Université du Maine **L3 SPI Info – 175EN001**

TP n° 1 : Une réalisation du TDA LISTE
par curseurs (faux pointeurs)
(D'après E. DELOZANNE, P. JACOBONI)

A Réaliser dans la Session « 20XX-TP1-TDA» d'Hop3x

Objectifs :

- Mise en œuvre d'un TDA en C
- Simulation de la mémoire dynamique et des pointeurs par un tableau et des curseurs
- Lecture d'une chaîne de caractères en C

Sujet :

Les listes sont toutes mémorisées dans un tableau MEMOIRE de taille fixée à la compilation et dont chaque case est un enregistrement à deux champs : le premier champ contient l'élément de la liste et deuxième champ contient l'indice de l'élément suivant de la liste dans le tableau MEMOIRE.

Exemple :

Ce tableau contient les listes l1 = (a b c) et l2 = (d e)
l1 est représentée par 5 et l2 par 1

	élément	suivant
1	d	7
2		4
3	c	0
4		6
5	a	8
6		0
7	e	0
8	b	3
9		10
10		2

1) En utilisant les primitives du TDA ELEMENT définir en C les opérations primitives du TDA LISTE en adoptant la représentation par faux-pointeur.

Remarque : vous utiliserez **sans les modifier** les fichiers du cours 1 disponibles sur UMTICE) où sont déclarées les primitives du TDA ELEMENT et du TDA LISTE ainsi que ceux où sont définies les primitives du TDA ELEMENT incarnés par des caractères.

2) Précisez comment vous glanez les cases libérées par la suppression d'éléments dans les listes. En d'autres termes on vous demande de réaliser un petit ramasse miettes (ou garbage collector ou glaneur de case mémoire) et de précisez la stratégie que vous adoptez pour récupérer la mémoire libérée (marquage des cases ou chaînage des cases disponibles).
3) Écrire un micro interpréteur permettant à l'utilisateur
   a) de créer une liste :
      > L = (a b c d e)
      (a b c d e)
      > toto = ()
      ()
   b) d'afficher une liste
      > L
      (a b c d e)
      > toto
      ( )
   c) de retourner **une copie** de la liste privée de son premier élément :
      >cdr(L)
      (b c d e)
      > cdr(toto)
      erreur : accès illégal pour une liste vide
   d) d'insérer l'élément <elt> en tête de liste (retourne une copie de la liste) :
      > LL = cons (e L)
      (e a b c d e)
      > L
      (a b c d e)
      > LL
      (e a b c d e)
      > cons(z toto)
      (z)
      > toto
      ( )

Test : Tester votre programme au moins sur les exemples donnés dans l'énoncé