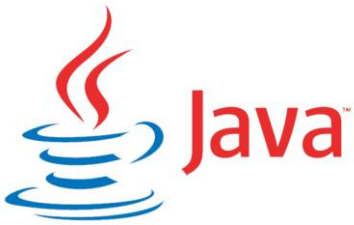
HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN
WÜRZBURG-SCHWEINFURT

FH·W-S



Programmieren II
Sommersemester 2016

Übungsblatt 9 (2 Punkte)

Byte-Streams

Datenflüsse werden in Java durch Streams abgebildet. Die in der Klassenbibliothek im Paket `java.io` vorhandenen abstrakten Klassen `OutputStream` und `InputStream` repräsentieren dabei den schreibenden bzw. den lesenden Endpunkt eines solchen Datenflusses. Je nach Quelle und Ziel des Datenflusses existieren unterschiedliche konkrete Implementierungen dieser abstrakten Klassen, z.B.

- `FileInputStream` bzw. `FileOutputStream` für das Lesen/Schreiben von Dateien
- `ByteArrayInputStream` bzw. `ByteArrayOutputStream` für das Lesen aus bzw. das Schreiben in ein Array von Bytes im Hauptspeicher
- …

Die kleinste auf einem solchen Stream übertragbare Informationseinheit ist ein Byte (daher auch Byte-Streams). Allerdings lassen sich auch komplexere Datentypen über einen Stream übertragen, sofern sich der Sender und der Empfänger hinsichtlich der Kodierung zu Bytes bzw. deren Umkehrung einig sind. Im Paket `java.io` sind mit `DataOutputStream` und `DataInputStream` zwei Klassen enthalten, die die elementaren Datentypen kodieren bzw. dekodieren können.

Aufgabe „exercise09“ (Abgabe 13.06. bis 16.06.2016, bZv-relevant)

Sie arbeiten in einem Team, das mit der Erstellung eines Online-Shops befasst ist. Sie sollen im Rahmen der Verkaufsabwicklung den Gesamtwert eines Warenkorbs ermitteln. Der Inhalt des Warenkorbs wird Ihnen als Byte-Stream zur Verfügung gestellt. Über diesen Stream werden kodiert die Positionsdaten übertragen. Jede Position im Warenkorb setzt sich aus

- einem Integer-Wert – das ist die Artikelnummer
- einem weiteren Integer-Wert – das ist die Anzahl der bestellten Stück dieses Artikels
- einem Double-Wert – das ist der Einzelpreis eines Stücks dieses Artikels

zusammen. Die Positionen des Warenkorbs folgen im Stream unmittelbar aufeinander, d.h.

`<Artikelnummer1><Anzahl1><Einzelpreis1><Artikelnummer2><Anzahl2><Einzelpreis2>…`

Ihre Aufgabe:

Im Verzeichnis `exercise09` finden Sie die abstrakte Klasse `AbstractCartCalculator`, die u.a. die abstrakte Methode

```
abstract public double totalPrice(InputStream in);
```

beinhaltet. Über die als Parameter übergebene `InputStream`-Instanz können Sie die Positionsdaten des Warenkorbs auslesen.

Bilden Sie bitte eine konkrete Unterklasse `CartCalculator`, die diese abstrakte Methode wie folgt implementiert:

- falls der übergebene InputStream `null` ist, wird 0.0 zurückgegeben.
- andernfalls werden aus dem Stream solange jeweils drei Werte (ein Tripel) je Position mit den Datentypen (int, int, double) ausgelesen, bis das Ende des Streams erreicht ist. Wird das Ende des Streams mitten in einem Tripel erkannt, so ist diese letzte Position nicht gültig und wird ignoriert. Der Rückgabewert ist die Summe der Preise aller gültigen Positionen. Ein Positionspreis ergibt sich durch das Produkt aus „Anzahl“ und „Einzelpreis“.

Bitte beachten Sie, dass der als Parameter übergebene InputStream zunächst nur Byte-Werte liefert. Die Dekodierung zu anderen Datentypen muss von Ihrer Methode geleistet werden. Außerdem müssen Sie sicherstellen, dass der übergebene InputStream nach dem Auslesen von Ihnen geschlossen wird.

Hinweise:

Um Ihre Implementierung überprüfen zu können, ist in der abstrakten Klasse `AbstractCartCalculator` eine konkrete Methode `test()` implementiert, die Sie in Ihrer Unterklasse `CartCalculator` z.B. in einer main-Methode aufrufen können:

```
public static void main(String[] args)
{
	CartCalculator calc = new CartCalculator();
	calc.test();
}
```

Die Testmethode nutzt die Klassen `ByteArrayInputStream` bzw. `ByteArrayOutputStream` für die Konstruktion eines Testfalls.