Softwaretechnologie, Übung 5

HENRY HAUSTEIN

Aufgabe 1

(a) Mit Mengen

```
1 public class Buch implements Comparable < Buch > {
private String isbn;
3 private String autor;
4 private String title;
   public Buch(String isbn, String autor, String title) {
       this.isbn = isbn;
       this.autor = autor;
8
9
       this.title = title;
10
11
12
    public Buch(String isbn) {
13
     this.isbn = isbn;
14
15
    public String toString() {
17
      return isbn + " - " + title;
18
19
20
    public String getISBN() {
21
      return isbn;
22
23
    public String getAuthor() {
24
25
       return autor;
26
27
28
    public String getTitle() {
       return title;
30
31
     public int compareTo(Buch buch){
       return this.isbn.compareTo(buch.getISBN());
34
35
36
     @Override
     public boolean equals(Object obj){
```

```
if (obj instanceOf Buch) {
38
          Buch buch = (Buch) obj;
          return isbn.equals(buch.isbn);
40
       }
41
42
       return false;
43
     }
44
45
     @Override
     public int hashCode(){
47
       return isbn.hashCode();
48
49 }
50
   public class Bibliothek {
52
53
     private Set < Buch > bestand = new TreeSet <>();
54
55
     public boolean sortiertesEinfuegen(Buch neuesBuch){
       //sortiert nach ISBN, automatisch sortiert
56
       return bestand.add(neuesBuch);
57
58
59
60
     public Buch sucheNachBuch(String isbn){
61
       for (Buch buch : bestand) {
62
          if (buch.getIsbn().equals(isbn)){
63
            return buch;
64
          }
65
       }
     }
66
67
     // Alterntive mit Iterator
     public Buch sucheNachISBN(String isbn) {
70
       Iterator < Buch > iter = bestand.iterator();
71
       while (iter.hasNext()) {
72
          Buch buch = iter.next();
73
          if (buch.getIsbn().equals(isbn)){
74
            return buch;
75
76
       }
77
78
79
     public Collection < Buch > sucheNachAutor(String autor) {
       Collection < Buch > authorBooks = new TreeSet < Buch > ();
80
81
       for (Book b : bestand) {
82
          if (b.getAuthor().equals(autor)) {
            authorBooks.add(b);
83
84
85
       }
86
       return authorBooks;
87
     }
88 }
```

(b) Quelltext

```
1 public Collection < String > getAllAuthors() {
    Set < String > result = new HashSet < String > ();
   for (Buch b : bestand) {
       result.add(b.getAuthor());
5
     return result;
7 }
9 public Map<String, Set<Buch>> bestandNachAutor() {
     Map < String , Set < Buch >> map = new TreeMap < String , Set < Buch</pre>
10
          >>();
     Collection < String > autorenliste = getAllAuthors();
11
     for (String autor : autorenliste) {
12
       Set < Buch > buecher = sucheNachAutor(autor);
13
       map.put(autor, buecher);
14
15
     }
16
     return map;
17 }
18
19 // Alternative mit nur einer benoetigten Funktion
   public Map<String, Set<Buch>> bestandNachAutor() {
21
     Map < String , Set < Buch >> map = new TreeMap < String , Set < Buch</pre>
          >>();
22
     for (Buch buch : bestand) {
23
       Set < Buch > buecher = map.get(buch.getAutor());
24
       if (buecher == null) {
          buecher = new TreeSet < Buch > ();
26
          map.put(buch.getAutor(), buecher);
27
     // Call-by-reference
28
     buecher.add(buch);
29
30
     }
31 }
```

Aufgabe 2

Datei Book. java

```
1 package collections3;
2
3 public class Book implements Comparable < Book > {
   private String isbn;
4
5
     private String author;
6
   private String title;
7
     public Book(String isbn, String author, String title){
8
9
       if (isbn == null | author == null | title == null) {
10
         throw new IllegalArgumentException("Fill the fucking
              arguments.");
```

```
} else {
11
12
         this.isbn = isbn;
13
         this.author = author;
14
         this.title = title;
      }
15
     }
16
17
18
     public Book(String isbn){
       if (isbn == null) {
19
20
         throw new IllegalArgumentException("Fill the fucking
              arguments.");
21
       } else {
22
         this.isbn = isbn;
23
         this.author = "";
         this.title = "";
24
25
       }
26
27
28
     public String getIsbn() {
29
      return this.isbn;
30
31
     public void setIsbn(String isbn) {
32
33
      this.isbn = isbn;
34
35
36
     public String getAuthor() {
37
      return this.author;
38
39
     public void setAuthor(String author) {
40
41
      this.author = author;
42
43
44
     public String getTitle() {
45
      return this.title;
46
47
     public void setTitle(String title) {
48
49
      this.title = title;
50
51
     public String toString() {
52
      return this.getTitle() + " by " + this.getAuthor() + " (ISBN "
53
             + this.getIsbn() + ")";
54
     }
55
     public int hashCode(){
56
57
      return isbn.hashCode();
58
59
```

```
60
          @Override
    61
          public boolean equals(Object o) {
            if (o instanceof Book) {
    62
              Book book = (Book) o;
    63
              return this.isbn.equals(book.getIsbn());
    64
            }
    65
    66
            return false;
    67
    68
    69
          @Override
    70
          public int compareTo(Book book) {
    71
            return this.isbn.compareTo(book.getIsbn());
    72
    73 }
Datei Library.java
     1 package collections3;
     2
     3 import java.util.*;
     4
     5 public class Library {
          private Map < String, Set < Book >> stock;
     6
     7
     8
          public Library(){
     9
            stock = new TreeMap < String, Set < Book >> ();
    10
    11
    12
          public Collection < Book > getStock(){
            Set < Book > output = new TreeSet < Book > ();;
    13
            for (Map.Entry < String, Set < Book >> entry : stock.entrySet()) {
    14
    15
              output.addAll(entry.getValue());
    16
            }
    17
            return output;
    18
    19
    20
          // alternativ, mit stock.values()
    21
          public Collection < Book > getStock() {
            Collection < Book > books = new TreeSet < Book > ();
    22
    23
            for (Collection < Book > b : this.stock.values()) {
    24
              books.addAll(b);
    25
            }
    26
            return books;
          }
    27
    28
    29
          public boolean insertBook(Book toInsertBook){
    30
            Set < Book > tempSet = new TreeSet < Book > ();
    31
            tempSet.add(toInsertBook);
    32
            if(stock.containsKey(toInsertBook.getAuthor())){
    33
              Set < Book > savedSet = stock.get(toInsertBook.getAuthor());
              if(savedSet.contains(toInsertBook)){
    34
                return false;
```

```
36
          } else {
37
            savedSet.add(toInsertBook);
38
            stock.put(toInsertBook.getAuthor(), savedSet);
            //replace - stellt sicher, dass Autor schon da ist
39
            stock.replace(toInsertBook.getAuthor(), savedSet);
40
          }
41
42
       } else {
43
          stock.put(toInsertBook.getAuthor(), tempSet);
44
45
       return true;
     }
46
47
     //inloop approved, aber sehr umstaendlich
48
49
     // alterantiv
50
     public boolean insertBook_2(Book book){
51
        if (!stock.containsKey(book.getAuthor())){
          stock.put(book.getAuthor(), new TreeMap < Buch > ());
52
53
       }
       Set < Buch > booksOfAuthor = stock.get(book.getAuthor());
54
55
       return booksOfAuthor.add(book);
56
     }
57
     // alternativ
58
     public boolean insertBook(Book newBook) {
59
60
        if (stock.containsKey(newBook.getAuthor())) {
61
          // Autor gibt es bereits
62
          Set < Book > authorBooks = stock.get(newBook.getAuthor());
          boolean result = authorBooks.add(newBook);
63
64
          stock.put(newBook.getAuthor(), authorBooks);
65
          return result;
66
       }
67
       else {
          // Autor muss noch hinzugefuegt werden
68
          Set <Book > authorBooks = new TreeSet <Book > ();
69
          authorBooks.add(newBook);
70
71
          stock.put(newBook.getAuthor(), authorBooks);
72
          return true;
73
       }
     }
74
75
76
     public Book searchForIsbn(String isbn){
77
       for (Map.Entry < String, Set < Book >> entry : stock.entrySet()) {
78
          for(Book book : entry.getValue()){
79
            if(book.getIsbn().equals(isbn)){
80
              return book;
81
82
          }
       }
83
84
       return null;
85
86
```

```
87
      // alternativ
 88
      public Book searchForIsbn_2(String isbn){
89
         for (Set < Buch > books : stock.values()){
           for (Book book : books ) {
90
             if (book.getIsbn().equals(isbn)){
91
               return book;
92
 93
             }
 94
           }
95
        }
 96
        return null;
 97
98
99
      public Map<String,Set<Book>> listStockByAuthor(){
100
        return stock;
101
102
103
      public Collection < Book > searchForAuthor(String author) {
104
         Set < Book > output = new TreeSet < Book > ();
105
         if(!stock.isEmpty()){
           if(stock.get(author) == null){
106
107
             return output;
108
           }
109
          return stock.get(author);
110
         } else {
111
           return output;
112
        }
113
      }
114
115
      // alternativ
      public Collection < Book > searchForAuthor(String author) {
116
         if (stock.get(author) == null) {
117
118
           //alternativ: return Collections.emptySet();
119
           return new TreeSet <Book > ();
120
121
         else {
122
           return stock.get(author);
123
        }
124
125 }
```