# Programmierung

## 07. Übungsblatt

Zeitraum: 08. – 12. Juni 2015

**Übung 1 (AGS 12.4.29)**

(a) Geben Sie einen Kombinator $A$ an, so dass $A\ t\ s\ u \Rightarrow_\beta^* s$ für alle Lambdaterme $t$, $s$ und $u$.
(b) Geben Sie einen Kombinator $B$ an, so dass $B\ t\ s \Rightarrow_\beta^* s\ t$ für alle Lambdaterme $t$ und $s$.
(c) Geben Sie einen Kombinator $C$ an, so dass $C\ C \Rightarrow_\beta C\ C$.
(d) Geben Sie einen Kombinator $D$ an, so dass $D \Rightarrow_\beta D$.
(e) Geben Sie einen Kombinator $E$ an, so dass $E\ E\ t \Rightarrow_\beta^* E\ t\ E$ für jeden Lambdaterm $t$.

**Übung 2 (AGS 12.4.31)**

(a) Berechnen Sie die Normalform des untenstehenden $\lambda$-Terms, indem Sie ihn *schrittweise* reduzieren. Geben Sie dabei vor jedem Schritt für die relevanten Teilausdrücke die Mengen der gebunden bzw. frei vorkommenden Variablen an.

$$(\lambda f\,x.f\,x)(\lambda z\,y.x\,y)x$$

(b) Gegeben sei die folgende Haskell-Funktion:

```
g :: Int -> Int -> Int
g a 0         = a
g a b
  | b == 1    = g (a + 1) (b - 1)
  | otherwise = g (a + 2) (b - 2)
```

Geben Sie einen $\lambda$-Term $\langle G\rangle$ an, so dass $\texttt{g} = \langle Y\rangle\langle G\rangle$ gilt.

(c) Gegeben sei der $\lambda$-Term

$$\langle F\rangle = \Big(\lambda f\,x\,y\,z.\langle\text{ite}\rangle\big(\langle\text{iszero}\rangle(\langle\text{pred}\rangle y)\big)\\ (\langle\text{mult}\rangle x z)\\ \big(\langle\text{add}\rangle\langle 1\rangle\big(f\,(\langle\text{add}\rangle x z)(\langle\text{pred}\rangle y)(\langle\text{sub}\rangle x z)\big)\big)\Big).$$

Berechnen Sie schrittweise die Normalform des Terms $\langle Y\rangle\langle F\rangle\langle 2\rangle\langle 2\rangle\langle 2\rangle$. Führen Sie im Rechenprozess zweckmäßige Abkürzungen der $\lambda$-Terme ein.

**Übung 3 (AGS 12.4.27)**

(a) Berechnen Sie die Normalform des untenstehenden $\lambda$-Terms, indem Sie ihn *schrittweise* reduzieren. Geben Sie dabei vor jedem Schritt für die relevanten Teilausdrücke die Mengen der gebunden bzw. frei vorkommenden Variablen an.

$$(\lambda xyz.\, y\, z\, x)(\lambda x.\, x\, y)(\lambda x.\, x)$$

(b) Gegeben sei die folgende Haskell-Funktion:

```
g :: Int -> Int -> Int -> Int
g 0 _ _ = 0
g n x y = (g (n - 1) x y) + (if n `mod` 2 == 0 then x else y)
```

Geben Sie einen $\lambda$-Term $\langle G\rangle$ an, so dass $\mathtt{g} = \langle Y\rangle\langle G\rangle$ gilt.

(c) Gegeben sei der $\lambda$-Term

$$\begin{aligned}\langle F\rangle = \lambda f\, n x y.\, \langle ite\rangle & \big(\langle iszero\rangle\, (\langle pred\rangle\, n)\big)\\ & (\langle add\rangle\, x\, y)\\ & \big(f\, (\langle pred\rangle\, n)(\langle mult\rangle\, x\, n)(\langle add\rangle\, y\, n)\big).\end{aligned}$$

Berechnen Sie schrittweise die Normalform des Terms $\langle Y\rangle\langle F\rangle\langle 2\rangle\langle 3\rangle\langle 5\rangle$. Führen Sie im Rechenprozess zweckmäßige Abkürzungen der $\lambda$-Terme ein.

**Zusatzaufgabe 1 (AGS 12.3.19)**

Folgende Definitionen seien gegeben:

```
data Tree a = Node a (Tree a) (Tree a) | Leaf a

mirror :: Tree a -> Tree a
mirror (Node x t1 t2) = Node x (mirror t2) (mirror t1)
mirror (Leaf x) = Leaf x

yield :: Tree a -> [a]
yield (Node _ t1 t2) = yield t1 ++ yield t2
yield (Leaf x) = [x]
```

Die folgende Aussage soll mittels struktureller Induktion über Bäumen bewiesen werden:

(A): Für jeden Typ `a` und jeden Baum `t :: Tree a` gilt

$$\texttt{reverse (yield t)} = \texttt{yield (mirror t)}\,.$$

Nutzen Sie folgende Eigenschaften: Für alle Typen `a`, Werte `x :: a` und Listen `xs, ys :: [a]` gilt

(E1): `reverse [x] = [x]`, und

(E2): `reverse (xs ++ ys) = reverse ys ++ reverse xs`.

Bearbeiten Sie die folgenden Teilaufgaben; geben Sie bei jeder Umformung die benutzte *Definition*, *Eigenschaft* bzw. *Induktionsvoraussetzung* an; quantifizieren Sie alle Variablen.

(a) Zeigen Sie den Induktionsanfang.

(b) Geben Sie die Induktionsvoraussetzung *vollständig* an.

(c) Zeigen Sie den Induktionsschritt.

**Zusatzaufgabe 2 (AGS 12.4.11 ⋆)**

(a) Gegeben sei folgender $\lambda$–Term:

$$(\lambda y.(\lambda x.x\ y))\,(\lambda y.x\ (\lambda x.(\lambda y.y)\ y))$$

Reduzieren Sie diesen Term bis seine Normalform erreicht ist. Schreiben Sie – bevor Sie einen Ableitungsschritt ausführen – für die relevanten (Teil-)Ausdrücke die Mengen der freien bzw. der gebundenen Vorkommen von Variablen auf.

(b) Eine Funktion $f : \mathbb{N} \times \mathbb{N}^+ \to \mathbb{N}^+$ sei wie folgt definiert:

$$\begin{aligned} f(x,y) &= 1 && \text{für } x = 0 \\ f(x,y) &= y && \text{für } x = 1 \\ f(x,y) &= f(x-1,y+1) \cdot f(x-2,y+2) && \text{für } x \geq 2 \end{aligned}$$

Geben Sie zur Funktion $f$ den zugehörigen $\lambda$-Term $\langle F \rangle$ an, so dass $\langle f \rangle = \langle Y \rangle \langle F \rangle$ gilt.
Sie dürfen ausschließlich die $\lambda$-Terme $\langle succ \rangle$, $\langle pred \rangle$, $\langle iszero \rangle$, $\langle ite \rangle$, $\langle mult \rangle$ und $\langle 0 \rangle$ als bekannt voraussetzen.