TP01 : Montée en compétences Lisp

Livrables attendus : un fichier lisp + un rapport présentant et argumentant le code lisp proposé.
Un bonus sera accordé lorsque deux solutions sont développées et comparées (itérative/récursive)

Date de remise : 5 octobre 2015 à 18H

Exercice 1 : Mise en condition

Ecrire sous forme préfixée :

$4 x^3 – 5 x^2 + 3 x + 1$

Représenter par une arborescence de type arbre généalogique, puis par un arbre binaire la liste :

(((a) b c d) e ((f) g) (((h))) (i (j)))

Ecrire les fonctions suivantes :

firstn (N L) : retourne la tête formée par les N premiers éléments de la liste L.

(firstn 2 '(2 3 4 5)) -> (2 3)

inter (L M) : retourne l'intersection des deux listes L et M

elim(L) : retourne la liste L après élimination des répétitions

(elim '(a z g a s z d g z s)) -> (a z g s d)

nbFeuilles (L) : retourne le nombre de feuilles d'un arbre généalogique représenté par une liste

(nbFeuilles '(r ((t)) y (g h) (j m l) p)) -> 9

monEqual : retourne t ou nil selon l'égalité de ses deux arguments.

Cette fonction existe sous le nom de « equal ». Vous vous aiderez de la fonction standard EQ et de prédicats. Afin de bien comprendre, testez :

(eq 'LUC 'LUC)

(eq 'LUC 'DANIEL)

(eq (car '(do re)) (cadr '(mi do sol)))

(eq '(d p f t r) '(d p f t r))

Faites les mêmes tests avec la fonction standard EQUAL. Expliquez les résultats obtenus

Exercice 2 : Objets fonctionnels

L'objectif est de voir comment utiliser `mapcar` avec des fonctions anonymes, des expressions *lambda*. Une fonction anonyme est une fonction sans nom qui peut être fournie en paramètre à toute fonction LISP. Le format est:

```
(lambda (paramètres)
  body)
```

où `<body>` est une suite d'instructions LISP. Par exemple:

```
(lambda (x)
  (* x 3))
```

est une fonction anonyme qui retourne le triple de l'argument passé en paramètre.

Ecrivez une fonction *list-pair*e qui retourne la liste des éléments par paire de deux listes fournies en paramètre. Utilisez `mapcar`.

Exemple:

```
(list-paire '(0 2 3 11) '(6 10 20 30))  -> ((0 6) (2 10) (3 20) (11 30))
```

Exercice 3 : *a-list*

Ecrivez la fonction décrite ci-dessous.

my-assoc

Arguments :
cl é : une S-Expression quelconque ;

a- l i st e : une liste d'association, c'est-à-dire une liste de la forme :

((cl é$_1$ val eur $_1$) (cl é$_2$ val eur $_2$) . . . (cl é$_n$ val eur $_n$))

Spécification :
(my- assoc cl e a- l i st e) retourne ni l si cl e ne correspond à aucune clé de la liste d'association, la paire correspondante dans le cas contraire; cette fonction existe sous le nom de assoc.

Exemples :

```
? (my-assoc 'Pierre
            '((Yolande 25) (Pierre 22) (Julie 45)))
(Pierre 22)

? (my-assoc 'Yves
```

```
'((Yolande aime Pierre)
  (Pierre aime Julie)
  (Julie aime Pierre)))
nil
```

Exercice 4 : gestion d'une base de connaissances en Lisp

On considère une base de connaissances sur des ouvrages littéraires définis par les propriétés suivantes : titre, auteur, année de parution, et nombre d'exemplaires vendus.

Soit la base BaseTest suivante :

```
(setq BaseTest '((" Le Dernier Jour d'un condamné ", Hugo, 1829, 50000)
                 (" Notre-Dame de Paris ", Hugo, 1831, 3000000)
                 (" Les Misérables ", Hugo, 1862, 2000000)
                 ("Le Horla ", Maupassant, 1887, 2000000)
                 (" Contes de la bécasse ", Maupassant, 1883, 500000)
                 ("Germinal ", Zola, 1885, 3000000)
))
```

A. Définir les fonctions de service :

auteur (ouvrage) : retourne l'auteur de l'ouvrage passé en argument

> (auteur '(" Notre-Dame de Paris ", Hugo, 1831, 3000000))

Hugo

titre (ouvrage) : retourne le titre de l'ouvrage passé en argument

> (titre '(" Notre-Dame de Paris ", Hugo, 1831, 3000000))

" Notre-Dame de Paris "

annee (ouvrage) : retourne l'année de l'ouvrage passé en argument

>(annee '(" Notre-Dame de Paris ", Hugo, 1831, 3000000))

1831

nombre (ouvrage) : retourne le nombre d'exemplaires vendus de l'ouvrage passé en argument

> (nombre '(" Notre-Dame de Paris ", Hugo, 1831, 3000000))

3000000

B. En utilisant ces fonctions de service si nécessaire, définir les fonctions suivantes :

FB1 : affiche tous les ouvrages

FB2 : affiche les ouvrages dont l'auteur est Hugo

FB3 : retourne la liste des titres d'ouvrage dont l'auteur est précisé en argument

FB4 : retourne le premier ouvrage paru en année X (X = un argument) ou nil

FB5 : retourne la liste des ouvrages dont le nombre d'exemplaires vendus dépasse 1000000 ou nil

FB6 : calcule et retourne la moyenne du nombre d'exemplaires vendus de l'auteur X (X = un argument)