Programmation fonctionnelle

Répétition 4

15 mars 2007

Correction des exercices proposés

1. Ecrire une fonction `frequency` prenant en argument une liste d'atomes `ls` et renvoyant une table d'apparition de chacun des atomes dans la liste `ls`. Cette table sera représentée par une liste de paires pointées, dont le car est un atome et le cdr la fréquence d'apparition de cet atome. Tous les atomes de `ls` apparaissent une et une seule fois dans la table.

`(frequency '(a b c b a b d a c b b))` ⇒ `((a . 3) (b . 5) (c . 2) (d . 1))`

2.* Soit la fonction `f` definie comme suit : si $n < 3$, alors $f(n) = n$. Sinon, $f(n) = f(n-1)+f(n-2)+f(n-3)$. Ecrire un code SCHEME qui implémente efficacement cette fonction.

Expressions à évaluer

3.$^\clubsuit$ Evaluer les expressions suivantes :

```
(((lambda (f) f) (lambda (x) x)) 5)

(((lambda (f) f) (lambda (x) (x x))) 5)

(((lambda (f) (f f)) (lambda (x) x)) 5)

((lambda (f x) (f (f x)))
 (lambda (x) (* x x))
 ((lambda () (* 2 2))))

((cadr (map (caddr (map (lambda (f)
                          (lambda (g)
                            (lambda (x) (f (g x)))))
                        (list car cadr caddr)))
            (list cdr cddr cdddr)))
 '(0 1 2 3 4 5))

((lambda (x) (list x (list (quote quote) x)))
 (quote (lambda (x) (list x (list (quote quote) x)))))
```

Spécification

4.♣ Spécifier la fonction suivante :

```
(define fct
  (lambda (f l)
    (cond
     ((null? l) 0)
     (else (+ (f (car l)) (fct f (cdr l)))))))
```

Quelle est la valeur de `(fct (lambda (x) (+ (* x x) 1)) '(1 2 3))` ?

Exercice sur les prédicats

5.♣ Ecrire une fonction `filter` prenant deux arguments, une liste `l` d'objets et un prédicat unaire `p`, renvoyant la liste des éléments de `l` pour lesquels `p` est vrai.

Définir alors au moyen de la fonction précédente, la fonction `greater` prenant comme arguments une liste `l` de nombres entiers et un nombre entier `a`, et renvoyant la liste des éléments de `l` strictement supérieurs à `a`.

Par exemple : `(greater '(5 2 7 4 3 6) 4)` $\Rightarrow$ `(5 7 6)`.

Récursion profonde sur les listes

Définir la procédure `count-all-1` à un seul argument qui compte, en profondeur, le nombre de 1 contenus dans l'argument.

```
(count-all-1 '(0 (1 2 (3 4 (1)) (3 (2 1) 1) 1) 0 (1 2 (1 2 3))))
```

renvoie 7.

Exercice sur les compositions de fonctions

6.⋆ Définir une fonction `compose-n` qui renvoie la fonction unaire donnée en argument `n` fois composée avec elle-même.

Variante: écrire une fonction `compose-fgf` qui prend comme argument une fonction f et renvoie une fonction qui prend comme argument une fonction g et qui renvoie la fonction $f \circ g \circ f$.

Variante 2: écrire une fonction `compose-fgab` qui prend comme argument deux fonctions f et g, ainsi que deux entiers a et b et renvoie la fonction

$$\underbrace{f \circ \ldots f}_{a} \circ \underbrace{g \circ \ldots g}_{b}$$

Variante 3 : écrire une fonction `compose-fa` qui prend comme argument une fonction f et deux entier a, b et renvoie la fonction

$$x \mapsto \underbrace{f \circ \ldots f}_{a}(b^x)$$

Exercices proposés

7.$^{\star\clubsuit}$ Ecrire un prédicat `prime ?` qui détermine si un entier strictement positif est premier ou non.

8.$^{\star\clubsuit}$ Un carré magique de dimension n est un tableau de taille $n \times n$ dans lequel chaque nombre de 1 à $n \times n$ apparaît une fois, et dont les sommes des lignes, des colonnes et des diagonales sont égales. Ecrire un prédicat magic ? qui vérifie si un carré de nombres est magique ou non. On représente un carré de nombres de dimension `n` par une liste de `n` listes contenant chacune `n` éléments.

`(magic? '((4 9 2) (3 5 7) (8 1 6)))` $\Rightarrow$ `#t`

`(magic? '((4 4 7) (5 3 7) (8 1 6)))` $\Rightarrow$ `#f`