PHILIPPS-UNIVERSITÄT MARBURG
Fachbereich Mathematik und Informatik
Prof. Dr. R. Loogen
M. Dieterle

D-35032 Marburg
Hans Meerwein Straße

12.Dezember 2007

9. Übung zu „Grundlagen des Compilerbaus", WS 2007/08

Abgabe der Aufgaben: Mi, 19.Dezember 2007 (vor der Vorlesung)

9.1 Attributgrammatik für Zuweisungen

5 Punkte

Die folgende Grammatik beschreibt eine Zuweisung an einen Bezeichner *id* von booleschem oder arithmetischem Typ. Dabei bezeichne *bval* einen Wahrheitswert und *numval* eine Zahl.

$$\begin{array}{rclr} G_z : Z & \rightarrow & id \texttt{ := } E & (1) \\ E & \rightarrow & bval & (2) \\ E & \rightarrow & numval & (3) \end{array} \qquad \begin{array}{rclr} E & \rightarrow & E + E & (4) \\ E & \rightarrow & E \,||\, E & (5) \\ E & \rightarrow & E \,\&\&\, E & (6) \\ E & \rightarrow & E = E & (7) \end{array}$$

(a) Nur Zahlen dürfen mit $+$, nur Wahrheitswerte mit && und || verknüpft werden. Der Vergleich $=$ ist nur für Argumente *gleichen* Typs definiert. Versehen Sie das Nonterminal E mit Attributen $t \in \{\text{Bool}, \text{Num}\}$ für den Typ des Ausdrucks und $err \in \{\text{true}, \text{false}\}$ als Typfehlerindikator.

(b) Mit den nebenstehenden Produktionen erzeugt eine erweiterte Grammatik G'_z eine ganze Liste von Zuweisungen. Auch Bezeichner können nun auf der rechten Seite verwendet werden.

$$\begin{array}{rclr} G'_z : Z & \rightarrow & id \texttt{ := } E \,;\, Z & (8) \\ E & \rightarrow & id & (9) \end{array}$$

Erweitern Sie die Attributierung, so dass die gesamte Liste von Zuweisungen überprüft wird. Bezeichner dürfen nur auf der rechten Seite enthalten sein, falls ihnen bereits vorher ein Wert zugewiesen wurde. Die Verwendung muss typkonform erfolgen, wobei durch eine erneute Zuweisung der Typ nicht verändert werden darf.

Bsp1: `a := True; a' := 3; a := a'=3;` mit $err.k_0 = false$
Bsp2: `a := True; a := a=3;` mit $err.k_0 = true$
Bsp3: `a := 3; a := a=3;` mit $err.k_0 = true$
(Dabei bezeichne k_0 den Wurzelknoten des Syntaxbaums)

Speichern Sie die Typinformationen in einer Hashtabelle mit den nebenstehenden Operationen. Verwenden Sie eine (leere) initiale Tabelle als inherites Attribut des Startsymbols Z (Parameter der Attributierung).

```
data Table = .. -- Hashtable
insert :: Table -> (key,value) -> Table
delete :: Table -> key -> Table
lookup :: Table -> key -> Maybe value
```

Bitte wenden!

9.2 Zwischencode für ein PSA-Programm

4 Punkte

(a) Ermitteln Sie die Übersetzung in Maschinencode ($trans(P)$) für das nebenstehende PSA-Programm P.

(b) Berechnen Sie die Konfigurationsfolge der abstrakten Maschine, die bei Ausführung des resultierenden MA-Programms und dem Startzustand $(1, \varepsilon, h_D)$ mit $h_D(i) = 0 \ \forall_{i \in \mathbb{N}}$ durchlaufen wird.

```
const X = 4;
var Y;
Y := 1;
while (Y * Y) < X do
    Y := Y + 1
```

9.3 Codelängenberechnung

3 Punkte

Definieren Sie passend zu den Übersetzungsfunktionen der Vorlesung eine Funktion zur Bestimmung der Codelänge:

$$cl : \mathit{AExp} \cup \mathit{BExp} \cup \mathit{Cmd} \rightarrow \mathbb{N}$$

Erläutern Sie, wo und wie diese Funktion für die Übersetzung von PSA-Programmen in MA-Code zu verwenden ist.