

Programmierung 2 - Sommersemester 2019

Prof. Dr. Markus Esch

Übungsblatt Nr. 18 Abgabe KW 23

1. Aufgabe

- (a) Implementieren Sie eine Methode `iterate`. Diese nimmt folgende Werte als Parameter entgegen:

- i x_0 = Startwert (double)
- ii n = Iterationszahl (int)
- iii $f : \mathbb{R} \rightarrow \mathbb{R}$ = Funktion

Die Funktion f soll wiederholt (n mal) wie folgt angewendet werden: $x_{i+1} = f(x_i)$. Das Ergebnis der Berechnung wird als Double-Wert zurückgegeben. Definieren Sie ein passendes funktionales Interface zur Übergabe der Funktion f .

- (b) Rufen Sie die Methode `iterate` mit folgenden Funktionen auf:

- i $f(x) = 2x$
- ii $f(x) = 0.5x$
- iii $f(x) = ax(x - 1)$, für ein gegebenes $a \in (0, 1)$

Implementieren Sie die Funktionen als Lambda-Ausdrücke.

2. 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) 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 aller 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: Nutzen Sie dazu eine variable Parameterliste.
- (h) 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 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` mit mindestens drei Filterkriterien

3. Aufgabe

Erweitern Sie Ihre Lösung für Aufgabe 1 von Übungsblatt Nr. 17 wie folgt:

- (a) Nutzen Sie die unter Aufgabe 1.b implementierte Top-Level-Klasse, um die Methode `applyAndPrint` mit einer Objekt-Methoden-Referenz zur Berechnung der Fakultät aufzurufen.
- (b) Nutzen Sie die unter 1.b implementierte Static Nested Class, um die Methode `applyAndPrint` mit einer statischen Methoden-Referenz zur Berechnung der Fakultät aufzurufen.