

Paradigmes de Programmation

Exercices en Scheme

Année Académique 2006-2007

Tom Mens et Johan Brichau

March 6, 2007

1 Évaluation d'expressions simples

Exercice 1.1 *Lesquelles des expressions suivantes sont syntaxiquement illégales en Scheme et pourquoi? (L'évaluation de certaines parmi ces expressions peuvent rapporter une erreur lors de leur exécution, malgré leur légalité syntaxique.)*

- `x`
- `(= y z)`
- `(3 + 4)`
- `sin(0)`
- `(= (= y z) 0)`
- `(define (f x) x)`
- `(define (f x) y)`
- `(define (f 'x) x)`
- `(define (f) 10)`
- `(f)`
- `(define (f #t) 4)`

Exercice 1.2 *Expliquez pourquoi les expressions suivantes renvoient une erreur lors de leur exécution.*

1. `(+ +)`
2. `(define (f n) (+ (/ n 3) 2))`
`(f 5 8)`
3. `(+ 5 (/ 1 0))`
4. `(sin 10 20)`
5. `(mafonction 10)`
6. `(define (mafonction x) (sin x x))`
`(mafonction 10 20)`
`(mafonction 10)`

Exercice 1.3 *Prédisez les expressions arithmétiques suivantes, évaluez-les et comparez les résultats.*

```

54
(+ 23 55)
(+ 23 44 99)
(+ 23 (- 55 44 33) (* 2 (/ 8 4)))
(/ 66 43)
(define a 3)
a
(/ 6 a)
(define b (+ a 1))
(+ a b (* a b))

```

Exercice 1.4 *Prédisez les expressions Booléennes suivantes, évaluez-les et comparez les résultats. Les variables a et b sont définis comme dans l'exercice précédent.*

```

(= 2 3)
(= 3 3)
(= a b)
(not (or (= 3 4) (= 5 6)))
(+ 2 (if (> a b) a b))

```

Exercice 1.5 *Prédisez les expressions conditionnelles suivantes, évaluez-les et comparez les résultats:*

```

(if (= 1 1) "waaw" "brrr")
(if (= 4 4) 5 6)
(if (> a b) a b)
(if (and (> b a) (< b (* a b))) b a)
(+ 2 (if (> a b) a b))
((if (< a b) + -) a b)
(cond ((= 1 1) "waaw 1")
      ((= 2 2) "waaw 2")
      ((= 3 3) "waaw once more")
      (else "waaw final"))
(* (cond ((> a b) a)
      ((< a b) b)
      (else -1))
  (+ a 1))

```

Exercice 1.6 *Exprimez l'expression arithmétique suivante en Scheme et évaluez-la:*

$$\frac{\frac{12}{19} + \frac{5+9}{2}}{(10+11) * \frac{20}{3}}$$

2 Fonctions et “special forms”

Exercice 2.1 Bien que `lambda`, `define`, `if`, `cond`, `or` et `and` suivent la syntaxe régulière, ils ne sont pas des fonctions régulières. Prouvez ceci.

Exercice 2.2 Écrivez deux fonctions `celcius-to-fahrenheit` et `fahrenheit-to-celcius`. La formule de conversion est $F = (C + 40) * 1,8 - 40$.

Exercice 2.3 Prédisez les appels de fonctions “special form” suivantes, évaluez-les et comparez les résultats:

```
(and #f (/ 1 0))
(and #t (/ 1 0))
(if #t 2 (/ 1 0))
(if #f 2 (/ 1 0))
(and #t #t #f (/ 1 0))
(and #t #t #t (/ 1 0))
```

3 Les variables locales

Exercice 3.1 *Le “special form” let n’est pas essentiel en Scheme. C’est du sucre syntaxique pour une expression en termes de lambda. Essayez de trouver cette expression.*

Exercice 3.2 *Prédisez la valeur de l’expression suivante:*

```
(let ((x 3))
  (let ((x (* x x)))
    (+ x x)))
```

Exercice 3.3 (*)

Pourquoi le programme suivant produit-il une erreur (et quelle erreur pensez-vous que c’est)? Fixez le programme de deux façons différentes.

```
(let ((x 1)
      (y (* 2 x)))
  (+ x y))
```

Exercice 3.4 *Prédisez le résultat du programme suivant:*

```
(define x 1)
(define y 2)
(define (f x y)
  (let ((x y)
        (y x))
    (begin
      (display x)
      (display y))))
(f 3 4)
```

Qu’est ce qui se passe quand on remplace let par let dans le programme ci-dessus?*

Exercice 3.5 *Prédisez le résultat du programme suivant:*

```
(define (test)
  (letrec
    ((pair? (lambda (n)
              (if (= n 0)
                  #t
                  (impair? (- n 1)))))
      (impair? (lambda (n)
                (if (= n 0)
                    #f
                    (pair? (- n 1)))))
      (begin
        (display (pair? 5))
        (display (impair? 5))
        (display (pair? 6))
        (display (impair? 6)))))
  (test))
```

Qu'est ce qui se passe quand on remplace `letrec` par `let` dans le programme ci-dessus? Qu'est ce qui se passe quand on remplace `letrec` par `let` dans le programme ci-dessus?*