

Universität Regensburg - Sommersemester 2004

Prof. Dr. Christian Wolff
Professur für Medieninformatik
Institut für Medien-, Informations- und Kulturwissenschaft



(Nachhol-)Klausur zur Lehrveranstaltung Einführung in die objektorientierte Programmierung mit Java

im Studiengang Informationswissenschaft

Dienstag, 8. Juni 2004

Allgemeine Hinweise

1. Bearbeitungszeit: 90 Minuten.
2. Maximal erreichbare Punktzahl: 90. Zu Ihrer Orientierung sind die erreichbaren Punkte bei jeder Frage genannt – bitte teilen Sie die Arbeitszeit entsprechend ein.
3. Schreiben Sie Ihren **Namen, Vornamen und Ihre Matrikelnummer (oder eine frei wählbare ID)** leserlich auf alle Klausurbögen, die Sie für Ihre Lösung verwenden - **bevor** Sie mit der Bearbeitung beginnen! Blätter ohne diese Angaben können nicht gewertet werden.
4. Verwenden Sie nur die bereitgestellten Klausurbögen.
5. Haken Sie ggf. nach Bearbeitung die Aufgaben auf der Angabe ab, um sicherzustellen, dass Sie keine Frage ausgelassen haben.
6. Benutzen Sie **keine Bleistifte, keine rotschreibenden Stifte und kein TippEx** (oder ähnliche Produkte).
7. Es sind **keine** weiteren Unterlagen (Skripte, Vorlesungsmitschriften, etc.) zugelassen.
8. Wenden Sie sich bei Unklarheiten in den Aufgabenstellungen immer an die Aufsichtsführenden. Hinweise und Hilfestellungen werden dann, falls erforderlich, offiziell für den gesamten Hörsaal durchgegeben. Aussagen unter „vier Augen“ sind ohne Gewähr.
9. Geben Sie keine mehrdeutigen (oder mehrere) Lösungen an. In solchen Fällen wird stets die Lösung mit der geringeren Punktzahl gewertet. Eine richtige und eine falsche Lösung zu einer Aufgabe ergeben also null Punkte.
10. Verändern Sie die Aufgabenstellung nicht, um Sie an Ihre Lösung „anzupassen“. Lösungen, die sich nicht an die vorgegebenen Aufgabenstellungen halten, werden mit null Punkten bewertet.



Fragen	Punkte
<p>Aufgabe 1 Gegeben sei folgende Problemstellung:</p> <p>Es soll eine Bestands- und Auftragsverwaltung für eine Schiffswerft entwickelt werden. Dazu liegt folgende Problembeschreibung vor:</p> <ul style="list-style-type: none">• Die Werft baut Fracht- und Personenschiffe (Fähren und Kreuzfahrtschiffe).• Alle Schiffe sind durch Name, Länge und Eigentümer gekennzeichnet und verfügen über ein Statusfeld mit den möglichen Werten beauftragt, im Bau, ausgeliefert. Die Werte für Eigentümer und Status müssen gelesen und geändert werden können.• Frachtschiffe sind zusätzlich durch Ihre Zuladung, Personenschiffe durch die Passagierzahl gekennzeichnet. Fähren können zusätzlich PKWs laden, Kreuzfahrtschiffe sind durch die Zahl ihrer Decks gekennzeichnet. <p>a) Entwerfen Sie in Java geeignete Klassen bzw. Schnittstellen für die obige Problemstellung. Verwenden Sie für die Typisierung von Eigenschaften primitive Datentypen bzw. den Zeichenkettentyp String. Geben Sie für eine der Klassen einen geeigneten Konstruktor an.</p> <p>b) Entwerfen Sie eine Werft-Klasse, in der die Schiffe der Werft verwaltet werden können. Der Bestand soll sich ändern, wenn ein Auftrag erteilt wird, ein Schiff fertig gebaut ist bzw. ausgeliefert wird. Dabei müssen Sie für erforderliche Methoden nur deren Signatur, aber keine Implementierung angeben.</p>	34 P.
<p>Aufgabe 2 Erklären Sie die Zugriffsmodifikatoren private, protected und public für Methoden und geben Sie ein Beispiel für ihre Verwendung.</p>	6 P.
<p>Aufgabe 3 Wie werden Java-Applets durch die virtuelle Java-Maschine eines Webbrowsers gesteuert? Wie unterscheiden Sie sich von Applikationen?</p>	6 P.



Fragen	Punkte
<p>Aufgabe 4 Erläutern Sie folgenden Klassenkopf („Inhalt“ und Funktion der Klasse sind ohne Belang).</p> <pre>public final class Integer extends Number implements Comparable</pre>	8 P.
<p>Aufgabe 5 Wandeln Sie die folgende Schachtelung von Bedingungsanweisungen in eine äquivalente Fallunterscheidung um.</p> <pre>// iZaehler sei eine Variable vom Typ int if(iZaehler == 10) { System.out.println("A"); } else { if(iZaehler == 8) { System.out.println("B"); } else { if(iZaehler == 9) { System.out.println("C"); } else { if(iZaehler == 6) { System.out.println("A"); } else { System.out.println("X"); } } } } // ...</pre>	8 P.



Aufgabe 7

28 P.

Schreiben Sie eine Java-Klasse **Sortierer**, die Kommandozeilenargumente einliest und sortiert ausgibt. Der eigentliche Sortiervorgang soll dabei in einer Methode **sortiere()** erfolgen. Ein Aufruf des Programms könnte wie folgt lauten:

```
java Sortierer      34    5    32    8    1    3456  8
Ausgabe:           1    5    8    8    32    34    3456
```

Hilfestellung:

Sie können Zeichenketten (Strings) mit der **parseInt()**-Methode der Klasse **Integer** in einen **int**-Wert umwandeln.

Die Sortierung kann nach einem beliebigen Verfahren erfolgen, z. B. durch wiederholten Vergleich und ggf. Tausch von Nachbarn in einem Array (**BubbleSort**).