Universität des Saarlandes
Fachrichtung 6.2 – Informatik
Prof. Dr. R. Wilhelm
Dr.-Ing. D. Kästner
Dipl.-Inf. M. Langenbach

10. Mai 2001

Lösungsvorschlag zum Übungsblatt 1: Übersetzerbau (SS 2001)

Aufgabe 1 (4 Punkte)

Einfache Typen werden durch `V` (void) und `I` (int) beschrieben. Klassennamen werden zwischen `L` und `;` eingebettet. Einem *n*-dimensionalen Feld werden *n* `[` vorangestellt. Signaturen von Methoden setzen sich zusammen aus dem Methodennamen, der Typliste der formalen Parameter in runden Klammern und der Rückgabewert[1].

(a) `m1()V`

(b) `m2(ILString;)V`

(c) `m3([I[[I)I`

(d) `m4([LObject;I[LString;)LObject;`

Aufgabe 2 (6 Punkte)

$$
\begin{array}{llll|ll}
\text{(a)} & \textit{code}\ \texttt{1+a*a}\ \varrho & = & \textit{code}\ \texttt{1}\ \varrho & = & \textbf{iconst\_1} \\
 & & & \textit{code}\ \texttt{a*a}\ \varrho & & \textbf{iload}\ \varrho(\texttt{a}) \\
 & & & \textbf{iadd} & & \textbf{iload}\ \varrho(\texttt{a}) \\
 & & & & & \textbf{imul} \\
 & & = & \textit{code}\ \texttt{1}\ \varrho & & \textbf{iadd} \\
 & & & \textit{code}\ \texttt{a}\ \varrho & & \\
 & & & \textit{code}\ \texttt{a}\ \varrho & = & \textbf{iconst\_1} \\
 & & & \textbf{imul} & & \textbf{iload}\ 2 \\
 & & & \textbf{iadd} & & \textbf{iload}\ 2 \\
 & & & & & \textbf{imul} \\
 & & & & & \textbf{iadd}
\end{array}
$$

[1] Als Präfix wird in der JVM noch zusätzlich der Klassenname verwendet, der hier aber unerheblich ist.

(b) *code* `a*(1+b)` ϱ = *code* `a` ϱ	= **iload** ϱ(`a`)
code `1+b` ϱ	**iconst_1**
imul	**iload** ϱ(`b`)
	iadd
= *code* `a` ϱ	**imul**
code `1` ϱ	
code `b` ϱ	= **iload** 2
iadd	**iconst_1**
imul	**iload** 3
	iadd
	imul

(c) *code* `a=(a+b)*(c+3*d)` ϱ	= **iload** ϱ(`a`)
	iload ϱ(`b`)
= *code* `(a+b)*(c+3*d)` ϱ	**iadd**
istore ϱ(`a`)	**iload** ϱ(`c`)
	iconst_3
= *code* `a+b` ϱ	**iload** ϱ(`d`)
code `c+3*d` ϱ	**imul**
imul	**iadd**
istore ϱ(`a`)	**imul**
	istore ϱ(`a`)
= *code* `a` ϱ	
code `b` ϱ	= **iload** 2
iadd	**iload** 3
code `c` ϱ	**iadd**
code `3` ϱ	**iload** 5
code `d` ϱ	**iconst_3**
imul	**iload** 7
iadd	**imul**
imul	**iadd**
istore ϱ(`a`)	**imul**
	istore 2

In den obigen Instruktionssequenzen wurde der Befehl **iconst_**x verwendet, $0 \leq x \leq 5$. In diesem Begehl ist der Wert der ganzen Zahl, die auf den Stack gelegt werden soll, bereits enthalten, so daß im Vergleich zu **bipush** x die Codegröße verringert werden kann. Um Konstanten x auf dem Stack abzulegen sollte demnach:

– **iconst_**x für $0 \leq x \leq 5$,

– **bipush** x für $6 \leq x \leq 255$ und

– **ldc** x^2 sonst

verwendet werden.

Aufgabe 3 (6 Punkte)

$$
\begin{aligned}
&\textit{code}\ \texttt{a=new int[10][][]}\ \varrho\\
&= \textit{code}\ \texttt{new int[10][][]}\ \varrho\\
&\quad \textbf{astore}\ \varrho(\texttt{a})
\end{aligned}
$$

$$
\begin{aligned}
&= \textit{code}\ \texttt{10}\ \varrho\\
&\quad \textbf{multianewarray}\ \texttt{[[[I}\ 1\\
&\quad \textbf{astore}\ \varrho(\texttt{a})\\
&= \textbf{bipush}\ 10\\
&\quad \textbf{multianewarray}\ \texttt{[[[I}\ 1\\
&\quad \textbf{astore}\ 13
\end{aligned}
$$

$$
\begin{array}{lll}
\multicolumn{3}{l}{\textit{code}\ \texttt{for (c=1; c<10; c++)...}\ \varrho}\\
= & & \textit{code}\ \texttt{c=1}\ \varrho\\
 & \ell_1: & \textit{code}\ \texttt{c<10}\ \varrho\\
 & & \textbf{ifeq}\ \ell_2\\
 & & \textit{code}\ \texttt{a[c]...}\ \varrho\\
 & & \textit{code}\ \texttt{c++}\ \varrho\\
 & & \textbf{goto}\ \ell_1\\
 & \ell_2: & \\
= & & \textbf{iconst\_1}\\
 & & \textbf{istore}\ 22\\
 & \ell_1: & \textbf{iload}\ 22\\
 & & \textbf{bipush}\ 10\\
 & & \textbf{isub}\\
 & & \textbf{iflt}\ \ell_3\\
 & & \textbf{iconst\_0}\\
 & & \textbf{goto}\ \ell_4\\
 & \ell_3: & \textbf{iconst\_1}\\
 & \ell_4: & \textbf{ifeq}\ \ell_2\\
 & & \textit{code}\ \texttt{a[c]...}\ \varrho\\
 & & \textbf{iload}\ 22\\
 & & \textbf{iconst\_1}\\
 & & \textbf{iadd}\\
 & & \textbf{istore}\ 22\\
 & & \textbf{goto}\ \ell_1\\
 & \ell_2: &
\end{array}
$$

$$
\begin{aligned}
&\textit{code}\ \texttt{a[c]=new int[c][2*c]...}\ \varrho\\
&= \textit{code}\ \texttt{a[c]}\ \varrho\\
&\quad \textit{code}\ \texttt{new int[c][2*c]}\ \varrho\\
&\quad \textbf{aastore}\\
&= \textbf{aload}\ 13\\
&\quad \textbf{iload}\ 22\\
&\quad \textit{code}\ \texttt{new int[c][2*c]}\ \varrho\\
&\quad \textbf{aastore}\\
&= \textbf{aload}\ 13\\
&\quad \textbf{iload}\ 22\\
&\quad \textbf{iload}\ 22\\
&\quad \textbf{iconst\_2}\\
&\quad \textbf{iload}\ 22\\
&\quad \textbf{imul}\\
&\quad \textbf{multianewarray}\ \texttt{[[[I}\ 2\\
&\quad \textbf{aastore}
\end{aligned}
$$

[2] Dies ist der **ldc**-Befehl in Jasmin-Assembler-Syntax. Der eigentliche JVM-Befehl erwartet an dieser Stelle einen Index in die Konstantentabelle.