Klausur: SS2008, Kurs: MB2, PR2 (Prof. Dr. Schimkat), Datum: Montag, 07.07.08

Vorname: Nachname: Matrikelnummer:

3. Prüfungsversuch: O ja / Ø nein

Aufgabe	1	2	3	4	5	6	Übung	Gesamt	LV-Note
Punkte max:	14	14	12	15	6	14	25	100	
Punkte									

Hinweise:
1) Beschriften Sie alle verwendeten Blätter nur einseitig -> die Rückseite ist freizulassen.
2) Falls Sie Teile Ihrer Lösung auf Extrablätter schreiben:
   * Geben Sie auf dem Extrablatt Ihren Namen und die Aufgabennummer an.
   * Verwenden Sie für jede Aufgabe ein eigenes Extrablatt.
3) Auf zusätzlichen Blättern lassen Sie links, rechts, oben und unten mindestens je 2 cm Rand und schreiben Sie in die obere rechte Ecke Ihren Namen.
4) Es sind die in der Lehrveranstaltung eingeführten Fachbegriffe zu verwenden.
5) Die Lösungen sind programmiertechnisch sinnvoll, verständlich und den jeweiligen Konventionen entsprechend auszuführen. Unnötig komplizierte oder unsinnige Programmkonstrukte führen zu Punktabzug.
6) Kommunikation jeder Art ist nur mit dem Dozenten gestattet.

Aufgabe 1 (14 Punkte):

Gegeben sei:

```
public interface Function {
    public abstract int value (int argument);
}
```

Schreiben Sie eine Klasse `FunctionPlotter`, deren Objekte als Komponenten in einer Swing-GUI verwendet werden können. Objekte dieser Klasse sollen in einem Bereich der Ausdehnung von 0 bis 500 in x- und von -200 bis 400 in y-Richtung eine mathematische Funktion als roten Linienzug vor einem grauen Gitternetz zeichnen. Das Gitternetz soll einen Linienabstand von 25 Pixeln besitzen. Die mathematische Funktion wird als Konstruktorparameter des Typs `Function` an das `FunctionPlotter`-Objekt übergeben.

Hinweise: (1) Sie dürfen annehmen, dass die Werte der math. Funktion nie kleiner als -200 oder größer als 400 werden. (2) Beachten Sie die Richtung der y-Achse!

Aufgabe 2 (14 Punkte):

Schreiben Sie unter Verwendung von Swing ein einfaches, aber vollständiges Programm, das nebeneinander zwei Komponenten des Typs `FunctionPlotter` (vgl. Aufgabe 1) in gleicher Größe anzeigt.

Wählen Sie als mathematische Funktionen *200*sin(0.1*x)* und *150*cos(0.05*x - PI/4)*.

Hinweis: Fügen Sie ggf. notwendige Typecasts so ein, dass die Kurve möglichst genau gezeichnet wird.

Aufgabe 3 (12 Punkte):

Die Klasse `PList`, die Personen-Objekte mit einer einfachen, nicht optimierten Implementierung einer verketteten Liste mit Head- und Tailknoten verwaltet, enthält den folgenden Code (`get(i)` gibt die i-te Person zurück):

```
1     ...
2     private class Node {
3           public Person person;
4           public Node next;
5     }
6     ...
7     public boolean equals (Object otherObject) {
8           ... // siehe Aufgabe)
9           for (int i=0; i < size(); i++) {
10                if (!get(i).equals (otherPList.get(i)) {
11                      return false;
12                }
13          }
14          return true;
15    }
```

a) [6] Ergänzen Sie die in Zeilennummer 8 ausgelassenen Codezeilen. (*Hinweis: die Namen otherObject und otherPList sind so richtig, d.h. kein Tippfehler*).

b) [6] Die Implementierung in den Zeilen 9..13 ist nicht besonders effizient. Ersetzen Sie die Zeilen 9..13 durch effizienten Code.

Aufgabe 4 (15 Punkte):

Geben Sie an, ob die folgenden Aussagen richtig oder falsch sind:

	richtig	falsch
XML-Dokumente müssen mit CSS dargestellt werden	O	X
XML-Dokumente müssen wohlgeformt sein	X	O
XML-Dokumente müssen valide sein	O	X
XML-Tags können Attribute besitzen	X	O
XML-Elemente müssen einen Inhalt besitzen	O	X
HashMap implementiert Collection	O	X
Collection wird von Set implementiert	O	X
auf Listenelemente kann per Index zugegriffen werden	X	O
Hashtabellen müssen so groß sein wie die Menge der Hashwerte	X	O
Die Methode hashCode() wird in Collection deklariert	O	X

Aufgabe 5 (6 Punkte):

a) [3] Beantworten Sie die folgenden Fragen:

Sind LayoutManager Swing-Komponenten?	O ja	X nein
Sind Swing-Komponenten LayoutManager ?	O ja	X nein
Ist JSplitPane ein LayoutManager ?	O ja	X nein

, [3] Geben Sie möglichst genau an, woran man erkennt, ob die Objekte einer Klasse TemperatureEvents aussenden können:

Aufgabe 6 (14 Punkte):

a) [4] Geben Sie die Regel an, ob und welche Codeabschnitte, die von verschiedenen Threads ausgeführt werden können, snychronisiert werden müssen.

b) [10] In einem Datencenter (des Typs `DataCenter`) wird einem möglichen Datenverlust dadurch vorgebeugt, dass alle Daten (des Typs `Data`) auf zwei verschiedenen Festplatten (des Typs `Disk`) gespeichert werden. Damit möglichst viele Anwendungen bedient werden können, können mehrere Threads gleichzeitig Daten mit `restore ()` auslesen und mit `save ()` abspeichern. Zum Lesen und Schreiben von Daten müssen jeweils die Nummern der beteiligten Festplatten und ein Schlüssel zum Auffinden der Daten angegeben werden.

Ändern Sie die Methode `restore ()` so, dass (b.1) sich verschiedene Threads nicht gegenseitig stören können und dass (b.2) im Datencenter ein möglichst hoher Datendurchsatz möglich ist (die entsprechenden Änderungen in `save ()` müssen nicht angegeben werden).

```
1   public class DataCenter {
2
3     private Disk [] diskArray;
4
5     public Data restore (int disk1, int disk2, long key) {
6           Data data1 = diskArray [disk1].read (key);
7           Data data2 = diskArray [disk2].read (key);
8           if (data1.compareTo (data2) == 0) {
9                 return data1;
10          } else {
11                return null;
12          }
13    }
14    public void save (int disk1, int disk2, Data data, long key) {
15          diskArray [disk1].write (data, key);
16          diskArray [disk2].write (data, key);
17    }
18
19  }
20  // zur Information:
21  interface Data extends Comparable {
22  }
23  interface Disk {
24    public Data read (long key);
25    public void write (Data data, long key);
26  }
```