Institut für Informatik
Prof. Dr.-Ing. Elke Pulvermüller
Lars Huning, M.Sc.

Universität Osnabrück, 12.04.2021
Testat bis 21.04.2021, 18:00 Uhr

Übungen zu Einführung in die Software-Entwicklung

Sommersemester 2021

Blatt 1

Übungsbetrieb

Dienstags nach der Vorlesung wird ein Aufgabenblatt auf der Veranstaltungsseite in Stud.IP hochgeladen, das bis einschließlich Mittwoch (letztes Testat um 17:30 Uhr) der darauffolgenden Woche zu bearbeiten ist.

Zur Vorbereitung auf das jeweilige Aufgabenblatt werden ebenfalls Dienstag nach der Vorlesung mehrere Übungsvideos hochgeladen. Diese dienen als praktische Ergänzung zum Vorlesungsinhalt und nehmen in der Regel Bezug auf die konkreten Übungsaufgaben die in der Woche anstehen.

Zusätzlich haben Sie die Gelegenheit Donnerstags von 8:30-10:00 Uhr und von 10:30-12:00 Uhr dem Übungsleiter Fragen zum Aufgabenblatt oder generelle Verständnisfragen zum Vorlesungsinhalt zu stellen. Diese Fragerunden finden rein online über das Videokonferenztool BigBlueButton statt. Um an der Konferenz teilzunehmen, klicken Sie in der Veranstaltungseite in Stud.IP auf den Reiter *Meetings* und wählen Sie das Meeting *Übung* aus. Achtung: Im Gegensatz zu den vorherigen Jahren wird an den Donnerstagsterminen kein Inhalt proaktiv vermittelt, es werden nur konkrete Fragen beantwortet. Für die proaktive Wissensvermittlung sind die Übungsvideos gedacht. „Passives“ Mithören bei den Übungen ist also nur begrenzt von Nutzen.

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 in einer Videokonferenz zum entsprechenden Zeitpunkt zu erklären,[1] sowie in das unten genannte Web-System hochzuladen. Beachten Sie dabei, dass das pro Aufgabenblatt nur eine Datei hochgeladen werden kann. Ein Neu-Upload ersetzt die vorherige Datei. Zur Bewertung wird immer die zuletzt hochgeladene Datei während des Testats genutzt.

Achten Sie bei der Bearbeitung der Aufgaben darauf, diese *vor* dem Testat einmal auf einem Uni-Rechner getestet zu haben, beispielsweise auf dem Rechner cip20. Näheres hierzu finden Sie im Abschnitt *Setup* weiter unten auf dem Aufgabenblatt.

Die Testate erfolgen in Zweierteams, zu denen man sich von Dienstag, 13.04. 14:00 Uhr, bis Donnerstag, 15.04. 18:00 Uhr, unter `https://binf-testate.informatik.uni-osnabrueck.de` mit dem RZ-Login anmelden kann.

[1] Für die Kommunikation mit Ihrem Tutor siehe den Abschnitt *Kommunikation* weiter unten auf diesem Aufgabenblatt

Fragen, Antworten und Ankündigungen

Um Fragen oder Probleme untereinander und mit den Tutoren diskutieren zu können, steht auf Stud.IP ein Forum zur Verfügung. Hier können Sie schriftlich Fragen stellen, falls Sie an den Übungsterminen keine Zeit haben oder diese überlastet gewesen sind.

ECTS Vergabe

Um die Zulassung zur Prüfung 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 Prüfung über die Vergabe der ECTS.

Bearbeitung der Aufgaben

Alle Programmieraufgaben sind in der Programmiersprache *Java* abzugeben. Nutzen Sie dabei Java Version 11, wie sie auch auf den Uni-Rechnern installiert ist. Eine Einführung in Java ist Teil der Übungsvideos, sowie Aufgabe 1.5. 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 (`https://www.oracle.com/technetwork/java/codeconvtoc-136057.html`). Machen Sie außerdem von der Java API Gebrauch. Bitte lösen Sie auch alle übrigen Aufgaben immer **schriftlich**.

Krankheitsfälle

Sollten Sie aus gesundheitlichen Gründen nicht an einem Testat teilnehmen können, müssen Sie bis spätestens zum nächsten Testat Ihrem Tutor ein ärztliches Attest einscannen und per E-Mail zukommen lassen. Sollte solch ein Attest ausbleiben, wird Ihr fehlendes Erscheinen als 0 Punkte für dieses Testat gewertet. Ab dem Vierten verpassten Testat gelten für ein Attest dieselben Anforderungen wie für einen Prüfungsrücktritt. Die Regelungen hierzu können Sie unter folgendem Link einsehen: `https://www.mathinf.uni-osnabrueck.de/fachbereich/pruefungsamt/hinweise_zum_pruefungsruecktritt.html`

Beachten Sie, dass ihr Testatpartner auch bei einer Krankschreibung von Ihnen wie gewöhnlich am Testat teilnehmen muss. Er oder sie benötigt weiterhin mindestens 50% der Punkte auf dem Aufgabenblatt zum Bestehen des Testats.

Alte Prüfungszulassungen

Falls Sie bereits in einem vorherigen Semester die Prüfungszulassung erworben haben, so bleibt diese grundsätzlich bestehen. Sie müssen also nicht nocheinmal am Testatbetrieb teilnehmen. Sollten Sie freiwillig nocheinmal am Testatbetrieb teilnehmen wollen, so erlischt Ihre alte Prüfungszulassung ab dem zweiten Testat.

Aufgabe 1.1: Kommunikation (0 Punkte)

Die Veranstaltung wird rein digital durchgeführt werden. Von Seiten des Übungs- und Testatssystems bestehen folgende Kommunikationsmöglichkeiten mit dem Übungsleiter und den Tutoren:

- Teilnahme an einer der beiden Fragerunden Do 8:30-10:00 Uhr oder Do 10:30-12:00 Uhr (Zugriff bereits oben beschrieben)
- Nutzung des Forums auf der Veranstaltungswebsite in Stud.IP
- E-Mail an den Übungsleiter / Tutoren (Diese Option ist nicht für fachliche Fragen gedacht, sondern für private organisatorische Angelegenheiten, die vertraulich behandelt werden sollen)
- Das Testat selber findet mit dem Videokonferenzsystem BigBlueButton (BBB) statt, welches Sie über den Reiter *Meetings* in Stud.IP auf der Veranstaltungswebsite erreichen können. Bis zum ersten Testat (d.h. spätestens bis zum 19.04. 8:00 Uhr), werden Sie hier ein privates Meeting für sich und Ihren Testatpartner finden. Sie können dieses Meeting anhand Ihrer beiden RZ-Kürzel erkennen. Zu der Testatzeit betreten Sie bitte beide dieses Meeting, in welches sich Ihr Tutor ebenfalls einloggen wird. Gerade am Anfang der Veranstaltung empfiehlt es sich, dass Sie bereits einige Minuten vorab im Meeting sind und die Funktionen von BigBlueButton testen (Audio, Video, Screen Sharing, Präsentation der Aufgaben).
- Die BBB Meetings für die Testate sind auch außerhalb dieser Zeiten nutzbar. Sie können diese z.B. Nutzen um während der Woche gemeinsam an den Übungsaufgaben zu arbeiten. Dabei ist zu Beachten, dass die Meetings nicht mit einem Passwort geschützt sind. D.h. jeder Teilnehmer der Stud.IP Veranstaltung kann auf jedes Meeting zugreifen. Das unerwünschte Teilnehmen in fremden Meetings anderer Testatgruppen ist untersagt. Sollten Sie einen unerwünschten Teilnehmer in Ihrem Meeting haben, der auch nach Aufforderung von Ihnen das Meeting nicht verlässt, können Sie diese Person dem Übungsleiter melden. Da in BBB jeder Teilnehmer mit seinem RZ-Kürzel angemeldet ist eine eindeutige Identifikation des Störenfrieds möglich.
- Die Uni bietet eine Messenger Alternative mit dem Namen *Element* an. Zugriff erlangen Sie unter `https://chat.virtuos.uni-osnabrueck.de` im Browser oder indem Sie die App unter `https://element.io/` runterladen. Für die App müssen Sie noch im Login-Screen den Server auf https://matrix.virtuos.uos.de ändern. Der Login erfolgt bei beiden Varianten mithilfe des RZ-Logins. Ihren Testatpartner finden Sie, wenn Sie in der Suche „@<RZ-Kürzel>:chat.virtuos.uni-osnabrueck.de“ eingeben, also z.B. @lhuning:chat.virtuos.uni-osnabrueck.de. Nachdem die Testate eingetragen sind, wird Ihr Tutor auch einen Element-Raum für Ihr Testat eröffnen. Dies ermöglicht Ihnen im Falle von Verbindungsproblemen in BBB während des Testats mit Ihrem Tutor über mobile Daten zu kommunizieren. Das RZ-Kürzel Ihres Tutors erfahren Sie, indem Sie das Studip Profil ihres Tutors öffnen und seine Mailadresse anschauen. Diese hat die Form „<RZ-Kürzel>@uni-osnabrueck.de“.
- Telefon: Für den Fall von Verbindungsproblemen während des Testats empfiehlt es sich Telefonnummern mit Ihrem Tutor auszutauschen.

Nachdem Ihr Testat eingetragen ist, sollte Ihr erster Schritt darin bestehen mit Ihrem Tutor zu kommunizieren. Schreiben Sie Ihrem Tutor eine Mail und greifen Sie auf Element zu um initial zwei Kommunikationswege mit Ihrem Tutor herzustellen. Die bevorzugte Kommunikationsvariante klären Sie dann indviduell mit Ihrem jeweiligen Tutor.

Aufgabe 1.2: Setup (0 Punkte)

Um eine möglichst hohe Vergleichbarkeit der Übungsaufgaben innerhalb der Veranstaltung zu gewährleisten, sollen während des Testats alle Programme auf dem Betriebssystem Ubuntu 18.04 (Desktop Variante) ausgeführt werden[2]. Es gibt zwei grundsätzliche Möglichkeiten wie Sie dazu vorgehen können:

1. Installation von Ubuntu auf einem Computer den Sie besitzen.
2. Nutzung von Ubuntu auf einem der Computer der Universität via Fernzugriff.

   Achtung: Für Aufgabe 1.3 müssen Sie diese Möglichkeit Ihrem Tutor präsentieren können. Für alle anderen Aufgaben der Veranstaltung ist es Ihnen freigestellt ob Sie Ihre Ergebnisse auf Ihrem eigenen Rechner oder einem der Uni-Rechner per Fernzugriff präsentieren.

Im Folgenden werden beide Möglichkeiten noch näher ausgeführt.

Installation von Ubuntu auf einem eigenen Rechner

Es gibt mehrere Möglichkeiten, wie Sie Ubuntu auf Ihrem Rechner installieren können:

- *Installation als einziges Betriebssystem:* Unter `https://ubuntu.com/tutorials/tutorial-install-ubuntu-desktop#1-overview` finden Sie eine Anleitung wie Sie Ubuntu von einer DVD oder einem boot-fähigem USB-Stick installieren können. Für die Erstellung solch einer DVD/eines USB-Sticks, finden Sie hier hilfreiche Tutorials: `https://ubuntu.com/tutorials?topic=desktop`. Das Betriebssytem können Sie z.B. hier runterladen: `http://releases.ubuntu.com/18.04.4/`
- *Installation als Dual-Boot Betriebssystem:* Eine Anleitung können Sie u.a. hier finden: `https://wiki.ubuntuusers.de/Dualboot/`
- *Installation innerhalb einer VM:*

  Schritt 1: Installation einer *Virtual Machine* (VM): Hierfür können Sie eine beliebige VM verwenden. Eine Option ist *VirtualBox*, welches Sie auf einem Linux-System nach dem folgenden Link installieren können: `https://wiki.ubuntuusers.de/VirtualBox/Installation/`. Auf einem Windows-System können Sie VirtualBox unter folgendem Link herunterladen und anschließend installieren: `https://www.virtualbox.org/wiki/Downloads`.

  Schritt 2: Downloaden eines ISO-Files mit Ubuntu 18.04: `http://releases.ubuntu.com/18.04.4/`

  Schritt 3: Ubuntu in der VM installieren und konfigurieren: `https://www.heise.de/tipps-tricks/Ubuntu-in-VirtualBox-nutzen-so-klappt-s-4203333.html`

[2]Version 18.04 wurde deswegen gewählt, da diese auch auf den Rechnern der Universität installiert ist und via Fernzugriff erreicht werden kann. Mittlerweile existiert die neuere Version 20.04, welche bzgl. des Inhaltes dieser Veranstaltung keine nennenswerte Unterschiede aufweisen sollte. Sie können also auch mit Version 20.04 arbeiten. In diesem Fall empfiehlt es sich aber vor dem Testat Ihre Programme einmal auf einem Cip-Rechner ausgeführt zu haben (siehe Aufgabe 1.3)

Zugriff auf die Rechner der Universität via Fernzugriff

Für das Arbeiten auf einem Rechner der Universität sind in dieser Veranstaltung vor allem drei Arten des Zugriffs relevant:

- Zugriff auf den entfernten Uni-Rechner via SSH: Unter Linux und MacOS Betriebssystemen können Sie hierfür den Terminalbefehl *ssh* verwenden. Unter Windows bietet sich die Verwendung des Programms *Putty* an. Bei erfolgreicher Verbindung erhalten Sie Zugriff auf ein Terminal auf dem Uni-Rechner.
- Zugriff auf den entfernten Uni-Rechner via SSH und X-Forwarding: Hierbei handelt es sich um eine modifzierte Variante des vorherigen Zugriffs. Während der vorherige Zugriff nur das Arbeiten auf dem Terminal erlaubt hat, erlaubt diese Variante es zusätzlich das *Graphical User Interface* (GUI) eines Programmes auf dem entfernten Rechner auf Ihrem Heimrechner anzuzeigen. So können Sie z.B. mit einem GUI-basiertem Texteditor, wie z.B. *Gedit*, auf dem Uni-Rechner arbeiten. Diese Zugriffsart ist ebenfalls mit dem *ssh* Befehl möglich, allerdings müssen Sie hierbei die Option *-X* zusätzlich angeben.

  Falls Sie Windows und Putty verwenden und z.B. Gedit bei Ihnen trotz X-Forwarding nicht auf dem lokalen Rechner angezeigt wird, können Sie dies z.B. durch die zusätzliche Installation von *Xming* nach der folgenen Anleitung lösen: `https://stackoverflow.com/questions/34932495/forward-x11-failed-network-error-connection-refused/44575703#44575703`. Xming können Sie z.B. hier runterladen: `https://sourceforge.net/projects/xming/files/Xming/6.9.0.31/Xming-6-9-0-31-setup.exe/download`.
- Dateienaustausch zwischen dem entfernten Uni-Rechner und Ihrem eigenen Rechner: Hierfür können Sie unter Linux und MacOS den Terminalbefehl *scp* verwenden. Unter Windows bietet sich das Programm *Filezilla* an.

Die entfernten Uni-Rechner die Ihnen zum Arbeiten zur Verfügung stehen heißen *cipX*, wobei *X* eine Zahl aus dem Intervalle [3-23] sein muss. Als Benutzername dient Ihr RZ-Login und als Passwort benutzen Sie ihr RZ-Passwort. So lautet ein Beispielaufruf über SSH z.B. *ssh lhuning@cip20.informatik.uos.de*.

Ausführlichere Informationen hierzu können Sie unter der folgenden Website finden: `https://www.inf.uni-osnabrueck.de/arbeitsgruppen/didaktik/lehrveranstaltungen/informatik_a_algorithmen_und_datenstrukturen/praktische_hinweise.html`

Aufgabe 1.3: Fernzugriff auf einen Uni-Rechner (20 Punkte)

- Zeigen Sie Ihrem Tutor wie Sie sich mithilfe von SSH und X-Forwarding auf dem Uni-Rechner unter der Adresse *cip10.informatik.uos.de* einloggen.
- Starten Sie auf dem Uni-Rechner den Texteditor *Gedit* und nutzen Sie diesen um eine leere Textdatei mit dem Namen *Testat1.txt* auf dem Uni-Rechner anzulegen. Speichern Sie diese Datei. (Gedit können Sie öffnen, indem Sie „gedit“ in das Terminal eintippen und Enter drücken).

- Nutzen Sie ein File Transfer Protocol, wie z.B. *SCP*, um die Datei *Testat1.txt* auf ihren Heimrechner zu übertragen. Schreiben Sie die Worte „Zuhause geschrieben“ in die Datei auf Ihrem Heimrechner. Anschließend benennen Sie die Datei um in *Testat1Home.txt*.
- Nutzen Sie ein File Transfer Protocol, wie z.B. *SCP*, um die Datei *Testat1Home.txt* auf *cip10.informatik.uos.de* hochzuladen. Öffnen Sie die Datei dort und verifizieren, dass Sie den Text „Zuhause geschrieben“ enthält.

Aufgabe 1.4: Terminal Grundlagen (20 Punkte)

Versetzen Sie sich in die Lage, auf einem Ubuntu Terminal, die folgenden UNIX-Kommandos während des Testats ohne Hilfsmittel einzusetzen:

Kommando	Erklärung
man java	Ausgabe der Manualseite zum Kommando *java*
ls -l	Auflisten aller Dateien und Verzeichnisse in Tabellenform (*list*)
mkdir Blatt1	Anlegen des Verzeichnisses *Blatt1* (*make directory*)
rmdir Blatt1	Löschen des (leeren) Verzeichnisses Blatt 1 (*remove directory*)
rm test.pdf	Löschen der Datei *test.pdf*
cd Blatt 1	Wechsel in das Verzeichnis *Blatt1* (*change directory*)
cd ..	Wechsel in das übergeordnete Verzeichnis
cd	Wechsel in das Heimatverzeichnis
pwd	Gibt das aktuelle Verzeichnis aus (*print working directory*)
cp <Pfad/Dateiname> .	Kopiert eine Datei in das aktuelle Verzeichnis
javac Test.java	Kompilieren der Programmdatei *Test.java*, Ergebnis ist die Datei *Test.class*
java Test	Ausführen des Programms *Test*
nano	Öffnet den Editor Nano
nano Test.java	Öffnet die Datei Test.java im nano Editor
gedit Test.java	Öffnet die Datei Test.java im gedit Editor

Sollten Sie eine Endlosschleife verursacht haben, können Sie ihr Programm mit der Tastenkombination *Strg c* abbrechen. Schließen Sie nicht achtlos das Terminalfenster, da der Prozess ansonsten im Hintergrund weiterläuft.

Aufgabe 1.5: Java Grundlagen (20 Punkte)

In der Veranstaltung *Algorithmen und Datenstrukturen* haben Sie u.a. die Programmiersprache C++ kennengelernt. In dieser Veranstaltung lernen Sie eine weitere Programmiersprache kennen: *Java*. Alle Programmieraufgaben innerhalb dieser Veranstaltung sollen mit der Version 11 von Java bearbeitet werden. Diese Aufgabe dient dazu, Sie mit den Grundlagen von Java vertraut zu machen. Beachten Sie, dass Sie für alle Java-Programme, die Sie innerhalb dieser Veranstaltung schreiben, Ihre Klassen, Variablen und Methoden *JavaDoc* konform kommentieren sollen. Eine kurze Einführung in JavaDoc wird Teil der Übungsvideos sein. Zudem können Sie ein kurzes Tutorial unter folgendem Link finden: `https://www.tutorialspoint.com/java/java_documentation.htm`. Weitere Tutorials und die API-Dokumentation zu Java finden Sie u.a. unter folgenden Links:

- `https://docs.oracle.com/javase/tutorial/`

- `https://www.tutorialspoint.com/java/index.htm`
- `https://docs.oracle.com/en/java/javase/11/docs/api/index.html`

Sofern Sie sich ein eigenes System in Aufgabe 1.2 aufgesetzt haben, müssen Sie ggf. noch Java installieren. Sie können dies über folgenden Terminalbefehl tun: *sudo apt-get install openjdk-11-jdk openjdk-11-demo openjdk-11-doc openjdk-11-jre-headless openjdk-11-source*

1. Erstellen Sie ein Java-Programm, welches nach dem Start auf dem Terminal „Hello World“ausgibt.
2. Erstellen Sie ein Java-Programm, welches beim Start als Kommandozeilenparameter eine Zahl $x > 0$ empfängt. Die Ausgabe des Programms soll von 1 bis x zählen. Dabei soll jede Zahl in einer eigenen Zeile ausgegeben werden.

   Achten Sie auf eine geeignete Fehlerbehandlung! Dies umfasst u.a., dass vom Programm nur positive Zahlen verarbeitet werden. Zudem darf der Nutzer nur genau einen Kommandozeilenparameter übergeben. Falls der Nutzer einen ungültigen Input eintippt, soll das Programm mit einer entsprechenden Fehlermeldung und einer Anleitung zum richtigen Aufruf beendet werden.

   Um die Fehlererkennung auch auf nachfolgenden Aufgabenblättern wiederverwenden zu können, sollen Sie diese in eine separate Methode, *boolean checkInput(String[] input)*, auslagern. Für die Fehlerkennung dürfen Sie davon ausgehen, dass der Nutzer nur Integer als Kommandozeilenparameter verwendet.[3]
3. Erklären Sie Ihrem Tutor die Funktionsweise der Terminalbefehle *java* und *javac*.
4. Erklären Sie Ihrem Tutor die Begriffe *Call-by-Value* und *Call-by-Reference*. Welches der beiden Konzepte ist in Java umgesetzt?

Aufgabe 1.6: 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)`

[3] Für das Erkennen, dass der Input kein Integer ist, bieten sich *Exceptions* in Java an. Diese behandeln wir allerdings erst später in der Vorlesung und müssen deswegen auf diesem Aufgabenblatt noch nicht berücksichtigt werden.

- `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 1.7: Fragen (10 Punkte)

Beantworten Sie Ihrer Tutorin / Ihrem Tutor Fragen zum Thema Objektorientierung.