Programmation applicative – L2
TD 1 : Premiers concepts d'évaluation – Correction

Guillaume Artignan {artignan@lirmm.fr}
Annie Chateau {chateau@lirmm.fr}
Hervé Dicky {dicky@lirmm.fr}
Christophe Dony {dony@lirmm.fr}
Bruno Paiva Lima da Silva {bplsilva@gmail.com}

1 Préambule : Algèbre de Boole

Exercice 1 *Donner les valeurs des expressions suivantes (on note V pour Vrai et F pour Faux) :*

```
V et ( non F )                              V
( V ou F ) et (( non V ) et F )             F
V ou X                                      V
```

```
 X | V ou X
-----------
 V |   V
-----------
 F |   V
```

```
( V ou X ) ou ( F et Y)                     V
```

```
 X | Y | V ou X | F et Y | (V ou X) ou (F et Y)
-----------------------------------------------
 V | V |   V    |   F    |          V
-----------------------------------------------
 F | V |   V    |   F    |          V
-----------------------------------------------
 V | F |   V    |   F    |          V
-----------------------------------------------
 F | F |   V    |   F    |          V
```

```
( non ( V et X ))                           non X
```

```
 X | V et X | (non (V et X))
---------------------------
 V |   V    |      F
---------------------------
 F |   F    |      V
```

```
(( non V ) ou ( non X ))                    non X
```

```
 X | non X | ((non V) ou (non X))
---------------------------------
 V |   F    |      F
---------------------------------
 F |   V    |      V
```

Construire une table de vérité pour les quatre dernières expressions.

2 Syntaxe de Scheme (notation parenthésée préfixée et S-expressions)

Exercice 2 *Transformer les expressions suivantes en notation préfixée :*

`5 - 4`	`(- 5 4)`
`2 - 6 + 10`	`(+ (- 2 6) 10)`
`2 + 5 * 3`	`(+ 2 (* 5 3))`
`3 * 5 - 7 / 2`	`(- (* 3 5) (/ 7 2))`
`5 + 3 * (4 - 2) + 5 / (2 + 1)`	`(+ 5 (* 3 (- 4 2)) (/ 5 (+ 2 1))`

Exercice 3 *En utilisant les noms de fonctions Scheme :* `+`, `-`, `*`, `/`, `<`, `=`, `sqrt` ($\sqrt{\ldots}$), `and`, `or`, `not`*, transformer les expressions algébriques suivantes en expressions Scheme :*

$1-(2-3)$	`(- 1 (- 2 3))`
$(1-2)-3$	`(- (- 1 2) 3)`
$3+7\times(2+1)$	`(+ 3 (* 7 (+ 2 1)))`
$a\times x^2+b\times x+c$	`(+ (* a x x) (* b x) c)`
$(a+1)\neq b$	`(not (= (+ a 1) b))`
$(a<b)$ *ou* $(a=b^2)$	`(or (< a b) (= a (* b b)))`
$\frac{a\times x+b}{c\times x+d}$	`(/ (+ (* a x) b) (+ (* c x) d))`
$\sqrt{b^2-4\times a\times c}$	`(sqrt (- (* b b) (* 4 a c)))`

Exercice 4 *Les expressions Scheme suivantes sont-elles syntaxiquement correctes ? Si ce n'est pas le cas, préciser ce qui ne va pas.*

`(+ 1 (* 2 3))`	`ok`
`+`	`ok (l'interpréteur répond : #<procedure:+> )`
`(+ 2 *)`	`le deuxième opérande n'est pas du bon type`
`(+ (/ 2 3) 1))(+ a b)`	`problème de parenthésage`
`(= 1)`	`il manque un opérande`

Exercice 5 *Donner la forme préfixée des expressions suivantes :*

a) $2+4x^2-3x^3$ `(+ 2 (- (* 4 x x) (* 3 x x x)))`

b) $\frac{sin(x+y)\times cos(x-y)}{1+cos(x+y)}$ `(/ (* (sin (+ x y)) (cos (- x y))) (+ 1 (cos (+ x y))))`

3 Les formes spéciales `define` et `lambda`

Exercice 6 *Définir en Scheme les fonctions suivantes à l'aide de lambda-expressions : :*

- $f : x \mapsto x^3$

```
(define f
  (lambda (x)
    (* x x x)))
```

- $g : y \mapsto 3y + 7$

```
(define g
  (lambda (x)
     (+ (* 3 x) 7)))
```

- $h : z \mapsto 3f(z) + g(z) + 1$

```
(define h
  (lambda (x)
     (+ (* 3 (f x)) (g x) 1)))
```

- *La fonction constante* $k2 : x \mapsto 2$.

```
(define k2 (lambda (x) 2))
```

- *La fonction identité* $id : x \mapsto x$.

```
(define id (lambda (x) x))
```

- *La fonction proj2 :* $(x, y) \mapsto y$.

```
(define proj2 (lambda (x y) y))
```

- *La fonction einstein :* $(u, v) \to \frac{u+v}{1+\frac{uv}{c^2}}$ *avec* $c = 300000$

```
(define einstein
  (lambda (u v)
    (\ (+ u v) (+ 1 (/ (* u v) (* 300000 300000))))))
```

Exercice 7 *Indiquer l'effet des fonctions suivantes :*

```
(define (f1) 0)
```

définit la fonction sans argument, constante égale à 0.

```
(define (f2 x) (* x x))
```

définit la fonction $f2 : x \mapsto x^2$.

```
(define (f3 x y) (+ (f2 x) (f2 y)))
```

définit la fonction $f3 : (x, y) \mapsto x^2 + y^2$.

```
(define (distc x1 y1 x2 y2) (f3 (- x1 x2) (- y1 y2)))
```

définit la fonction distc : $(x_1, y_1, x_2, y_2) \mapsto (x_1 - x_2)^2 + (y_1 - y_2)^2$, *autrement dit le carré de la distance euclidienne entre deux points du plan euclidien, de coordonnées* (x_1, y_1) *et* (x_2, y_2) *respectivement.*

Exercice 8 *Suivre le cheminement de Scheme sur l'exemple suivant. Que se passe-t-il à chacune de ces commandes ?*

```
> (define pi 3.14)                      rien ne s'affiche (définition)
> pi                                    3.14 (évaluation)
> (define (carre x) (* x x))            rien ne s'affiche (définition)
> (carre 5)                             25 (évaluation)
> (define (cercle r) (* pi (carre r)))  rien ne s'affiche (définition)
> (cercle 12)                           452.16 (évaluation)
```

4 La forme spéciale `if`

Exercice 9 *Définissez la fonction* `abs` *qui renvoie la valeur absolue d'un nombre. Écrivez ensuite la fonction* `care-div` *qui fait la division de 2 nombres en vérifiant au préalable que le diviseur n'est pas nul.*

```
(define abs
  (lambda (x)
    (if (>= x 0) x (- 0 x))))

(define care-div
  (lambda (a b)
    (if (= b 0)
      (display "Erreur : le diviseur est nul")
      (/ a b))))
```

5 Expressions et fonctions non récursives

Exercice 10 *Donner la fonction Scheme* `fahrenheit->celsius` *qui permet de convertir une température donnée en degré Fahrenheit en température Celsius en utilisant la formule suivante :*

$$F(C) = \frac{9}{5}(C + 40) - 40$$

Attention, il faut "inverser" la formule !

```
(define fahrenheit->celsius
  (lambda (t) (- (/ (* (+ t 40) 5) 9) 40)))

(define celsius->fahrenheit
  (lambda (t) (- (/ (* (+ t 40) 9) 5) 40)))
```

Écrire une fonction `liquide-cel?` *pour savoir si une température donnée en degrés celsius correspond à un état de l'eau qui n'est ni glace ni vapeur.*

```
(define liquide-cel?
  (lambda (t) (and (> t 0) (< t 100))))
```

Donner une fonction `liquide-fah?` *pour savoir si une température donnée en degrés fahrenheit correspond à un état de l'eau qui n'est ni glace ni vapeur.*

```
(define liquide-fah?
  (lambda (t) (liquide-cel? (fahrenheit->celsius t))))
```

Exercice 11 *Donner les fonctions* `puissance-2, puissance-4, puissance-6` *et* `puissance-9`*, qui, étant donné un argument x, calculent respectivement x^2, x^4, x^6 et x^9.*

Soit on définit toutes les fonctions sur le même modèle, à savoir

```
(define puissance-4 (lambda (x) (* x x x x)))
```

en mettant le "bon" nombre de `x`, soit on essaie d'anticiper un peu la notion de généralisation, en "réutilisant" les fonctions définies au fûr et à mesure :

```
(define puissance-2 (lambda (x) (* x x)))

(define puissance-4 (lambda (x) (* (puissance-2 x) (* puissance-2 x))))
ou
(define puissance-4 (lambda (x) (puissance-2 (puissance-2 x))))

(define puissance-6 (lambda (x) (* (puissance-2 x) (puissance-4 x))))

(define puissance-9 (lambda (x) (* x (puissance-2 (puissance-4 x)))))
```

discuter s'il y a le temps d'une généralisation possible en utilisant les chiffres de la décomposition binaire de l'exposant.

Exercice 12 *Écrire les fonctions* `unité, dizaine` *et* `centaine` *dont les évaluations sont respectivement égales au chiffre des unités, au chiffre des dizaines et au chiffre des centaines du nombre donné comme paramètre.*

Cet exercice fait appel à la notion de numération en base 10, et à la division euclidienne (rappeler ce qu'est la division euclidienne, à toutes fins utiles). Faire remarquer que le chiffre des unités est le reste de la division euclidienne par 10, que le chiffre des dizaines est le reste de la division euclidienne par 10^2 moins le chiffre des unités, le tout divisé par 10, et ainsi de suite... Introduire la fonction `modulo` de Scheme, ou bien la fonction `remainder`.

```
(define unite
  (lambda (x)
    (modulo x 10)))

(define dizaine
  (lambda (x)
    (/ (- (modulo x 100) (unite x)) 10)))

(define centaine
  (lambda (x)
    (/ (- (modulo x 1000) (* 10 (dizaine x)) (unite x)) 100)))
```

Autre solution : le chiffre des dizaines est aussi le chiffre des unités du quotient de la division euclidienne par 10, le chiffre des centaines le chiffre des unités de la division euclidienne par 100...

```
(define dizaine
  (lambda (x)
    (unite (quotient x 10))))

(define centaine
  (lambda (x)
    (unite (quotient x 100))))
```

Exercice 13 *Écrire les fonctions* `min-2` *et* `max-2` *qui, étant donnés deux nombres, donnent respectivement le plus petit et le plus grand des deux nombres.*

```
(define min-2 (lambda (x y) (if (< x y) x y)))
(define max-2 (lambda (x y) (if (> x y) x y)))
```

Écrire les fonctions `min-3` *et* `max-3` *qui, étant donnés deux nombres, donnent respectivement le plus petit et le plus grand des trois nombres.*

```
(define min-3 (lambda (x y z) (min-2 (min-2 x y) z)))
(define max-3 (lambda (x y z) (max-2 (max-2 x y) z)))
```