Aufgabenblatt 5

Allgemeine Anmerkungen

Ihre Lösung für dieses Aufgabenblatt ist bis Freitag, 15.5., 13h durch `git commit` und `push` abzugeben. Mit der Angabe wird die leere Datei `CelestialSystemIndexMap.java` mitgeliefert. Wenn Sie zusätzlich zu den gefragten Klassen, weitere Klassen oder Interfaces definieren, achten Sie darauf, dass die Klassennamen mit `My` beginnen, um Konflikte mit späteren Aufgabenblättern zu vermeiden.

Ziel

Ziel der Aufgabe ist die Anwendung der Konzepte: dynamisches Binden, toString, Hash-Tabelle (siehe Skriptum Seiten 70-91).

Gegebenes Interface

Gegeben ist folgendes Interface:

```
// Represents an index of celestial systems. The key of the index
// is a single celestial body and the associated value is the
// celestial system to which the celestial body belongs. For
// example, Io belongs to the system (Jupiter, Io, Europa,
// Ganymede, Kallisto).
public interface CelestialSystemIndex {

    // Adds a system of bodies to the index.
    // Adding a system adds multiple (key, value) pairs to the
    // index, one for each body of the system, with the same
    // value, i.e., reference to the celestial system.
    // An attempt to add a system with a body that already exists
    // in the index leaves the index unchanged and the returned
    // value would be 'false'.
    // The method returns 'true' if the index was changed as a
    // result of the call and 'false' otherwise.
    boolean add(CelestialSystem system);

    // Returns the celestial system with which a body is
    // associated. If body is not contained as a key, 'null'
    // is returned.
    CelestialSystem get(CelestialBody body);

    // Returns 'true' if the specified 'body' is listed
    // in the index.
    boolean contains(CelestialBody body);
```

```
    // Returns 'true' if 'o' is of the same class as 'this' and
    // 'this' and 'o' contain an equal set of
    // (key, value) pairs, i.e. each (key, value) pair of 'this'
    // is contained (i.e. has an equal counterpart) in 'o' and
    // vice versa. Two (key, value) pairs are equal if they have
    // equal keys and equal values.
    boolean equals(Object o);

}
```

Aufgaben

1. Überschreiben Sie in den Klassen `CelestialBody` und `CelestialSystem` zusätzlich zu `toString` auch die Methoden `equals` und `hashCode`. Zwei `CelestialBody`-Objekte sind im Sinn von `equals` gleich, wenn ihr Name gleich ist. Zwei Objekte vom Typ `CelestialSystem` sind gleich, wenn sie die gleichen Himmelskörper enthalten (unabhängig von der Reihenfolge). Achten Sie darauf, dass `equals` und `hashCode` zusammenpassen und die in `Object` beschriebenen Eigenschaften erfüllen.
2. Benennen Sie Ihre bisherige Klasse `CelestialSystemIndex` aus Aufgabenblatt 4 in `CelestialSystemIndexTree` um und definieren Sie obiges Interface.
3. Vervollständigen Sie eine Klasse `CelestialSystemIndexMap`, die das Interface `CelestialSystemIndex` mittels Hash-Tabelle implementiert. Verwenden Sie dabei keine vorgefertigten Klassen aus dem Java-Collection-Framework, sondern orientieren Sie sich an den Beispielen aus dem Skriptum. Wählen Sie eine geeignete Form der Kollisionsbehandlung. Testen Sie die Implementierung mit eigenen Testfällen. Die Klasse `CelestialSystemIndexMap` soll einen parameterlosen Konstruktor haben.
4. Überschreiben Sie in der Klasse `CelestialSystemIndexMap` die Methoden `toString`, `equals` und `hashCode`. `CelestialSystemIndexMap` nutzt `equals` für den Vergleich von Schlüsselwerten.
5. Freiwillige Zusatzaufgabe: Definieren Sie `CelestialSystemIndexTree` in abgewandelter Form als neue Klasse `CelestialSystemIndexTreeVariant`, sodass diese das gegebene Interface `CelestialSystemIndex` implementiert (das Original `CelestialSystemIndexTree` soll erhalten bleiben). Achten Sie insbesondere auf die Beschreibung der Methode `add`. Diese Zusatzaufgabe wird mit maximal einem zusätzlichen Bonuspunkt bewertet.

Die Positionen und Geschwindigkeitsvektoren können beim Testen (0,0,0) gesetzt werden. Sie können für die Klasse `CelestialBody` einen entsprechenden zusätzlichen Konstruktor schreiben.

Zusatzfragen

Bitte beantworten Sie folgende Zusatzfragen als Kommentare in der Datei `CelestialSystemIndexMap`.

1. Wie ändert sich das Verhalten von `CelestialSystemIndexMap`, wenn Sie in Ihrer Lösung die Implementierung der Methoden `equals` und `hashCode` aus der Klasse `CelestialBody` löschen?
2. Was passiert, wenn Sie nur `hashCode` löschen?
3. Welche Bedingungen gelten allgemein für die Methoden `equals` und `hashCode`?
4. Gilt in Ihrer Lösung, dass `x.toString().equals(y.toString())` den Wert `true` liefert, wenn `x.equals(y)` den Wert `true` liefert? Welche Probleme können entstehen, wenn diese Bedingung nicht erfüllt ist? (Anmerkung: Es war in diesem Aufgabenblatt nicht verlangt, dass Ihre Lösung die Bedingung erfüllen muss.)
5. Was könnte man ändern, damit neben `CelestialSystemIndexMap` auch `CelestialSystemIndexTree` das Interface `CelestialSystemIndex` implementieren kann?