Hochschule Karlsruhe
University of Applied Sciences

Fakultät für **Informatik**
und Wirtschaftsinformatik

Prof. Dr. Carsten Sinz
Lehrgebiet Software-Entwicklung

Übungsblatt 5 zur Vorlesung Programmieren

Ausgabe: Mo 27.11.2023 – Abgabe: So 07.12.2023

Aufgabe 1 (Abstrakter Datentyp Liste):

Implementieren Sie den Datentyp "Abstrakte Liste" für Listen ganzer Zahlen (`int`). Gehen Sie dazu wie folgt vor:

a) Ergänzen Sie in der Datei `IntList.java` Rümpfe für die Methoden

- `public void addFirst(int x)`
- `public Integer getFirst()`
- `public boolean dropFirst()`
- `public void addLast(int x)`
- `public Integer getLast()`
- `public boolean dropLast()`
- `public void remove(int x)`
- `public boolean contains(int x)`
- `public int size()`
- `public boolean isEmpty()`

Die Spezifikation der Methoden soll so wie in der Vorlesung sein. Für die in der Vorlesung nicht vorgestellten Methoden `getFirst`, `dropFirst`, `getLast`, `dropLast` soll folgendes gelten: `getFirst` liefert das erste Element der Liste zurück, falls es ein solches gibt, ansonsten `null`, `getLast` soll analog funktionieren. `dropFirst` entfernt das erste Element aus der Liste, `dropLast` das letzte. Wenn ein Element aus der Liste entfernt wurde, wird `true` zurückgegeben, ansonsten (leere Liste) `false`.

b) Erstellen Sie für die Methode `dropLast` Visualisierungen der Listenzellen und der `head`-Referenz wie auf den Folien der Vorlesung (Prog1_K7, S. 7ff). Geben Sie die Verkettung der Listenzellen vor und nach Ausführung von `dropLast` an für die drei Fälle: (i) leere Liste, (ii) einelementige Liste, (iii) Liste mit drei Elementen.

c) Fügen Sie der Klasse `IntList` eine Methode `public String toString()` hinzu. Die String-Repräsentation einer Liste soll immer mit einer öffnenden eckigen Klammer beginnen, danach die Elemente durch Komma und Leerzeichen getrennt enthalten und mit einer schließenden eckigen Klammer enden. Die Liste, welche die Elemente 1, 2 und 3 (in dieser Reihenfolge) enthält, sollte daher die Repräsentation `[1, 2, 3]` besitzen, die leere Liste die Repräsentation `[]`.

d) Fügen Sie der Klasse `IntList` einen Iterator, wie in der Vorlesung beschrieben, hinzu, der es erlaubt vorwärts durch die Liste zu gehen.

e) **(optional)** Fügen Sie der Klasse `IntList` eine Methode `public void reverse()` hinzu. Diese soll eine Liste umkehren (d.h. das erste Element wird zum letzten etc.). Ihre Imple-

mentierung soll keinen neuen Speicherplatz mittels `new` anfordern, d.h. es soll sich um einen *in-place*-Algorithmus handeln.

Ihre Implementierung soll so geschrieben sein, dass sich die `main`-Methode der Klasse `IntListApp` ausführen lässt und das gewünschte Ergebnis liefert. Außerdem sollen sämtliche Tests aus der Klasse `IntListTest` das Ergebnis "ok" melden. Zur Ausführung der Tests rufen Sie die `main`-Methode der Klasse `IntListTest` aus.

Aufgabe 2 (Sichtbarkeit von Variablen, Klassen und Methoden):

In Ihrer Implementierung für Aufgabe 1 sind verschiedene Elemente mit den Sichtbarkeitsmodifikatoren `public` oder `private` markiert. Begründen Sie, warum Sie diese so für sinnvoll erachten oder passen Sie die Modifikatoren gegebenenfalls an.