Universität Karlsruhe (TH)

Lehrstuhl für Programmierparadigmen

Compilerpraktikum SS 2009 — http://pp.info.uni-karlsruhe.de/
Dozent: Prof. Dr.-Ing. G. Snelting — snelting@ipd.info.uni-karlsruhe.de
Betreuer: Matthias Braun — braun@ipd.info.uni-karlsruhe.de
Betreuer: Jürgen Graf — graf@ipd.info.uni-karlsruhe.de

Übungsblatt 3 | Ausgabe: 06.05.2009 | Besprechung: 13.05.2009

Hinweise:

- Auf der Homepage `http://pp.info.uni-karlsruhe.de/lehre/SS2009/compprakt/` finden Sie nun auch eine weitere Sammlung von Beispielprogrammen für den Test des abstrakten Syntaxbaumes.

Aufgabe 1: *Abstrakter Syntaxbaum*

1.1 *Modellierung*

Überlegen Sie sich aus welchen Elementen der abstrakte Syntaxbaum bestehen sollte und erstellen Sie ein kleines Dokument in dem jedes Element in 2-3 Sätzen beschrieben wird. Formulieren Sie eine entsprechende abstrakte Algebra, wie sie in der Vorlesung Compiler 1 vorgestellt wurde (siehe `http://pp.info.uni-karlsruhe.de/lehre/WS200809/compiler/compiler_zusatzfolien_syntax.pdf` Seite 6-7).

1.2 *Planung*

Zeigen Sie an folgendem Beispielprogramm, wie Ihr geplanter AST aussehen sollte. Was sind die Unterschiede zum entsprechenden Parsebaum?

```
1   public class A {
2
3     public int x;
4     public int y;
5
6     public int calc(int x, int y) {
7       int sum;
8
9       if (x > y) {
10        sum = x + y;
11      } else {
12        sum = x * y;
13      }
14
15      return sum;
16    }
17  }
```

1.3 *Implementierung*

Erweitern Sie ihren Parser aus dem letzten Aufgabenblatt entsprechenden, so dass ein abstrakter Syntaxbaum erzeugt wird.

Aufgabe 2: *Code zählen*

Entwickeln Sie ein Programm, dass ihren abstrakten Syntaxbaum durchläuft und für jede Klasse die Anzahl der enthaltenen Attribute sowie der enthaltenen Methoden ausgibt. Für jede Methode soll zudem die Anzahl der Parameter, der lokalen Variablen, der enthaltenen Statements und der enthaltenen Ausdrücke ausgegeben werden.

Verwenden Sie dabei folgendes Format:
`Class` **<ClassName>**`: A(`**<NumAttributes>**`), M(`**<NumMethods>**`)`
`Method` **<MethodName>**`: P(`**<NumParams>**`), L(`**<NumLocalVars>**`), S(`**<NumStatements>**`), E(`**<NumExpressions>**`)`

Beispiel:
Die entsprechende Ausgabe für das Programm aus Aufgabe 1 wäre somit:
`Class` **A**`: A(`**2**`), M(`**1**`)`
`Method` **calc**`: P(`**2**`), L(`**1**`), S(`**4**`), E(`**14**`)`

Lassen Sie sich diese Information für alle Beispielprogramme ausgeben und vergleichen Sie ihre Ausgaben mit denen der anderen Gruppen.