Musterlösung V. Übungsblatt “Informatik I” WS 95/96

Ioannis Potikoglu, Christian Schulte

Aufgabe V.1 (Definitionen) `f` muß wie folgt gebunden werden:

1. `(define f 5)`
2. `(define f (lambda () 5))`
3. `(define f (lambda (n) 5))`
4. `(define f (lambda () (lambda () 5)))`
5. `(define f (lambda () (lambda (n) 5)))`

Aufgabe V.2 (Exponentialfunktion) Für den Fall, daß `m` gerade ist, wird die Prozedur zweimal mit denselben Argumenten aufgerufen. Das ist der Grund wieso diese Formulierung langsam ist.

$$\text{exp}(n,m) = \begin{cases} 1 & \text{falls } m = 0 \\ \text{exp}(n,m/2) * \text{exp}(n,m/2) & \text{falls } m > 0 \text{ und } 2|m \\ n * \text{exp}(n,m-1) & \text{sonst} \end{cases}$$

Der Zeitaufwand der Prozedur ist $O(m)$.

Aufgabe V.3 (Fixpunktberechnung)

```
(define (gut-genug? x fx)
  (if (< (abs (- x fx)) 0.0001)
      #t
      #f))

(define (fixpunkt f x)
  (let ((fx (f x)))
    (if (gut-genug? x fx)
        fx
        (fixpunkt f fx))))
```

Ein Test liefert:

```
(fixpunkt cos 1)
=>.7391155620964187
```

Aufgabe V.4 (Listen)

1. Reihenfolge der Operationen: cdr car cdr cdr car cdr car

   `(car (cdr (car (cdr (cdr (car (cdr l1)))))))`

2. Reihenfolge der Operatioen: car car

   `(car (car l2))`

3. Reihenfolge der Operationen: cdr car cdr car cdr car cdr car cdr car cdr car (6 Paare)

   ```
   (car (cdr (car (cdr (car (cdr (car (cdr (car (cdr (car (cdr l3))))))))))))
   ```

Aufgabe V.5 (Elementauswahl)

```
(define (element xs n)
  (if (null? xs)
      #f
      (if (= n 1)
          (car xs)
          (element (cdr xs) (- n 1)))))
```

Aufgabe V.6 (Liniensegmente)

```
(define (erzeuge-punkt x y)
  (cons x y))
(define (x-koord p)
  (car p))
(define (y-koord p)
  (cdr p))

(define (erzeuge-linie p1 p2)
  (if (and (pair? p1) (pair? p2))
      (cons p1 p2)
      (error "Zugriffsoperationen nicht wohldefiniert")))

(define (start-punkt l)
  (car l))
(define (end-punkt l)
  (cdr l))
(define (mittelpunkt l)
  (let ((x1 (x-koord (start-punkt l)))
        (y1 (y-koord (start-punkt l)))
        (x2 (x-koord (end-punkt l)))
        (y2 (y-koord (end-punkt l))))
     (erzeuge-punkt (/ (+ x1 x2) 2) (/ (+ y1 y2) 2))))
```

Aufgabe V.7 (Definition von `car` und `cdr`)

1. `(define (cdr z) (z (lambda (p q) q)))`

2. Wir werden die folgenden Abkürzungen verwenden:

$$\begin{aligned}
A_1 &:= \texttt{(lambda (m) (m x y))} \\
A_2 &:= \texttt{(lambda (p q) p)} \\
A_3 &:= \texttt{(lambda (x y) } A_1\texttt{)} \\
A_4 &:= \texttt{(lambda (z) (z } A_2\texttt{))} \\
G &:= \{(\texttt{cons}, A_3), (\texttt{car}, A_4)\}
\end{aligned}$$

Wir werten den Ausdruck `(car (cons x y))` in der globalen Umgebung G aus.

Damit erhalten wir folgenden Semantikbaum:

$$
\begin{array}{lll}
\texttt{(car (cons x y))}, \emptyset, G & \longrightarrow & v \\
\hline
\texttt{car}, \emptyset, G & \longrightarrow & A_4\Box\emptyset \\
\texttt{(cons x y)}, \emptyset, G & \longrightarrow & A_1\Box\{(\mathrm{x}, v), (\mathrm{y}, w)\} \\
\hline
\quad \texttt{cons}, \emptyset, G & \longrightarrow & A_3\Box\emptyset \\
\quad \mathrm{x}, \emptyset, G & \longrightarrow & v \\
\quad \mathrm{y}, \emptyset, G & \longrightarrow & w \\
\quad A_1, \{(\mathrm{x}, v), (\mathrm{y}, w)\} & \longrightarrow & A_1\Box\{(\mathrm{x}, v), (\mathrm{y}, w)\} \\
\hline
\texttt{(z } A_2\texttt{)}, \{(\mathrm{z}, A_1\Box\{(\mathrm{x}, v), (\mathrm{y}, w)\})\} & \longrightarrow & v \\
\hline
\mathrm{z}, \{(\mathrm{z}, A_1\Box\{(\mathrm{x}, v), (\mathrm{y}, w)\})\}, G & \longrightarrow & A_1\Box\{(\mathrm{x}, v), (\mathrm{y}, w)\} \\
A_2, \{(\mathrm{z}, A_1\Box\{(\mathrm{x}, v), (\mathrm{y}, w)\})\}, G & \longrightarrow & A_2\Box\{(\mathrm{z}, A_1\Box\{(\mathrm{x}, v), (\mathrm{y}, w)\})\} \\
\texttt{(m x y)}, \underbrace{\{(\mathrm{x}, v), (\mathrm{y}, w), (\mathrm{m}, \cdots)\}}_{:=U} & \longrightarrow & v \\
\hline
\quad \mathrm{m}, U, G & \longrightarrow & A_2\Box\{(z, \cdots)\} \\
\quad \mathrm{x}, U, G & \longrightarrow & v \\
\quad \mathrm{y}, U, G & \longrightarrow & w \\
\quad \mathrm{p}, \{(\mathrm{m}, \cdots), (\mathrm{x}, v), (\mathrm{y}, w)\} & \longrightarrow & v \\
\hline
\end{array}
$$

Der zweite Teil geht analog.

Aufgabe V.8 (Größter gemeinsamer Teiler) Die Prozedur sieht wie folgt aus:

```
(define (ggt-core n m)
  (if (= n 0) m
      (if (> n m)
          (ggt-core (- n m) m)
          (ggt-core (- m n) n))))

(define (ggt n m)
  (if (> m 0)
      (ggt-core n m)
       n))
```

Zuerst zeigen wir das folgende:

$$Teiler(n) \cap Teiler(m) = Teiler(n-m) \cap Teiler(m) \;\; falls \;\; n > m$$

Sei $n > m$.

1. $Teiler(n) \cap Teiler(m) \subseteq Teiler(n-m) \cap Teiler(m)$

   Sei $d \in Teiler(n) \cap Teiler(m)$ dann

   $$d \mid n \Leftrightarrow \exists q_1 \in N : n = dq_1$$

   und

   $$d \mid m \Leftrightarrow \exists q_2 \in N : n = dq_2$$

   Es folgt:

   $\exists q_3 \in N$ nämlich $q_3 = (q_1 - q_2)$, so daß $n - m = dq_3$

2. $Teiler(n) \cap Teiler(m) \supseteq Teiler(n-m) \cap Teiler(m)$

   Mit $d|(n-m)$ und $d|m$ folgt $d|n$ da $n = (q_1 - q_2)d$ wobei $(n-m) = q_1 d$ und $m = q_2 d$.

   Aus 1. und 2. folgt die Behauptung.

Damit haben wir, daß die Prozedur in jedem Schritt die Invariante $Teiler(n) \cap Teiler(m) = Teiler(n_i) \cap Teiler(m_i)$ einhält, wobei die m_i, n_i die Werte der Variablen n,m in der i-ten Instanz der Funktion `ggt-core` sind. Dies lässt sich sehr einfach durch Induktion zeigen. Hierbei wird das oben gezeigte beim Induktionsschritt angewendet.
Weiterhin terminiert die Prozedur, da $(n_i + m_i)$ für wachsendes i streng monoton fallend ist.
Ist die Abruchbedingung für $i = k$ erreicht, so wird $m_k = ggt(n, m)$ ausgegeben, womit die Prozedur das gewünschte Eingabe-/Ausgabeverhalten hat.
Der Aufwand der Prozedur ist $O(n)$ (für $n > m$).

Zusatzaufgabe V.9 (Rekursion) Der Ausdruck lautet:

```
(define Y
  (lambda (S)
    (let
        ((H (lambda (G) (lambda (n) ((S (G G)) n)))))
      (H H))))
```

Damit läßt sich zum Beispiel die Fakultätsfunktion berechnen:

```
(define fak
  (Y (lambda (fak) (lambda (n) (if (< n 1) 1 (* n (fak (- n 1))))))))
```