— Übungsblatt 1 (T + P) —

Aufgabe 1.1 (T) (Serialisierung in Java – Vorgegebene Programme nachvollziehen)

Unter den bereitgestellten Programm-Vorlagen finden Sie die Java-Klasse `Ticket`, deren Objekte Tickets bei einem Online-Dienst repräsentieren sollen. Schauen Sie sich die Klasse genau an.
Die Klasse ist mit privaten Instanzvariablen für

- die Nummer des Tickets,
- den Namen des Ticket-Besitzers bzw. der Ticket-Besitzerin,
- die Kundennummer des Ticket-Besitzers bzw. der Ticket-Besitzerin,
- die Bezeichnung des Events, für das das Ticket gültig ist, und
- den Preis des Tickets in EUR

sowie mit

- einem Konstruktor, der die Ticket-Nummer, den Kund(inn)en-Namen, die Kund(inn)en-Nummer, das Event und den Preis des Tickets als Parameter übergeben bekommt und das Ticket-Objekt initialisiert,
- einer öffentlichen Zugriffsmethode, mit der die Kund(inn)ennummer eines Ticket-Objekts ausgelesen werden kann,
- einer `toString`-Methode, die das Ticket-Objekt in eine geeignete `String`-Darstellung konvertiert

ausgestattet. Die Objekte der Klasse können serialisiert werden!

Im Vorlagen-Verzeichnis finden Sie außerdem eine ausführbare Java-Klasse `TicketEdit` die es ermöglicht, nach entsprechenden Benutzer-Konsolen-Eingaben mehrere Ticket-Objekte zu erzeugen und diese serialisiert in der Datei `Tickets.dat` zu speichern. Sie startet dabei mit der Ticket-Nummer 20210200000001 und nummeriert alle Tickets fortlaufend durch. Machen Sie sich mit dem Programm-Code der Klasse vertraut.
Ein typischer Programmablauf beim Start dieser Klasse könnte z. B. wie folgt aussehen (eingegebene Daten jeweils rechts):

```
Anzahl der zu erzeugenden Tickets:              17
Daten fuer Ticket-Nummer 20210200000001
Name des Ticket-Besitzers:                      Lieselotte
Kundennummer des Ticket-Besitzers:              1000905
Event:                                          Kammertheater
Preis des Tickets in EUR:                       20
Daten fuer Ticket-Nummer 20210200000002
Name des Ticket-Besitzers:                      Lieselotte
Kundennummer des Ticket-Besitzers:              1000905
```

```
Event:                                          Stones-Konzert
Preis des Tickets in EUR:                       150
Daten fuer Ticket-Nummer 20210200000003
Name des Ticket-Besitzers:                      Friedlind
Kundennummer des Ticket-Besitzers:              1000935
Event:                                          KSC
Preis des Tickets in EUR:                       12.50

...

Daten fuer Ticket-Nummer 20210200000004
Name des Ticket-Besitzers:                      Peter
Kundennummer des Ticket-Besitzers:              1000904
Event:                                          Staatstheater
Preis des Tickets in EUR:                       25
17 Tickets in die Datei Tickets.dat geschrieben
```

Sie können auch die im Vorlagen-Verzeichnis zur Verfügung gestellte Testdaten-Datei `testdaten.txt` verwenden (z. B. über die Zwischenablage oder indem Sie auf Betriebssystem-Ebene die Standardeingabe beim Programmstart mit dem Zeichen „<“ auf diese Datei umsteuern), um eine Datei `Tickets.dat` zu erzeugen.

Im Vorlagen-Verzeichnis finden Sie darüber hinaus eine weitere ausführbare Java-Klasse `TicketCheck`. Diese ermöglicht es, alle Ticket-Objekte, die in der Datei `Tickets.dat` serialisiert gespeichert sind, einzulesen und im Klartext auf den Bildschirm auszugeben. Zum Einstz kommen dabei zwei Methoden:

- Die Methode

  ```
  public static Ticket[] liesTicketsAus (String dateiname),
  ```

  liest alle Tickets aus der namentlich genannten serialisierten Datei ein und speichert sie in einem entsprechend dimensionierten Array, das als Ergebnis zurückgeliefert wird.

- Die `main`-Methode verwendet diese Methode, um die Daten aus `Tickets.dat` in ein Array einzulesen und die Komponenten des Ergebnis-Arrays auf den Bildschirm auszugeben.

Machen Sie sich mit dem Programm-Code der Klasse vertraut.
Ein typischer Programmablauf beim Start dieser Klasse könnte z. B. wie folgt aussehen:

```
Die Datei Tickets.dat enthaelt 17 Tickets
Ticket fuer Lieselotte (Kd-Nr. 1000905)
        fuer das Event << Kammertheater >> zum Preis von 20.0 EUR
Ticket fuer Lieselotte (Kd-Nr. 1000905)
        fuer das Event << Stones-Konzert >> zum Preis von 150.0 EUR
Ticket fuer Friedlind (Kd-Nr. 1000935)
        fuer das Event << KSC >> zum Preis von 12.5 EUR
   ...

Ticket fuer Peter (Kd-Nr. 1000904)
        fuer das Event << Staatstheater >> zum Preis von 25.0 EUR
```

Aufgabe 1.2 (P) (Server mit Objekt-Auslieferungs-Dienst)

Schreiben Sie eine Java-Server-Applikation unter Verwendung von direkter Netzwerkprogrammierung mit Sockets und Streams, die es ermöglicht, Ticket-Objekte serialisiert an Clients, die den Server kontaktieren, zu verschicken.

a) Schreiben Sie dazu eine Server-Klasse, deren `main`-Methode ein `ServerSocket` mit dem Port 9898 verbindet. In einer Endlosschleife soll dann für jeden Client, der eine Verbindung aufbaut, nebenläufig ein Thread-Objekt, das den eigentlichen Dienst realisiert, erzeugt und gestartet werden.

b) Schreiben Sie dann die Thread-Klasse, die den eigentlichen Dienst implementiert.

Im Konstruktor der Klasse sollen Ströme (ein `DataInputStream` und ein `ObjectOutputStream`) über das Socket zum Client geöffnet werden.

In der `run`-Methode sollen die vom Client empfangenen Kundennummern abgearbeitet werden. Dazu sollen in einer geeigneten Schleife jeweils

- über den `DataInputStream` eine vom Client geschickte Kundennummer gelesen werden,
- mit Hilfe der Methode `liesTicketsAus` (siehe Aufgabe 2) alle in der Datei `Tickets.dat` gespeicherten Objekte gelesen werden und
- diejenigen Ticket-Objekte, die die gerade bearbeitete Kundennummer betreffen, serialisiert an den Client geschickt werden.

Die Schleife soll abgebrochen werden, wenn der Client als Kundennummer -1 geschickt hat.

Aufgabe 1.3 (P) (Client als Objekt-Empfänger)

Schreiben Sie eine Java-Client-Applikation, die es ermöglicht alle Tickets für bestimmte Kunden unter Angabe ihrer Kundennummer beim Server über den Ticket-Objekt-Auslieferungs-Dienst abzurufen.

Ihr Programm soll zunächst die Verbindung zum Dienst auf dem Server herstellen und über geeignete Ströme die gewünschten Informationen abfragen. Dazu soll wiederholt jeweils über Tastatur eine Kundennummer eingelesen, diese an den Server geschickt und die vom Server gesendeten Objekte empfangen und im Klartext auf den Bildschirm ausgegeben werden. Die Client-Anwendung soll beendet werden, wenn als Kundennummer der Wert -1 eingegeben wird.

Ein typischer Client-Programmablauf könnte z. B. wie folgt aussehen (eingegebene Daten jeweils rechts):

```
Abfragen von Tickets beim Ticket-Server
Kunden-Nummer (-1 fuer Abbruch):                                    1000905
Alle Tickets fuer Kunden-Nummer 1000905:
Ticket fuer Lieselotte (Kd-Nr. 1000905)
        fuer das Event << Kammertheater >> zum Preis von 20.0 EUR
Ticket fuer Lieselotte (Kd-Nr. 1000905)
        fuer das Event << Stones-Konzert >> zum Preis von 150.0 EUR
```

```
Kunden-Nummer (-1 fuer Abbruch):                                    1000913
Alle Tickets fuer Kunden-Nummer 1000913:
Ticket fuer Gudrun (Kd-Nr. 1000913)
        fuer das Event << Kino >> zum Preis von 10.0 EUR

Kunden-Nummer (-1 fuer Abbruch):                                    1000945
Alle Tickets fuer Kunden-Nummer 1000945:
Ticket fuer Markus (Kd-Nr. 1000945)
        fuer das Event << Kinderkino >> zum Preis von 5.0 EUR
Ticket fuer Markus (Kd-Nr. 1000945)
        fuer das Event << Kinderchorkonzert >> zum Preis von 15.0 EUR
Ticket fuer Markus (Kd-Nr. 1000945)
        fuer das Event << Kreuzfahrt >> zum Preis von 150.0 EUR

Kunden-Nummer (-1 fuer Abbruch):                                    -1
Ticket-Client beendet.
```

Aufgabe 1.4 (T) (Zusammenhang der Programme aus den Aufgaben 1.1 bis 1.3)

a) Erläutern Sie grafisch, wie die Klassen bzw. Programme aus den Aufgaben 1.1 bis 1.3 zusammenhängen. Geben Sie dabei insbesondere an,

- welche Arten von Netzwerkverbindungen (Sockets und Streams) eingesetzt werden,
- welche Arten von Daten jeweils während des Programmablaufs übertragen werden,
- welcher Art dabei jeweils Datenquelle und Ziel sind,
- wie die Programme interagieren (insbesondere im Hinblick auf die abzuwickelnden Protokolle) und
- welche Maßnahmen im Programmentwurf bzw. in der Implementierung ergriffen werden müssen, um die Interaktion geregelt ablaufen zu lassen.

b) Überlegen Sie sich, welche Probleme auftreten können, wenn diese Programme gleichzeitig (eventuell sogar mehrfach parallel) auf einem verteilten System eingesetzt werden. Welche zusätzliche Mechanismen werden benötigt, um diese Probleme in den Griff zu bekommen?