

	<h1>Universität Karlsruhe (TH)</h1> <h2>Lehrstuhl für Programmierparadigmen</h2>	
	Compilerpraktikum SS 2010 Dozent: Prof. Dr.-Ing. G. Snelting Betreuer: Matthias Braun Betreuer: Sebastian Buchwald	http://pp.info.uni-karlsruhe.de/ snelting@ipd.info.uni-karlsruhe.de matthias.braun@kit.edu sebastian.buchwald@kit.edu
Übungsblatt 6	Ausgabe: 19.05.2010	Besprechung: 02.06.2010

Aufgabe für die *nächsten 2* Wochen ist die Transformation des AST in die Firm Zwischensprache. Für ein kleineres komplettes Beispiel zur JFirm-Benutzung siehe auch im Wiki unter: http://pp.info.uni-karlsruhe.de/lehre/SS2010/compprakt/wiki/index.php/FirmGraph_Aufbau. Außerdem kann man sich den in JFirm enthaltenen Brainfuck-Compiler ansehen.

Aufgabe 1: Typen

- Schreiben Sie eine Funktion die zu jedem ihrer einfachen Typen im AST einen entsprechenden Firm-Typ erzeugt. Wie gehen Sie mit den Typen `boolean` und `void` um?
- Schreiben Sie eine Funktion die Methodentypen erzeugt.
- Schreiben Sie eine Funktion die Klassentypen für ihre Klassen erzeugt.

Aufgabe 2: Funktionen erzeugen

- Schreiben Sie eine Funktion die initiale Firm-Graphen (also ohne weitere Knoten erzeugt) für Methoden in ihrem AST anlegt.
- Lassen Sie sich `.vcg`-Dateien für ihre Graphen erzeugen.
- Wie behandeln Sie Funktionsparameter?
- Wie behandeln Sie lokale Variablen?
- Implementieren Sie die `return`-Anweisung. Statt komplette Ausdrücke für die Rückgabewert umzusetzen können Sie zunächst `0`-Konstanten erzeugen.

Aufgabe 3: Ausdrücke

- Implementieren Sie eine Funktion die alle Literal-Ausdrücke in Firm-Knoten umsetzt.
- Schreiben Sie eine Funktion, die für einen arithmetischen Ausdruck im AST einen Firm-Graph aufbaut.
- Implementieren Sie eine Funktion die für Vergleiche einen Firm-Graph aufbaut.
- Implementieren Sie Zugriffe auf lokale Variablen und Funktionsparameter.
- Was ist das besondere an der linken Seite einer Zuweisung? Wie behandeln Sie diesen Fall?
- Schreiben Sie eine Funktion die Knoten für die Adressen eines Feldzugriffs und Arrayzugriffs erstellt. Erzeugen Sie die entsprechend nötigen Load und Store Knoten um Lese- und Schreibzugriffe umzusetzen.
- Implementieren Sie die `Call`- und `New`-Ausdrücke.

Aufgabe 4: Ausdrücke

- Implementieren Sie die `if`-Anweisung.

- Erzeugen Sie ein externes Entity namens `print_int` und implementieren Sie `System.out.println` als Aufruf dieser externen Funktion.
- Implementieren Sie die `while`-Anweisung.

Aufgabe 5: Zusatzaufgaben

Freiwillige Zusatzaufgaben.

5.1 x86-Backend

- Implementieren Sie zunächst eine Lowering-Phase die `sel`-Knoten in Adressrechnung umwandelt und `alloc`-Knoten in entsprechende Aufrufe aus der C-Bibliothek (z.B. `calloc`). Sie können auch den vorgegebenen Code von <http://pp.info.uni-karlsruhe.de/lehre/SS2010/compprakt/wiki/index.php/JFirm> in der Sektion Backend anpassen.
- Implementieren Sie eine „Laufzeitbibliothek“ in C um die `System.out.println`-Funktion umzusetzen. Erweitern Sie die Lowering-Phase, so dass `println`-Aufrufe in Aufrufe ihrer Bibliotheksfunktion geändert werden.
- Nutzen Sie das eingebaute x86-Backend in Firm um Code zu erzeugen.