOcc : entier, nbre doccurence de e
Resultat : nbOcc

Debut
        nbOcc <- 0
        Tantque (l != vide)
                Si l->info = e Alors
                        nbOcc++
                FinSi
                l<- l-> succ
        FinTq
        retourner nbOcc
Fin

Nb_occurence_rec(l: liste, e : entier): entier
données : l : liste a parcourir
                e : element recherché
Resultat : nbre dOccurences de e

Debut
        Si l= vide Alors retourner 0
        Si l->info = e Alors
                retourner 1+ Nb_occurence_rec(l->succ, e)
        Sinon
                retourner Nb_occurence_rec(l->succ, e)
        FinSi

        //retourner (l->info=e ? 1:0)+nb_occurence_rec(l->succ,e)
Fin

Position_it1(l: liste, e : entier): entier
données : l : liste a parcourir
                e : element pour lequel on souhaite retourner sa pos
Var loc : pos : entier, position de e
Resultat :  e

Debut
        pos<-0 // 0 est la position du premier element
        Tanque (l!= vide et l->info != e ) faire
                pos++
                l<-l->succ
        FinTq
        Si l=vide Alors retourner -1
        retourner pos
Fin
```

```
Position_it2(l: liste, e : entier): entier
données : l : liste a parcourir
                    e : element pour lequel on souhaite retourner sa pos
Var loc : pos : entier, position de e
Resultat :  e

Debut
        pos<-0 // 0 est la position du premier element
        Tanque (l!= vide ) faire
                Si (l-> info =  e) Alors retourner pos
                pos++
                l<-l->succ
        FinTq
        retourner -1
Fin

Position_rec(l: liste, e : entier): entier
données : l : liste a parcourir
                    e : element pour lequel on souhaite retourner sa pos
var loc : pos : entier
Resultat :  position de e dans l

Debut
        Si l = vide Alors retourner -1
        Si l->info = e Alors retourner 0
        pos <- Poition_rec(l->succ, e)
        Si pos != -1 Alors retourner pos++
        Sinon retourner -1
Fin

max_liste_it(l: liste): entier
données : l : liste a parcourir
Var loc : max : entier
Resultat :  max

Debut
        Si l = vide retourner ERR
        max <- l->info
        Tantque l->succ!= vide faire
                Si l->succ->info > max Alors
                        max = l->succ->info
                FinSi
                l<-l->succ
        FinTq
        retourner max
Fin
```

```
max_liste_rec(l: liste): entier
données : l : liste a parcourir
Var loc : max : entier
Resultat :  max

Debut
        Si l = vide retourner ERR
        Si l->succ = vide Alors retourner l->info
        max <- max_liste_rec(l->succ)
        retourner l->info>max ? l->info: max
        // Si l->info >max retourner l->info
        // Sinon retourner max

Fin
```