P. Thiemann, M. Neubauer Wintersemester 2004/2005

Compilerbau

http://proglang.informatik.uni-freiburg.de/teaching/compilerbau/2004/

Übungsblatt 6

Abgabe: 8.12.2004

Aufgabe 1 (Typüberprüfung mit Hilfe von Attributgrammatiken):

Die folgende Attributgrammatik dient zur Typberechnung einfacher arithmetischer Ausdrücke mit Operatoren `+`, `-`, `*`, `/` und `div` und Konstanten vom Typ *int* und *float*. Zur Vereinfachung sind alle Integer- bzw. Float-Konstanten durch das Terminal `i` bzw. `f` repräsentiert. Die Werte für das Attribut *typ* sind $V = \{int, float\}$.

$$\begin{aligned}
&\langle \circ E \to E\ Aop\ T\rangle\,.\,\mathrm{typ} = \\
&\quad f_{\langle \circ E \to E\ Aop\ T\rangle}(\langle E \to \circ E\ Aop\ T\rangle\,.\,\mathrm{typ}, \langle E \to E\ Aop\ \circ T\rangle\,.\,\mathrm{typ}) \quad \text{s.u.} \\
&\langle \circ T \to T\ Mop\ F\rangle\,.\,\mathrm{typ} = \\
&\quad f_{\langle \circ T \to T\ Mop\ F\rangle}(\langle T \to T\ \circ Mop\ F\rangle\,.\,\mathrm{op}, \langle T \to \circ T\ Mop\ F\rangle\,.\,\mathrm{typ}, \langle T \to T\ Mop\ \circ F\rangle\,.\,\mathrm{typ}) \quad \text{s.u.}
\end{aligned}$$

$$\begin{array}{ll}
\langle \circ E \to T\rangle\,.\,\mathrm{typ} = f_{\langle \circ E \to T\rangle}(\langle E \to \circ T\rangle\,.\,\mathrm{typ}) & f_{\langle \circ E \to T\rangle} = \mathrm{id} \\
\langle \circ T \to F\rangle\,.\,\mathrm{typ} = f_{\langle \circ T \to F\rangle}(\langle T \to \circ F\rangle\,.\,\mathrm{typ}) & f_{\langle \circ T \to F\rangle} = \mathrm{id} \\
\langle \circ P \to \mathtt{i}\rangle\,.\,\mathrm{typ} = f_{\langle \circ P \to \mathtt{i}\rangle} & f_{\langle \circ P \to \mathtt{i}\rangle} = int \\
\langle \circ P \to \mathtt{f}\rangle\,.\,\mathrm{typ} = f_{\langle \circ P \to \mathtt{f}\rangle} & f_{\langle \circ P \to \mathtt{f}\rangle} = float \\
\langle \circ P \to \mathtt{(}\ E\ \mathtt{)}\rangle\,.\,\mathrm{typ} = f_{\langle \circ P \to \mathtt{(}\ E\ \mathtt{)}\rangle}(\langle P \to \mathtt{(}\ \circ E\ \mathtt{)}\rangle\,.\,\mathrm{typ}) & f_{\langle \circ P \to \mathtt{(}E\mathtt{)}\rangle} = \mathrm{id} \\
\langle \circ F \to P\rangle\,.\,\mathrm{typ} = f_{\langle \circ F \to P\rangle}(\langle F \to \circ P\rangle\,.\,\mathrm{typ}) & f_{\langle \circ F \to P\rangle} = \mathrm{id} \\
\langle \circ Aop \to \mathtt{+}\rangle\,.\,\mathrm{op} = f_{\langle \circ Aop \to \mathtt{+}\rangle} & f_{\langle \circ Aop \to \mathtt{+}\rangle} = \mathtt{+} \\
\langle \circ Aop \to \mathtt{-}\rangle\,.\,\mathrm{op} = f_{\langle \circ Aop \to \mathtt{-}\rangle} & f_{\langle \circ Aop \to \mathtt{-}\rangle} = \mathtt{-} \\
\langle \circ Mop \to \mathtt{*}\rangle\,.\,\mathrm{op} = f_{\langle \circ Mop \to \mathtt{*}\rangle} & f_{\langle \circ Mop \to \mathtt{*}\rangle} = \mathtt{*} \\
\langle \circ Mop \to \mathtt{/}\rangle\,.\,\mathrm{op} = f_{\langle \circ Mop \to \mathtt{/}\rangle} & f_{\langle \circ Mop \to \mathtt{/}\rangle} = \mathtt{/} \\
\langle \circ Mop \to \mathtt{div}\rangle\,.\,\mathrm{op} = f_{\langle \circ Mop \to \mathtt{div}\rangle} & f_{\langle \circ Mop \to \mathtt{div}\rangle} = \mathtt{div}
\end{array}$$

$$f_{\langle \circ E \to E\ Aop\ T\rangle}(a_1, a_2) = \begin{cases} int & \text{wenn } a_1 = a_2 = int \\ float & \text{sonst} \end{cases}$$

$$f_{\langle \circ T \to T\ Mop\ F\rangle}(a_1, a_2, a_3) = \begin{cases} int & \text{wenn } a_1 = \mathtt{*} \wedge a_2 = a_3 = int \\ int & \text{wenn } a_1 = \mathtt{div} \wedge a_2 = a_3 = int \\ float & \text{sonst} \end{cases}$$

Gib Typen für alle Teilausdrücke des arithemtischen Ausdrücke

$$42 \mathtt{*} (3 \mathtt{/} (12\, \mathtt{div}\, 5)) \mathtt{+} 3$$

an, indem Du die Attributdekoration des Ableitungsbaumes ermittelst.

Aufgabe 2 (Attributgrammatik zur Textformatierung):

Durch die folgende Attributgrammatik wird eine Layoutregel für eine einfache Textformattierungssprache definiert. Dies geschieht, indem zu jeder textlichen Einheit (Buchstabe,...) das kleinste Rechteck bestimmt wird, in die sie hinein paßt. Eine Hoch- bzw. Tiefstellung relativ zu einem Rechteck (Bsp: $\boxed{\text{M}}^{\boxed{\text{x}}}$) soll geschehen, indem die Grundlinie (Oberkante) von $\boxed{\text{x}}$ bei 1/3 (2/3) der Höhe des Rechtecks $\boxed{\text{M}}$ liegt. Ist zu einem Rechteck ein Subskript und ein Superskript vorhanden, so werden deren Rechtecke an der halben Höhe des Bezugsrechtecks ausgerichtet. Dabei wird ferner der Inhalt des hoch- bzw. tiefgestellten Rechtecks entsprechend der Schachtelungstiefe der Hoch- bzw. Tiefstellung verkleinert. Die Bestimmung dieser Rechtecke leistet die folgende Attributierung.

Es sei **Att** = **Syn** ∪ **Inh** mit **Syn** = $\{w, h\}$ und **Inh** = $\{l\}$ (w Breite, h Höhe, l Level — Schachtelungstiefe). $\mathcal{G}$ hat die folgenden Regeln und Attributgleichungen.

$$\begin{array}{ll}
S \leftarrow E & E.l = 1 \\[4pt]
E \leftarrow c & E.w = d(E.l, w_0) \\
 & E.h = d(E.l, h_0) \\[4pt]
E \leftarrow EE & E.w = E_1.w + E_2.w \\
 & E.h = \max\{E_1.h, E_2.h\} \\
 & E_1.l = E.l \\
 & E_2.l = E.l \\[4pt]
E \leftarrow \{E\} & E.w = E_1.w \\
 & E.h = E_1.h \\
 & E_1.l = E.l
\end{array}$$

$$\begin{array}{ll}
E \leftarrow E \ \mathbf{sup}\ E & E.w = E_1.w + E_2.w \\
 & E.h = \frac{1}{3}E_1.h + E_2.h \\
 & E_1.l = E.l \\
 & E_2.l = E.l + 1 \\[4pt]
E \leftarrow E \ \mathbf{sub}\ E & E.w = E_1.w + E_2.w \\
 & E.h = \frac{1}{3}E_1.h + E_2.h \\
 & E_1.l = E.l \\
 & E_2.l = E.l + 1 \\[4pt]
E \leftarrow E \ \mathbf{sub}\ E \ \mathbf{sup}\ E & E.w = E_1.w + \max\{E_2.w, E_3.w\} \\
 & E.h = E_2.h + E_3.h \\
 & E_1.l = E.l \\
 & E_2.l = E.l + 1 \\
 & E_3.l = E.l + 1
\end{array}$$

Die Interpretation der Attributsorten h und w sind die rationalen Zahlen; von l die natürlichen Zahlen. Die Funktion $d(n, x)$ verkleinert x um einen von n abhängigen Faktor, z.B. $d(n, x) = x/n$ oder $d(n, x) = x\sqrt{2}^{-n}$.

(i) Verbesse die Gleichungen zum Attribut h.

(ii) Führe zwei weitere Attribute x und y ein, deren Werte die Position der linken unteren Ecke des zugehörigen Rechtecks bestimmen. Der Nullpunkt liegt beim Startsymbol, d.h. die Attributgleichungen ändern sich zu

$$S \leftarrow E \quad E.l = 1 \quad E.x = 0 \quad E.y = 0$$

Füge bei restlichen Produktionen die entsprechenden Gleichungen hinzu.

(iii) In welcher Reihenfolge können die Attribute berechnet werden, um einen vollständige Attributierung zu erhalten?

Aufgabe 3 (Parser für wohlgeformte XML-Dokumente):

Implementiere mit Hilfe eines LR-Parsergenerators einen XML-DOM-Parser[1], der nicht-wohlgeformte XML-Dokumente zurückweisst.

Benutze als Lexer den zu Blatt 2, Aufgabe von Dir implementierten SAX-Parser. Teste die Wohlgeformheitskriterien, die im XML-Standard definiert sind[2], mit Hilfe passender Attributionsfunktionen. Falls Dir keine DOM-Bibliothek zur Verfügung steht, genügt es zu signalisieren, ob das Eingabedokument wohlgeformt ist oder nicht.

[1] `http://www.w3.org/DOM/`
[2] `http://www.w3.org/TR/REC-xml`