

Übungen zu Einführung in die Informatik I

Aufgabe 45 **Modellierung und Implementation eines Bücherregals**

In Aufgabe 42 von Übungsblatt 10 haben wir bereits die Klasse Buch modelliert und implementiert. Nun wollen wir dazu ein Bücherregal realisieren.

- a) *„Mein Bücherregal kann leer sein oder eine Anzahl von Büchern enthalten. Wenn es voll ist, kann es erweitert werden. In mein Bücherregal kann man ein neues Buch einstellen, nach einem Buchtitel suchen und ein Buch mit einem gewünschten Buchtitel entnehmen. Außerdem hat mein Bücherregal ein Verzeichnis der enthaltenen Buchtitel.“*

Identifizieren Sie aus dieser Beschreibung eines Bücherregals die Dienste, die eine Klasse BuecherRegal zur Verfügung stellen sollte, überlegen Sie sich, welche Attribute zur Bereitstellung dieser Dienste nötig sind, und geben Sie einen detaillierten Entwurf der Klasse in UML an.

- b) (P) Implementieren Sie die Klasse BuecherRegal in Java. Stützen Sie sich dabei auf den Reihungstyp Buch[] und entscheiden Sie sich für eine von mehreren möglichen Varianten bei den Fragen,
- wie Sie ein leeres Regal füllen (z. B. von links, von rechts oder ohne eine solche Ordnung),
 - ob das Regal nach der Entnahme von Büchern Lücken aufweisen soll oder nicht und wann Sie gegebenenfalls entstandene Lücken wieder schließen,
 - ob beim vollen Regal die Erweiterung explizit angefordert werden muss oder ob die Erweiterung automatisch beim Einstellen des nächsten Buches erfolgt.

Aufgabe 46 **Implementation des Bücherregals auf sortierter Reihung**

In der vorhergehenden Aufgabe haben wir die Klasse BuecherRegal modelliert und implementiert. Nun wollen wir für diese Klasse eine neue Implementation angeben, die sich auf Reihungen abstützt, in denen die Bücher lexikographisch nach dem Titel sortiert werden. Die neue Implementation soll den Klassennamen BuecherRegalSortiert erhalten.

- a) Ändern Sie die Implementation der Operationen stelleEin(), findeBuchZumTitel() und entferneBuchZumTitel() so, dass die Sortiertheit der Bücher sichergestellt bzw. zum schnellen Auffinden der Bücher ausgenutzt wird.
- b) Implementieren Sie eine weitere Operation druckeAutorenTitelVerzeichnis(), die zu jedem Buch den Autor und den Titel ausdruckt. Dabei sollen die Autoren in lexikographischer Reihenfolge erscheinen. Falls es mehrere Bücher zum selben Autor gibt, sollen diese nach dem Titel sortiert ausgedruckt werden.

Hinweise:

- Für Vergleiche von Objekten der Klasse `String` steht Ihnen in Java die Instanz-Methode `int compareTo(String s)` zur Verfügung: Sind `string1` und `string2` zwei Objekte vom Typ `String`, dann liefert der Aufruf `string1.compareTo(string2)` einen Wert kleiner, gleich oder größer Null, falls `string1` lexikographisch kleiner, gleich oder größer als `string2` ist.
- Auf der Homepage des Programmierpraktikums finden Sie zu diesem Übungsblatt eine Datei `Buecher.txt`, die eine Folge von Aufrufen der Methode `stelleEin()` enthält. Sie können den Inhalt dieser Datei an eine geeignete Stelle Ihrer `main`-Methode kopieren, um eine „realistische Umgebung“ zum Testen Ihrer Methoden zu erhalten, ohne selbst unzählige Autorennamen und Buchtitel eingeben zu müssen.

Aufgabe 47 (H) Das N-Damenproblem

(1+8+1+1 Punkte)

Aufgabenstellung des Damenproblems ist es, n Damen so auf einem Schachbrett der Größe $n \times n$ zu positionieren, dass keine zwei Damen sich gegenseitig nach den Regeln des Schach schlagen können (jede Figur kann dabei jede andere angreifen). Für das normale Schachbrett mit $8 \times 8 = 64$ Feldern sollen demzufolge 8 Damen positioniert werden.

- Wählen Sie eine Methode um Ihr Schachbrett zu verwalten und initialisieren Sie es geeignet.
- Implementieren Sie eine Lösung des Damenproblems für $n = 1$ und $n \geq 4$ in Java unter Verwendung des nachfolgenden Algorithmus.

Um Listen in Java zu erstellen können Sie z.B. folgende Java-Funktionen verwenden (s. auch <http://java.sun.com/j2se/1.5.0/docs/api/java/util/ArrayList.html>):

```
// Eine leere ArrayList vom Typ Integer erzeugen
ArrayList<Integer> list = new ArrayList<Integer>();
// Die Zahl fünf an die Liste anhängen
list.add(5);
// Die Zahl drei an die Liste anhängen
list.add(3);
// Die Position der Zahl fünf in der Liste zurückgeben
int indexOfFive = list.indexOf(5);
// Die Zahl fünf aus der Liste entfernen und zurückgeben
int five = list.remove(indexOfFive);
// Das erste Element der Liste durch die Zahl fünf ersetzen
list.set(0, five);
```

- Die Zahl n durch 12 teilen und den Rest merken.
 - Eine geordnete Liste der geraden Zahlen von 2 bis n aufstellen.
 - Falls der Rest 3 oder 9 ist, die Zahl 2 an das Ende der Liste verschieben.
 - Die geordneten ungeraden Zahlen von 1 bis n an die Liste anhängen. Falls der Rest 8 ist, diese paarweise vertauschen (z.B. 3, 1, 7, 5, 11, 9, ...).
 - Falls der Rest 2 ist, die Positionen von 1 und 3 austauschen und die Zahl 5 an das Ende der Liste verschieben.
 - Falls der Rest 3 oder 9 ist, die Zahlen 1 und 3 an das Ende der Liste verschieben.
 - Der i -te Eintrag der Liste entspricht nun der Spaltenposition der Dame, die in der i -ten Zeile positioniert wird.
- Implementieren Sie eine geeignete Funktion `print`, die Ihnen das Schachbrett mit den darauf positionierten Damen ausgibt (z.B. Q für *Feld mit Dame* und + für *leeres Feld*).

- d) Testen Sie Ihre Implementierung für $n = 8$ und $n = 14$ und geben Sie die Programmausgabe mit ab.