ISIMA 1ère année

1er septembre 2010
Durée : 2 heures
Documents autorisés

PROGRAMMATION FONCTIONNELLE

Exercice 1 :

On supposera que l'on dispose d'une fonction `sqrt` telle que l'évaluation de l'expression `(sqrt x)` retourne la racine carrée d'un nombre positif ou nul `x`.

1°) Ecrire une fonction `DM` ayant comme arguments une liste `E` représentant un ensemble E d'éléments de type quelconque et une fonction `d` représentant une distance dans E (c'est-à-dire une fonction de $E \times E$ dans $\mathbb{R}_+$) et qui est telle que l'évaluation de l'expression `(DM E d)` retourne la distance moyenne entre les tous les éléments de `E` pris deux à deux, en utilisant la fonction `d` comme distance dans `E`.

Par exemple, l'évaluation de `(DM '(1 2 4) (lambda (x y) (abs (- x y))))`, où la fonction `abs` désigne la fonction valeur absolue dans $\mathbb{R}$, doit retourner 2 (la moyenne de | 1 – 2 |, | 1 – 4 | et | 2 – 4 |).

Solution :

```
(define DM (lambda (E d)
   (let* ((n (length E)) (nc (/ (* n (- n 1)) 2)))
      (/ (SD E d) nc) ) ))
(define SD (lambda (E d)
   (if (null? E) 0
       (+ (SD2 (car E) (cdr E) d) (SD (cdr E) d)) ) ))
(define SD2 (lambda (p E d)
   (apply + (map (lambda (x) (d p x)) E) )))
```

2°) Ecrire en utilisant la fonction `DM` une fonction `DMC2` ayant comme argument un entier `n` et telle que l'évaluation de l'expression `(DMC2 n)` retourne la distance moyenne entre les n^2 points du plan de coordonnées entières comprises entre 1 et `n` en utilisant la distance euclidienne $d(A, B) = \sqrt{(x_B - x_A)^2 + (y_B - y_A)^2}$.

Solution :

```
(define sqr (lambda (x) (* x x)))
```

```
(define deucl (lambda (p1 p2)
   (sqrt (+ (sqr (- (car p1) (car p2)))
            (sqr (- (cadr p1) (cadr p2))) )) ))
(define DMC2 (lambda (n) (DM (carre n) deucl)))
(define carre (lambda (n)
   (let ((Ln1 (CreerLn1 n)))
(map (lambda (x) (map (lambda (y) (list x y)) Ln1)) Ln1) ) ))
(define CreerLn1 (lambda (n)
   (if (= n 1) '(1) (cons n (CreerLn1 (- n 1)))) ))
```

Exercice 2 :

Ecrire une fonction `S` qui a deux arguments :

- une liste "plate" (sans sous liste), `LP`, de longueur p ;
- et une liste "structurée" (avec éventuellement des sous listes), `LS`, de longueur $s \leq p$;

et telle que l'évaluation de `(S LP LS)` retourne un couple dont les deux termes sont respectivement :

- une liste structurée comme `LS` et qui contient les s premiers éléments de `LP` ;
- une liste plate contenant les p - s éléments restants de `LP`.

Exemple :

L'évaluation de `(S '(1 2 3 4 5) '(a (a (a))))` doit retourner :

```
((1 (2 (3))) (4 5))
```

Solution :

```
(define S (lambda (LP LS)
   (if (null? LS) (list () LP))
       (if (list (car LS))
          (let* ((C1 (S LP (car LS)))
                 (P (car C1))
                 (RLP (cadr C1))
                 (C2 (S RLP (cdr LS))) )
             (list (cons P (car C2)) (cadr C2)) ) )
          (let ((C2 (S (cdr LP) (cdr LS))))
             (list (cons (car LP) (car C2)) (cadr C2)) ) ) ) ))
```