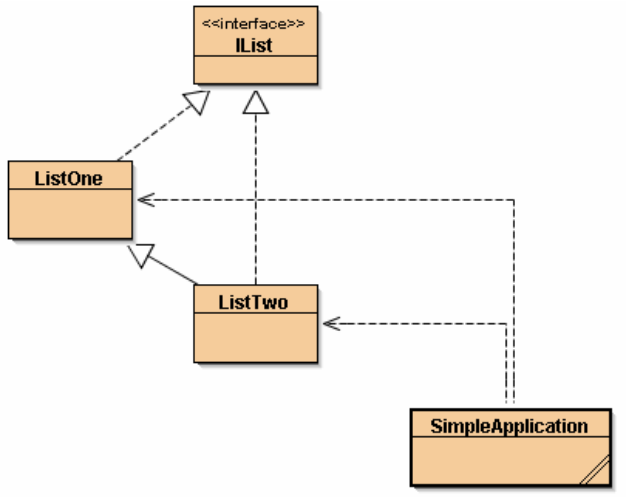
Programmieren 2
Übung Semesterwoche 3

Definieren Sie ein Interface IList für eine Liste, die Objekte des Typs Integer (Wrapper Klasse) aufnehmen kann. Dieses Interface spezifiziert folgende Methoden:

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

    /*
     * 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.
     */
    boolean remove(Integer value);

    /*
     * 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.
     */
    boolean removeValue(int value);

    /*
     * 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.
     */
    boolean removeValues(int value);

    /*
     * Diese Methode gibt die Anzahl Elemente in der Liste zurück.
     */
    int size();

    /*
     * Diese Methode erhält einen ganzzahligen Wert. Sie gibt true zurück, falls ein
     * Integer Objekt mit diesem Wert existiert, false sonst.
     */
    boolean exists(Integer value);

    /*
     * Diese Methode gibt alle Werte der Integer Objekte in der Liste aus.
     */
    void print();
}
```

Implementieren Sie das Interface IList in zwei Klassen:

- ListOne: Diese Klasse verwendet ArrayList, um die Liste zu implementieren.
- ListTwo: Diese Klasse verwendet LinkedList, um die Liste zu implementieren.




```
import java.util.*;

public class ListOne implements IList
{
    private ArrayList<Integer> list;

    public ListOne()
    {
        list = new ArrayList<Integer>();
    }

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

    /*
     * 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)
    {
        //Iterator it = list.iterator();
        int count=0;
        for(Integer i: list)
        {
            if(i.equals(value))
            {
                list.remove(count);
                return true;
            }
            count++;
        }
        return false;
    }
```

```
 /*
     * 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 remove(value);   //???
    }

    /*
     * 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 returnvalue=true;

        for(Integer i: list)
        {
            if(i.equals(value))
            {
                if(remove(value)==false)
                {
                    returnvalue = false;
                }
            }
        }
        return returnvalue;
    }

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

    /*
     * Diese Methode erhält einen ganzzahligen Wert. Sie gibt true zurück, falls ein
     * Integer Objekt mit diesem Wert existiert, false sonst.
     */
    public boolean exists(Integer value)
    {
        for(Integer i: list)
        {
            if(i.equals(value))
            {
                return true;
            }
        }
        return false;
    }

    /*
     * Diese Methode gibt alle Werte der Integer Objekte in der Liste aus.
     */
    public void print()
    {
        System.out.println("Values in list:");

        for(Integer i: list)
        {
            System.out.println(i);
        }
    }
}
```

```
import java.util.*;

public class ListTwo extends ListOne implements IList
{
    private LinkedList<Integer> list;

    public ListTwo()
    {
        list = new LinkedList<Integer>();
    }

}
```

Schreiben Sie eine Anwendung, in der Sie das Interface IList verwenden, um Integer Objekte abzuspeichern. Stellen Sie sicher, dass Sie möglichst wenig verändern müssen, um an Stelle der Implementation ListOne die Implementation ListTwo zu verwenden.

```
public class SimpleApplication
{
    public static void main(String[] args)
    {
        ListOne lo = new ListOne();
        lo.add(10);
        lo.add(20);
        lo.add(30);
        lo.print();

        ListTwo lt = new ListTwo();
        lt.add(10);
        lt.add(20);
        lt.add(30);
        lt.print();
    }
}
```