

INFORMATIK I
Inoffizielle Probeklausur

15. JANUAR 2005

Vorname:

Nachname:

Matrikelnummer:

Tutorium:

Zur Probeklausur sind *keine* Hilfsmittel zugelassen. Die Bearbeitungszeit beträgt 60 Minuten. Bitte tragen Sie zunächst Ihren Namen, Ihre Matrikelnummer und Ihre Tutoriumsnummer sowie auf jede Seite Ihren Namen, Ihre Matrikelnummer und Ihr Tutorium ein.

Die Probeklausur ist komplett und geheftet abzugeben. Sie dürfen die Rückseiten der Aufgabenblätter als Konzeptpapier benutzen. Verwenden Sie *kein* eigenes Papier. Nur Lösungen, die sich auf dem entsprechenden Aufgabenblatt oder seiner Rückseite befinden, gehen in die Bewertung ein.

Die errungene Punktzahl und Note dienen allein der Selbstkontrolle. Sie gehen insbesondere nicht in die Bewertung Ihrer Übungsblätter ein. Art, Anzahl und Schwierigkeiten der Aufgaben in der Probeklausur sind fiktiv. Jegliche Zusammenhänge mit der eigentlichen Klausur sind unbeabsichtigt.

Unkostenbeitrag: € 0,50.

Aufgabe	1	2	3	4	5	Summe
Maximal	8	10	12	22	8	60
Erreicht						

Note:

1. Verständnis- und Wissensfragen (8 Punkte)

1.1 Wahr-/Falsch-Fragen (6 Punkte)

Kreuzen Sie an, ob die Aussage wahr (*W*) oder falsch (*F*) ist.

Hinweis: Jedes korrekte Kreuz zählt 0,5 Punkte, jedes falsche Kreuz bewirkt 0,5 Punkte Abzug! Die Teilaufgabe wird mindestens mit 0 Punkten bewertet.

- | | | |
|-----------------------------------|-----------------------------------|---|
| <input type="checkbox"/> <i>W</i> | <input type="checkbox"/> <i>F</i> | Quicksort hat im schlimmsten Fall eine Theoretische Effizienz von $O(n \log n)$. |
| <input type="checkbox"/> <i>W</i> | <input type="checkbox"/> <i>F</i> | Ein Algorithmus mit einer Theoretischen Effizienz von $O(n^2)$ ist für alle Eingaben schneller als ein Algorithmus mit einer Theoretischen Effizienz von $O(n^3)$. |
| <input type="checkbox"/> <i>W</i> | <input type="checkbox"/> <i>F</i> | Die Taktfrequenz des Prozessors in einem Von-Neumann-Rechner stellt den sogenannten „Flaschenhals“ dar. |
| <input type="checkbox"/> <i>W</i> | <input type="checkbox"/> <i>F</i> | Signalparameter unterliegen keinerlei zeitlichen Veränderungen. |
| <input type="checkbox"/> <i>W</i> | <input type="checkbox"/> <i>F</i> | Folgender Java-Code liefert keinen Fehler: <code>for(;;) {;}</code> |
| <input type="checkbox"/> <i>W</i> | <input type="checkbox"/> <i>F</i> | In einem Hamiltonschen Kreis ist jede Ecke eines ungerichteten Graphen mindestens einmal oder mehrmals enthalten. |
| <input type="checkbox"/> <i>W</i> | <input type="checkbox"/> <i>F</i> | Ein Graph ist genau dann azyklisch, wenn er keinen Eulerschen Zyklus enthält. |
| <input type="checkbox"/> <i>W</i> | <input type="checkbox"/> <i>F</i> | Ein gerichteter Baum hat genau eine Ecke e mit $ \bullet e = 0$. |
| <input type="checkbox"/> <i>W</i> | <input type="checkbox"/> <i>F</i> | Der Euklidische Algorithmus berechnet den größten gemeinsamen Teiler zweier natürlicher Zahlen. |
| <input type="checkbox"/> <i>W</i> | <input type="checkbox"/> <i>F</i> | Klassennamen werden in einem UML-Klassendiagramm unterstrichen dargestellt. |
| <input type="checkbox"/> <i>W</i> | <input type="checkbox"/> <i>F</i> | Die Sonne steuert das Wachstum der Pflanzen. |
| <input type="checkbox"/> <i>W</i> | <input type="checkbox"/> <i>F</i> | Der Lautstärkereger des Radios regelt dessen Lautstärke. |

1.2 Wissensfragen (2 Punkte)

a) Definieren Sie Information im Sinne der Vorlesung. (1 Punkt)

b) Definieren Sie System im Sinne der Vorlesung. (1 Punkt)

2. Algebren (10 Punkte)

2.1 Kantorowitsch-Baum (5 Punkte)

Gegeben sei folgender arithmetischer Ausdruck in Infix-Form:

$$(a + b) * ((c - d) * 5) - 2.$$

Hinweis: Beachten Sie die übliche Vorrangregelung der Operatoren.

- Zeichnen Sie den zugehörigen Kantorowitsch-Baum. (3 Punkte)
- Bestimmen Sie die Postfix-Form des Ausdrucks. (2 Punkte)

2.2 Boolesche Algebra (5 Punkte)

Wandeln Sie den Ausdruck

$$\left(((a \wedge b) \vee (\neg a \wedge \neg b)) \wedge \neg(a \vee c) \right) \vee (b \wedge c)$$

unter Verwendung der Gesetze der Booleschen Algebra in konjunktive Normalform (KNF) um.

3. Relationen (12 Punkte)

3.1 Halbgruppen & Monoide (4 Punkte)

Überprüfen Sie, ob es sich bei den folgenden Strukturen um Halbgruppen, Monoide oder keines von beidem handelt. Beweisen Sie Ihre Aussage.

- a) Die ganzen Zahlen \mathbb{Z} mit der Verknüpfung \star : $a \star b := a + 3b$. (2 Punkte)
- b) Die Zahlen vom Typ `int` in Java mit dem Plus-Operator `+`. (2 Punkte)

3.2 Graphen (8 Punkte)

Gegeben sei folgender Graph $G = (E, K)$ mit $E := \{1, 2, 3, 4, 5\}$ und

K : 1 : [3, 4]
 2 : [1]
 3 : [4, 5]
 4 : [5]
 5 : [1]

a) Wie nennt man diese Darstellungsform? (1 Punkt)

b) Zeichnen Sie den zugehörigen Graphen. (1 Punkt)

c) Führen Sie den Floyd-Warshall-Algorithmus zur Berechnung der reflexiven transitiven Hülle für die durch den Graphen gegebene Relation durch. Tragen Sie dazu die Ursprungsrelation in das erste angegebene Feld ein und führen Sie dort den ersten Schritt des Algorithmus aus. Führen Sie dann den Algorithmus der Reihe nach für die Spalten 1,2,3,4 und 5 durch und tragen Sie die jeweiligen Zwischenergebnisse in die dafür vorgesehenen Felder ein. Markieren Sie jeweils die hinzugekommene(n) Kante(n) durch Einkreisen! (6 Punkte)

		1	2	3	4	5					
	1										
	2										
	3										
	4										
	5										

~>

		1	2	3	4	5					
	1										
	2										
	3										
	4										
	5										

~>

		1	2	3	4	5					
	1										
	2										
	3										
	4										
	5										

~>

		1	2	3	4	5					
	1										
	2										
	3										
	4										
	5										

~>

		1	2	3	4	5					
	1										
	2										
	3										
	4										
	5										

~>

		1	2	3	4	5					
	1										
	2										
	3										
	4										
	5										

4. Java (22 Punkte)

4.1 Suche in einer sortierten Liste (10 Punkte)

Schreiben Sie eine Methode

```
static boolean search(int[] list, int n)
```

die in einer *aufsteigend sortierten* Liste (hier als Array `list` übergeben) aus positiven ganzen Zahlen das Element `n` sucht. Falls `n` gefunden wird, soll `true` zurückgegeben werden; falls nicht, `false`.

Die Methode soll eine Theoretische Effizienz von $O(\log n)$ haben.

Hinweis: `list.length` liefert die Länge der Liste bzw. des Arrays `list`.

Implementierungen mit einer Theoretischen Effizienz von $O(n)$ und höher werden mit 0 Punkten bewertet.

4.2 Quelltext-Analyse (12 Punkte)

Gegeben sei folgendes Java-Programm:

```
class Program1{
    public static void main(String args[]) {
        int a1 = 0, a2 = 0, a3 = 0, a4 = 0, a5 = 0, a6 = 0;
        int number;
        for (int i = 0; i < 600; i++) {
            number = (i % 6) + 1;          /* 1 */
            switch(number) {
                case 1 : a1++;             /* 2 */
                case 2 : a2++;             /* 3 */
                case 3 : a3++;             /* 4 */
                case 4 : a4++;             /* 5 */
                case 5 : a5++;             /* 6 */
                case 6 : a6++;             /* 7 */
            }
        }
        int sum = a1 + a2 + a3 + a4 + a5 + a6;
        Out.println("Variable 1 : " + a1);
        Out.println("Variable 2 : " + a2);
        Out.println("Variable 3 : " + a3);
        Out.println("Variable 4 : " + a4);
        Out.println("Variable 5 : " + a5);
        Out.println("Variable 6 : " + a6);
        Out.println("Insgesamt : " + sum);
    }
}
```

a) Welche Ausgabe ist zu erwarten? (3,5 Punkte)

Variable 1: _____

Variable 2: _____

Variable 3: _____

Variable 4: _____

Variable 5: _____

Variable 6: _____

Insgesamt : _____

b) Begründen Sie Ihre Aussage. (4 Punkte)

c) Wie müsste das Programm verändert bzw. erweitert werden, damit es eine Simulation eines 6-seitigen Würfels mit gleicher Verteilung ergibt? Beschränken Sie sich dabei nur auf die Zeilen, die mit einem Kommentar `/**/` markiert sind! (3,5 Punkte)

```
/* 1 */ _____  
/* 2 */ _____  
/* 3 */ _____  
/* 4 */ _____  
/* 5 */ _____  
/* 6 */ _____  
/* 7 */ _____
```

d) Welche Funktion erfüllt `default` in einem `switch`-Block? (1 Punkt)

5. UML (8 Punkte)

Homer J. Simpson hat sich entschlossen zu studieren. Da er so vergesslich ist, musste ihm Marge ein Aktivitätsdiagramm seines Tages anfertigen, damit er sich auch an der Universität zurechtfindet.

Sein Tagesablauf sieht folgendermassen aus: Zunächst geht er zum Frühstückstisch und isst solange bis er nicht mehr hungrig ist. Danach hat er mehrere Möglichkeiten zur Universität von Springfield zu gelangen, die er allerdings vom Wetter abhängig macht: Wenn es nicht regnet, geht er zu Fuß; wenn es regnet, aber nicht glatt ist, fährt er mit dem Auto; wenn es regnet und glatt ist, fährt er mit dem Bus.

Wenn er zu Fuß geht, macht er zwischendurch einen Zwischenstop bei Moe's und kippt 2 Bierchen, macht sich dann direkt auf den Weg zur Universität; Wenn er mit dem Bus oder dem Auto fährt, fährt er direkt zur Universität.

An der Universität hört er Montags, Mittwochs und Donnerstags den ganzen Tag „Atomkraftwerksicherheit für Fortgeschrittene“ (abgekürzt: AKS II). Dienstags hat er Tutorium in „Effizient ohne Überarbeitung“ (abgekürzt: EÜ), Freitags hätte er „Körperliche Fitness für Anfänger“ (abgekürzt: KF I), wo er aber nur hinget, wenn eine Prüfung stattfindet. Nach den Veranstaltungen fährt er nach Hause.

Da Marge leider im Moment auf Schulung ist und Homer eine ganze Flasche Ketchup über sein altes UML-Aktivitätsdiagramm geschüttet hat, müssen Sie nun ein neues erstellen.

Hinweis: Sie können davon ausgehen, dass die Klassen `Homer`, `Terminkalender` & `Wetter` bereits existieren und mit den benötigten Methoden ausgestattet sind.

