Informatik II (D-ITET)
Frühjahrssemester 2012
Prof. F. Mattern
S. Mayer (simon.mayer@inf.ethz.ch)

ETH Zürich
Institut für Pervasive Computing
Gruppe für Verteilte Systeme
http://vs.inf.ethz.ch

Übungsserie Nr. 3

Ausgabe: 14. März 2012
Abgabe: 21. März 2012

Hinweise

Für diese Serie benötigen Sie das Archiv
`http://vs.inf.ethz.ch/edu/FS2012/I2/downloads/u3.zip`.

1. Aufgabe: (5 Punkte) Objekte, Strings und Referenzen

Strings gibt es in Java in zwei Varianten: den unveränderlichen (immutable) `String` und den veränderlichen (mutable) `StringBuffer`.

Unveränderliche Objekte können nach ihrer Erzeugung nicht mehr modifiziert werden. Dadurch sind dem Compiler und der Laufzeitumgebung Optimierungen möglich, so dass sich z.B. mehrere Objekte den selben Speicher teilen können. Falls jedoch eine Modifikation nötig ist, muss stattdessen eine modifizierte Kopie angelegt werden, was extra Laufzeit und Speicher kostet. Deshalb hängt es vom konkretem Anwendungsfall ab, welcher Typ effizienter ist.

Analysieren Sie die Klasse *u3a1.StringObject*.

(1a) *(1 Punkt)* Was ist die Ausgabe des Programms?

(1b) *(4 Punkte)* Beschreiben Sie, welche Referenzen und welche Objekte an den jeweils markierten Stellen im Programm zur Laufzeit existieren. Nehmen Sie dabei an, dass nicht mehr erreichbare Objekte im Speicher erhalten bleiben, also keine Garbage Collection durchgeführt wird.

Verwenden Sie folgende Notation mit fortlaufenden IDs:

- Objekt: *(Objekt-ID, Typ, Inhalt)*
- Referenz: *(Name, Objekt-ID)*

Zum Beispiel führt der folgende Code

```
String str = "foo";
StringBuffer buf = new StringBuffer("foobuffer");
// Markierung 1
```

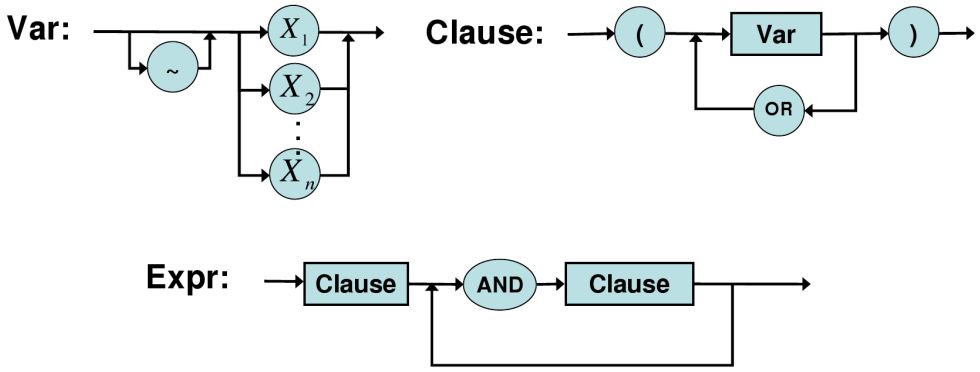
zu folgenden Objekten und Referenzen an der *Markierung 1*:

```
(1, String, "foo")
(2, String, "foobuffer")
(3, StringBuffer, "foobuffer")

(str, 1)
(buf, 3)
```

2. Aufgabe: (4 Punkte) Syntaxdiagramm

Betrachten Sie folgendes Syntaxdiagramm:



(2a) *(2 Punkte)* Geben Sie an, welche der folgenden Ausdrücke nach dem Diagramm *Clause* korrekt erzeugt werden können.

	erzeugbar	**nicht erzeugbar**		**erzeugbar**	**nicht erzeugbar**
X_2	○	○	$\sim (X_1 \text{ OR } \sim X_2)$	○	○
$(\sim X_1)$	○	○	$(X_2) \text{ OR } (\sim X_1 \text{ OR } X_2)$	○	○

(2b) *(2 Punkte)* Geben Sie an, welche der folgenden Ausdrücke nach dem Diagramm *Expr* korrekt erzeugt werden können.

	erzeugbar	**nicht erzeugbar**
$(X_1 \text{ OR } X_2) \text{ AND } (\sim X_1)$	○	○
$(X_1) \text{ AND } (\sim X_1 \text{ OR } \sim X_2) \text{ AND } (X_2)$	○	○

3. Aufgabe: (7 Punkte) Syntaxchecker

(3a) *(1 Punkt)* Ergänzen Sie das Syntaxdiagramm auf Seite 63 (Folie 152) im Skript, sodass leere Bäume und leere Teilbäume generiert werden können. Ein leerer Baum bzw. Teilbaum soll dabei durch ein '-' repräsentiert werden. Achten Sie darauf, dass durch Ihre Modifikation keine ungültigen Bäume generiert werden können.

(3b) *(6 Punkte)* Die Klasse *u3a3.KD* soll ein Syntaxchecker für Wurzelbäume in Klammerdarstellung werden. Vervollständigen Sie die Implementierung.

Hinweis: Schreiben Sie für *Baum*, *Nachfolger* und *Knoten* je eine eigene Funktion, die mit Hilfe der jeweils anderen Funktionen und analog zum jeweiligen Syntaxdiagramm den String sequentiell überprüft und die Anzahl der überprüften Zeichen zurück gibt.