ingenieur
wissenschaften
htw saar

Hochschule für
Technik und Wirtschaft
des Saarlandes
University of
Applied Science

Programmierung 2 - Sommersemester 2020

Prof. Dr. Peter Birkner

Übungsblatt Nr. 18
Abgabe KW 25

1. Aufgabe
   Erweitern Sie die Klasse Lager aus Übungsblatt 9 (Programmierung 1) wie folgt:
   (a) Implementieren Sie eine `getSorted`-Methode, welche die Artikel im Lager als sortierte Liste zurückgibt. Die Methode soll so implementiert sein, dass das Sortierkriterium als Parameter an die Methode übergeben werden kann. Nutzen Sie zur Übergabe des Sortierkriteriums das funktionale Interface `java.util.function.BiPredicate<Artikel, Artikel>`.
   (b) Implementieren Sie eine Methode `filter`. Diese soll alle Artikel des Lagers zurückgeben, welche ein bestimmtes Filterkriterium erfüllen. Das Filterkriterium wird als Parameter an die Methode übergeben. Wählen Sie zur Übergabe des Sortierkriteriums ein passendes funktionales Interface aus `java.util.function`.
   (c) Implementieren Sie eine Methode `applyToArticles`, die eine an die Methode übergebene Operation auf alle Artikel im Lager anwendet. Wählen Sie zur Übergabe der Operation ein passendes funktionales Interface aus `java.util.function`.
   (d) **Wichtiger Hinweis:** Benutzen Sie für diesen Aufgabenteil die Datei Ueb18Fassade.java, die Sie auf der Moodle-Seite finden, d.h. halten Sie sich an die vorgegebenen Schnittstellen und beachten Sie die darin enthaltenen JavaDoc-Kommentare! Sie können gerne weitere Klassen hinzufügen, wenn Sie diese brauchen.
   Testen Sie die neuen Methoden der `Lager`-Klasse mit Hilfe von Lambda-Ausdrücken wie folgt:
      i Implementieren Sie jeweils einen Lambda-Ausdruck, der die Artikel im Lager nach folgenden Kategorien sortiert:
         (a) Unterkategorie (alphabetisch)
         (b) Bestand
         (c) Preis
      ii Reduzieren Sie den Preis alle Artikel um 10%.
      iii Fügen Sie allen Artikelbeschreibungen das Suffix **(Sonderangebot)** hinzu.
      iv Erzeugen Sie einen Lambda-Ausdruck, der die beiden Operationen ii und iii konkateniert.
   (e) Implementieren Sie eine Methode `applyToSomeArticles`, die eine Operation auf die Artikel anwendet, welche ein bestimmtes Kriterium erfüllen. Die Operation und das Filterkriterium werden als Parameter an die Methode übergeben. Wählen Sie dazu passende funktionale Interfaces aus `java.util.function`.
   (f) Implementieren Sie eine Methode `getArticles`, die eine sortierte Liste der Artikel zurückgibt, welche ein bestimmtes Suchkriterium erfüllen. Such- und Sortierkriterium werden als

Parameter an die Methode übergeben. Wählen Sie dazu passende funktionale Interfaces aus `java.util.function`.

(g) Implementieren Sie eine Methode `filterAll`, die eine beliebige Menge an Filterkriterien als Parameter entgegennimmt und die Artikel des Lagers zurück gibt, die alle Filterkriterien erfüllen. Wählen Sie dazu ein passendes funktionales Interface aus `java.util.function`. Hinweis: Nutzten Sie dazu eine variable Parameterliste.

(h) **Wichtiger Hinweis:** Benutzen Sie für diesen Aufgabenteil die Datei Ueb18Fassade.java, die Sie auf der Moodle-Seite finden, d.h. halten Sie sich an die vorgegebenen Schnittstellen und beachten Sie die darin enthaltenen JavaDoc-Kommentare! Sie können gerne weitere Klassen hinzufügen, wenn Sie diese brauchen.

Testen Sie die neuen Methoden der `Lager`-Klasse mit Hilfe von Lambda-Ausdrücken wie folgt:

i Erhöhen Sie den Preis aller CDs um 10%.

ii Reduzieren Sie den Preis aller Artikel, von denen nur noch maximal zwei Exemplare im Bestand sind um 5%.

iii Reduzieren Sie alle Bücher eines gegebenen Autors um 5%.

iv Erzeugen Sie einen Lambda-Ausdruck der die beiden Operationen i und ii kombiniert.

v Fragen Sie eine Liste aller Bücher, sortiert nach Autor, ab.

vi Testen Sie die Methode `filterAll` wie folgt: Filtern Sie alle Bücher von einem bestimmten Autor, deren Preis in einem bestimmten Bereich liegt (also von minPreis bis maxPreis).