```java
package oop3_dat3_u;

import java.util.List;

/**
 * Implementieren Sie das Interface IList in zwei Klassen:
 * Diese Klasse verwendet ArrayList, um die Liste zu implementieren.
 *
 * Ich habe das ganze ein wenig abgeändert, damit es möglichst keine
 * Code-Duplikationen gibt...
 * @author Christian Bontekoe <christian.bontekoe@stud.hslu.ch>, Felix Rohrer
<felix.rohrer@stud.hslu.ch>
 */
public class MyList implements IList
{

    private List<Integer> myList;

    public MyList(List<Integer> newList)
    {
        myList = newList;
    }

    /**
     * add()
     * Diese Methode erhält ein Integer-Objekt und gibt nichts zurück.
     * Das Objekt wird in die Liste eingefügt.
     */
    public void add(Integer value)
    {
        myList.add(value);
    }

    /**
     * remove()
     * Diese Methode erhält ein Integer-Objekt. Dieses wird (ein Vorkommen) aus
     * der Liste entfernt. Wenn das Objekt entfernt werden konnte, gibt die
     * Methode true zurück, andernfalls false.
     */
    public boolean remove(Integer value)
    {
        return myList.remove(value);
    }

    /**
     * removeValue()
     * Diese Methode erhält einen ganzzahligen Wert und entfernt ein
     * Integer-Objekt mit diesem Wert. Konnte ein solches Objekt entfernt
     * werden, gibt die Methode true zurück, false sonst.
     */
    public boolean removeValue(int value)
    {
        return myList.remove((Integer) value);
    }

    /**
     * removeValues()
     * Diese Methode erhält einen ganzzahligen Wert und entfernt alle
     * Integer-Objekte mit diesem Wert. Konnten alle Objekte entfernt werden,
     * gibt die Methode true zurück, false sonst.
     */
    public boolean removeValues(int value)
    {
        boolean res = false;
        while (this.removeValue(value)) {
            res = true;
        }
        return res;
    }

    /**
     * size()
     * Diese Methode gibt die Anzahl Elemente in der Liste zurück.
     */
    public int size()
    {
        return myList.size();
    }

    /**
     * exists()
     * Diese Methode erhält einen ganzzahligen Wert. Sie gibt true zurück, falls
     * ein Integer-Objekt mit diesem Wert existiert, false sonst.
     */
    public boolean exists(int value)
    {
        return myList.contains((Integer) value);
    }

    /**
     * print()
     * Diese Methode gibt alle Werte der Integer-Objekte in der Liste aus.
     */
    public void print()
    {
        for (Integer i : myList) {
            System.out.println(i);
        }
    }
}
```

```java
package oop3_dat3_u;

import java.util.ArrayList;

/**
 * Implementieren Sie das Interface IList in zwei Klassen:
 * Diese Klasse verwendet ArrayList, um die Liste zu implementieren.
 *
 * Ich habe das ganze ein wenig abgeändert, damit es möglichst keine
 * Code-Duplikationen gibt...
 * @author Christian Bontekoe <christian.bontekoe@stud.hslu.ch>, Felix Rohrer
<felix.rohrer@stud.hslu.ch>
 */
public class ListOne extends MyList
{

    public ListOne()
    {
        super(new ArrayList<Integer>());
    }
}
```

```java

package oop3_dat3_u;
import java.util.LinkedList;
/**
 * Implementieren Sie das Interface IList in zwei Klassen:
 * Diese Klasse verwendet LinkedList, um die Liste zu implementieren.
 *
 * Ich habe das ganze ein wenig abgeändert, damit es möglichst keine
 * Code-Duplikationen gibt...
 * @author Christian Bontekoe <christian.bontekoe@stud.hslu.ch>, Felix Rohrer
<felix.rohrer@stud.hslu.ch>
 */
public class ListTwo extends MyList
{
    public ListTwo()
    {
        super(new LinkedList<Integer>());
    }
}
```

```java
package oop3_dat3_u;

/**
 * Implementieren Sie das Interface IList in zwei Klassen:
 * Diese Klasse verwendet ArrayList, um die Liste zu implementieren.
 *
 * Ich habe das ganze ein wenig abgeändert, damit es möglichst keine
 * Code-Duplikationen gibt...
 * @author Felix Rohrer <felix.rohrer@stud.hslu.ch>
 */
public class Main
{

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args)
    {
        // ListOne: ArrayList
        System.out.println("----- List One / ArrayList ----------------------------");
        ListOne myListOne = new ListOne();
        myListOne.add((Integer) 1);
        myListOne.add((Integer) 2);
        myListOne.add((Integer) 3);
        myListOne.add((Integer) 5);
        myListOne.add((Integer) 10);
        myListOne.add((Integer) 23);
        myListOne.add((Integer) 5);
        myListOne.add((Integer) 25);

        System.out.println("myListOne:");
        myListOne.print();

        System.out.println("Remove Integer 3");
        myListOne.remove((Integer) 3);
        myListOne.print();

        System.out.println("Remove value 23");
        myListOne.removeValue(23);
        myListOne.print();

        System.out.println("Remove all 5");
        myListOne.removeValues(5);
        myListOne.print();

        // List Two: LinkedList
        System.out.println("----- List Two / LinkedList ----------------------------");
        ListTwo myListTwo = new ListTwo();
        myListTwo.add((Integer) 1);
        myListTwo.add((Integer) 2);
        myListTwo.add((Integer) 3);
        myListTwo.add((Integer) 5);
        myListTwo.add((Integer) 10);
        myListTwo.add((Integer) 23);
        myListTwo.add((Integer) 5);
        myListTwo.add((Integer) 25);

        System.out.println("myListTwo:");
        myListTwo.print();

        System.out.println("Remove Integer 3");
        myListTwo.remove((Integer) 3);
        myListTwo.print();

        System.out.println("Remove value 23");
        myListTwo.removeValue(23);
        myListTwo.print();

        System.out.println("Remove all 5");
        myListTwo.removeValues(5);
        myListTwo.print();
    }
}
```