PHILIPPS-UNIVERSITÄT MARBURG
Fachbereich Mathematik und Informatik
Prof. Dr. R. Loogen
J.Berthold

D-35032 Marburg
Hans Meerwein Straße
Lahnberge

23.November 2006

6. Übung zu „Grundlagen der funktionalen Programmierung“,

Abgabe: 30.November 2006 vor der Vorlesung **WS06/07**

Aufgaben

6.1 Inferieren Sie, falls möglich, (syntaxgesteuert) Typen für die folgenden Ausdrücke **4 Punkte**

(a) $fix\, f.\lambda x\, y.if\, (\underline{=}\, y\, \underline{0})\ \underline{0}\ (\underline{+}\ (f\, \underline{true}\, \underline{0})\ (f\, \underline{1}\, \underline{0}))$

(b) $let\ f\ =\ \lambda x\, y.if\, (\underline{=}\, y\, \underline{0})\ \underline{0}\ (\underline{+}\ (f\, \underline{true}\, \underline{0})\ (f\, \underline{1}\, \underline{0}))\ in\ f$

Bearbeiten Sie zunächst den inneren Teil, der in beiden Ausdrücken gleich ist.

6.2 **Bestimmung des allgemeinsten Unifikators** **3 Punkte**

Überführen Sie die folgenden Typgleichungen mit Hilfe der Inferenzregeln für Typgleichungen jeweils in gelöste Form (d.h. bestimmen Sie den allgemeinsten Unifikator) bzw. begründen Sie, dass diese nicht existiert.

(a) $\alpha_1 \to (\alpha_1 \to \alpha_2) = \beta_1 \to (\beta_2 \to \beta_2)$

(b) $\alpha_1 \to (\alpha_1 \to \alpha_2) = (\gamma_1 \to \gamma_2) \to \gamma_2$

(c) $\beta_1 \to (\beta_2 \to \beta_2) = (\gamma_1 \to \gamma_2) \to \gamma_2$

6.3 **Bestimmung eines allgemeinsten Typs** **5 Punkte**

Berechnen Sie mit dem in der Vorlesung vorgestellten Typrekonstruktionsalgorithmus $\mathcal{TR}$ den allgemeinsten Typ von

$$length = \text{fix}\ l.\lambda x.if\ \ (\underline{null}\ x)\ \underline{0}\ (\underline{succ}\ (l\ (\underline{tail}\ x)))$$

Zu den Aufgaben 1 und 3 sei eine Typannahme A_0 vorgegeben, welche die Typen der verwendeten Konstanten enthält:

Bool und Int seien Basistypen und $List :: Types \to Types$ ein einstelliger Typkonstruktor, sowie

$$\begin{array}{rcl}
A_0(if) & = & \forall\alpha.\underline{Bool} \to \alpha \to \alpha \to \alpha \\
A_0(\underline{true}) & = & \underline{Bool} \\
A_0(\underline{0}) = A_0(\underline{1}) & = & \underline{Int} \\
A_0(\underline{succ}) & = & \underline{Int} \to \underline{Int} \\
A_0(\underline{+}) & = & \underline{Int} \to \underline{Int} \to \underline{Int} \\
A_0(\underline{=}) & = & \underline{Int} \to \underline{Int} \to \underline{Bool} \\
A_0(\underline{null}) & = & \forall\alpha.List\ \alpha \to \underline{Bool} \\
A_o(\underline{tail}) & = & \forall\alpha.List\ \alpha \to List\ \alpha
\end{array}$$