Université Mohammed V
Faculté des Sciences de Rabat
Département d'Informatique

Année Universitaire 2014/2015
MODULE SD (SMI4)

Travaux Dirigés de Structures de Données

[TD n°1 : " Types Abstraits de Données :Vecteurs et Listes "]

Objectifs : - *Initier à la manipulation des types abstraits de données (TADs).*
- *Application aux TAD : Vecteurs et Listes*

Exercice 1

On désire implanter, en langage C, **le type abstrait de données Vecteur**, défini dans le cours. On rappelle, ci-dessous, une spécification minimale du **TAD Vecteur :**

```
Type Vecteur
Utilise Entier, Elément
Opérations
     vect          : Entier → Vecteur
     changer_ième: Vecteur x Entier x Elément → Vecteur
     ième          : Vecteur x Entier → Elément
     taille        : Vecteur → Entier
Préconditions
     vect(i) est_défini_ssi i ≥ 0
     ième(v,i) est_défini_ssi 0 ≤ i < taille(v)
     changer_ième(v,i,e) est_défini_ssi 0 ≤ i < taille(v)
Axiomes
     Soit, i, j : Entier, e : Elément, v : Vecteur
     si 0 ≤ i < taille(v) alors ième(changer_ième(v,i,e),i) = e
     si 0 ≤ i < taille(v) et 0 ≤ j < taille(v) et i ≠ j
          alors ième(changer_ième(v,i,e),j) = ième(v,j)
     taille(vect(i)) = i
     taille(changer_ième(v,i,e)) = taille(v)
```

Il existe plusieurs méthodes pour représenter les vecteurs. Parmi celles-ci une méthode qui consiste à représenter un vecteur sous la forme d'un tableau. On considère ici, un vecteur comme est une structure regroupant un entier, pour la taille, et un pointeur sur les éléments du vecteur.

a)- Donner la déclaration complète, en C, de la structure de données représentant le TAD Vecteur d'entiers, c-à-d :
- Donner les déclarations nécessaires pour définir **le type Vecteur d'entiers** ;
- Donner les déclarations, en C, **des primitives** du TAD Vecteur d'entiers.

b)- Définir les primitives (*ou opérations de base*) du TAD Vecteur d'entiers, c'est à dire : pour chaque primitive, écrire la fonction C correspondante.

c)- Au TAD Vecteur d'entiers on ajoute les deux opérations auxiliaires suivantes:
- **`produit_scalaire`** qui calcule le produit scalaire de deux vecteurs d'entiers.
- **`decalage_circulaire`** qui prend un vecteur et retourne le vecteur dans lequel tous les éléments sont décalés d'une place vers la droite (*le dernier est inséré à la 1ère place*).

Ecrire en langage C les deux opérations ci-dessus.

Exercice2

On considère ***la représentation simplement chaînée*** pour le TAD **`Liste`** défini en cours pour une liste de nombres entiers.

a)- Ecrire la fonction **`affiche`** qui affiche les éléments d'une liste chaînée **`L`**.

b)- Ecrire la fonction **`copie`** qui crée une copie d'une liste chaînée **`L`**.

c)- Ecrire la fonction **`concat`** qui concatène deux listes L_1 et L_2 (*sans qu'elles soient modifiées*).

d)- Ecrire la fonction **`supprime_occur`** qui supprime toutes les occurrences d'une valeur **`val`** dans une liste **`L`**.

e)- Ecrire la fonction récursive **`compte_occur`** qui calcule le nombre d'occurrences d'un élément **`e`** dans une liste **`L`**

f)- Ecrire la fonction **`purge`** qui purge une liste **`L`** (*i.e., enlève les doublons*).

g) -Ecrire la fonction récursive **`inverse`** qui crée la liste miroir **`LM`** d'une liste **`L`**.

Exercice 3

Une liste triée est une liste dont les éléments sont toujours rangés dans un ordre défini par une clé (*un champ ou une combinaison de champs d'un élément*). Pour simplifier, on considère des listes de nombres entiers.

a)- Ecrire la fonction **`insere_tri`** qui insère un élément dans une liste triée **`LT`**.

b)- Ecrire la fonction **`fusion_tri`** qui fusionne deux listes triées **`LT1`** et **`LT2`** (*sans qu'elles soient modifiées*).

Exercice 4

a)- Ecrire la fonction **`insere_avant`** qui insère, dans ***une liste doublement chaînée*** **`Ld`**, un entier **`e`** avant un noeud d'adresse **`pN`**.

b)- (***Question facultative***) Ecrire la fonction **`supprime_poccur`** qui supprime, dans une liste doublement chaînée **`Ld`**, la première occurrence d'un élément connaissant l'adresse **`pN`** de son noeud.

Exercice 5

Refaire l'exercice 3 ci-dessus, en considérant ***une liste chaînée circulaire*** **`Lc`**.