

### Aufgabe 13 *Einführung in die Programmierung mit Java*

#### Hinweise

- Die Abgabe dieser Übungsaufgaben muss bis spätestens Sonntag, den 21. Februar 2021 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 Plagieren verirken 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 13.1    *Zugriff auf Dateien (2 Punkte)*

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

- (a) von der Eingabe zwei Dateinamen einliest,
- (b) aus der ersten Datei eine durch Kommas separierte Liste von Zahlen einliest und
- (c) die sortierte Liste von Zahlen in die zweite Datei schreibt.

Ein Beispielablauf des Programms könnte wie folgt aussehen.

Bitte geben Sie den Dateinamen einer Datei mit Zahlen ein: *test.txt*  
[200,300,100]

Bitte geben Sie den Dateinamen zum Speichern der Liste ein: *bla.txt*

Die Inhalte der Dateien `input.txt` und `output.txt` sind 200,300,100 und 100,200,300.

**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 der Client eine Liste von Zahlen sendet, diese Liste dann vom Server sortiert wird, und an den Client zurück geschickt wird.

### Aufgabe 13.2     *Datenaustausch via Sockets zwischen Threads (3 Punkte)*

Schreiben Sie eine Klasse **Client** und eine Klasse **Server**, die beide die **Thread**-Klasse erweitern; außerdem, schreiben Sie eine Klasse **ClientServerCommunication**, welche je eine Instanz dieser Threads in der **main**-Methode startet.

- Die **run**-Methode des **Server**-Threads erzeugt zuerst einen Server-Socket auf dem lokalen Rechner (unter Benutzung eines fixen Ports). Danach wartet dieser Thread auf Verbindungen (vom **Client**-Thread), um Sequenzen von **String**-Repräsentationen von Zahlen zu empfangen, wobei jede Zahl eine eigene Zeile benutzt—es sei denn das Befehlswort “**sort**” wurde gesendet. Nachdem die Zeichenkette “**sort**” empfangen wurde, soll die (zuletzt) eingegebene Liste von Zahlen sortiert an den Client zurückgesendet werden.
- Die **run**-Methode der **Client**-Klasse soll eine Verbindung zum **Server**-Thread aufbauen, wiederholt Zahlen bzw. das Wort “**sort**” von der Eingabe einlesen, diese Zeichenketten an der Server senden, und nach senden des Wortes “**sort**” die sortierte Liste der letzten Zahlenfolge empfangen und auf der Konsole ausgeben.
- Die **main**-Methode der Klasse **ClientServerCommunication** soll lediglich eine Instanz von **Client** und **Server** erzeugen und anschließend starten.

Ein Beispielablauf könnte wie folgt aussehen.

```
... and go!
Connection established
3
received 3
sending 3
2
sending 2
received 2
1
sending 1
received 1
sort
sending sort
received sort
Great to have this sorted. [1, 2, 3]
1337
sending 1337
received 1337
02
sending 02
received 02
1
sending 1
received 1
sort
sending sort
received sort
Great to have this sorted. [1, 2, 1337]
```