INFO0054 - Programmation fonctionnelle

Répétition 4: Les spécificités de la programmation fonctionnelle

Jean-Michel Begon

2 Mars 2016

Closure

Exercice 1.
Définir une fonction `affine-map` qui prend deux réels `a` et `b` en argument et qui renvoie la fonction $f : x \in R \rightarrow ax + b$.

Exercice 2.
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

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

Continuations

Exercice 3.
Écrire une fonction `range` qui prend deux entiers comme argument, `min` et `max`, et qui renvoie la liste des entiers compris entre ces deux bornes (inclues).

```
(range -2 6) => '(-2 -1 0 1 2 3 4 5 6)
```

Exercice 4.
Réécrire la fonction `range` sous la forme d'un itérateur paresseux : `xrange` prend deux entiers comme argument, `min` et `max`, et renvoie la première fonction de la suite de $m = \min(0, max - min + 1)$ éléments vérifiants $f_i = (m_i \,.\, f_{i+1})$ et $f_{m+1} =' ()$.

Exercice 5.
Ecrire une fonction `it->ls` qui prend en argument une paire d'un itérateur paresseux et qui renvoie la liste des itérés.

```
(it->ls ((xrange -2 6))) => '(-2 -1 0 1 2 3 4 5 6)
```

Les listes variables d'arguments

Exercice 6.
Ecrire une fonction `plus` à un nombre variable d'arguments et qui renvoie la somme de ces éléments.

Exercice 7.
Ecrire une fonction `linear-map` à un nombre variable d'arguments réels et qui renvoie une fonction prenant une liste de réel en argument et renvoyant le produit scalaire entre cette liste et les arguments encapsulés.

Exercice 8.
Ecrire une fonction `curry` qui prend deux arguments, une fonction $f : D_1 \times \ldots \times D_n \to Y$ et un élément $d_1 \in D_1$ et qui renvoie une fonction $h : D_2 \times \ldots \times D_n \to Y$ telle que $(h\ d_2 \ldots d_n) = (f\ d_1\ d_2 \ldots d_n)$.

Les structures de données

Exercice 9.
Proposer une implémentation d'une file à l'aide de deux listes.