Institut für Informatik
Prof. Dr.-Ing. Elke Pulvermüller
Dipl.-Systemwiss. Mathias Menninghaus

Universität Osnabrück, 04.04.2017
http://www-lehre.inf.uos.de/~binf
Testat bis 12.04.2017, 14:00 Uhr

Übungen zu Informatik B

Sommersemester 2017

Blatt 1

Übungsbetrieb

Dienstags in der Vorlesung wird ein Aufgabenblatt verteilt, das bis einschließlich Mittwoch (letztes Testat um 13:30 Uhr) der darauffolgenden Woche zu bearbeiten ist. Die Aufgabenblätter finden sich auch auf der Veranstaltungswebseite (`http://www-lehre.inf.uos.de/~binf`).

Die Übungen finden Donnerstags zu zwei unterschiedlichen Terminen (08:00 - 10:00 Uhr, Raum 66/E33 und 10:00 - 12:00 Uhr, Raum 66/E34) mit dem gleichen Inhalt statt. In den Übungen werden das neue Aufgabenblatt besprochen und die Lösungen zum alten Aufgabenblatt diskutiert.

Testatbetrieb

Begleitend zur Veranstaltung finden wöchentliche, 30-minütige Testate bei den Tutoren der Veranstaltung statt. Das aktuelle Aufgabenblatt ist bis zum Testattermin zu bearbeiten und dem jeweiligen Tutor zum entsprechenden Zeitpunkt in Raum 93/E10 vorzulegen sowie in das unten genannte Web-System hochzuladen.

Achten Sie bei der Bearbeitung der Aufgaben darauf, diese *vor* dem Testat einmal auf einem Uni-Rechner getestet zu haben, beispielsweise in Raum 93/E11, der zum Bearbeiten der Aufgaben zur Verfügung steht.

Die Testate erfolgen in Zweierteams, zu denen man sich von Mittwoch, 05.04. 14:00 Uhr, bis Freitag, 07.04. 12:00 Uhr, unter `https://binf-testate.informatik.uni-osnabrueck.de` mit dem RZ-Login anmelden kann.

Fragen, Antworten und Ankündigungen

Um Fragen oder Probleme untereinander und mit den Tutoren diskutieren zu können, steht auf Stud.IP ein Forum zur Verfügung.

Scheinvergabe

Um die Zulassung zur Klausur zu erhalten, müssen alle bis auf eins der ausgegebenen Übungsblätter erfolgreich bearbeitet (mindestens 50% der Punkte) und dem Tutor präsentiert werden. Zum Abschluss der Veranstaltung entscheidet eine Klausur über die Scheinvergabe.

Bearbeitung der Aufgaben

Kommentieren Sie Ihren gesamten Quellcode `javadoc`-konform und achten Sie auf geeignete Klassen-, Methoden- und Variablennamen. Bei der Formatierung des Quellcodes können Sie sich nach den Java-Code-Conventions von Oracle (Sun) richten, den Link finden Sie auf der Übungsseite. Machen Sie außerdem von der Java API Gebrauch. Bitte lösen Sie auch alle übrigen Aufgaben immer **schriftlich**.

Aufgabe 1.1: Sichtbarkeit (20 Punkte)

Betrachten Sie die Datei `Fraglich.java`. Geben Sie an den markierten Stellen 1 bis 15 die Werte der Variablen `a`, `b` und `c` an. Denken Sie bei Arrays daran, den Inhalt der einzelnen Einträge anzugeben. Erklären Sie Ihrem Tutor **schriftlich** die Begriffe *call-by-value* und *call-by-reference* in Java.

Aufgabe 1.2: Fibonacci (30 Punkte)

Die Fibonacci Zahl $f(n)$ zu einer ganzen Zahl $n \geq 2$ sei definiert durch

$$f_n = f_{n-1} + f_{n-2} \text{ für } n \geq 2 \text{ mit } f_0 = 0 \text{ und } f_1 = 1$$

Implementieren Sie die Klasse `Fibonacci`. Ein Objekt der Klasse `Fibonacci` besitzt die Instanzmethode `next()`, die, beginnend mit $f(2)$, immer die nächste Zahl der Fibonacci-Folge zurück gibt. Beim ersten Aufruf von `next` wird also `1`, beim zweiten `2`, beim dritten `3`, dann `5`, `8` usw. zurück gegeben.

Implementieren Sie anschließend die Klasse `FibonacciPrint`. Wenn die Klasse über die Kommandozeile aufgerufen wird, soll ihr als Parameter genau eine ganze Zahl *n* übergeben werden. Diese wird in Java automatisch in das `String[]` der `main` Methode übergeben. Von `FibonacciPrint` sollen alle Fibonacci-Zahlen von $f(0)$ bis $f(n)$ wie folgt formatiert auf der Kommandozeile ausgegeben werden:

```
| n |  f(n)  |
+---+--------+
|  0|       0|
```

Benutzen Sie für die Standard Ein- und Ausgabe Java-Bordmittel und nicht die `AlgoTools.jar` aus Informatik A. Einen `String s` aus dem der `main`-Methode übergebenen `String`-Array können Sie mit `Integer.parseInt()` in ein `int` umwandeln. Ignorieren Sie dabei Exceptions vom Typ `NumberFormatException`. Achten Sie auf die Behandlung von weiteren Fehlern.

Aufgabe 1.3: Fraction (30 Punkte)

Implementieren Sie die Klasse `Fraction` mit den beiden Instanzvariablen `numerator` und `denominator` zur Repräsentation eines Bruches.

Es soll zwei Konstruktoren geben, die den Bruch bei der Instanziierung kürzen. Einer, der eine ganze Zahl annimmt und den Nenner auf 1 setzt und ein zweiter, der mit Nenner und Zähler aufgerufen wird. Verketten Sie die Konstruktoren.

Implementieren Sie zusätzlich die folgenden Instanzmethoden, die bei der Ausführung der entsprechenden numerischen Operation immer eine neue `Fraction` erzeugen.

- `Fraction multiply(int factor)`
- `Fraction multiply(Fraction factor)`

- `Fraction divide(Fraction divisor)`
- `Fraction multiply(Fraction... factors)`

Dabei erwartet die Methode `multiply(Fraction... factors)` eine variable Liste von `Fraction` Instanzen. D.h. sie kann mit beliebig vielen Argumenten vom Typ `Fraction` aufgerufen werden. Machen Sie es möglich, eine Instanz mit der Methode `String toString()` als `String` (z.B.`1/4`) auszugeben.

Testen Sie Ihre neue Klasse mit Hilfe einer separaten Testklasse. Das Testprogramm soll einige Instanzen der Klasse `Fraction` erzeugen und jede der programmierten Operationen mindestens einmal testen. Es soll automatisiert eine Ausgabe erfolgen, ob der jeweilige Test erfolgreich war oder nicht.

Aufgabe 1.4: Fragen (20 Punkte)

Beantworten Sie Ihrer Tutorin / Ihrem Tutor Fragen zur Veranstaltung Informatik B.