Prof. Dr. Volker Linnemann
Dr. Sven Groppe
Universität zu Lübeck
Institut für Informationssysteme

Lübeck, den 25.1.2008

Vorlesung „Programmieren“

Wintersemester 2007/2008

11. Übungsblatt

1. Generizität (Programmieraufgabe) (23 Punkte)

Folgender Java-Quelltext sei gegeben, der einige benutzerdefinierte generische Klassen verwendet:

```
import java.util.*;

public class Test {
    public Test() {
    }

    public static void main(String[] args) {
        List<Integer> l = new LinkedList<Integer>();
        l.add(new Integer(1));
        l.add(new Integer(2));
        l.add(new Integer(3));
        l.add(new Integer(4));
        l.add(new Integer(5));
        Output<Integer> oi=new Output<Integer>();
        Map<Integer> mi = new Map<Integer>();
        System.out.print("Inhalt der Integer-Liste:");
        l = mi.map(l,oi);
        System.out.println();
        AddConstant<Integer> ac =
             new AddConstant<Integer>(2, new IntegerOperations());
        l = mi.map(l,ac);
        System.out.print("Inhalt der Integer-Liste:");
        l = mi.map(l,oi);
        System.out.println();
        Square<Integer> si = new Square<Integer>(new IntegerOperations());
        l = mi.map(l,si);
        System.out.print("Inhalt der Integer-Liste:");
        l = mi.map(l,oi);
        System.out.println();

        List<Double> ld = new LinkedList<Double>();
        ld.add(new Double(1));
        ld.add(new Double(2));
        ld.add(new Double(3));
        ld.add(new Double(4));
        ld.add(new Double(5));
        Output<Double> od=new Output<Double>();
        Map<Double> md = new Map<Double>();
        System.out.print("Inhalt der Double-Liste:");
        ld = md.map(ld,od);
        System.out.println();
        AddConstant<Double> acd =
             new AddConstant<Double>(2.0, new DoubleOperations());
        ld = md.map(ld,acd);
        System.out.print("Inhalt der Double-Liste:");
        ld = md.map(ld,od);
        System.out.println();
        Square<Double> sd = new Square<Double>(new DoubleOperations());
        ld = md.map(ld,sd);
        System.out.print("Inhalt der Double-Liste:");
        ld = md.map(ld,od);
        System.out.println();

        List<String> ls = new LinkedList<String>();
```

```
        ls.add("1");
        ls.add("2");
        ls.add("3");
        ls.add("4");
        ls.add("5");
        Output<String> os=new Output<String>();
        Map<String> ms = new Map<String>();
        System.out.print("Inhalt der String-Liste:");
        ls = ms.map(ls,os);
        System.out.println();
        AddConstant<String> acs =
             new AddConstant<String>("!", new StringOperations());
        ls = ms.map(ls,acs);
        System.out.print("Inhalt der String-Liste:");
        ls = ms.map(ls,os);
        System.out.println();
        Square<String> squares = new Square<String>(new StringOperations());
        ls = ms.map(ls,squares);
        System.out.print("Inhalt der String-Liste:");
        ls = ms.map(ls,os);
        System.out.println();
    }
}
```

Die Ausführung der `main(String[] args)`-Methode ergibt folgende Ausgabe:

```
Inhalt der Integer-Liste:1 2 3 4 5
Inhalt der Integer-Liste:3 4 5 6 7
Inhalt der Integer-Liste:9 16 25 36 49
Inhalt der Double-Liste:1.0 2.0 3.0 4.0 5.0
Inhalt der Double-Liste:3.0 4.0 5.0 6.0 7.0
Inhalt der Double-Liste:9.0 16.0 25.0 36.0 49.0
Inhalt der String-Liste:1 2 3 4 5
Inhalt der String-Liste:!1 !2 !3 !4 !5
Inhalt der String-Liste:!!11!1 !!22!2 !!33!3 !!44!4 !!55!5
```

Leider sind die benutzerdefinierten Klassen verloren gegangen. Ihre Aufgabe ist es nun, diese benutzerdefinierten Klassen zu entwickeln, wobei Generizitäts-Konstrukte sinnvoll eingesetzt werden sollen. Weiterhin sollen weder Klassennamen noch Methodennamen und Signaturen verändert werden, so dass der angegebene Java-Quelltext übersetzbar bleibt und bei Ausführung die gleiche Ausgabe erzeugt.

Aufgaben:

a) Die Klasse `Map` soll genau eine Methode enthalten, die nacheinander alle einzelnen Elemente einer Liste mittels einer Methode bearbeitet und das aktuelle Element durch das Ergebnis der Methode in der Liste ersetzt. (5 Punkte)

b) Die Klasse `Output` soll eine Methode für die Klasse `Map` zur Verfügung stellen, um die einzelnen Elemente einer Liste durch Leerzeichen getrennt auszugeben. (2 Punkte)

c) Die Klasse `AddConstant` soll eine Methode für die Klasse `Map` zur Verfügung stellen, um eine vorher übergebene Konstante zu den einzelnen Elementen einer Liste hinzu zu addieren. (3 Punkte)

d) Die Klasse `Square` soll eine Methode für die Klasse `Map` zur Verfügung stellen, um das Quadrat der einzelnen Elemente einer Liste zu bilden. (3 Punkte)

e) Die Klasse `IntegerOperations` soll zwei Methoden zur Verfügung stellen, die zwei `Integer`-Objekte addiert bzw. multipliziert. (2 Punkte)

f) Die Klasse `DoubleOperations` soll zwei Methoden zur Verfügung stellen, die zwei `Double`-Objekte addiert bzw. multipliziert. (2 Punkte)

g) Die Klasse `StringOperations` soll zwei Methoden zur Verfügung stellen, die zwei `String`-Objekte aneinanderhängt bzw. eine Art Multiplikation durchführt.

Die „Multiplikation“ von zwei Strings $a_1 \dots a_n$ und $b_1 \dots b_m$, wobei a_i und b_i die einzelnen Zeichen der beiden Strings darstellen, sei folgendermaßen definiert:

$a_1\ b_1\ \ldots\ b_m\ a_2\ b_1\ \ldots\ b_m \quad \ldots \quad a_n\ b_1\ \ldots\ b_m$

Zum Beispiel ist die „Multiplikation" von „`123`" und „`ab`" gleich „`1ab2ab3ab`". (3 Punkte)

h) Verwenden Sie an geeigneter Stelle Vererbung. (3 Punkte)

Bemerkungen:

- Dieses ist eine Programmieraufgabe. Bitte geben Sie die Lösung *sowohl* auf **Papier** *als auch* den Java-Quelltext in elektronischer Form per **Email** an Ihren Tutor ab.

Bemerkungen:

- Jede Seite soll oben rechts den Namen der Abgebenden und die Übungsgruppennummer (wichtig!) enthalten.
- Lösungen für die Übungsaufgaben sind (in der Regel) zu zweit abzugeben.
- Kommentieren Sie Ihre Lösungen! Besteht eine Lösung aus mehreren Zetteln, so sind diese zusammen zu heften. Bitte keine Hüllen, Mappen, o.ä..
- Bitte schicken Sie *Programmieraufgaben zusätzlich zur Abgabe auf Papier in elektronischer Form per Email* an Ihren jeweiligen Tutor.
- Kommentieren Sie ihren Quelltext bei Programmieraufgaben. Dabei sollen keine Trivialitäten kommentiert werden, also bitte keine Kommentare wie

  ```
  x=5; // Wir weisen nun der Variablen x den Wert 5 zu
  ```

  sondern sinnvolle Kommentare, die Ideen des Quelltextabschnittes beschreiben oder auf Unteraufgaben (z. B. a), b), …) hinweisen.
- **Hinreichende Bedingung für die Zulassung zur Klausur:** 50% der erreichbaren Punkte bei jedem Übungszettel (bis auf zwei) und einmaliges Vorrechnen in der Übung
- **Zertifikatskriterium:** Das Bestehen der Klausur am Ende des Semesters

Abgabetermin: Freitag, 1.2.2008, nach der Vorlesung