Mathematisches Institut
der Universität München
H. Schwichtenberg
L. Crosilla, M. Seisenberger

Wintersemester 2002/2003
Blatt 7

Übungen zum Ferienkurs über Nichtnumerisches Programmieren

Aufgabe 26. Sei

$$\varphi := \frac{1+\sqrt{5}}{2}, \qquad \hat{\varphi} := \frac{1-\sqrt{5}}{2}$$

Man zeige

(a) (Euler). $\mathsf{Fib}(n) = \frac{1}{\sqrt{5}}(\varphi^n - \hat{\varphi}^n)$ (Hinweis: φ und $\hat{\varphi}$ lösen $x^2 - x - 1 = 0$).

(b) $\mathsf{Fib}(n)$ ist diejenige ganze Zahl, die von $\varphi^n/\sqrt{5}$ den geringsten Abstand hat. (Hinweis: Man schätze den numerischen Wert von $\hat{\varphi}^n$ ab).

(c) Man verwende dies zum Schreiben einer Prozedur `fib` zur Berechnung der `Fibonacci`-Zahlen mit Zeitkomplexität $O(\log(n))$.

Aufgabe 27. (a) Man schreibe eine Prozedur `(show-frame k)`, die die Variablenbelegung im aktuellen Rahmen (mit Index `k`) sowie den Index des Elternrahmens anzeigt.

(b) Man schreibe eine Prozedur `(show-var-env from-index to-index)`, die die gegenwärtige Variablenumgebung von den Indizes `from-index` bis `to-index` (einschliesslich) anzeigt.

Aufgabe 28. Die Prozeduren `eval` und `app` des Interpreters rufen sich gegenseitig auf. Man vervollständige die Definition von `eval`

```
(define (eval w k)
  (let ((key (keyword w)))
    (cond
     ((eq? 'true key) w)
     ((eq? 'false key) w)
     ((eq? 'number key) w)
     ((eq? 'symbol key) (var-env-lookup (cadr w) k))
     (else
      (let* ((ws (val-to-list w))
             (w0 (car ws))
             (key0 (keyword w0)))
        (if (and (eq? 'symbol key0)
                 (memq (cadr w0) syntactic-keywords))
            ...
            ;sonst Anwendungsterm
            (let* ((vs (do ((l (cdr ws) (cdr l))
                            (res '() (cons ?? res)))
                           ((null? l) (reverse res))))
                   (v0 ??))
              (app ?? ??))))))))
```

und (unter Verwendung von `insert-frame` und `make-frame`) die Definition von `app`

```
(define (app v vs)
  (let ((key (keyword v)))
    (cond
      ((eq? 'proc key)
       (let* ((k (proc-to-frame-index v))
              (w (proc-to-lambda-expr v))
              (vars (lambda-expr-to-vars w))
              (w1 (lambda-expr-to-body w)))
         (eval ?? ??)))
      ((eq? 'prim-proc key)
       (let* ((symbol (cadr v))
              (info (assq symbol prim-procs-on-atoms)))
         (if info
             (atom-to-atom-val
               (apply (cadr info) (map cadr vs)))
             ;sonst primitive Prozedur auf Werten
             (apply (cadr (assq symbol prim-procs-on-values))
                    vs)))))))
```

Abgabetermin. Donnerstag, den 10. Oktober 2002, in der Übung oder (besser) per email an `seisenb@mathematik.uni-muenchen.de`