Klausur: Programmierkurs 1, Java, HTML

C 013, A5

11. Oktober 2004

Name: .....
Vorname: .....
Matrikel-Nr.: .....
Studienfach: .....

Hinweise zur Bearbeitung

a. Die Klausur besteht aus **4 Aufgaben**, für die insgesamt **66 Punkte** vergeben werden. Die Klausur gilt als bestanden, wenn davon **mindestens 33 Punkte** (d.h. 50 %) erreicht werden.

b. Eigenes Papier und sonstige Hilfsmittel (z.B. Taschenrechner, Palm, Notebook, Handy) sind nicht zugelassen. Die Rückseiten der Klausurbögen dürfen verwendet werden.

c. Alle Lösungswege und Rechengänge sind grundsätzlich zu dokumentieren. **Lösungen ohne Lösungsweg werden nicht gewertet.**

Viel Erfolg !

Unterschrift des Kandidaten:

Bitte nicht ausfüllen:

Aufgabe	1	2	3	4	Summe
Punkte					

Bestanden	Korrektor
ja | nein	

Aufgabe 1 (17 Punkte)

a. Schreiben Sie ein HTML-Formular (keine vollständige Seite) entsprechend den folgenden Vorgaben:

- Bei Submitierung des Formulars soll das CGI-Programm `http://www.uni-mannheim.de/formular.cgi` ausgeführt werden.
- Es soll zwei Texteingabefelder mit der Bezeichnung '`Vorname`' bzw. '`Familienname`' geben. Diese Felder sollen den Namen '`given`' bzw. '`family`' haben und jeweils Platz für 30 Zeichen in der Anzeige haben.
- Die Felder sollen in einer Tabelle in drei Zeilen und zwei Spalten erscheinen.
- Die Tabellenzellen sollen mit einem Rand versehen sein.
- Der Submit-Button soll mit '`Abschicken`' beschriftet sein.

(9 Punkte)

b. Schreiben Sie das Formular und die Tabelle XML-konform um, d.h., schreiben Sie sie als XHTML Formular und Tabelle (wenn Sie es noch

nicht getan haben). (4 Punkte)

c. Geben Sie CSS-Regeln für folgende Elemente und Darstellungen an:

- Ein `strong`-Element innerhalb von `blockquote` soll mit blauer, fetter Schrift dargestellt werden.
- Ein `strong` innerhalb von `em` soll mit kursiver, gelber Schrift dargestellt werden.

(4 Punkte)

Aufgabe 2 (13 Punkte)

a. Was versteht man unter dem Überladen von Methoden? (3 Punkte)

b. Was versteht man unter dem Überschreiben von Variablen oder Methoden? (3 Punkte)

c. Was ist der Inhalt der Variablen `s` nach Ausführung des folgenden Programmfragments (bitte markieren) (2 Punkte)

```
int a = 4;
String s = a.toString();
```

- `"4"`,
- `a`,
- kein Inhalt, Kompilationsfehler.

d. Was ist der Inhalt der Variablen `s` nach Ausführung des folgenden Programmfragments (bitte markieren) (2 Punkte)

```
Object b = new Long(4);
String s = b.toString();
```

- `"4"`,
- `b`,
- `null`, Laufzeit-Fehler.

e. Was bedeuten die Modifizierer `public` und `private`? (3 Punkte)

Aufgabe 3 (22 Punkte)

Schreiben Sie ein einfaches Java Programm `Wetter.java`, mit dem Sie das Wetter zu einem bestimmten Zeitpunkt in einer Datei protokollieren und anzeigen können. Ein Wetter-Datensatz soll aus einem Datum und einer Temperatur bestehen. Der Inhalt der Datei soll intern in der `List`-Variablen `protokoll` gespeichert und verwaltet werden (z.B. als `ArrayList`).
Hinweise: Nützliche Klassen sind z.B. `BufferedReader`, `PrintWriter`, `FileReader` und `FileWriter`.

a. Implementieren Sie einen geeigneten Konstruktor (ohne Parameter) für die Klasse `Wetter`, der die Variable `protokoll` initialisiert. (2 Punkte)

b. Implementieren Sie Methoden `addTemperature(String temp)` und `addTemperature(String datum, String temp)`, die die Wetterinfos zu der `List` hinzufügt. Falls `datum` nicht angegeben wird, soll das aktuelle Datum aus `java.util.Date` (ohne weitere Formatierung) genommen werden. (3 Punkte)

c. Das Wetterprotokoll in `protokoll` kann zeilenweise oder als ganzes Objekt in eine Datei gespeichert bzw. davon gelesen werden. Entscheiden Sie sich in Teil d) für eine Implementierung und begründen Sie hier Ihre Wahl durch Angabe der Vor- und Nachteile. (2 Punkte)

d. Implementieren Sie `writeTemperatureFile( String filename)` sowie `readTemperatureFile( String filename)`. Die erste Methode soll die internen Daten in die angegebene Datei `filename` schreiben. Die zweite soll die Temperaturdaten aus der angegebenen Datei `filename` lesen und intern in `protokoll` ablegen.

(7 Punkte)

e. Implementieren Sie eine `main()` Methode. Diese soll zunächst ein `Wetter` Objekt erzeugen und dann, falls existent, die bisherigen Da-

ten aus der Datei `wetter.pro` lesen. Dann soll der in der Kommandozeile angegebene Temperatur- und Datums-Wert an den Wetterdatensatz angefügt werden und zum Schluss sollen die Daten wieder in die gleiche Datei zurückgeschrieben werden. Wenn kein neuer Datensatz angefügt wird (beim Weglassen der Kommandozeilen-Parameter) soll das Wetter-Protokoll am Bildschirm angezeigt werden. Beispiel für die Anzeige: (8 Punkte)

```
4. Oktober  6:00 - 10 Grad
4. Oktober 12:00 - 20 Grad
4. Oktober 2004 18:00 - 22 Grad
```

Beispiele für den Aufruf:

```
java Wetter "4. Oktober 6:00" "10 Grad"
java Wetter "20 Grad"
```

Aufgabe 4 (14 Punkte)

Gegeben seien ein Interface `Obst` und zwei Implementierungen `Apfel` und `Zitrus` (siehe nächste Seite). In einem Feld `obstkorb` befinden sich mehrere Äpfel und Zitrusfrüchte:

```
Obst[]    obstkorb = new Obst[ anzahl ];
       obstkorb[0] = new Apfel("Elstar","rot","herb");
       obstkorb[1] = new Apfel("Boskop","gruen","bitter");
       obstkorb[2] = new Zitrus("Orange","orange","suess");
       obstkorb[3] = new Apfel("GoldParmaene","gelb","suess");
       obstkorb[5] = new Zitrus("Limone","gruen","bitter");
       ...
obstkorb[anzahl-1] = new Zitrus("Zitrone","gelb","bitter");
```

a. Schreiben Sie ein Java Fragment (z.B. eine Schleife), mit dem Sie den Sortennamen (aus `getSorte()`) jedes `gruen`en Obststücks im `obstkorb` anzeigen (d.h. mit `println` ausgeben) können. (4 Punkte)

b. Schreiben Sie ein Java Fragment, mit dem Sie den Sortennamen jedes `herb`en `Apfel`s im `obstkorb` anzeigen können. (4 Punkte)

c. Anstelle eines Interfaces `Obst` könnte man unter Umständen auch eine Superklasse `Obst` oder eine abstrakte Klasse `Obst` verwenden und davon `Apfel` bzw. `Zitrus` ableiten. Was sind die Vor- und Nachteile der drei Möglichkeiten? (6 Punkte)

```
public interface Obst {
 public String getSorte();
 public String getFarbe();
 public String getGeschmack();
}

public class Apfel implements Obst {
 public Apfel(String sorte, String farbe, String geschmack) {
   ...
 }
 ...
}

public class Zitrus implements Obst {
 public Zitrus(String sorte, String farbe, String geschmack) {
   ...
 }
 ...
}
```