



Name der Prüfung: Probeklausur - Informatik I

Datum: 31. Januar 2013

Bearbeitungszeit: 90 Minuten

Uhrzeit: 10:00 Uhr

Institut: Fachbereich Informatik
Computergrafik

Prüfer: Hendrik Lensch

Vom Prüfungsteilnehmer auszufüllen:

Name: _____ **Vorname:** _____

Matrikelnummer:

Studiengang: _____ **Abschluss:** _____

Datum, Unterschrift des Prüfungsteilnehmers

Mit meiner Unterschrift bestätige ich, dass ich prüfungsfähig bin und die auf der nachfolgenden Seite genannten Hinweise gelesen und verstanden habe. Weiterhin ist mir bekannt, dass die Prüfung nur gewertet werden kann, wenn ich mich zu der Prüfung ordnungsgemäß angemeldet habe und auf der Liste der angemeldeten Studierenden stehe. Nur mit Unterschrift findet eine Bewertung der Klausur statt.

Hinweise zur Prüfung:

- Die Hinweise zur Klausur befinden sich auf der folgenden Seite.

Viel Erfolg!

Aufg.	1	2	3	4	5	6	7	8	9	10	Summe
Pkt.	7	7	9	9	10	7	8	8	9	6	80
Err.											
Korr.											

Gesamtnote: _____

Hinweise zur Klausur:

- Erlaubte Hilfsmittel sind ausschließlich dokumentenechte Stifte in schwarz oder blau (insbesondere keine Bleistifte). Rot und grün sind nicht zu verwenden und den Korrektoren vorbehalten.
- Taschen, Rucksäcke o.Ä. dürfen nicht mit an den Platz genommen werden. Diese sind vorne im Hörsaal abzustellen. Handys, Smartphones, Laptops, usw. müssen während der Klausur ausgeschaltet in der Tasche, dem Rucksack, etc. sein und dürfen nicht mit an den Platz genommen werden.
- Halten Sie Ihren Studierendenausweis zur Identitätskontrolle bereit.
- Auf dem Titelblatt sind Vorname, Nachname, Matrikelnummer, Studiengang und angestrebter Abschluss anzugeben.
- Die Klausur besteht aus 10 Aufgaben bzw. 12 Blättern.
- Die ausgeteilten Klausurunterlagen sind von den Teilnehmerinnen/Teilnehmern auf Vollständigkeit zu überprüfen.
- Sofern die Teilnehmerinnen/ der Teilnehmer vor dem Abgabezeitpunkt die Klausurarbeit abgeschlossen haben, kann dieselbe abgegeben und der Klausorraum verlassen werden. In den letzten zehn Minuten vor dem Abgabezeitpunkt ist dies nicht mehr gestattet, um allen Teilnehmerinnen/Teilnehmer ein ungestörtes Arbeiten bis zum Schluss zu ermöglichen.
- Vor dem Verlassen des Klausorraumes sind die entsprechenden Unterlagen bei den Aufsichtsführenden abzugeben. Bei zeitweiligem Verlassen des Klausorraumes wird die Abwesenheitszeit im Protokoll festgehalten.
- Bei Täuschung und Täuschungsversuchen sowie Ordnungsverstößen wird die Klausur als „nicht ausreichend“ (5,0) gewertet. In besonders schweren Fällen, wie z.B. bei wiederholtem Täuschungsversuch oder dem unzulässigen Zusammenwirken mehrerer Personen oder dem Einsatz unzulässiger technischer Hilfsmittel kann der Prüfungsausschuss die/den Studierende(n) von der Erbringung weiterer Prüfungsleistungen ausschließen.
- Gesamtpunktzahl: 80.
- Mit 50% der maximal erreichbaren Punkte ist die Klausur sicher bestanden.
- Bei jeder Aufgabe sind die maximal erreichbaren Punkte angegeben.
- Ausschließlich die Antworten auf den ausgegebenen, zusammengehefteten Blättern sind für die Bewertung ausschlaggebend. Zusatzblätter sind bei der Klausuraufsicht erhältlich und werden angeheftet.
- Die Klausurergebnisse werden in das Hochschulportal eingestellt.
- Die geforderten (Programmier-)Aufgaben sind in Java-Code zu lösen, außer es ist in der Aufgabenstellung anders gefordert.

Aufgabe 1 (Datentypen und Konvertierung)

7 Punkte

Welche der unten folgenden Zuweisungen sind gültig?

```
int i = 5; char c = 'c'; float f = 2.3f; double d = 4; String s = "s";
```

- a) `i = i + 2.3f;`
- b) `i = i + f;`
- c) `d = d + (int) d;`
- d) `c = (char) ((int) f + (int) c);`
- e) `c = (char) (f + c);`
- f) `s = s + c;`
- g) `c = c + s;`

Aufgabe 2 (Zahlendarstellung)

7 Punkte

Gegeben sind die Zahlen `int a = 23` und `int b = 42`.

a) Wie werden `a`, `b` intern dargestellt?

`a` =

`b` =

b) Führen Sie mit der internen Darstellung die Rechnung $a - b = c$ durch. Geben Sie Zwischenschritte an.

c) Geben Sie die Darstellung von `c` und von `-c` mit Zwischenschritten an.

Aufgabe 3 (Matrizen)

9 Punkte

- a) Wie kann eine Matrix in Java repräsentiert werden?
- b) Schreiben Sie eine Methode `int[][] matrix(int[][] ary)` welche eine Matrix transponiert

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 0 & 1 & 2 \end{pmatrix}^T = \begin{pmatrix} 1 & 5 & 9 \\ 2 & 6 & 0 \\ 3 & 7 & 1 \\ 4 & 8 & 2 \end{pmatrix} \quad (1)$$

- c) Diskutieren Sie Laufzeit der Prozedur `int[][] matrix(int[][] ary)` für eine quadratische Matrix und geben Sie die asymptotische Laufzeit in $O(\dots)$ notation an.

Aufgabe 4 (Listen)

9 Punkte

Gegeben sind die Klassen `ListElement.java` und `LineareListe.java`:

```
1  [ ListElement.java ]
2
3  public class ListElement{
4      int first;
5      ListElement next;
6
7      public ListElement(int first , ListElement next){
8          this.first = first;
9          this.next = next;
10     }
11 }

1
2  [ LineareListe.java ]
3
4  public class LineareListe{
5      ListElement head;
6
7      LineareListe(){
8          this.head = null;
9      }
10 }
```

- a) Schreiben Sie eine Methode `void printListReverse( )` in der Klasse `LineareListe` welche eine Liste in umgekehrter Reihenfolge zurück gibt. Dabei soll nur ein Mal über die Liste iteriert werden.
- b) Was bewirken die folgenden Codezeilen auf einer Liste `x` angewendet?

```
1  public LineareListe mysterio()
2  {
3      if (this.head == null){
4          return null;
5      }
6      ListElement x = this.head.next;
7      for (int i = 0; x != null && i < 4; x = x.next){
8          this.head = x;
9      }
10     return this;
11 }
```

Erklären Sie Ihre Lösung!

- c) Ist die Methode `mysterio()` deterministisch, determinierend, terminierend?

Aufgabe 5 (Bäume)

10 Punkte

- a) Definieren Sie die Klassen für einen binären Baum.
- b) Nennen Sie die zentrale Eigenschaft binärer Suchbäume und zeichnen Sie ein Beispiel mit mindestens 6 Knoten.
- c) Schreiben Sie eine Methode `int minimum(btree t)`, welche das Minimum in einem binären Suchbaum sucht und ausgibt.
- d) Beschreiben Sie mit (Pseudo)-Code die binäre Suche auf einem binären Suchbaum und führen Sie diese für ein (nicht triviales) Beispiel ihrer Wahl durch.
- e) Diskutieren Sie Laufzeit der binären Suche und geben Sie die asymptotische Laufzeit in $O(\dots)$ notation an.

Aufgabe 6 (Heaps)

7 Punkte

Gegeben sei die Zahlenfolge 23, 1, 6, 19, 14, 18, 8, 24, 15

- a) Fügen Sie die Zahlen der gegebenen Zahlenfolge nacheinander in einen anfangs leeren Heap ein.
- b) Erklären Sie in wenigen Sätzen wie man mit Hilfe eines Heaps Zahlen sortiert (HeapSort-Algorithmus).
- c) Führen Sie den HeapSort-Algorithmus an dem obigen Heap durch, allerdings nur für die ersten 4 Zahlen. Achten Sie darauf, dass Ihr Vorgehen nachvollziehbar ist.

Aufgabe 7 (Kurzfragen)

8 Punkte

Antworten Sie auf die folgenden Fragen einfach mit **ja** oder **nein**, d.h. ohne Begründung. (Bewertung: bf 1 Punkt für jede richtige Antwort; -1 Punkt für eine falsche Antwort; minimal 0 Punkte für die gesamte Aufgabe)

1. Jede **while**-Schleife kann auch als rekursive Methode mit Hilfe von **if**-Anweisungen programmiert werden. (ja / nein)
2. Beim **Quicksort**-Verfahren ist der Zeitaufwand für die Sortierung abhängig von der Wahl des Pivot-Elements. (ja / nein)
3. Bei der Klassenvereinbarung in Java muss der Programmierer mindestens einen Konstruktor selbst implementieren. (ja / nein)
4. Ein Binärbaum mit $n \geq 1$ Knoten besitzt genau $n - 1$ Kanten. (ja / nein)
5. In Java kann jede **while**-Schleife durch eine **for**-Schleife implementiert werden. (ja / nein)
6. In Java können die Zeilen von zweidimensionalen Arrays unterschiedliche Länge haben. (ja / nein)
7. Stacks können nicht mit Hilfe von Arrays implementiert werden. (ja / nein)
8. $\log_{100} n = O(\log_2 n)$ (ja / nein)

Aufgabe 8 (Auswahl in Arrays)

8 Punkte

Gegeben sei eine zufällige Zahlenfolge `int[] arr`.

- a) Schreiben Sie eine Methode `int countNumber( int[] arr, int num )`, die zählt, wie häufig die Zahl `num` in dem Array vorkommt.
- b) Schreiben Sie eine Methode `int[] findNumberPos( int[] arr, int num )`, die ein Array mit den Positionen der Zahl `num` in dem Feld zurückgibt.

Aufgabe 9 (Rekursion vs. Iteration)

9 Punkte

Das Skalarprodukt $s(x, y)$ von Vektoren $x, y \in \mathcal{R}^n$ ist definiert durch $s(x, y) = \sum_{k=1}^n x_k y_k$. In Java wird ein solcher reeller Vektor als ein Array mit n Elementen vom Typ `double` implementiert.

- a) Schreiben Sie eine *rekursive* Methode zur Berechnung des Skalarproduktes, welche das Ergebnis als `double` zurückliefert. Sie können davon ausgehen, dass beide Arrays gleiche Länge haben.
- b) Von welchem Typ ist die Rekursion?
- c) Lässt sich die vorliegende Rekursion auch in eine iterative Methode umwandeln? Begründen Sie Ihre Antwort kurz.

Aufgabe 10 (Klassen und Objekte)

6 Punkte

Gegeben sei folgende einfache Klasse zur Speicherung von Daten der Spieler eines Tennisturniers:

```
1 public class TurnierSpieler {
2     public String name;
3     public int    alter;
4     public int    altersdifferenz(int alter) {
5         return Math.abs(this.alter - alter);
6     }
7 }
```

a) Erläutern Sie den Aufbau der Klasse graphisch (in UML). **(2 Punkte)**

b) Was bewirken die folgenden Anweisungen?

```
1 TurnierSpieler maier;
2 maier = new TurnierSpieler();
```

Warum benötigt man die zweite Anweisung (Zeile 2) überhaupt? **(2 Punkte)**

c) Schreiben Sie eine Codezeile, welche dem Spieler einen Namen zuweist. **(2 Punkte)**