TECHNISCHE UNIVERSITÄT MÜNCHEN
FAKULTÄT FÜR INFORMATIK

Lehrstuhl für Sprachen und Beschreibungsstrukturen
Praktikum Grundlagen der Programmierung
F. Forster, T. Gawlitza, A. Flexeder

WS 2007/2008
Lösungsvorschläge zu Blatt 6
23. November 2007

Übungen zu Praktikum Grundlagen der Programmierung

Aufgabe 33 (Ü) Vergleiche und Referenzen in Java

```
public class Person {
    private String vorname;
    private String nachname;

    // Konstruktor
    public Person(String vn, String nn) {
        vorname = vn;
        nachname = nn;
    }

    // einfacher Gleichheitstest (für die Übung ausreichend)
    public boolean equals(Person person) {
        if (this == person)
            return true;
        return vorname.equals(person.vorname) && nachname.equals(person.nachname);
    }
}

public class Vergleiche {
    public static void main(String[] args) {
        // a)
        int a = 1;
        int b = 1;
        String c = new String("a");
        String d = new String("a");

        System.out.println("a␣==␣a␣:␣" + (a == a)); // true
        System.out.println("a␣==␣b␣:␣" + (a == b)); // true

        System.out.println("c␣==␣c␣:␣" + (c == c)); // true
        System.out.println("c␣==␣d␣:␣" + (c == d)); // false
        System.out.println("c.equals(c)␣:␣" + c.equals(c)); // true
        System.out.println("c.equals(d)␣:␣" + c.equals(d)); // true

        System.out.println();

        // b)
        System.out.println("Person:");
        Person e = new Person("Bodo", "Bierfass");
        Person f = new Person("Bodo", "Bierfass");
        System.out.println("e==e␣:␣" + (e == e)); // true
        System.out.println("e==f␣:␣" + (e == f)); // false
        System.out.println("e.equals(e)␣:␣" + e.equals(e)); // true
        System.out.println("e.equals(f)␣:␣" + e.equals(f)); // true
    }
}
```

Aufgabe 34 Doppelt verkettete Liste (Lösungsvorschlag)

```
public class Entry {
    String element;
    Entry next;
    Entry previous;

    Entry(String element, Entry next, Entry previous) {
        this.element = element;
        this.next = next;
        this.previous = previous;
    }

    public String getElement() {
        return element;
    }

    public String toString() {
        return "[Entry␣:␣" + element + "]";
    }
}

public class LinkedList {
    private Entry header;

    public LinkedList() {
        header = new Entry(null, null, null);
        header.previous = header.next = header;
    }

    private void addBefore(String o, Entry e) {
        if (o!=null){
            Entry newEntry = new Entry(o, e, e.previous);
            newEntry.previous.next = newEntry;
            newEntry.next.previous = newEntry;
        }
    }

    public void add(String o) {
        addBefore(o, header);
    }

    public String toString() {
        String retVal = "[";
        for (Entry e = header.next; e != header; e = e.next) {
            retVal = retVal + e.getElement();
            if (e.next != header)
                // Man ist nicht bem letzten Element, also "," ausgeben
                retVal = retVal + ",";
        }
        return retVal + "]";
    }
}

public class ListTest {
    public static void main(String[] args) {
        LinkedList list = new LinkedList();

        list.add(new String("Das"));
        list.add(new String("ist"));
        list.add(new String("eine"));
        list.add(new String("Test-List!"));

        System.out.println("Die␣Liste␣:␣" + list);
    }
}
```

Aufgabe 35 Textdokumente (Lösungsvorschlag)

```
public class Absatz {
    private String str;
    private boolean fett;
    private boolean kursiv;

    public Absatz(String str, boolean fett, boolean kursiv) {
        this.str = str;
        this.fett = fett;
        this.kursiv = kursiv;
    }

    public Absatz(String str) {
        this(str,false,false);
    }

    public String toString() {
        return str;
    }
}

public class AbsatzListe {
    protected Absatz absatz;
    protected AbsatzListe next;

    public AbsatzListe(Absatz a, AbsatzListe n) {
        absatz = a;
        next = n;
    }

    public AbsatzListe(Absatz a) {
        this(a, null);
    }
}

public class Seite {
    private int abstandAbsatz;
    private AbsatzListe anfang, ende;

    public Seite() {
    }

    public void setAbstandAbsatz(int d) {
        abstandAbsatz= d;
    }

    public int getAbstandAbsatz() {
        return abstandAbsatz;
    }

    public void addAbsatz(Absatz a) {
        if (anfang == null && ende == null) {
            anfang = ende = new AbsatzListe(a);
        } else {
            AbsatzListe tmp = new AbsatzListe(a);
            ende.next = tmp;
            ende = tmp;
        }
    }

    public String toString() {
        String d = "\n";
        for (int i = 0; i < abstandAbsatz; i++)
            d = d + "\n";

        String retVal = "";
        for (AbsatzListe l = anfang; l != null; l = l.next) {
            retVal = retVal + l.absatz.toString();
            if (l.next != null)
                retVal = retVal + d;
        }
        return retVal;
    }
}
```