Une dernière fonction récursive : un évaluateur d'expression d'un langage

Université Montpellier-II - UFR - ULIN201 - Langages Applicatifs -
Chapitre 10 :
C.Dony

17 Une dernière fonction récursive

Réalisons un langage de programmation défini par sa fonction d'interprétation de ses expressions. Dotons le minimalement pour débuter.

– Des types de base et leurs fonctions primitives
  - nombres
  - booléens
  Des identificateurs
  Une conditionnelle, nommée "si", ayant la même sémantique que la traditionnelle conditionnelle.
– Une syntaxe. Pour simplifier, prenons celle de Scheme. Nous disposons d'un analyseur syntaxique tout prêt, implanté par la fonction "read".
– Une sémantique des construction. Prenons celles de Scheme et mettons la en oeuvre via une fonction d'interprétation "eval304". Rendons la utilisable via une boucle "toplevel" implantée par la fonction "scheme304".

```
(define (scheme304)

  ;; sans fonction utilisateur
  ;; sans environnement
  ;; sans gestion de la pile

  (let ((**global-env (list (list '+ +) (list '- -) (list '* *) (list
'/ /) (list '= =) (list 'not not)))
        (**primitives '(+ - * / = not))
        (**fin? #f))

    (define (eval304 e)
      (cond ((or (number? e) (boolean? e) (string? e) (procedure? e))
             e)
            ((symbol? e) (env-value e))
            ((list? e)
             (cond ((eq? (car e) 'fin) (set! **fin? #t) 'Au-revoir)
                   ((eq? (car e) 'si) (eval-si e))
                   ((primitive? (car e)) (apply (eval304 (car e)) (evlis (cdr e))))
                   (#t (error (car e) "fonction primitive inconnue"))))
            (#t (error e "construction_inconnue"))))

    (define (primitive? f) (memq f **primitives))

    (define (env-value symbol)
      (let ((liaison (assq symbol **global-env)))
        (if liaison
            (cadr liaison)
            (erreur symbol "variable_indéfinie: "))))

    (define (eval-si e)
      (if (eval304 (cadr e))
          (eval304 (caddr e))
          (eval304 (cadddr e))))

    (define (evlis l)
      (map eval304 l))
```

```
    (do ()
      (**fin? 'A_bientôt)
      (print (eval304 (read))))))

(scheme304)
```