Prof. Dr. Arnd Poetzsch-Heffter
Dr.-Ing. Jan Schäfer

Technische Universität Kaiserslautern
Fachbereich Informatik
AG Softwaretechnik

Übungsblatt 7: Übersetzer und sprachverarbeitende Werkzeuge (SS 2011)

Hand Out: 1. Juni 2011
Hand In: 8. Juni 2011

Aufgabe 1 Anwendung von Typregeln

Zeigen Sie mit den Typregeln der Vorlesung (Folien 86–88), ob die folgenden Ausdrücke typkorrekt sind oder nicht. Bauen Sie dazu einen Ableitungsbaum auf.

a) $\Sigma \vDash \text{Assign}(y, \text{Relation}(x, \text{RelOp}(\text{Plus}()), \text{LongConst}(5))) : \text{boolean}$,
mit $\Sigma = \{y \rightarrow \text{boolean}, x \rightarrow \text{ClassType}(C)\}$

b) $\Sigma \vDash \text{Assign}(y, \text{Invoc}(y, m, y)) : \text{InterfaceType}(J)$,
mit $\Sigma = \begin{array}{l}\{\ y \rightarrow \text{InterfaceType}(I), \text{InterfaceType}(J) \leq \text{InterfaceType}(I), \\ \ \ (I, m) \rightarrow (\text{InterfaceType}(I), \text{InterfaceType}(J))\ \}\end{array}$

Aufgabe 2 Typanalyse von Ausdrücken

Für die Typanalyse implementieren Sie ein Attribut `typeOf`. Es ist auf beliebigen `Expression` definiert und liefert deren `Type` zurück. Verwenden Sie `resolveName`, um die Typen von Variablen, statischen Klassenattributen und statischen Methodenaufrufen herauszufinden.

Aufgabe 3 Kontextbedingungen

Um die Überprüfung von MiniJava-Programmen zu vervollständigen müssen Sie noch folgende Kontextbedingungen überprüfen.

- Jede `CompilationUnit` muss genau eine Klasse `Main` mit einer statischen Methode **`static void`** `main()` enthalten.
- Jede Methode und jedes Attribut ist **`static`**.
- Es gibt drei Typen von Scopes in MiniJava-Programmen: Die `CompilationUnit`, jede Klasse und jede Methode bilden einen Scope. Pro Scope darf jeder Bezeichner nur einmal deklariert werden.
- Jeder Methodenaufruf hat die korrekte Anzahl an Parametern und jeder aktuelle Parameter hat einen, zum formalen Parameter, kompatiblen Typ.
- Jede nicht-**`void`** Methode hat mindestens ein **`return`**. Der Typ des Ausdrucks hinter dem **`return`** ist kompatibel zum Rückgabetyp der Methode.
- Jede **`void`** Methode hat nur **`return`**-Anweisungen ohne Ausdruck.
- Nach einem **`return`** folgt keine weitere Anweisung im selben Block.
- Die Bedingungen von **`if`** und **`while`** haben den Typ **`boolean`**.
- Die linke und die rechte Seite einer Anweisung haben kompatible Typen.
- Jeder `Expression` hat einen Typ.

Dies alles soll im Attribut `contextCorrect`, welches eine `CompilationUnit` nimmt und ein **`boolean`** liefert, überprüft werden.

Zur Realisierung definieren Sie sich weitere Attribute und evtl. weitere Katjasorten. Dokumentieren Sie kurz jedes Attribut und seine Abhängigkeiten in einer separaten Datei.