Bereich: Datenstrukturen (1)

Ziehung der Lottozahlen	**Musterlösung**
Package: `de.dhbwka.java.exercise.collections`	**Klasse:** `Lottery`

```java
package de.dhbwka.java.exercise.collections;

import java.util.Random;
import java.util.Set;
import java.util.TreeSet;

/**
 * Part of lectures on 'Programming in Java'. Baden-Wuerttemberg
 * Cooperative State University.
 *
 * (C) 2016-2018 by W. Geiger, T. Schlachter, C. Schmitt, W. Suess
 *
 * @author DHBW lecturer
 * @version 1.1
 */
public class Lottery {

   public static void main( String[] args ) {

      // Set => no duplicates
      // TreeSet => ordered!
      Set<Integer> numbers = new TreeSet<>();

      // store number that was added last
      Integer addNumber = null;
      Random r = new Random();

      // Remember: usage of Set prevents duplicates
      while ( numbers.size() < 7 ) {
         numbers.add( addNumber = r.nextInt( 49 ) + 1 ); // Autoboxing
      }

      // remove last added number => that's the bonus number!
      numbers.remove( addNumber );

      for ( Integer no : numbers ) {
         System.out.print( no + " " );
      }
      System.out.println( "Bonus number: " + addNumber );

   }

}
```

Bereich: Datenstrukturen (1)

Bücherei	**Musterlösung**
Package: `de.dhbwka.java.exercise.collections`	**Klasse:** `Library`

```java
package de.dhbwka.java.exercise.collections;

/**
 * Part of lectures on 'Programming in Java'. Baden-Wuerttemberg
 * Cooperative State University.
 *
 * (C) 2016-2018 by W. Geiger, T. Schlachter, C. Schmitt, W. Suess
 *
 * @author DHBW lecturer
 * @version 1.1
 */
public class Book {

   public static final int TITLE = 0;
   public static final int AUTHOR = 1;
   public static final int YEAR = 2;
   public static final int PUBLISHER = 3;
   public static final int[] CRITERIA =
         { Book.TITLE, Book.AUTHOR, Book.YEAR, Book.PUBLISHER };

   private String title;
   private String author;
   private int year;
   private String publisher;

   public Book() {
   }

   public Book( String title, String author, int year, String publisher ) {
      super();
      this.title = title;
      this.author = author;
      this.year = year;
      this.publisher = publisher;
   }

   @Override
   public String toString() {
      return this.title + ";" + this.author + ";" + this.year + ";"
            + this.publisher;
   }

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

   public void setTitle( String title ) {
      this.title = title;
   }
```

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

   public void setAuthor( String author ) {
      this.author = author;
   }

   public int getYear() {
      return this.year;
   }

   public void setYear( int year ) {
      this.year = year;
   }

   public String getPublisher() {
      return this.publisher;
   }

   public void setPublisher( String publisher ) {
      this.publisher = publisher;
   }
}
```

```
package de.dhbwka.java.exercise.collections;

import java.util.Comparator;

/**
 * Part of lectures on 'Programming in Java'. Baden-Wuerttemberg
 * Cooperative State University.
 *
 * (C) 2016-2018 by W. Geiger, T. Schlachter, C. Schmitt, W. Suess
 *
 * @author DHBW lecturer
 * @version 1.0
 */
public class BookComparator implements Comparator<Book> {

   private int order; // Selects Attribute to compare

   public BookComparator( int order ) {
      this.order = order;
   }
   @Override
   public int compare( Book b1, Book b2 ) {
      switch (this.order) {
         case Book.TITLE:  return b1.getTitle().compareTo( b2.getTitle() );
         case Book.AUTHOR: return b1.getAuthor().compareTo( b2.getAuthor() );
         case Book.YEAR:   return b1.getYear() - b2.getYear();
         case Book.PUBLISHER:
            return b1.getPublisher().compareTo( b2.getPublisher() );
      }
      return 0;
   }
}
```

```
package de.dhbwka.java.exercise.collections;

import java.awt.BorderLayout;
import java.awt.FlowLayout;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.PrintWriter;
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JTextField;

/**
 * Part of lectures on 'Programming in Java'. Baden-Wuerttemberg
 * Cooperative State University.
 *
 * (C) 2016-2018 by W. Geiger, T. Schlachter, C. Schmitt, W. Suess
 *
 * @author DHBW lecturer
 * @version 1.1
 */
public class Library {

   private static final String[] orderCriteria =
         { "Title", "Author", "Year", "Publisher" };

   private JFrame frame;
   private String filename = "books.txt";
   private JTextField[] inputFields;
   private List<Book> books = new ArrayList<>();

   public Library() {
      this.loadBooks();

      // Input fields incl. labels
      JPanel panInput = new JPanel();
      panInput.setLayout( new GridLayout( 4, 2, 5, 5 ) );
      this.inputFields = new JTextField[Library.orderCriteria.length];
      for ( int i = 0; i < Library.orderCriteria.length; i++ ) {
         panInput.add( new JLabel( Library.orderCriteria[i] ) );
         this.inputFields[i] = new JTextField( "" );
         panInput.add( this.inputFields[i] );
      }


      // Continued on next page
```

```
      // save button incl. event handling
      JButton btnSave = new JButton( "Save entry" );
      btnSave.addActionListener( new ActionListener() {
         @Override
         public void actionPerformed( ActionEvent e ) {
            Library.this.saveBook( Library.this.inputFields[0].getText(),
                  Library.this.inputFields[1].getText(),
                  new Integer( Library.this.inputFields[2].getText() ),
                  Library.this.inputFields[3].getText() );
            for ( JTextField field : Library.this.inputFields ) {
               field.setText( "" );
            }
         }
      } );

      // sort buttons incl. event handling
      JPanel panSort = new JPanel( new FlowLayout() );
      panSort.add( new JLabel( "Ordered output:" ) );

      for ( int i = 0; i < Library.orderCriteria.length; i++ ) {
         JButton but = new JButton( Library.orderCriteria[i] );

         // artificial attribute
         but.setActionCommand( Integer.toString( Book.CRITERIA[i] ) );
         but.addActionListener( new ActionListener() {
            @Override
            public void actionPerformed( ActionEvent e ) {
               Library.this.sort( Integer.parseInt( e.getActionCommand() ) );
            }
         } );

         panSort.add( but );
      }
      this.frame = new JFrame( "Library" );
      this.frame.setLayout( new BorderLayout() );
      this.frame.add( panInput, BorderLayout.NORTH );
      this.frame.add( btnSave, BorderLayout.CENTER );
      this.frame.add( panSort, BorderLayout.SOUTH );
      this.frame.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
      this.frame.setSize( 500, 190 );
      this.frame.setVisible( true );
   }

   public void saveBook( String title, String author, int year,
         String publisher ) {
      Book book = new Book( title, author, year, publisher );
      this.books.add( book );

      try ( PrintWriter pw = new PrintWriter(
            new FileWriter( new File( this.filename ), true ) ); ) {
         pw.println( book ); // uses toString of Book
      } catch ( Exception ex ) {
         System.err
               .println( "Write error: " + ex.getLocalizedMessage() );
      }
   }
```

```
   public void loadBooks() {
      try ( BufferedReader br = new BufferedReader(
            new FileReader( new File( this.filename ) ) ); ) {
         while ( br.ready() ) {
            String[] parts = br.readLine().split( ";" );
            if ( parts.length == 4 ) {
               this.books.add( new Book( parts[0], parts[1],
                     new Integer( parts[2] ), parts[3] ) );
            }
         }
      } catch ( Exception ex ) {
         System.err.println( "Read error: " + ex.getLocalizedMessage() );
      }
   }

   /**
    * Order books by order criteria and display ordered list
    */
   public void sort( int order ) {
      Collections.sort( this.books, new BookComparator( order ) );
      // uses Library.toString()
      JOptionPane.showMessageDialog( this.frame, this,
            "Books ordered by " + Library.orderCriteria[order],
            JOptionPane.INFORMATION_MESSAGE );
   }

   /**
    * All books as a single multi line String
    *
    * @return string with one book per line
    */
   @Override
   public String toString() {
      StringBuffer output = new StringBuffer( "" );
      for ( Book book : this.books ) {
         output.append( book + System.lineSeparator() );
      }
      return output.toString();
   };

   public static void main( String[] args ) {
      new Library();
   }

}
```