Christian-Albrechts-Universität zu Kiel
Institut für Informatik
Lehrstuhl für Programmiersprachen und Übersetzerkonstruktion
Prof. Dr. M. Hanus, F. Reck

Institut für Informatik

3. Übung zur Vorlesung „Informatik I“

Abgabe: Freitag, 14. November, 14:00 Uhr

Bitte geben Sie Ihre Lösungen sowohl schriftlich im Schrein der Informatik, als auch über iLean ab!

Präsenzaufgabe 1

Es seien die folgenden Terme zu der in der Vorlesung vorgestellten Signatur gegeben:

- $t_1 =$ `(+ (* 3 (+ x 1)) (- (* x 2) 1))` und
- $t_2 =$ `(+ 2 (* x (- (* y 3) (+ (/ (* 4 z) 2) 1))))`

(a) Es sei $\sigma(\texttt{x}) =$ `(* x 2)`, $\sigma(\texttt{y}) =$ `(- (* 3 z) 5)` und $\sigma(\texttt{z}) =$ `3`. Führen Sie die Substitutionen $t_1\sigma$ und $t_2\sigma$ durch.

(b) Führen Sie die Termersetzungen

- $t_1[\texttt{(* y 2)}]_{\langle 1,2,1\rangle}$
- $t_1[\texttt{(* z 5)}]_{\langle 2\rangle}$
- $t_2[\texttt{5}]_{\langle 2,2,1,2\rangle}$
- $t_2[\texttt{(- y 2)}]_{\langle 1\rangle}$
- $t_2[\texttt{7}]_{\langle 2,2,2\rangle}$

durch.

Präsenzaufgabe 2

Seien die Funktionen `add1` und `add2` definiert durch:

```
(define (add1 n m)
  (if  (= n 0)
       m
       (inc (add1 (dec n) m))))

(define (add2 n m)
  (if  (= n 0)
       m
       (add2 (dec n) (inc m))))
```

Die beiden Funktionen addieren jeweils zwei natürliche Zahlen. Sie basieren beide auf der elementaren Inkrementierung `inc` und Dekrementierung `dec` definiert durch:

```
(define (inc n) (+ n 1))

(define (dec n) (- n 1))
```

(a) Erläutern Sie mit Hilfe des Substitutionsmodells die Berechnungsprozesse, die bei der Auswertung von `(add1 2 3)` und `(add2 2 3)` in Scheme erzeugt werden.

(b) Sind diese Prozesse rekursiv oder iterativ? In welchem Zusammenhang steht die Struktur des rekursiven Aufrufes mit der Art des Berechnungsprozesses?

Aufgabe 3 4 Punkte

Die aus Serie 2, Präsenzaufgabe 1 bekannten Funktionen `newif` und `newfak` sind definiert durch

```
(define (newif b x y)
  (cond (b x)
        (else y)))

(define (newfak x)
  (newif (= x 0)
         1
         (* x (newfak (- x 1)))))
```

Werten Sie den Term `(newfak 1)` in Normalordnung aus.

Aufgabe 4 8 Punkte

Sei die Funktion `f` definiert durch

```
(define (f x y)
   (if  (= x 0)
        (f y (f x y))
        x))
```

Werten Sie den Term `(f 0 1)` sowohl in applikativer als auch in Normalordnung aus. Beachten Sie, dass die Sonderform `(if p a1 a2)` immer erst `p` auswertet und dann zu `a1` bzw. `a2` reduziert (sowohl in applikativer als auch in Normalordnung).

Aufgabe 5 (Programmieraufgabe) 8 Punkte

In der Vorlesung haben Sie die folgende, rekursive Funktion zur schnellen Berechnung von Potenzen kennengelernt:

```
(define (schnell-pot b n)
  (cond ((= n 0)      1)
        ((gerade? n)  (quadrat (schnell-pot b (/ n 2))))
        (else         (* b (schnell-pot b (- n 1))))))


(define (gerade? n)
  (= (remainder n 2) 0))
```

Schreiben Sie eine Funktion `(schnell-pot-iter b n)`, die das schnelle Potenzieren in einem iterativen Prozess durchführt.