Inst. f. Praktische Informatik **Übungen zu Übersetzerbau WS 2003/2004**

Prof. Dr. H. Mössenböck **Übung 6**

Name _____ Matr. Nr. _____

Übungsgruppe: Punkte _____ korr. _____

❑ 1 (Wöß)	Do 10^{15} - 11^{45}	
❑ 2 (Wöß)	Do 12^{00} - 13^{30}	Letzter Abgabetermin:
❑ 3 (Rammerstorfer)	Do 17^{15} - 18^{45}	Donnerstag, 8.1.2004, 8^{15} Uhr

Codeerzeugung – Teil 2 (24 Punkte)

Vervollständigen Sie nun Ihren Compiler, indem Sie auch die fehlenden Teile der Codeerzeugung gemäß der in den Unterlagen ausgegebenen Spezifikation der *MicroJava*-VM hinzufügen. Alle weiteren nötigen Klassen befinden sich im Package `ssw.mj.codegen`.

Um nun Sprünge innerhalb des Codes realisieren zu können, benötigen wir zunächst sogenannte *Labels*, also Sprungmarken, mit denen die Ziele der Sprünge greifbar gemacht werden. Führen Sie dazu die Klasse `Label` ein:

```
class Label {
  boolean defined;          // is destination of jump already defined?
  int adr;                  // jump destination address (defined) or
                            // head of unresolved forward jumps (!defined)
  void put();               // generates code for a jump to this label
  void here();              // defines this label to be at the current pc position
  void setTo (Label dest);  // defines this label to be at the position of dest
}
```

Außerdem muss die Klasse *Item* wie folgt erweitert werden:

```
class Item {
  static final int       // item kinds
    Con    = 0,           // constant
    Local  = 1,           // local variable
    Static = 2,           // global variable
    Stack  = 3,           // expression on stack
    Fld    = 4,           // field of inner class
    Elem   = 5,           // array element
    Meth   = 6,           // method call
    Cond   = 7;           // condition

  int kind;              // Con, Local, Static, Stack, Fld, Elem, Meth, Cond
  Struct type;           // item type
  int adr;               // Con:Wert; Local,Static,Fld,Meth:Adresse; Cond: Operator
  Obj obj;               // Meth: method object from symboltable
  Label tLabel, fLabel;  // Cond: True Jumps, False Jumps
}
```

Testen Sie Ihren Generator ausführlich. Lassen Sie Ihre Testprogramme auch tatsächlich auf der MJ-VM (`java ssw.mj.Run <obj-Datei>`) laufen.

Auf der Übungsseite finden Sie ein Testprogramm (`TestProgram.mj`), das Ihr Compiler korrekt übersetzen muss. Läuft das Programm korrekt ab, so wird die Zeichenfolge „1234“ ausgegeben.

Ebenfalls auf der Übungsseite finden Sie die MicroJava-VM (`MJVM.jar`).

Die MicroJava-VM-Datei `Run.java` ist aber auch schon im `UB-UE06-Angabe.zip` Archiv enthalten. Sie müssen diese in dasselbe Verzeichnis wie den Compiler (`Compiler.java`, `Parser.java`, ...) – also `ssw.mj` – entpacken und dann compilieren.

Sie starten die VM mit einer der folgenden Kommandozeilen:

```
> java ssw.mj.Run [-debug] <Objectfile>
> java -jar mjvm.jar [-debug] <Objectfile>
```

Die MicroJava-Objektdatei muss dem in den Unterlagen angegebenen Format entsprechen. Die Standard-Ein- und Ausgabe erfolgt auf der Konsole. Ist die Option `-debug` spezifiziert, so wird ein Trace des Programmablaufs im folgenden Format am Bildschirm ausgegeben:

```
Position im Code (Bytenr.): Befehl           Parameter
                          | Expressionstack
```