TU Berlin
Software and Business Engineering
Prof. Dr. Ingo Weber
Dr. Tobias Heindel
Richard Hobeck

WiSe 2021/22
Aufgabe 12
Block 4
24. Januar 2022

Abgabetermin: 30. Januar 2022

Aufgabe 12
Programmieren I

Hinweise

- Die Abgabe dieser Übungsaufgaben muss bis spätestens Sonntag, den 30. Januar 2022 um 23:59 Uhr im ISIS-Kurs erfolgt sein. Es gelten die Ihnen bekannten Übungsbedingungen.
- Lösungen zu diesen Aufgaben sind als gezippter Projektordner abzugeben. Eine Anleitung zum Zippen von Projekten finden Sie auf der Seite des ISIS-Kurses. *Bitte benutzen Sie einen Dateinamen der Form* `VornameNachname.zip`.
- Bitte beachten Sie, dass Abgaben im Rahmen der Übungsleistung für die Zulassung zur Klausur relevant sind. Durch Plagiieren verwirken Sie sich die Möglichkeit zur Zulassung zur Klausur in diesem Semester.

Zur Ein- und Ausgabe in Dateien mit Streams Eine der häufigsten Anwendung von Streams sind Lese- und Schreibeoperationen auf Dateien.

Aufgabe 12.1 *Zugriff auf Dateien* (2 Punkte)

Schreiben Sie eine Klasse `FileIO` mit einer `main`-Methode, die

(a) von der Eingabe drei Dateinamen einliest,

(b) die ersten zwei Dateien enthalten je eine durch Kommas separierte Liste von Zahlen.

Schreiben Sie eine Liste von Zahlen in die Datei mit dem dritten Namen, in der nur diejenigen Zahlen vorkommen, die in beiden Listen vorkommen. Sie dürfen Duplikate ignorieren.
Ein Beispielablauf des Programms könnte wir folgt aussehen.

```
Bitte geben Sie den Dateinnamen einer Datei mit Zahlen ein:
foo.txt
Bitte geben Sie den Dateinnamen einer weiteren Datei mit Zahlen ein:
bar.txt
Bitte geben Sie den Dateinnamen zum Speichern der gemeinsamen Zahlen ein:
foobar.txt
```

Wenn die Inhalte der Dateien `foo.txt` und `bar.txt` `200,300,100` bzw. `200,300,400` sind, dann ist der Inhalt der Datei `foobar.txt` `200,300`.

Zu Sockets In der Vorlesung haben Sie einfache Beispiele für Sockets kennengelernt. In der folgenden Aufgabe geht es darum, den einfachen Client und den einfachen Server so abzuändern, dass mehrere Clients gestartet werden, je eine Interaktion mit dem Server abschließen, und dann die Clients alle schließen.

Aufgabe 12.2 *Server mit mehreren Clients* **(3 Punkte)**

Schreiben Sie eine Klasse `Client` und eine Klasse `Server`, die beide die `Thread`-Klasse erweitern; außerdem, schreiben Sie eine Klasse `ClientServerCommunication`, welche eine Instanz der `Server`-Klasse und mehrere (>5) Instanzen der `Client`-Klasse erzeugt, die (nacheinander) mit dem Server kommunizieren. Die Kommunikation ist jeweils das Sender einer Ziffer von der Standardeingabe (bzw. der Buchstabe `x`, um die Kommunikation zu beenden).
Ein Beispielablauf könnte wie folgt aussehen.

```
Verbindung zu 127.0.0.1 an Port 8081.
Bitte Ziffer eingeben ('x' für Exit): 1
server: 1
Bitte Ziffer eingeben ('x' für Exit): 0
server: 0
Bitte Ziffer eingeben ('x' für Exit): 4
server: 4
Bitte Ziffer eingeben ('x' für Exit): x
Verbindung zu 127.0.0.1 an Port 8081.
Bitte Ziffer eingeben ('x' für Exit): x
Verbindung zu 127.0.0.1 an Port 8081.
Bitte Ziffer eingeben ('x' für Exit): x
Verbindung zu 127.0.0.1 an Port 8081.
Bitte Ziffer eingeben ('x' für Exit): x
Verbindung zu 127.0.0.1 an Port 8081.
Bitte Ziffer eingeben ('x' für Exit): x
Verbindung zu 127.0.0.1 an Port 8081.
Bitte Ziffer eingeben ('x' für Exit): x

Process finished with exit code 0
```