

Funktionale Programmierung

9. Übungsblatt

Prof. Dr. Margarita Esponda

1. Aufgabe (2 Punkte)

Wie kann eine Konstanten-Funktion, die äquivalent zur Haskell-Funktion **const** ist, als **Lambda**-Ausdruck definiert werden?

Anwendungsbeispiele in Haskell:

```
const 3 7 => 3
const "abe" 7 => "abe"
```

2. Aufgabe (6 Punkte)

Schreiben Sie folgende rekursive Funktion als **Lambda**-Ausdruck um, und testen Sie die Funktion mit dem Argument 4 (d.h. äquivalent zu $g\ 4$).

```
g 0 = 1
g n = 1 + (g (n-1))*3
```

3. Aufgabe (8 Punkte)

Wir können im Lambda-Kalkül sowie in unserem algebraischen Datentyp **ZInt**, positive und negative Zahlen mit Hilfe von Zahlenpaaren (aus zwei natürlichen Zahlen), darstellen. Die Lambda-Abstraktion **$\lambda z.z\ a\ b$** stellt dann das Tupel **(a, b)** dar, mit **a** und **b** natürliche Zahlen, und kann als die Zahl **b - a** interpretiert werden.

Beispiele:

```
 $\lambda z.z\ 0\ 0$  stellt das Tupel (0,0) dar, und wird als die Zahl 0 interpretiert.
 $\lambda z.z\ 0\ 1$  ... (0, 1) ... 1
 $\lambda z.z\ 1\ 0$  ... (1, 0) ... -1
 $\lambda z.z\ 7\ 2$  ... (7, 2) ... -5
```

Die Summe von zwei ganzen Zahlen kann wie folgt berechnet werden:

$$(a, b) + (c, d) \Rightarrow ((a+c), (b+d))$$

Beispiel: $(3, 0) + (0, 3) \Rightarrow (3, 3)$, was die ganze Zahl 0 darstellt.

Ein Lambda-Ausdruck, der zwei ganze Zahlen zusammenaddiert, kann wie folgt definiert werden:

$$\lambda x\ y. (\lambda z. z\ (A\ (xT)\ (yT))\ (A\ (xF)\ (yF)))$$

mit **A** gleich die Summe zweier natürlicher Zahlen

a) Definieren Sie eine Lambda-Abstraktion, die bei Eingabe der Zahl **x** die Zahl **-x** zurückgibt.

b) Definieren Sie die Subtraktions-Funktion.

Die Darstellung von positiven und negativen Zahlen mit Hilfe von Tupeln führt zu einer nicht eindeutigen Darstellung der Zahlen.

Beispiel: Die negative Zahl -3 kann als (8, 5) oder (3, 0) dargestellt werden.

- c) Definieren Sie einen Lambda-Ausdruck, der eine beliebige Zahl (a,b) in ein Tupel der Form (n, 0) (negative Zahl) oder (0, m) (positive Zahl) umwandelt.

Beispiel: $\lambda z.z\ 3\ 5 \Rightarrow \lambda z.z\ 0\ 2$
 $\lambda z.z\ 7\ 2 \Rightarrow \lambda z.z\ 5\ 0$

4. Aufgabe (6 Punkte)

Definieren Sie für positive und negative Zahlen **Lambda**-Ausdrücke, die die Vergleichsoperationen ($>$) und (\neq) berechnen.

5. Aufgabe (4 Punkte)

Definieren Sie einen Lambda-Ausdruck, der die Länge von zwei Listen vergleicht (\leq) und die Werte 1, 0 oder -1 zurückgibt, je nachdem, ob die erste Liste kleiner, gleich oder größer ist.

6. Aufgabe (6 Punkte)

- a) Vervollständigen Sie die in der Vorlesung definierten Lambda-Ausdrücke für Listen mit einer Funktion, die ein Element in einer Liste sucht und entsprechende Wahrheitswerte zurück gibt.
- b) Definieren Sie eine Lambda-Funktion, die ein Element aus einer Liste löscht bzw. der Semantik folgender Haskell-Funktion entspricht:

```
remove y [] = []  
remove y (x:xs) | y==x    = xs  
                | otherwise = x: remove y xs
```

Wichtiger Hinweis:

Die Lösungen sollen elektronisch (nur Whiteboard-Upload) abgegeben werden. Es ist **keine verspätete Abgabe per Email erlaubt**.