B2.1 Angewandte Programmierung (SL) - 1. Zug-SoSe2018

Ub02 (Streams)

Übung 2: Streams

12 Punkte

Lernziele:

1. Arbeiten mit **Java-Generics**
2. Verwendung von **Collection-Klassen**
3. **Codewiederholung vermeiden**
4. **Streams**
5. **Dokumentation im JavaDoc-Style**

Aufgabe 2a: Bestellungen (6 Punkte)

Die Aufgabe 1 soll um Bestellungen erweitert werden. Zu jedem Kunde können mehrere Bestellungen aufgeben werden. Jede Bestellung besteht aus einem Produkt (= Weinsorte). Die **Klasse Bestellung** erhält die Attribute:

```
private String bestellnummer; // Eindeutige Bestellnummer
private int anzahl; // Anzahl der Flaschen unabhängig von Verpackungseinheit
private Date bestelldatum; // Bestelldatum
private double gesamtPreis; // Brutto ohne Rabattabzug
private double gewaehrterRabatt = 0.0; // Rabatt abhängig vom Kunden
private double gesamtPreisNetto; // mit Rabattabzug
private WeinSorte wein;
```

Die Klasse Bestellung implementiert das **Interface java.lang.Comparable <Bestellung>**, das bedeutet, Sie müssen eine Implementierung für die ***compareTo****()*-Methode angeben. Die ***compareTo()*-Methode** soll *Bestellungen* nach **Datum** vergleichen und zwar:

- Wenn beide Bestellungen gleiches Datum haben, wird 0 zurückgegeben.
- Wenn das Datum des Parameters vor dem Datum der Vergleichsobjektes liegt, wird -1 zurückgegeben.
- Wenn das Datum des Parameters hinter dem Datum der Vergleichsobjektes liegt, wird +1 zurückgegeben.

Ergänzen Sie das Menü aus Aufgabe 1 um folgende zusätzliche Menüpunkte:

- Bestellung erfassen

- Bestellnummer suchen
- Bestellungen nach Datum absteigend sortiert ausgeben

Beim Erfassen der Bestellung muss diese einem Kunden zugeordnet werden. Die Eingabedaten müssen auf Plausibilität geprüft werden!

Die Suche nach einer Bestellnummer soll die damit verbundene Bestellung auf der Konsole ausgeben.

Die nach Datum sortierte Liste betrifft die Bestellungen eines Kunden, d. h. es soll nach Kundennummer gesucht und dann alle Bestellungen des Kunden ausgegeben werden.

Mit Programmstart sollen mindestens 6 Kunden (pro Kundentyp 2) mit einer unterschiedlichen Anzahl an Bestellungen vorhanden sein!

Aufgabe 2b: Streams (6 Punkte)

Die Kunden- und Bestelldaten sollen dauerhaft im Dateisystem gespeichert werden. Erweitern Sie dazu das Menü um folgende Punkte:

- Exportieren aller Kunden als CSV-Datei
- Speichern vom Kunden mit Bestellungen
- Laden von Kunden mit Bestellungen

Eine CSV-Datei ist eine Textdatei mit Trennzeichen (*CSV = comma seperated value*).

Das Speichern und Laden von Kundendaten soll mit Hilfe einer Objekt-Datei (Object Stream) realisiert werden. Beim Laden von Kunden mit Bestellungen werden die aktuellen Kunden und Bestellungen verworfen und durch die Daten aus der Datei ersetzt.

In allen drei Fällen soll es möglich sein den Ort der Datei für das Exportieren, Speichern und Laden selbst über die Eingabe auf der Konsole zu bestimmen. Mögliche Fehler sind abzufangen.

Hinweise:

- Bitte **keine Umlaute** und **kein ß** in **Bezeichnern** oder **Dateinamen** verwenden!
- Bitte **entfernen** Sie im Quellcode **alle** Zeilen, die **Annotationen** (beginnend mit "@", beispielsweise "@Override", "@SuppressWarnings", etc.) enthalten. Bitte nicht JavaDoc entfernen!
- Falls Sie es nicht schon getan haben, legen Sie **Pakete** an!
- Halten Sie die **main**-Funktion **schmal**!
- ***Dokumentieren*** Sie Ihre Anwendung im ***JavaDoc - Style*** !
- ***Bei Nichtbefolgung gibt es Punktabzug!***

Zuletzt geändert: Sunday, 13. May 2018, 21:39