Universität Augsburg
Institut für Informatik

Deckblatt: Übung zur Vorlesung Informatik 2 SS 2018
Fakultät für Angewandte Informatik
Lehrprofessur für Informatik
Prof. Dr. Lorenz, Marius Brendle, Johannes Metzger

Hinweis: Es sind alle Felder auszufüllen! Abgabe der Übungsblätter immer **mittwochs** (Ausnahme wenn Feiertag: donnerstags) bis **spätestens 12:00 Uhr** in die entsprechend gekennzeichneten Briefkästen der Veranstaltung im Erdgeschoss des Instituts für Informatik (Gebäude N). Zuwiderhandlung wird mit Strafe geahndet! (Punktabzug)

Übungsblatt	

(hier die Nummer des bearbeiteten **Übungsblatts eintragen**)

	Übung 01 (1057 N) Montag 08:15 - 09:45 Uhr (Isabell Rücker)
	Übung 02 (1056 N) Montag 14:00 - 15:30 Uhr (Henning Cui)
	Übung 03 (1057 N) Montag 15:45 - 17:15 Uhr (Josef Kircher)
	Übung 04 (1054 N) Montag 17:30 - 19:00 Uhr (Mosaab Slimani)
	Übung 05 (1057 N) Montag 17:30 - 19:00 Uhr (David Hacker)
	Übung 06 (1055 N) Dienstag 12:15 - 13:45 Uhr (André Schweiger)
X	Übung 07 (1054 N) Dienstag 17:30 - 19:00 Uhr (Benjamin Sertolli)
	Übung 08 (1057 N) Dienstag 17:30 - 19:00 Uhr (Dat Le Thanh)
	Übung 09 (1054 N) Mittwoch 08:15 - 09:45 Uhr (Erik Pallas)
	Übung 10 (1055 N) Mittwoch 08:15 - 09:45 Uhr (Moritz Feldmann)
	Übung 11 (1054 N) Mittwoch 10:00 - 11:30 Uhr (Denise Böhm)
	Übung 12 (1056 N) Donnerstag 08:15 - 09:45 Uhr (Florian Magg)
	Übung 13 (1054 N) Donnerstag 15:45 - 17:15 Uhr (Marvin Drexelius)
	Übung 14 (1054 N) Donnerstag 17:30 - 19:00 Uhr (Patrick Eckert)
	Übung 15 (1057 N) Donnerstag 17:30 - 19:00 Uhr (Alexander Szöke)
	Übung 16 (1057 N) Freitag 08:15 - 09:45 Uhr (Philipp Braml)
	Übung 17 (1054 N) Freitag 10:00 - 11:30 Uhr (Elisabeth Korndörfer)
	Übung 18 (1054 N) Freitag 12:15 - 13:45 Uhr (Philipp Häusele)
	Übung 19 (1056 N) Freitag 12:15 - 13:45 Uhr (Maximilian Demmler)
	Übung 20 (1054 N) Freitag 14:00 - 15:30 Uhr (Florian Straßer)

(hier die eingeteilte **Übungsgruppe ankreuzen**)

Teamnummer	6

(hier die Nummer des eingeteilten **Teams eintragen**)

Tarik Selimovic
Anton Lydike
Dominic Cesnak

(hier die **Vor- und Nachnamen** aller Teammitglieder **eintragen**)

Aufgabe		
Aufgabe		
Aufgabe		
Aufgabe		
Gesamt		(vom Tutor auszufüllen)

Übungsblatt 9

33)

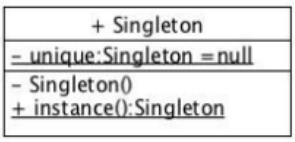
a)

```
package aufgabe33a;

public class A {
    private static A unique = null;

    private A() {
    }

    public static A instance() {
        return unique != null ? unique : (unique = new A());
    }
}
```

b)

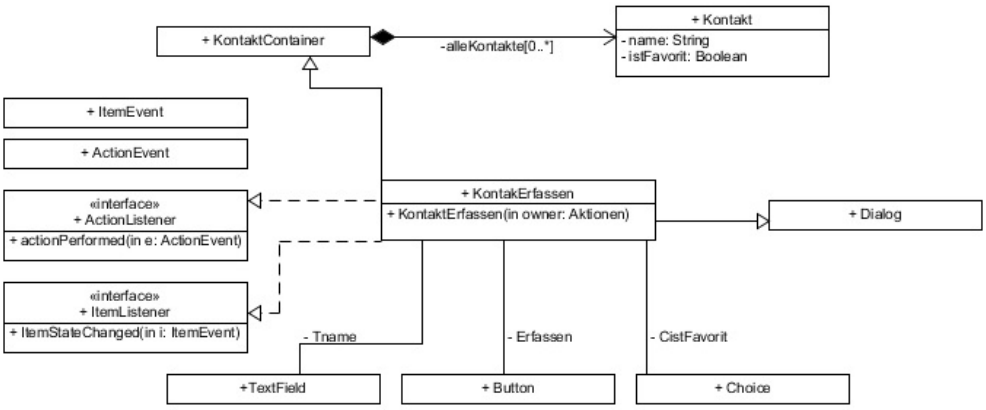
```
package aufgabe33b;

import java.util.*;

public class A implements Observer {

    private int countChanges = 0;
    private B b;

    public A (B b) {
        super();
        this.b = b;
        b.addObserver(this);
    }

    @Override
    public void update(Observable o, Object arg) {
        countChanges++;
    }
}

class B extends Observable {}
```

c)

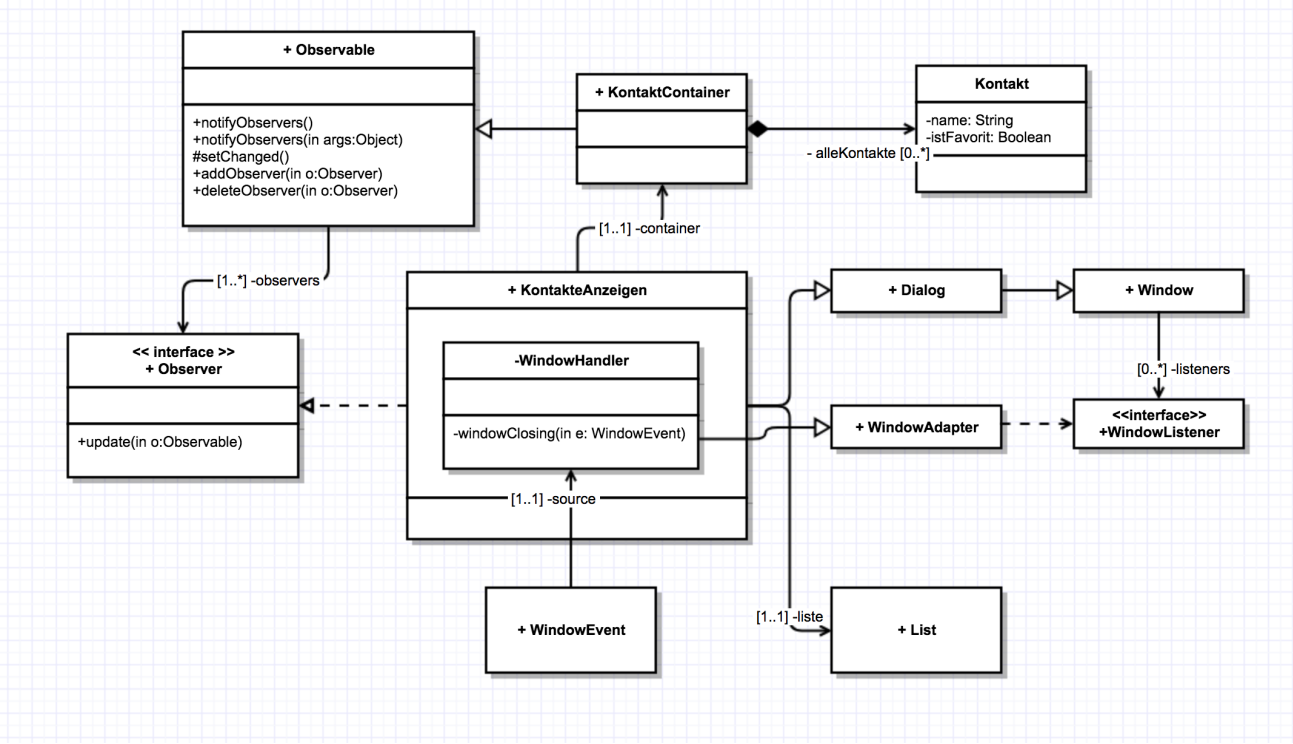


34)

a)


+ KontaktContainer
-alleKontakte[0..*]
+ Kontakt
- name: String
- istFavorit: Boolean
+ ItemEvent
+ ActionEvent
«interface»
+ ActionListener
+ actionPerformed(in e: ActionEvent)
«interface»
+ ItemListener
+ ItemStateChanged(in i: ItemEvent)
+ KontakErfassen
+ KontaktErfassen(in owner: Aktionen)
+ Dialog
- Tname
- Erfassen
- CistFavorit
+TextField
+ Button
+ Choice

b)


+ Observable
+notifyObservers()
+notifyObservers(in args:Object)
#setChanged()
+addObserver(in o:Observer)
+deleteObserver(in o:Observer)
+ KontaktContainer
Kontakt
-name: String
-istFavorit: Boolean
- alleKontakte [0..*]
[1..1] -container
[1..*] -observers
+ KontakteAnzeigen
-WindowHandler
-windowClosing(in e: WindowEvent)
[1..1] -source
<< interface >>
+ Observer
+update(in o:Observable)
+ Dialog
+ Window
[0..*] -listeners
+ WindowAdapter
<<interface>>
+WindowListener
+ WindowEvent
[1..1] -liste
+ List

35)

a)

```
package aufgabe35a;

import java.util.*;
import java.util.zip.DataFormatException;

public class GeometricObjectContainer implements Iterable<GeometricObject> {

    private static GeometricObjectContainer unique = null;
    private ArrayList<GeometricObject> objects;

    private GeometricObjectContainer () {
        objects = new ArrayList<GeometricObject>();
    }

    public static GeometricObjectContainer instance() {
        return unique == null ? (unique = new GeometricObjectContainer()) : unique;
    }

    public void linkObject(GeometricObject o) throws DataFormatException{
        if (objects.contains(o)) throw new DataFormatException("Already added");

        objects.add(o);
    }

    public void unlinkObject(GeometricObject o) {
        objects.remove(o);
    }

    @Override
    public Iterator<GeometricObject> iterator() {
        return objects.iterator();
    }
}

abstract class GeometricObject {}
```

b)

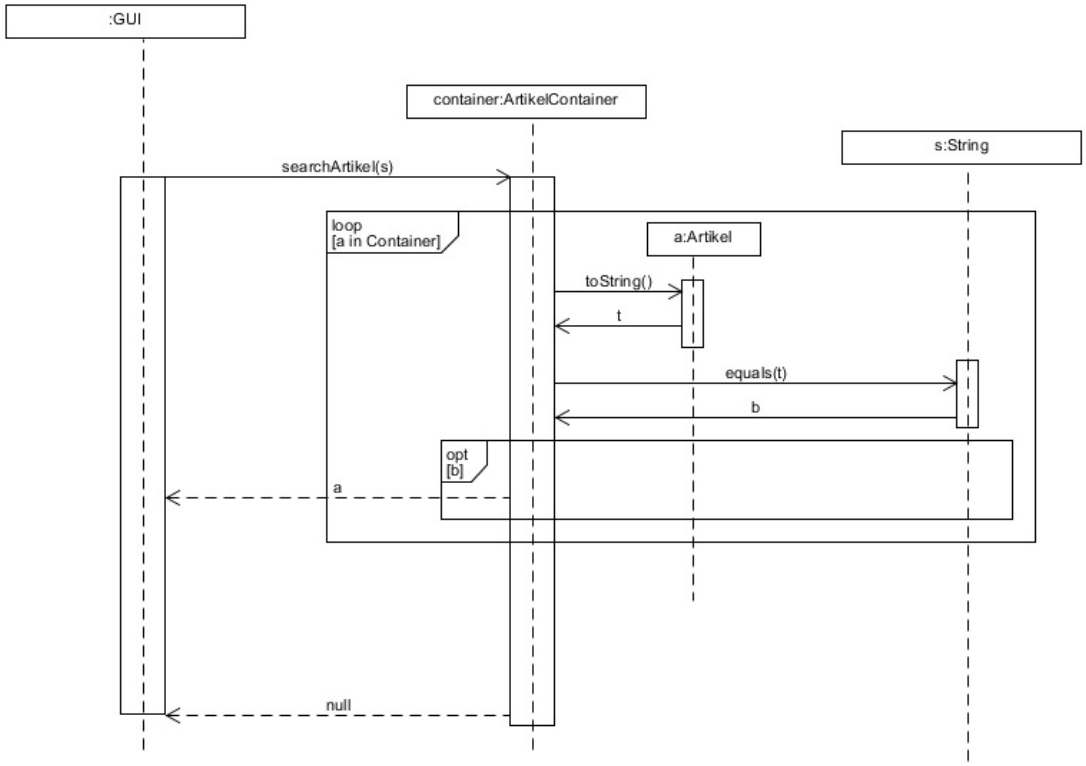
```
package aufgabe35b;

public class BuchContainer {
    private static BuchContainer unique;
    private java.util.ArrayList<Buch> alleBücher = new java.util.ArrayList<Buch>();

    private BuchContainer() {}

    public static BuchContainer instance() {
        return unique != null ? unique : (unique = new BuchContainer());
    }
}

class Buch {}
```

c)

```
package aufgabe35c;

import java.util.*;

public class MitarbeiterContainer implements Iterable<Mitarbeiter> {
    private static MitarbeiterContainer unique = null;

    private ArrayList<Mitarbeiter> alleMitarbeiter;

    private MitarbeiterContainer () {
        alleMitarbeiter = new ArrayList<Mitarbeiter>();
    }

    public static MitarbeiterContainer instance() {
        return unique != null ? unique : (unique = new MitarbeiterContainer());
    }

    public void linkMitabreiter(Mitarbeiter m) {
        if (alleMitarbeiter.contains(m))
        alleMitarbeiter.add(m);
    }

    public void unlinkMitarbeiter(Mitarbeiter m) {
        alleMitarbeiter.remove(m);
    }

    @Override
    public Iterator<Mitarbeiter> iterator() {
        return alleMitarbeiter.iterator();
    }
}

class Mitarbeiter {}
```

36)

a)


:GUI
container:ArtikelContainer
s:String
searchArtikel(s)
loop
[a in Container]
a:Artikel
toString()
t
equals(t)
b
opt
[b]
a
null