INLOOP Softwaretechnologie, Collections III (Library with Map)

Henry Haustein

Datei `Book.java`

```java
// Nahezu identischer Code wie fuer Collections II (Library with Set)
package collections3;

public class Book implements Comparable<Book> {
  private String isbn;
  private String author;
  private String title;

  public Book(String isbn, String author, String title) {
    this.isbn = isbn;
    this.author = author;
    this.title = title;
  }

  public Book(String isbn) {
    this.isbn = isbn;
    this.author = "";
    this.title = "";
  }

  public String getIsbn() {
    return this.isbn;
  }

  public String getAuthor() {
    return this.author;
  }

  public String getTitle() {
    return this.title;
  }

  public String toString() {
    return this.title + " by " + this.author + " (ISBN " + this.isbn + ")";
  }

  public int compareTo(Book book){
```

```
38      return this.getIsbn().compareTo(book.getIsbn());
39    }
40
41    public boolean equals(Object o) {
42      if (o instanceof Book) {
43        Book book = (Book) o;
44        return this.isbn.equals(book.getIsbn());
45      }
46      return false;
47    }
48
49    public int hashCode() {
50      return this.isbn.hashCode();
51    }
52  }
```

Datei `Library.java`

```
1   package collections3;
2
3   import java.util.*;
4
5   public class Library {
6     private Map<String, Set<Book>> stock;
7
8     public Library() {
9       this.stock = new TreeMap<String, Set<Book>>();
10    }
11
12    public boolean insertBook(Book newBook) {
13      if (stock.containsKey(newBook.getAuthor())) {
14        // Autor gibt es bereits
15        // Hinzufuegen des Buches zur bereits bestehenden Menge
16        Set<Book> authorBooks = stock.get(newBook.getAuthor());
17        boolean result = authorBooks.add(newBook);
18        stock.put(newBook.getAuthor(), authorBooks);
19        return result;
20      }
21      else {
22        // Autor muss noch hinzugefuegt werden
23        // Neuerstellung der Menge von Buechern des Autors
24        Set<Book> authorBooks = new TreeSet<Book>();
25        authorBooks.add(newBook);
26        stock.put(newBook.getAuthor(), authorBooks);
27        return true;
28      }
29    }
30
31    public Book searchForIsbn(String isbn) {
32      // Durchlaufen aller Autoren (= Keys der Map) und
33      // anschliessendes Durchlaufen aller Buecher eines Autors
34      for (String author : stock.keySet()) {
```

```
35        Set<Book> authorBooks = stock.get(author);
36        for (Book b : authorBooks) {
37          if (b.getIsbn().equals(isbn)) {
38            return b;
39          }
40        }
41      }
42      return null;
43    }
44
45    public Collection<Book> searchForAuthor(String author) {
46      // Wenn es einen Autor nicht gibt, soll eine leere Menge
47      // zurueckgegeben werden
48      if (stock.get(author) == null) {
49        return new TreeSet<Book>();
50      }
51      else {
52        return stock.get(author);
53      }
54    }
55
56    public Map<String, Set<Book>> listStockByAuthor() {
57      return stock;
58    }
59
60    public Collection<Book> getStock() {
61      Collection<Book> books = new TreeSet<Book>();
62      // stock.values() gibt eine Collection der Values der Map
63      // zurueck, also eine Menge von Mengen von Buechern bzw.
64      // Collection<Collection<Book>>
65      for (Collection<Book> b : this.stock.values()) {
66        // addAll fuegt eine ganze Collection an Buechern auf
67        // einmal hinzu
68        books.addAll(b);
69      }
70      return books;
71    }
72  }
```