

```

1 package oop3_dat3_u;
2
3 import java.util.List;
4
5 /**
6  * Implementieren Sie das Interface IList in zwei Klassen:
7  * Diese Klasse verwendet ArrayList, um die Liste zu implementieren.
8  *
9  * Ich habe das ganze ein wenig abgeändert, damit es möglichst keine
10 * Code-Duplikationen gibt...
11 * @author Christian Bontekoe <christian.bontekoe@stud.hslu.ch>, Felix Rohrer
12 * <Felix.rohrer@stud.hslu.ch>
13 */
14 public class MyList implements IList
15 {
16     private List<Integer> myList;
17
18     public MyList(List<Integer> newList)
19     {
20         myList = newList;
21     }
22
23     /**
24     * add()
25     * Diese Methode erhält ein Integer-Objekt und gibt nichts zurück.
26     * Das Objekt wird in die Liste eingefügt.
27     */
28     public void add(Integer value)
29     {
30         myList.add(value);
31     }
32
33     /**
34     * remove()
35     * Diese Methode erhält ein Integer-Objekt. Dieses wird (ein Vorkommen) aus
36     * der Liste entfernt. Wenn das Objekt entfernt werden konnte, gibt die
37     * Methode true zurück, andernfalls false.
38     */
39     public boolean remove(Integer value)
40     {
41         return myList.remove(value);
42     }
43
44     /**
45     * removeValue()
46     * Diese Methode erhält einen ganzzahligen Wert und entfernt ein
47     * Integer-Objekt mit diesem Wert. Konnte ein solches Objekt entfernt
48     * werden, gibt die Methode true zurück, false sonst.
49     */
50     public boolean removeValue(int value)
51     {
52         return myList.remove((Integer) value);
53     }
54
55     /**
56     * removeValues()
57     * Diese Methode erhält einen ganzzahligen Wert und entfernt alle
58     * Integer-Objekte mit diesem Wert. Konnten alle Objekte entfernt werden,
59     * gibt die Methode true zurück, false sonst.
60     */
61     public boolean removeValues(int value)
62     {
63         boolean res = false;
64         while (this.removeValue(value)) {
65             res = true;
66         }
67         return res;
68     }
69
70     /**
71     * size()
72     * Diese Methode gibt die Anzahl Elemente in der Liste zurück.
73     */
74     public int size()
75     {
76         return myList.size();
77     }
78
79     /**
80     * exists()
81     * Diese Methode erhält einen ganzzahligen Wert. Sie gibt true zurück, falls
82     * ein Integer-Objekt mit diesem Wert existiert, false sonst.
83     */
84     public boolean exists(int value)
85     {
86         return myList.contains((Integer) value);
87     }
88
89     /**
90     * print()
91     * Diese Methode gibt alle Werte der Integer-Objekte in der Liste aus.
92     */
93     public void print()
94     {
95         for (Integer i : myList) {
96             System.out.println(i);
97         }
98     }
99 }

```

```
1 package oop3_dat3_u;
2
3 import java.util.ArrayList;
4
5 /**
6  * Implementieren Sie das Interface IList in zwei Klassen:
7  * Diese Klasse verwendet ArrayList, um die Liste zu implementieren.
8  *
9  * Ich habe das ganze ein wenig abgeändert, damit es möglichst keine
10 * Code-Duplikationen gibt...
11 * @author Christian Bontekoe <christian.bontekoe@stud.hslu.ch>, Felix Rohrer
12 * <felix.rohrer@stud.hslu.ch>
13 */
14 public class ListOne extends MyList
15 {
16     public ListOne()
17     {
18         super(new ArrayList<Integer>());
19     }
20 }
```

```
1
2 package oop3_dat3_u;
3 import java.util.LinkedList;
4 /**
5  * Implementieren Sie das Interface IList in zwei Klassen:
6  * Diese Klasse verwendet LinkedList, um die Liste zu implementieren.
7  *
8  * Ich habe das ganze ein wenig abgeändert, damit es möglichst keine
9  * Code-Duplikationen gibt...
10 * @author Christian Bontekoe <christian.bontekoe@stud.hslu.ch>, Felix Rohrer
    <felix.rohrer@stud.hslu.ch>
11 */
12 public class ListTwo extends MyList
13 {
14     public ListTwo()
15     {
16         super(new LinkedList<Integer>());
17     }
18 }
```

```

1 package oop3_dat3_u;
2
3 /**
4  * Implementieren Sie das Interface IList in zwei Klassen:
5  * Diese Klasse verwendet ArrayList, um die Liste zu implementieren.
6  *
7  * Ich habe das ganze ein wenig abgeändert, damit es möglichst keine
8  * Code-Duplikationen gibt...
9  * @author Felix Rohrer <felix.rohrer@stud.hslu.ch>
10 */
11 public class Main
12 {
13
14     /**
15      * @param args the command line arguments
16      */
17     public static void main(String[] args)
18     {
19         // ListOne: ArrayList
20         System.out.println("----- List One / ArrayList -----");
21         ListOne myListOne = new ListOne();
22         myListOne.add((Integer) 1);
23         myListOne.add((Integer) 2);
24         myListOne.add((Integer) 3);
25         myListOne.add((Integer) 5);
26         myListOne.add((Integer) 10);
27         myListOne.add((Integer) 23);
28         myListOne.add((Integer) 5);
29         myListOne.add((Integer) 25);
30
31         System.out.println("myListOne:");
32         myListOne.print();
33
34         System.out.println("Remove Integer 3");
35         myListOne.remove((Integer) 3);
36         myListOne.print();
37
38         System.out.println("Remove value 23");
39         myListOne.removeValue(23);
40         myListOne.print();
41
42         System.out.println("Remove all 5");
43         myListOne.removeValues(5);
44         myListOne.print();
45
46         // List Two: LinkedList
47         System.out.println("----- List Two / LinkedList -----");
48         ListTwo myListTwo = new ListTwo();
49         myListTwo.add((Integer) 1);
50         myListTwo.add((Integer) 2);
51         myListTwo.add((Integer) 3);
52         myListTwo.add((Integer) 5);
53         myListTwo.add((Integer) 10);
54         myListTwo.add((Integer) 23);
55         myListTwo.add((Integer) 5);
56         myListTwo.add((Integer) 25);
57
58         System.out.println("myListTwo:");
59         myListTwo.print();
60
61         System.out.println("Remove Integer 3");
62         myListTwo.remove((Integer) 3);
63         myListTwo.print();
64
65         System.out.println("Remove value 23");
66         myListTwo.removeValue(23);
67         myListTwo.print();
68
69         System.out.println("Remove all 5");
70         myListTwo.removeValues(5);
71         myListTwo.print();
72     }
73 }

```