## Bereich: Aufzählungstypen

| **Monate** | **Musterlösung** |
|---|---|
| **Package:** `de.dhbwka.java.exercise.enums` | **Klasse:** `Months` |

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

import java.util.Calendar;

/**
 * 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 enum Months {

   JANUARY("Januar", 31, "Hartung, Eismond"),
   FEBRUARY("Februar", 28,
         "Hornung, Schmelzmond, Taumond, Narrenmond, Rebmond, Hintester"),
   MARCH("M\u00E4rz", 31, "Lenzing, Lenzmond"),
   APRIL("April", 30, "Launing, Ostermond"),
   MAY("Mai", 31, "Winnemond*, Blumenmond"),
   JUNE("Juni", 30, "Brachet, Brachmond"),
   JULY("Juli", 31, "Heuert, Heumond"),
   AUGUST("August", 31, "Ernting, Erntemond, Bisemond"),
   SEPTEMBER("September", 30, "Scheiding, Herbstmond"),
   OCTOBER("Oktober", 31, "Gilbhart, Gilbhard, Weinmond"),
   NOVEMBER("November", 30, "Nebelung, Windmond, Wintermond"),
   DECEMBER("Dezember", 31, "Julmond, Heilmond, Christmond, Dustermond");

   private String name;
   private int days;
   private String altNames;

   private Months( String name, int days, String altNames ) {
      this.name = name;
      this.days = days;
      this.altNames = altNames;
   }

   @Override
   public String toString() {
      return "Der " + this.name + " hat " + this.days
            + " Tage und hie\u00DF fr\u00FCher '" + this.altNames + "'";
   }

   public String getName() {
      return this.name;
   }
```

```
   public int getDays() {
      return this.days;
   }

   public String getAltNames() {
      return this.altNames;
   }

   public static void main( String[] args ) {
      int thisMonth = Calendar.getInstance().get( Calendar.MONTH );
      for ( Months month : Months.values() ) {
         if ( month.ordinal() == thisMonth ) {
            System.out.println( month );
         }
      }
      System.out.println( Months.values()[thisMonth] );
   }

}
```

## Bereich: Aufzählungstypen

| **Kartenspiel** | **Musterlösung** |
|---|---|
| **Package:** `de.dhbwka.java.exercise.enums.cards` | **Klasse:** `CardGame` |

```java
package de.dhbwka.java.exercise.enums.cards;

/**
 * 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 enum Suit {

   DIAMONDS("Karo", 9),
   HEART("Herz", 10),
   SPADE("Pik", 11),
   CLUBS("Kreuz", 12);

   private final String name;
   private final int value;

   private Suit( String name, int value ) {
      this.name = name;
      this.value = value;
   }

   @Override
   public String toString() {
      return this.name;
   }

   public String getName() {
      return this.name;
   }

   public int getValue() {
      return this.value;
   }

}


// Continued on next page
```

```java
package de.dhbwka.java.exercise.enums.cards;

/**
 * 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 enum CardValue {

   SEVEN("7", "7"),
   EIGHT("8", "8"),
   NINE("9", "9"),
   JACK("B", "Bube"),
   QUEEN("D", "Dame"),
   KING("K", "K\u00F6nig"),
   TEN("10", "10"),
   ACE("A", "Ass");

   private final String name;
   private final String longName;

   private CardValue( String name, String longName ) {
      this.name = name;
      this.longName = longName;
   }

   @Override
   public String toString() {
      return this.longName;
   }

   public String getName() {
      return this.name;
   }

   public String getLongName() {
      return this.longName;
   }

}

// Continued on next page
```

```java
package de.dhbwka.java.exercise.enums.cards;

/**
 * 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 PlayingCard implements Comparable<PlayingCard> {

   private Suit suit;
   private CardValue value;

   public PlayingCard() {
   }

   public PlayingCard( Suit suit, CardValue value ) {
      this.suit = suit;
      this.value = value;
   }

   @Override
   public String toString() {
      return this.suit + " " + this.value;
   }

   @Override
   public int compareTo( PlayingCard ok ) {
      return (new Integer( this.getOrderValue() )
            .compareTo( ok.getOrderValue() ));
   }

   private int getOrderValue() {
      return this.suit.ordinal() * 10 + this.value.ordinal();
   }

   public Suit getSuit() {
      return this.suit;
   }

   public void setSuit( Suit suit ) {
      this.suit = suit;
   }

   public CardValue getValue() {
      return this.value;
   }

   public void setValue( CardValue value ) {
      this.value = value;
   }

}
```

```java
package de.dhbwka.java.exercise.enums.cards;

import java.util.ArrayList;
import java.util.Collections;
import java.util.List;

/**
 * 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 CardGame {

   private List<PlayingCard> deck;

   public CardGame() {
      this.deck = new ArrayList<>();
      for ( Suit f : Suit.values() ) {
         for ( CardValue w : CardValue.values() ) {
            this.deck.add( new PlayingCard( f, w ) );
         }
      }
   }

   public void shuffle() {
      Collections.shuffle( this.deck );
   }

   public void sort() {
      Collections.sort( this.deck );
   }

   public PlayingCard get() {
      if ( this.deck.isEmpty() ) {
         return null;
      }
      return this.deck.remove( 0 );
   }

   public List<PlayingCard> all() {
      return this.deck;
   }

}

// Continued on next page
```

```java
package de.dhbwka.java.exercise.enums.cards;

/**
 * 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 TestGame {

   public static void main( String[] args ) {
      CardGame deck = new CardGame();
      deck.shuffle();
      PlayingCard card = deck.get();
      PlayingCard heartSeven = new PlayingCard( Suit.HEART, CardValue.SEVEN );

      System.out.println( "10 Karten ziehen und vergleichen:" );
      for ( int i = 0; i < 10; i++ ) {
         System.out.println( card + " verglichen mit " +
               heartSeven + ": " + card.compareTo( heartSeven ) );
         card = deck.get();
      }
      deck.sort();

      System.out.println( "\u00DCbrige Karten sortiert:" );
      for ( PlayingCard cd : deck.all() ) {
         System.out.println( cd );
      }
   }

}
```

## Bereich: Aufzählungstypen

| **Sortierkriterien** | **Musterlösung** |
|---|---|
| **Package:** `de.dhbwka.java.exercise.enums.library` | **Klasse:** `Attributes` |

```java
package de.dhbwka.java.exercise.enums.library;

/**
 * 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 enum Attributes {

   TITLE("Title"),
   AUTHOR("Author"),
   YEAR("Year"),
   PUBLISHER("Publisher");

   private String name;

   private Attributes( String name ) {
      this.name = name;
   }

   public String getName() {
      return this.name;
   }

}
```

```java
package de.dhbwka.java.exercise.enums.library;

/**
 * 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 {

   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;
   }

}

// Continued on next page
```

```java
package de.dhbwka.java.exercise.enums.library;

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 1.1
 */
public class BookComparator implements Comparator<Book> {

   private Attributes order; // Selects Attribute to compare

   public BookComparator( Attributes order ) {
      this.order = order;
   }

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

}

// Continued on next page
```

```java
package de.dhbwka.java.exercise.enums.library;

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 JFrame frame;
   private String filename = "books.txt";
   private List<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 ArrayList<>();
      for ( Attributes att : Attributes.values() ) {
         JTextField field = new JTextField( "" );
         this.inputFields.add( field );
         panInput.add( new JLabel( att.getName() ) );
         panInput.add( field );
      }

      // 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.get( 0 ).getText(),
                  Library.this.inputFields.get( 1 ).getText(),
                  new Integer( Library.this.inputFields.get( 2 ).getText() ),
                  Library.this.inputFields.get( 3 ).getText() );

            // Reset fields
            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 ( Attributes att : Attributes.values() ) {
         JButton but = new JButton( att.getName() );
         but.setActionCommand( att.toString() ); // artificial attribute
         but.addActionListener( new ActionListener() {
            @Override
            public void actionPerformed( ActionEvent e ) {
               Library.this.sort( Attributes
                     .valueOf( ((JButton) e.getSource()).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() );
      }
   }
```

```
   // Continued on next page

   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( Attributes order ) {
      Collections.sort( this.books, new BookComparator( order ) );
      // uses Library.toString()
      JOptionPane.showMessageDialog( this.frame, this,
            "Books ordered by " + order.getName(),
            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();
   }

}
```