Aufgabe 8: Indirekte Rekursion

MicroJava unterstützt zwar (direkte) Rekursion, bei der eine Methode sich selbst aufrufen kann. Es unterstützt aber keine indirekte Rekursion, bei der zum Beispiel eine Methode foo eine Methode bar aufruft und diese wiederum foo. Warum wird indirekte Rekursion von MicroJava nicht unterstützt?

Lösung

Der MicroJava-Compiler ist ein Einpass-Compiler, der ein Programm streng sequentiell abarbeitet. Wenn er auf die Verwendung eines Namens stößt, muss dieser vorher deklariert worden sein. Das ist aber bei indirekter Rekursion nicht möglich.

```
void foo() {
   if (...) bar();
   ...
}

void bar() {
   if (...) foo();
   ...
}
```

Bei der Abarbeitung von foo ist bar noch nicht deklariert. Würde man die beiden Methoden in umgekehrter Reihenfolge deklarieren, hätte man das gleiche Problem: Bei der Abarbeitung von bar wäre foo noch nicht deklariert.

Mehrpass-Compiler bauen hingegen eine interne Programmrepräsentation auf (z.B. einen abstrakten Syntaxbaum). Dabei werden im ersten Pass zwar die Deklarationen verarbeitet und in die Symbolliste eingetragen, aber die Namen in den Anweisungen werden noch nicht in der Symbolliste gesucht. In einem zweiten Pass werden dann die Anweisungen verarbeitet. Zu diesem Zeitpunkt sind alle Deklarationen bereits bekannt, so dass indirekte Rekursion möglich wird.

Einpass-Compiler lösen das Problem der indirekten Rekursion oft über "Forward-Deklarationen". Dabei kann die Signatur einer Methode ohne Code deklariert werden, zum Beispiel:

```
void bar(); forward;

void foo() {
   if (...) bar();
   ...
}

void bar() {
   if (...) foo();
   ...
}
```

In diesem Fall ist die Deklaration von bar bereits in foo bekannt. Bei der tatsächlichen Deklaration von bar muss allerdings überprüft werden, ob die Parameter mit der Forward-Deklaration übereinstimmen. Forward-Deklarationen sind in MicroJava nicht vorgesehen.