Der Code befindet sich in der ZIP Datei. Zusätzlich haben wir den Code wie gewünscht exportiert (zwei PDFs). Diese befinden sich ebenfalls im ZIP.

```java
//Die drei von Nebenan
package Uebungsaufgaben.Uebung02;

import org.junit.jupiter.api.DisplayNameGenerator;

import java.util.Arrays;

class Dance {
    private String name;
    private String beat;
    private Figure[] figures;

    //Constructors
    public Dance(String name, String beat) {
        this.name = name;
        this.beat = beat;

    }

    public Dance(String name, String beat, Figure[] figures) {
        this.beat = beat;
        this.name = name;
        this.figures = figures;

    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getBeat() {
        return beat;
    }
```

```
38
39      public void setBeat(String beat) {
40          this.beat = beat;
41      }
42
43      public Figure[] getFigures() {
44          return figures;
45      }
46
47      public void setFigures(Figure[] figures) {
48          this.figures = figures;
49      }
50  }
51
52  class Figure {
53      private String name;
54      private String description;
55      private Figure[] sequence = new Figure[0];
56
57      public Figure(String name, String description
    ) {
58          this.name = name;
59          this.description = description;
60          this.sequence = null;
61      }
62
63      public Figure(String name, Figure[] sequence) {
64          this.name = name;
65          this.sequence = sequence;
66          this.description = null;
67      }
68
69      public boolean add( Figure inFigure ) {
70          if ( sequence == null )                  //
    keine Umwandlung von Beschreibungsfiguren in
    Sequenzfiguren erlaubt
71              return false;
72          if ( !inFigure.contains(this) ) {        //
```

```
 73             Figure[] neuSequenz = new Figure[
    sequence.length+1];
 74             for ( int i = 0; i < sequence.length;
    i++ ) {
 75                 neuSequenz[i] = sequence[i];
 76             };
 77             neuSequenz[sequence.length] = inFigure
    ;
 78             return true;
 79         }
 80         return false;
 81     }
 82 
 83 
 84     //
 85     // protected
 86     //
 87 
 88     protected boolean contains( Figure inFigure
     ) {
 89         if ( sequence == null ) {
 90             return false;
 91         }
 92         for ( Figure seqItem : sequence ) {
 93             if ( seqItem == inFigure )
 94                 return true;
 95             else {
 96                 if ( seqItem.contains(inFigure) )
 97                     return true;
 98             }
 99         }
100         return false;
101     }
102 }
103 
104 class StandardDance extends Dance{
105 
106     public StandardDance(String name, String beat
```

```
106 ) {
107         super(name, beat);
108     }
109 
110     public StandardDance(String name, String beat
    , Figure[] figures) {
111         super(name, beat, figures);
112     }
113 }
114 
115 class LatinDance extends Dance{
116 
117 
118     public LatinDance(String name, String beat) {
119         super(name, beat);
120     }
121 
122     public LatinDance(String name, String beat,
    Figure[] figures) {
123         super(name, beat, figures);
124     }
125 }
126 
127 class DanceDatabase {
128     public static void main(String[] args) {
129         Figure basicMove = new Figure("Basic Move"
    , "Hier könnte IHRE Werbung stehen!");
130         Figure fan = new Figure("Fan", "Hier
    könnte IHRE Werbung stehen!");
131         Figure promenade = new Figure("Promenade"
    , "Hier könnte IHRE Werbung stehen!");
132         Figure spin_turn = new Figure("Spin Turn"
    , "Hier könnte IHRE Werbung stehen!");
133         Figure natural_turn = new Figure("Natual
    Dance", "Hier könnte IHRE Werbung stehen!");
134         Figure chasse = new Figure("Chasse", "Hier
     könnte IHRE Werbung stehen!");
135         Figure whisk = new Figure("Whisk", new
```

```
135 Figure[]{promenade, chasse});
136
137
138         LatinDance jive = new LatinDance("Jive", "
    4/4" );
139         LatinDance rumba = new LatinDance("Rumba"
    , "4/4");
140         LatinDance chachacha = new LatinDance("
    ChaChaCha", "4/4");
141         StandardDance tango = new StandardDance("
    Tango", "4/4");
142         LatinDance quickstep = new LatinDance("
    Quickstep", "4/4");
143         StandardDance waltz = new StandardDance("
    Waltz", "3/4");
144
145         jive.setFigures(new Figure[]{basicMove});
146         rumba.setFigures(new Figure[]{basicMove,
    fan});
147         chachacha.setFigures(new Figure[]{
    basicMove, fan});
148         tango.setFigures(new Figure[]{basicMove,
    promenade});
149         quickstep.setFigures(new Figure[]{
    basicMove, spin_turn});
150         waltz.setFigures(new Figure[]{spin_turn,
    whisk, natural_turn});
151
152         whisk.add(promenade);
153         whisk.add(chasse);
154
155
156
157     }
158 }
159
```

```java
package Uebungsaufgaben.Uebung02;

public class Out {
    public static void out(boolean booleanValue) {
        System.out.println(booleanValue);
    }

    public static void out(int integerparam) {
        System.out.println(integerparam);
    }

    public static void out(double doubleparam) {
        System.out.println(doubleparam);
    }

    public static void out(char charparam) {
        System.out.println(charparam);
    }

    public static void out(String stringparam) {
        System.out.println(stringparam);
    }

    public static void out(Object objectparam) {
        System.out.println(objectparam);
    }

}

/*
zu Aufgabenteil b:
Dies ist möglich, da die Klasse Objekt (
bereitgestellt von Oracle [Java]) mit jedem anderen
 Datentyp (wie String, int, char etc.) umgehen kann
 */


```