MIAS 1 et MASS 1 – Programmation fonctionnelle

TP n° 7: corrigé

```
(load "pack-graph.sc")


; carre : dessine un carre de cote lg, centre sur l'origine du repere

(define carre
  (lambda (lg)
    (define co (/ lg 2))
    (begin
      (position-pen (- co) co)
      (draw-line-to co co)
      (draw-line-to co (- co))
      (draw-line-to (- co) (- co))
      (draw-line-to (- co) co))))



; anime : affiche un carre dont le cote n varie de 400 a 0
; on dessine d'abord un carre blanc de cote n+2 qui efface le carre
; dessine a l'appel recursif precedent, puis on dessine le carre de cote
; n et on appelle recursivement la fonction avec un cote n-2.

(define anime
  (lambda ()
    ; on utilise une fonction auxiliaire
    (define anime-aux
      (lambda (n)
        (if (<= n 0) #f
            (begin
              (white-pen)
              (carre (+ n 2))
              (black-pen)
              (carre n)
              (anime-aux (- n 2))))))
    (begin
      (clear-graphics)
      (anime-aux 400))))



; segment : trace un segment de droite entre les points (x,y) et (z t)

(define segment
  (lambda (x y z t)
    (begin
      (position-pen x y)
      (draw-line-to z t))))
```

```
(define fractale
  (lambda (x1 y1 x2 y2 k n)
    (let ( (x3 (/ (+ (* 3 x1) x2) 4))
           (y3 (/ (+ (* 3 y1) y2) 4))
           (x4 (/ (- (* 2 (+ x1 x2)) (* k (- y2 y1))) 4))
           (y4 (/ (+ (* 2 (+ y1 y2)) (* k (- x2 x1))) 4))
           (x5 (/ (+ (* 2 (+ x1 x2)) (* k (- y2 y1))) 4))
           (y5 (/ (- (* 2 (+ y1 y2)) (* k (- x2 x1))) 4))
           (x6 (/ (+ x1 (* 3 x2)) 4))
           (y6 (/ (+ y1 (* 3 y2)) 4)) )
      (if (= n 0)
          (segment x1 y1 x2 y2)
          (begin
            (fractale x1 y1 x3 y3 k (- n 1))
            (fractale x3 y3 x4 y4 k (- n 1))
            (fractale x4 y4 x5 y5 k (- n 1))
            (fractale x5 y5 x6 y6 k (- n 1))
            (fractale x6 y6 x2 y2 k (- n 1)) )))))



; Cette fractale est une variante du dragon qu'on obtient en effectuant
; l'un des deux appels  recursifs sur un segment dont on donne les coordonnees
; dans le sens indirect.
; Par exemple, on effectue le  2e appel recursif sur le segment [(a2,b2); (a3,b3)]
; au lieu du segment [(a3,b3); (a2,b2)]

(define dessin
  (lambda (a1 b1 a2 b2 n)
    (if (= n 0)
        (segment a1 b1 a2 b2)
        (let ( (a3 (/ (+ a1 b1 a2 (- b2)) 2))
               (b3 (/ (+ a2 b2 b1 (- a1)) 2)) )
          (begin
            (dessin a1 b1 a3 b3 (- n 1))
            (dessin a3 b3 a2 b2 (- n 1)))))))
```