Aufgabenblatt #5

In Ihrer Übungsumgebung unter Linux im Informatik-Labor finden Sie das Projekt Aufgabenblatt 5. Dieses Projekt besteht aus fünf Aufgaben, von denen Sie drei Aufgaben erweitern müssen. Die vierte und fünfte Aufgabe werden als Ad-hoc Aufgaben in der Übung gestellt und dürfen bis dahin nicht verändert werden.

Abgabe: Die Abgabe muss bis Montag, den 12.01. um 15:59 erfolgen. Alle Änderungen am Projekt in der Übungsumgebung, die bis zu diesem Zeitpunkt von Ihnen vorgenommen wurden, werden von uns als Abgabe betrachtet. Verspätete Abgaben werden nicht akzeptiert.

Bitte beachten Sie folgende Punkte:

- Dieses Aufgabenblatt wird benotet.
- Verändern Sie bitte nicht die Ordnerstruktur oder die Dateinamen!
- Ändern Sie nur die Inhalte der benötigten Dateien!
- Die Antworten zu den Zusatzfragen können bei den jeweiligen Aufgaben als Kommentare geschrieben werden. Achten Sie dabei auf korrekte Kommentare!

Aufgabe 1: Ausnahmen

Implementieren Sie in der gegebenen Klasse `Book`:

- Einen Konstruktor, der vier Parameter übergeben bekommt: Buchtitel (`String`), Autorenliste (`String`), Seitenanzahl (`int`), Preis in Cents (`long`).
- Getter-Methoden für jede Objektvariable.
- Eine Setter-Methode für eine Änderung am Preis.

Der Konstruktor wirft eine `InvalidBookException` wenn zumindest einer der folgenden Fälle eintritt:

- Der Buchtitel ist gleich `null`.
- Die Autorenliste ist gleich `null`.
- Die Seitenanzahl ist kleiner als 0.
- Der Preis ist kleiner als 0.

Die Setter-Methode wirft eine `InvalidPriceException`, wenn der Preis kleiner als 0 ist. Implementieren Sie zusätzlich in der `main`-Methode der Klasse `Book` einen kleinen Test, bei dem Sie

fünf `Book`-Objekte anlegen und deren Inhalt auf der Kommandozeile ausgeben. Dieser Test wird von uns nicht überprüft und soll nur als Grundlage für die Zusatzfragen dienen!

Zusatzfragen:

- Welchen Vorteil bieten Ausnahmen (Exceptions) gegenüber Rückgabewerten?
- Welches Konstrukt wird bei der Behandlung von Ausnahmen verwendet und wie erfolgt dabei die Abarbeitung des Programms?
- Wie würden Sie Fehlersituationen lösen, wenn Sie keine Ausnahmen verwenden könnten?
- Welche Exception aus der Java-API könnten Sie bei diesem Programm verwenden um falsche Argumentwerte anzuzeigen?

Aufgabe 2: Ausnahmen, Assertions

Erweitern Sie in Aufgabe 2 die Klasse `Fraction` um folgende Funktionalität:

- Im Konstruktor werden der Zähler (numerator) und der Nenner (denominator) gesetzt. Ist der Wert für den Nenner gleich 0, dann wird eine `InvalidDenominatorException` geworfen.
- In der Methode `add` wird ein als Parameter übergebener Bruch zu dem Bruch addiert. Dabei können sowohl eine `InvalidDenominatorException` als auch eine `NullObjectException` (wenn der übergebene Bruch gleich `null` ist) geworfen werden.

Die Methoden `gcd` und `reduce` müssen nicht erweitert aber bei der Präsentation verstanden werden.

Zusatzfragen:

- Wo und warum würden Sie in dieser Aufgabe Assertions verwenden?
- Wann würden Sie Assertions Exceptions vorziehen?
- Wann würden Sie Exceptions Assertions vorziehen?

Aufgabe 3: Listen

In der Klasse `IntList` haben Sie eine Listenimplementierung gegeben. Implementieren Sie folgende Methoden:

- `search`: Sucht in der Liste nach einem übergebenen Wert. Wird der Wert gefunden, dann wird `true` zurückgeliefert. Ansonsten wird `false` zurückgeliefert. Diese Methode muss iterativ in der Klasse `IntList` implementiert werden.

- `first`: Liefert den Wert des ersten Elements in der Liste zurück.
- `reverse`: Dreht die Liste um. Die Methode muss iterativ implementiert werden. Die eigentliche Iteration sollte in der Klasse `ListNode` durchgeführt werden. Bei dieser Methode werden keine (!) neuen Knoten erzeugt und die Werte der Knoten dürfen nicht überschrieben werden. Die Umkehrung der Liste wird nur durch die Neuverkettung der vorhandenen Knoten erreicht!
- `reverseRecursive`: Dreht die Liste um. Die Methode muss rekursiv implementiert werden. Die eigentliche Rekursion sollte in der Klasse `ListNode` durchgeführt werden. Bei dieser Methode werden keine (!) neuen Knoten erzeugt und die Werte der Knoten dürfen nicht überschrieben werden. Die Umkehrung der Liste wird nur durch die Neuverkettung der vorhandenen Knoten erreicht!

Zusatzfrage:

- Sie haben ein Listenobjekt `list` mit drei Knoten mit den Inhalten 1, 2 und 3 gegeben. Auf diese Liste wird `System.out.println(list);` aufgerufen. Erklären Sie genau (Element für Element), wie die Ausgabe entsteht.

Aufgabe 4: Ad-hoc Aufgabe

Aufgabe 4 wird direkt in den Übungen vorgestellt und bearbeitet.

Aufgabe 5: Ad-hoc Aufgabe

Aufgabe 5 wird direkt in den Übungen vorgestellt und bearbeitet.