TD8 – Soutien - Refaire TD sur les listes

1. Comparaison de listes

Ecrire en pseudo-langage la fonction **récursive** qui réalise cette comparaison en renvoyant vrai si les listes sont égales, faux sinon.

On peut travailler directement sur l1 et l2 car elles sont données en entrée !

```
Fonction egales-rec (l1, l2 : liste) : booléen
Début
Si l1=nil et l2=nil
     alors retourner(vrai)
     sinon Si (l1=nil et l2≠nil) ou (l1≠nil et l2=nil)
               Alors retourner(faux)
               Sinon retourner(l1^.val=l2^.val et egales-rec(l1^.suiv,l2^.suiv))
           FinSi
FinSi
Fin
```

2. Suppression de toutes les occurrences d'un élément dans une liste

Ecrire en pseudo-langage une fonction **itérative** qui réalise ces suppressions.

```
Procédure supprime (E/S l : liste, E e : entier)
Var r, p : liste
Début
r←l
Tantque r<>nil Faire
  Si l=r et r^.val=e
     alors p←l
           l←l^.suiv
           r←l
           récupérer(p)
     sinon Si (r^.suiv<>nil) et (r^.suiv^.val=e)
               alors p←r^.suiv
                     r^.suiv←p^.suiv
                     récupérer(p)
               sinon r←r^.suiv
           FinSi
  FinSi
FintTantQue
Fin
```

3. Liste chainée - 5 pts

Soit `t` un tableau de `n` entiers trié dans l'ordre croissant dont les valeurs peuvent être répétées. On veut écrire une fonction qui transforme ce tableau en une liste chainée `l` triée dans l'ordre croissant dont les valeurs sont uniques (on ne garde qu'une seule occurrence de chaque valeur de t).

3.1. Expliquer en français le principe de la fonction `transforme(t,n)`.
3.2. Ecrire en pseudo-langage la fonction `transforme(t,n)`.

```
type  tab = tableau[1..max] d'entiers
      liste = ^cellule
      cellule = Enregistrement
                  val : entier
                  suiv : liste
                FinrEnregistrement
```

```
Fonction transforme (t : tab , n : entier) : liste
Var   i : entier
      l, q : liste
Début
i← 1
  TantQue i<= n Faire
      Si i=1
           alors l← allouer(cellule)
                 l^.val← t[1]
                 q← l
           sinon q^.suiv← allouer(cellule)
                 q← q^.suiv
                 q^.val← t[i]
      FinSi
      TantQue (i< n) et (t[i+1]=t[i]) Faire
            i← i+1
      FinTantQue
      i← i+1
  FintanQue
q^.suiv← nil
retourner(l)
Fin
```