h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES
fbi
FACHBEREICH INFORMATIK

Programmieren II

Praktikum 1: Datei- und Ausnahmeverarbeitung

Wintersemester 2023/2024

Prof. Dr. Arnim Malcherek

Allgemeine Hinweise zum Praktikum:

- Bereiten Sie die Aufgaben unbedingt zu Hause oder in einem freien Labor vor. Das beinhaltet:
  - Entwurf der Lösung
  - Codieren der Lösung in einem Qt-Creator-Projekt
- Die Zeit während des Praktikums dient dazu, die Lösung testieren zu lassen sowie eventuelle Korrekturen vorzunehmen.
- Das Praktikum dient auch zur Vorbereitung der praktischen Klausur am Ende des Semesters. Versuchen Sie also in Ihrem eigenen Interesse, die Aufgaben selbständig nur mit Verwendung Ihrer Unterlagen bzw. Ihres bevorzugten C++-Buches und ohne Codefragmente aus dem Netz zu lösen.
- Die Lösung wird nur dann testiert, wenn
  - sie erklärt werden kann bzw. Fragen zur Lösung beantwortet werden können.
  - das Programm ablauffähig und die Lösung nachvollziehbar ist.
  - die Hinweise oder Einschränkungen aus der Aufgabenstellung befolgt wurden.
- Zur Erinnerung hier noch einmal die Regeln des Praktikums, die schon in der Vorlesung besprochen wurden:
  - Sie können in Gruppen arbeiten. Ich empfehle 2er Gruppen, damit Sie genügend selbst programmieren. **Die Abgabe erfolgt aber einzeln. Sie müssen das Coding also im Detail kennen und verstehen.**
  - Ein Testat gibt es nur zum jeweiligen Termin.
  - Es gibt keine Noten. Die Bewertung ist lediglich erfolgreich / nicht erfolgreich.
  - Das Praktikum ist Zulassungsvoraussetzung für die Klausur. Hierfür müssen alle sechs Praktika testiert sein.

Lernziele: Sie lernen Text- und Binärdateien - auch komplexerer Art - sicher und effizient zu verarbeiten.

Wir wollen ein Verwaltungsprogramm für das Reisebüro ‚Upandaway' erstellen. Im ersten Praktikum legen wir das Grundgerüst und schreiben ein Programm, mit dem wir existierende Daten aus einem Fremdsystem übernehmen können.

Aufgabe 1

Legen Sie Klassen zu dem Klassendiagramm aus Abbildung 1 an.

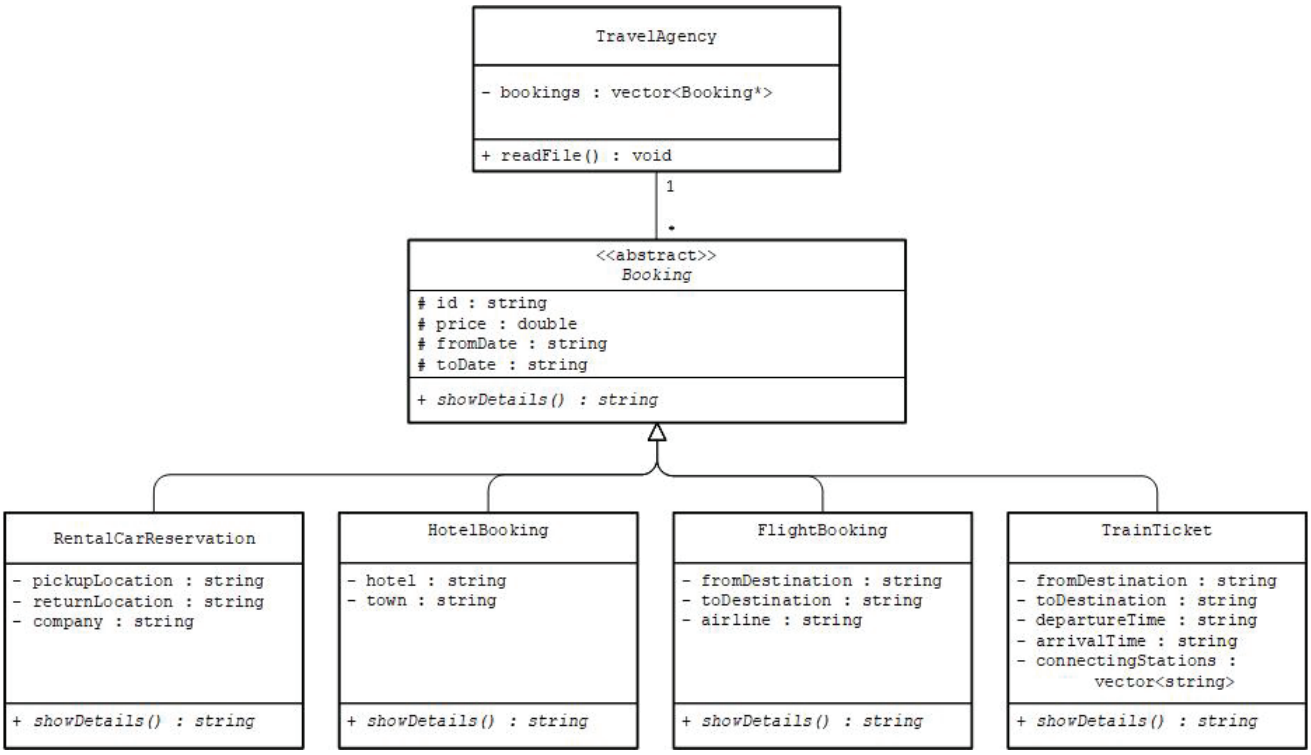

Abbildung 1: Klassendiagramm

Die Methode `showDetails` soll die Inhalte einer Buchung jeweils in sinnvoll formatierter Form ausgeben (siehe Beispiele).

- `Flugbuchung von FRA nach GRU mit Lufthansa am 04.09.2021. Preis: 2892.04 Euro`
- `Hotelreservierung im Residence 1898 in Warschau vom 02.06.2021 bis zum 03.06.2021. Preis: 256.02 Euro`
- `Mietwagenreservierung mit Europcar. Abholung am 13.03.2021 in Hamburg Flughafen. Rückgabe am 21.03.2021 in Aarhus Flughafen. Preis: 765.11 Euro`
- `Zugticket von Mannheim Hbf nach Magdeburg Hbf am 19.10.2021 von 10:04 bis 16:23 über Erfurt Hbf, Halle(Saale) Hbf. Preis: 161.15 Euro`

Achten Sie überall auf korrekte Destruktoren. Copy-Konstruktoren benötigen wir in diesem Praktikum noch nicht.

Aufgabe 2

- Legen Sie im `main` Programm eine Instanz der Klasse `TravelAgency` an und rufen anschließend die Methode `readFile()` auf. Bei der Datenübernahme aus dem Altsystem wurden alle Daten in einer JSON-Datei *bookings.json* zusammengepackt. Die Datei liegt auf Moodle und sollte selbsterklärend sein. In der Methode `readFile()` lesen Sie diese Datei ein.
- Legen Sie jeweils die passenden Objekte auf dem Heap an, und speichern Sie sie im Attribut `bookings` Ihrer Instanz von `TravelAgency`.
- Rufen Sie für jede Buchung einmal die Methode `showDetails` auf.
- Geben Sie anschließend eine Meldung aus, wie viele Buchungen Sie jeweils von welchem Typ und mit welchem Gesamtwert angelegt haben.
  Beispielausgabe: `Es wurden 14 Flugbuchungen im Wert von 33213,45 €, 34 Mietwagenbuchungen im Wert von 22314,84 € 40 Hotelreservierungen im Wert von 5463.32 € und 5 Zugbuchungen im Wert von 754,31 €, angelegt (die Zahlenwerte sind nur Beispiele und passen nicht zur Datei).`

Aufgabe 3

Legen Sie jetzt in `TravelAgency` die Methode `readBinaryFile` an. Fügen Sie in main eine Abfrage ein, welche Datei eingelesen werden soll, und rufen Sie dann die entsprechende Methode auf. In `readBinaryFile` lesen Sie die Binärdatei *bookingsBinary.bin* ein. Diese ist folgendermaßen aufgebaut:

- Attribut 1: Der Typ der Buchung ist ein einzelnes `char`-Zeichen (F für Flight, R für RentalCar, H für Hotel und T für Train).
- Attribut 2: Die Id ist als 38-stelliges `char`-Array abgelegt.
- Attribut 3: Der Preis ist im Format `double` vorhanden.
- Attribut 4 und 5: Start- und Enddatum sind jeweils 8-stellige `char`-Arrays im Format yyyyMMdd.
- Attribut 6 und 7 für Flüge: Die Flughafenkürzel sind dreistellige `char`-Arrays
- Attribut 8 für Flüge: Der Name der Fluglinie ist als 15-stelliges `char`-Array vorhanden. Inhalte sind entweder nach 15 Stellen abgeschnitten oder mit Leerzeichen aufgefüllt.
- Attribut 6, 7 und 8 für Rental Car Reservations: Die Namen der Abhol- und Rückgabestationen sowie der Firmenname sind in dieser Reihenfolge jeweils 15-stellige `char`-Arrays. Inhalte sind entweder nach 15 Stellen abgeschnitten oder aufgefüllt mit Leerzeichen.
- Attribut 6 und 7 für Hotelbuchungen sind die Namen des Hotels und der Stadt jeweils als 15-stellige `char`-Arrays. Inhalte sind entweder nach 15 Stellen abgeschnitten oder aufgefüllt mit Leerzeichen.
- Attribut 6 und 7 für Train Tickets: Die Namen des Start- und Zielbahnhofs sind in dieser Reihenfolge jeweils 15-stellige `char`-Arrays. Inhalte sind entweder nach 15 Stellen abgeschnitten oder aufgefüllt mit Leerzeichen.
- Attribut 8 und 9 für Train Tickets: Die Abfahrts- und Ankunftszeit als 5-stelliges `char`-Array (z.B. 10:04)

- Attribut 10 für Train Tickets: Enthält die Anzahl der Verbindungsbahnhöfe als int-Variable (4 Byte)
- Attribute >=11 für Train Tickets: Enthält die Namen der Verbindungsbahnhöfe jeweils als 15-stellige `char`-Arrays. Inhalte sind entweder nach 15 Stellen abgeschnitten oder aufgefüllt mit Leerzeichen.

Verfahren Sie jetzt genauso wie in Aufgabe 2: Lesen Sie die Datei ein, und legen Sie die Objekte an. Geben Sie wieder die gleichen Meldungen wie in Aufgabe 2 aus.

Aufgabe 4

Fügen Sie über `try-throw-catch` eine Fehlerprüfung beim Lesen der JSON-Datei ein, die die folgenden Prüfungen vornimmt:

- Alle Attribute müssen gefüllt sein. Leere Attribute sind nicht erlaubt.
- Das Attribut `Price` muss einen korrekten numerischen Wert enthalten.
- Flughafenkürzel dürfen nur aus drei Zeichen bestehen.

Falls eine Prüfung nicht erfüllt ist, soll eine Ausnahme geworfen werden. Die Ausnahme soll an geeigneter Stelle abgefangen und eine Nachricht ausgegeben werden, die die Art des Fehlers und die betroffene Zeilennummer der Datei enthält. Dann soll der Benutzer die Möglichkeit erhalten, die Datei zu korrigieren (außerhalb des Programms) und erneut einzulesen, ohne dass das Programm neu gestartet werden muss. Natürlich darf es dabei nicht zu doppelten Buchungen kommen, und aller Heap-Speicher, der nicht mehr benötigt wird, muss wieder freigegeben werden. Beispielhafter Ablauf in Abbildung 2.

```
JSON- oder Binaerdatei lesen?
j eingeben fuer JSON-Datei
b eingeben fuer Binaerdatei
j
Falsches Flughafenkuerzel in Zeile 24
Haben Sie die Datei korrigiert? (j/n)
j
JSON- oder Binaerdatei lesen?
j eingeben fuer JSON-Datei
b eingeben fuer Binaerdatei
j
Es wurden 47 Flugbuchungen im Wert von 52878.5,
25 Hotelreservierungen im Wert von 18406.3 und
15 Mietwagenbuchungen im Wert von 6072.06 eingelesen.

JSON- oder Binaerdatei lesen?
j eingeben fuer JSON-Datei
b eingeben fuer Binaerdatei
```

Abbildung 2: Beispielablauf