

TD : Listes simplement chaînées

Exercice 1

En considérant la définition d'une Liste simplement chaînée suivante

```
type
  Liste = ^Cellule
  Cellule = structure
    elem : <type>
    succ : ^Cellule
  Fin structure
```

Donner l'implémentation de chacune des fonctions suivantes :

1. **ListeCopier(L : Liste) : Liste** qui permet de retourner une copie de la liste donnée comme argument.
2. **ListeComparer(L1 : Liste, L2 : Liste) : booléen** qui permet de retourner vrai si les deux listes ont les mêmes éléments dans le même ordre et de retourner faux dans le cas contraire.
3. **ListePurger(var L : Liste)** Cette fonction permet de purger une liste (supprimer les doublons).
4. **ListeRemplacer(var L : Liste, a : <type>, b : <type>) : entier** Cette fonction permet de remplacer toutes les occurrences d'un élément **a** par l'élément **b** et renvoyer le nombre de remplacements.
5. **Concatener(L1 : Liste, L2 : Liste) : Liste** : Cette fonction permet de retourner une nouvelle liste contenant les éléments de la liste L1 suivis par les éléments de la liste L2 (en gardant le même ordre des éléments).

Exercice 2

On se propose de modéliser la gestion des patients dans un cabinet médical. Un patient est caractérisé par: le nom, le prénom et un champ **rdv** (pour rendez-vous) de type entier indiquant si le patient a un rendez-vous ou pas: 0 si le patient est sans rendez-vous, 1 si la patient est avec rendez-vous.

Avant d'être consultés par le médecin, les patients sont entrés dans une salle d'attente qui sera modélisée par une liste simplement chaînée de patients. Une secrétaire fait entrer les patients ayant un **rdv=1** selon leur ordre d'arrivée, ensuite elle fait entrer les autres patients (ceux dont le **rdv=0**) selon leur ordre d'arrivée aussi.

1. Définir le type **Patient**.
2. Définir le type **Cellule** dans le cadre d'une liste simplement chaînée de patients.
3. Définir le type **Liste_patients** pouvant contenir une liste de patients
4. Dans une deuxième étape, on vous demande d'écrire les procédures suivantes :
 - 4.1. **Procédure AjoutPatient (var L :Liste_patients, nom :chaine, prenom :chaine, rdv :entier)** , qui permet d'ajouter un nouveau patient à la fin de la liste L
 - 4.2. **Procédure RendezVous (L :Liste_patients, Var rdv :entier, Var sansRdv :entier)**, qui compte le nombre de patients avec rendez-vous (dans la variable rdv), et le nombre de patients sans rendez-vous (dans la variable sansRdv).
 - 4.3. **Procédure SupprimePatient (Var L :Liste_patients)** , qui permet de faire entrer un patient en consultation. Cette opération est effectuée de la manière suivante : s'il n'y a aucun patient avec rendez-vous alors c'est le premier patient de la liste qui est supprimé. Sinon on supprime le premier patient qui a un rendez-vous.
 - 4.4 **Procédure ConsulterSalleAttente (L :Liste_patients)** , qui affiche tout d'abord les Patients avec rendez-vous, ensuite les patients sans rendez-vous.