Klausur: SS2007, Kurs: MB2, PR2 (Prof. Dr. Schimkat), Datum: Dienstag, 18.07.07

Vorname Nachname: Matrikelnummer: 8

3. Prüfungsversuch: O ja / ☒ nein

Aufgabe	1	2	3	4	5	6	Übung	Gesamt	LV-Note
Punkte max:	11	7	15	17	11	14	25	100	XXXXXX
Punkte	8	3	7	5	8	3	23	57	3,0

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.

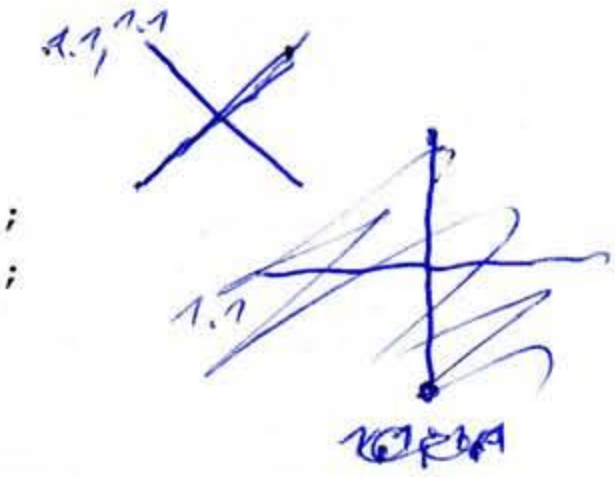
Aufgabe 1 (11 Punkte):

Schreiben Sie unter Verwendung von Swing ein einfaches, aber vollständiges Programm, das den Text `Hello Swing` mit einer JLabel-Komponente auf einer grafischen Oberfläche anzeigt. 8

Aufgabe 2 (7 Punkte):

Gegeben sei die folgende Klasse zur Verwendung als Komponente in einer Swing-GUI. Sie soll als Container andere Komponenten enthalten können und diese vor einem Hintergrund mit gekreuzten Linien darstellen. Die Klasse enthält leider mehrere Fehler. Korregieren Sie den gezeigten Teil des Quellcodes so, dass die Ausgabe der offenbar gewünschten Darstellung am nächsten kommt. 3

```
class CrossedLinesPanel extends JPanel {
      public void paint (Graphics g) {
            g.drawLine (1.1,   1.1, 999.9, 777.7);
            g.drawLine (1.1, 777.7, 999.9,   1.1);
      }
      ...
}
```



Aufgabe 3 (15 Punkte):

Schreiben Sie eine Klasse `BlauGelbSchalter` als Komponente für eine Swing-GUI. Jeder `BlauGelbSchalter` enthält zwei Schaltflächen, deren Anklicken dazu führt, dass die Hintergrundfarbe einer Zielkomponente auf Blau bzw. Gelb geändert wird. Die Zielkomponente vom Typ `javax.swing.JComponent` wird als Konstruktorparameter an den `BlauGelbSchalter` übergeben. 7

Aufgabe 4 (17 Punkte):

5

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 b)
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) [5] Skizzieren Sie den Zusammenhang aller Variablen, Referenzen und Objekte (a1) nach der Anweisung `Plist people = new Plist ();` und (a2) nach den weiteren Anweisungen

```
people.insert (0, new Person ("Henry"));
people.insert (0, new Person ("Susanna"));
```

2

b) [4] Ergänzen Sie die in Zeilennummer 8 ausgelassenen Codezeilen

1

c) [8] Erklären Sie stichwortartig, weshalb der Code in den Zeilen 9 bis 13 schlecht ist und geben Sie eine bessere Implementierung dieses Codeabschnitts an. (*Hinweis: die Namen otherObject und otherPList sind so richtig, d.h. kein Tippfehler*).

2

8/3

Aufgabe 5 (11 Punkte):

a) [3] Entsprechend ihrer Lage in der Containmenthierarchie lassen sich Komponenten einer grafischen Oberfläche in verschiedene Arten einteilen. Füllen Sie die folgende Tabelle mit den Namen der Komponentenarten und entsprechenden Klassen, die im Aufgabentext dieser Klausur nicht auftauchen, aus:

Komponentenart	Beispielklasse aus Swing
Top-Level-Container	JFrame
intermediate container	JPanel
atomare Komponente	JLabel

b) [5] Beantworten Sie die folgenden Fragen:

Sind LayoutManager Swing-Komponenten? O ja ☒ nein

Sind Swing-Komponenten LayoutManager ? O ja ☒ nein

Ist JTabbedPane ein LayoutManager ? O ja ☒ nein

Wie heisst die gemeinsame Superklasse von JButton und JMenuItem, die den genannten Klassen in der Vererbungshierarchie am nächsten steht? AbstractButton

Was ist ein tagging Interface? Ein Interface ohne abstracte Methoden. bzw ein leeres Interface.

5

Ein Beispiel für ein tagging Interface ist java.util.Cloneable

c) [3] Geben Sie möglichst genau an, woran man erkennt, ob die Objekte einer Klasse `ItemEvent`s aussenden können:

Ø

?

Aufgabe 6 (14 Punkte):

3/0

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.

3

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  }
```

import ... III

```
~~public~~ class Aufgabe1 {
    public static void main(String args[]){
        JFrame frame = new JFrame("HelloSwing");
        Container cont = frame.getContentPane();
        cont.add(new HelloSwing());
*       frame.setVisible(true);
    }
}
class HelloSwing extends JPanel{
    public HelloSwing(){
        ~~add(new JLabel("Hello Swing"));~~
        add(new HelloLabel());
    }
}
class HelloLabel extends JLabel{
    public HelloLabel(){
        setText("Hello Swing");
    }
}
```

8

* frame.setSize(400, 200); (✓)

Aufgabe 7

Component.

```
class CrossedLinesPanel extends JPanel{
    public void paint(Graphics g){
        super.paintComponent(g);
        g.drawLine(1, 1, 999, 777);
        g.drawLine(1, 1, 999, 778);
        g.drawLine(1, 778, 999, 1);
    }
}
```

1000

3

Aufgabe 3

final II JPanel (1) II

```
~~public~~ class BlauGelbSchalter extends JButton {
    public BlauGelbSchalter(JComponent comp){
        addActionListener(new ActionListener(){
            public void actionPerformed(ActionEvent
                e){
                if(comp.getBackground().equals
                    (Color.yellow))
                    comp.setBackground(Color.blue);
                else
                    comp.setBackground(Color.yellow);
            }
        });
    }
}
```

Komposition aus zwei Schaltflächen (siehe Aufgabe) fehlt (-4)

(1) Schaltfläche trägt keine Beschriftung

8

7

~~ROME~~

Aufgabe 4

~~offtopic~~

b) 8 if ~~(~~(!(otherObject.isInstanceOf(PList))){
9 return false;
10 }

1

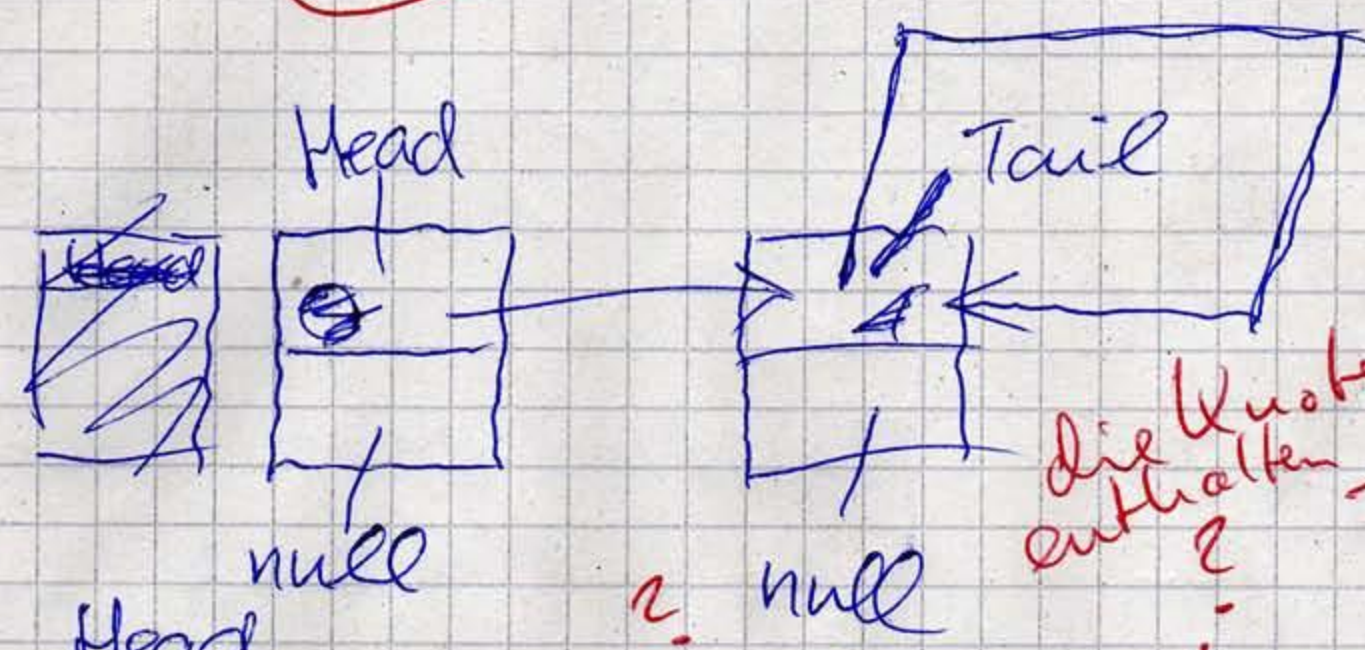
c) Der Code ist schlecht, da anhand von i ~~n~~ durch die Liste iteriert wird und auch die Schleife von i gesteuert wird. ungenau

2

while(n

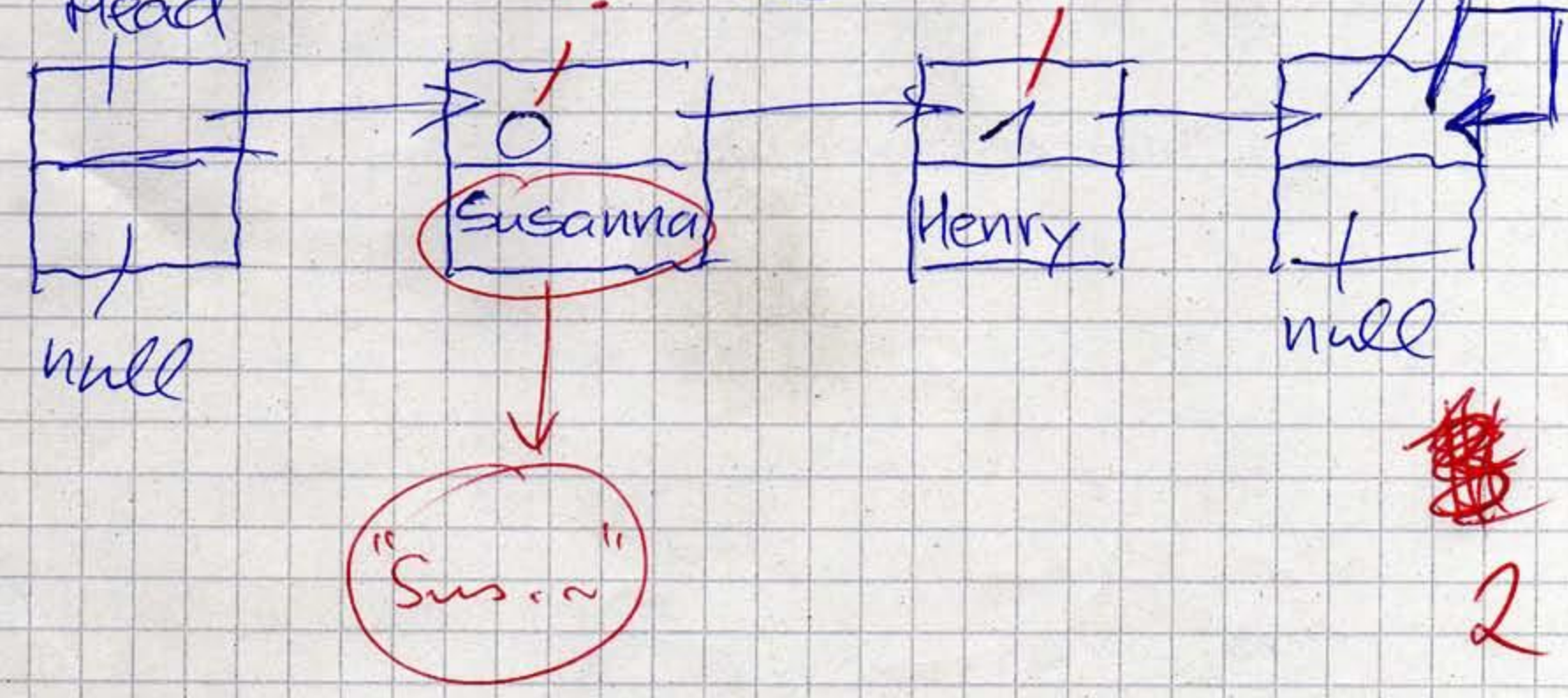
2

a) a1:



die Knoten haben keinen Index!

a2:



2

Aufgabe 6

a) Wenn mehrere Threads auf ein Objekt oder eine Methode zugreifen, sollte synchronisiert werden, da sonst ein Deadlock entstehen könnte.

0

b)

b1:

~~public Data restore (int disk1, int disk2, long key) {~~

~~Data data1 = diskArray [disk1]. read (key);~~

~~Data data2 = diskArray [disk2]. read (key);~~

~~Data data1 = (disk1 < disk2)~~ ?

3

```
public Data restore ( . . . ) {
    Data data1 = . . .
    Data data2 = . . .
    synchronized (data1) {
        if (data1)
        synchronized (data2) {
            if (data1.compareTo (data2) == 0) {
                :
            }
        }
    }
}
```

Deadlock-Gefahr

b2: ~~public synchronized void Save (...) {~~

~~}~~

~~}~~ public synchronized Data restore(. . .) {

}