Informatik I **Wintersemester 2018/19**
Prof. Dr. Carsten Damm Georg-August-Universität Göttingen
Dr. Henrik Brosenne Institut für Informatik

Übungsblatt 09

E-Learning

Absolvieren Sie die Tests bis Di., 08.01.2019, 23:55 Uhr.

Die Tests sind in der Stud.IP-Veranstaltung *Informatik I* unter *Lernmodule* hinterlegt.

Sie können einen Test **nur einmal durchlaufen**. Sobald Sie einen Test starten steht Ihnen nur eine **begrenzte Zeit** zu Verfügung, um den Test zu bearbeiten.

Alle Punkte, die Sie beim Test erreichen, werden ihnen angerechnet.

ILIAS – 20 Punkte

BubbleSort, SelectionSort

Absolvieren Sie den Test *Informatik I - ILIAS 09 Teil 1.*
(20 Punkte)

Hinweis

Wenn Sie den Test einmal vollständig durchlaufen haben kommen Sie auf die Seite *Testergebnisse*. Starten Sie den Test erneut aus Stud.IP, ist jetzt auch eine Schaltfläche *Testergebnisse anzeigen* vorhanden, die auf diese Seite führt.

Auf der Seite *Testergebnisse* können Sie sich unter *Übersicht der Testdurchläufe* zu jedem Testdurchlauf *Details anzeigen* lassen.

In der Auflistung der Aufgaben führt der Titel einer Aufgabe zu einer **Musterlösung** für die jeweilige Aufgabe.

ILIAS 4-Minuten-Aufgaben – 12 Punkte

Absolvieren Sie die Tests *Informatik I - ILIAS 09 Teil 2, Teil 3* und *Teil 4.*
(12 Punkte)

Als Vorbereitung für ähnliche Aufgaben in der E-Klausur sollten Sie den Test *Informatik I - ILIAS 09 Teil 5* absolvieren.
(0 Punkte)

LON-CAPA – 12 Punkte

Verkettete Listen, Binäre Bäume

Absolvieren Sie im Test *Informatik I - LON-CAPA* die *Übung 09.*
(12 Punkte)

Übung

Abgabe bis Di., 08.01.2019, 18 Uhr.

Werfen Sie Ihre Lösung in den Zettelkästen Ihrer Gruppenübung. Für die Übungen im Nordbereich stehen die Zettelkästen im Sockelgeschoß (Ebene -1) **oder** auf dem Flur vor dem Seminarraum auf Ebene 0 des Instituts für Informatik.

Wenn Ihre Übung im Südbereich stattfindet, klären Sie mit Ihrem Tutor wo die Lösungen abzugeben sind.

Achten Sie darauf, dass Ihr **Name**, Ihre **Gruppe** und Ihre **Matrikelnummer** auf **jedem** Blatt stehen!

Falls Ihre Lösung mehrere Blätter umfasst, heften Sie diese bitte zusammen.

Aufgabe 1 – 21 Punkte

Binäre Suche

Viele Funktionen lassen sich relativ einfach direkt berechnen, die jeweilige Umkehrfunktion aber nicht. Ein Beispiel dafür ist die Quadratfunktion $f(x) = x^2$ und deren Umkehrfunktion, die Quadratwurzel $f^{-1}(x) = \sqrt{x}$.

Für streng monoton steigende Funktionen, d.h. einen Funktion f bei der aus $x > y$ folgt, dass $f(x) > f(y)$ gilt, kann die Idee der binären Suche zur Bestimmung der Umkehrfunktion verwendet werden. D.h. $f^{-1}(y)$ wird bestimmt, indem man ein x mit $f(x) = y$ sucht. Betrachtet man während der Suchen einen Kandidaten x mit $f(x) \neq y$, kann man folgern, weil f streng monoton steigt, ob der gesuchte Wert kleiner oder größer x ist. Dann braucht man nur noch in diesem Bereich weitersuchen.

1. Die Quadratwurzel kann in der Regel bei der Berechnung in Java mit einem Gleitkommatyp nicht exakt bestimmt werden.

   Formulieren und begründen Sie ein sinnvolles Kriterium, wann das Ergebnis genau genug ist.
   (8 Punkte)

2. Geben Sie den Java-Code zweier `double`-Methoden an. Die erste Methode bekommt genau einen `double`-Wert übergeben und bestimmt dessen Quadratwurzel, indem die zweite Methode, mit allen erforderlichen Argumenten, aufgerufen wird. Die zweite Methode führt rekursiv eine binäre Suche auf den Funktionswerten der Quadratfunktion durch.
   (13 Punkte)

Hinweis

- Für negative Argumente kann z.B. `Double.NaN` zurückgeliefert werden.
- Für $y \leq 1$ gilt $0 < \sqrt{y} \leq 1$.
- Für $y > 1$ gilt $1 < \sqrt{y} < y$.

Praktische Übung

Abgabe der Prüfsumme bis Di., 08.01.2019, 23:55 Uhr.
Testat von Di., 15.01.2019., 8-10 Uhr bis Fr., 18.01.2019, 18-20 Uhr.

Hilfe zum Bearbeiten der praktischen Übungen können Sie grundsätzlich jeden Tag in den Rechnerübungen bekommen, insbesondere in den Rechnerübungen am **Montag**, in denen **keine Testate** stattfinden.

Hinweise zu den praktischen Übungen, insbesondere zur **Abgabe der Prüfsumme** und zur **Durchführung der Testate**, sind in der Stud.IP-Veranstaltung *Informatik I* unter *Dateien→Übungsblätter* hinterlegt.

Aufgabe 1 – 10 Punkte

Mehrfache Rekursion

Programmieren Sie eine zweifach rekursive Methode `int summe(int[] v, int von, int bis)`, die die Summe der Werte im Feld `v` von Index `von` bis Index `bis` berechnet. Ist der Basisfall erreicht wird der zugehörige Wert zurückgeliefert, sonst wird der Indexbereich in zwei möglichst gleich große Teilbereiche zerlegt, die Funktion ruft sich selbst für jeden dieser beiden Teilbereiche auf und verarbeitet die Ergebnisse.

Programmieren Sie eine Applikation, die eine Folge ganzer Zahlen auf der Kommandozeile übergeben bekommt, diese in einem Feld speichert, das komplette Feld mit der Methode `summe` addiert und das Ergebnis ausgibt.

Versehen Sie den Quelltext mit ausführlichen Kommentaren, die den Programmablauf erläutern.
(10 Punkte)

Hinweis
`https://www.stud.informatik.uni-goettingen.de/info1/java/Summe.java`

Aufgabe 2 – 25 Punkte

Geld

Betrachten Sie die folgende Klasse `Geld`.

```
public class Geld {
  private int euro, cent;

  // Konstruktor
  public Geld(int e, int c) {
    euro = e;
    cent = c;
  }
}
```

Ergänzen Sie die Klasse `Geld` auf folgende Weise.

1. Erweitern Sie den Konstruktor, sodass sichergestellt ist, dass die Variable `cent` nur Beträge kleiner 100 enthält, auch wenn Argumente mit Cent-Beträgen größer 99 übergeben werden.

   Lagern Sie die entsprechende Funktionalität in eine `private` Methode aus, die vom Konstruktor benutzt wird.
   (4 Punkte)

   Hinweis
   Sie können davon ausgehen, dass der Konstruktor nur mit nicht negativen Werten aufgerufen wird.

2. Erweitern Sie die Klasse `Geld` um eine Methode `boolean equals(Geld g)`. Für zwei Objekte `betrag1` und `betrag2` der Klasse `Geld` liefert `betrag1.equals(betrag2)` den Wert `true` zurück, wenn `betrag1` und `betrag2` denselben Geldbetrag darstellen. Andernfalls wird `false` zurück geliefert.

   Zwei Geldbeträge sind gleich wenn die Werte für Euro und Cents jeweils gleich sind.
   (4 Punkte)

3. Schreiben Sie eine Methode `public void add(Geld g)`. Für zwei Objekte `betrag1` und `betrag2` der Klasse `Geld` soll beim Aufruf von `betrag1.add(betrag2)` der Wert von `betrag2` zu `betrag1` addiert werden.

   Beachten Sie, dass zwei Geldsummen zusammen mehr als 99 Cent haben können. Stellen Sie sicher, dass die Variable `cent` nur Beträge kleiner 100 enthält. Benutzen Sie dazu die Methode aus Aufgabenteil 1.
   (4 Punkte)

4. Überschreiben Sie die Methode `public String toString()`. Es soll einen String zurück geliefert werden, der den Geldbetrag in Euro und Cent repräsentiert. Z.B. soll für einen Betrag mit 7 Euro und 41 Cent die Zeichenkette `7,41 Euro` zurück geliefert werden.
   (4 Punkte)

   Hinweis. Beachten Sie, dass es auch cent-Werte zwischen 0 und 9 gibt.

5. Erweitern Sie die Klasse `Geld` um ein Test-Unit, dass die Klasse ausgiebig testet.
   (4 Punkte)

6. Versehen Sie das Programm mit ausführlichen Kommentaren, die den Programmablauf erläutern.
   (5 Punkte)