

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 Ceanak

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

Aufgabe		
Aufgabe		
Aufgabe		
Aufgabe		
<b>Gesamt</b>		

(vom Tutor auszufüllen)

# Übungsblatt 3

---

9)

---

A)

```
package aufgabe9;

import aufgabe9.Circle;
import aufgabe9.Point;

import java.util.zip.DataFormatException;

public class A {

    public static void main(String args[]) {

        if(args.length % 3 != 0) {
            System.out.println("Die Anzahl der Eingabeparameter ist nicht durch 3 teilbar!");
            System.exit(0);
        }

        int n = 0;
        for (int i = 0; i < args.length; i = i+3) {
            try {
                double x = Double.parseDouble(args[i]);
                double y = Double.parseDouble(args[i+1]);
                double r = Double.parseDouble(args[i+2]);

                Point p = new Point(x,y);
                Circle c = new Circle(r, p);
                n++;
            } catch (NumberFormatException e) {
                System.err.println("Couldn't convert!");
            } catch (DataFormatException e) {
                System.err.println("Could not create circle!" + e.getMessage());
            }
        }

        System.out.println("Es wurde(n) " + n + " Kreis(e) erzeugt.");
    }
}
```

**B)**

```

package aufgabe9;

import aufgabe9.Contact;

import java.util.zip.DataFormatException;

public class B {

    public static void main(String args[]) {
        if (args.length < 1) {
            System.err.println("Fehlende Eingabeparameter!");
            System.exit(0);
        }
        Contact contact;
        try {
            contact = new Contact(args[0]);

        } catch (DataFormatException e) {
            System.err.println("Kontakt konnte nicht erstellt werden: " + e.getMessage());
            return;
        }

        if (args.length > 1) {
            try {
                contact.setBirthday(args[1]);
            } catch (DataFormatException e) {
                System.err.println("Geburtstag konnte nicht gesetzt werden: " + e.getMessage());
            }
        }
        for (int i = 2; i < args.length; i++) {
            try {
                contact.addTelephone(args[i]);
            } catch (DataFormatException e) {
                System.err.println("Telefon konnte nicht gesetzt werden: " + e.getMessage());
            }
        }

        for (String telephone : contact.getTelephone()) {
            System.out.println(telephone);
        }
    }
}

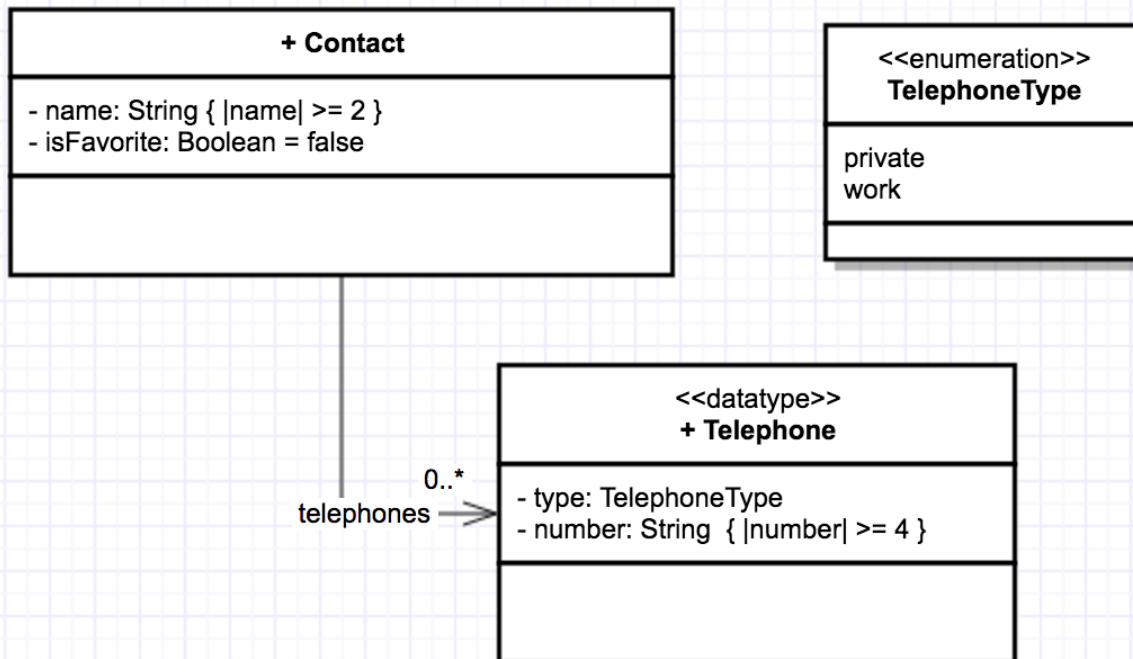
```

**10)****a)**

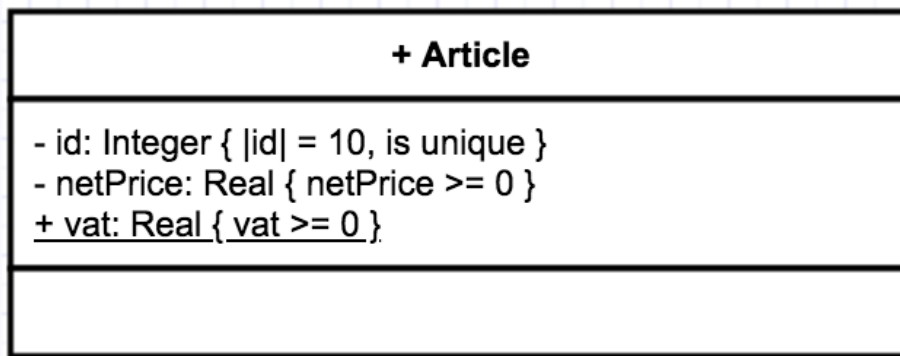
+ Nachricht
- beschriftung : String { 1 <  beschriftung  <= 20 }
- type : MessageType

<<enumeration>> MessageType
synchronous asynchronous create

b)

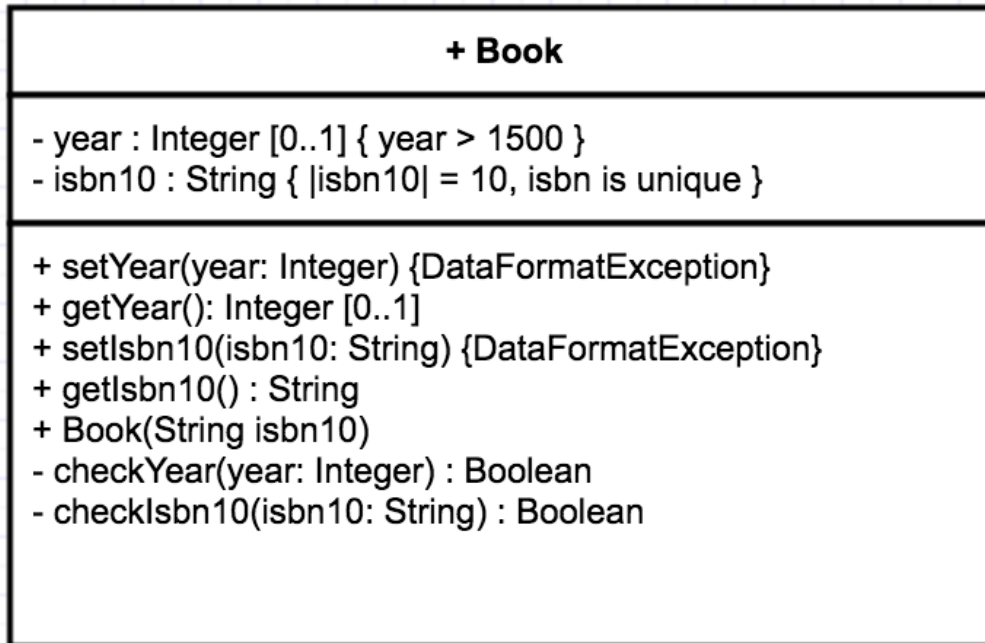


c)

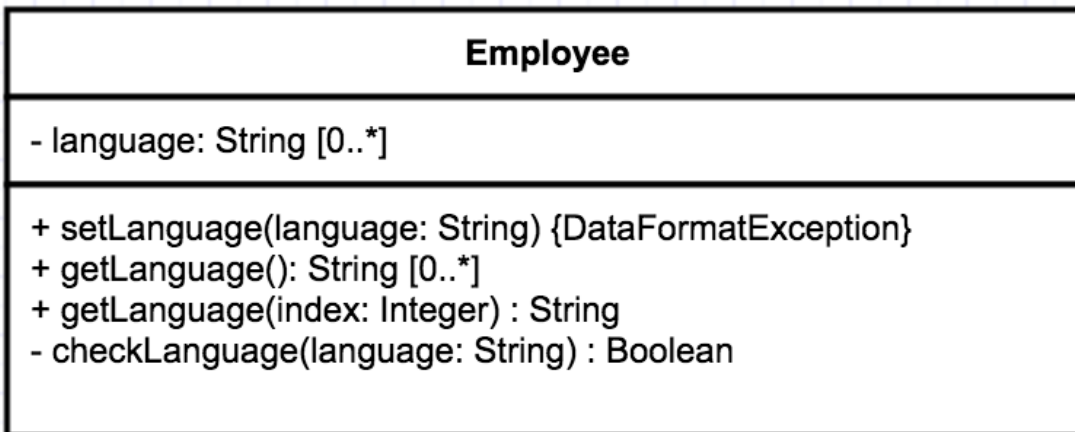


**11)**

a)



b)



## 12)

### Hobby)

```
package aufgabe12;

import java.util.zip.DataFormatException;

public class Hobby {

    private String name;
    private int priority = 1;

    public Hobby(String name) throws DataFormatException {
        this.setName(name);
    }

    public Hobby(String name, int priority) throws DataFormatException {
        this.setName(name);
        this.setPriority(priority);
    }

    public String getName() {
        return name;
    }

    public int getPriority() {
        return priority;
    }

    private void setName(String name) throws DataFormatException {
        if(checkName(name)) {
            this.name = name;
        } else {
            throw new DataFormatException("Name ist ungueltig!");
        }
    }

    public void setPriority(int priority) throws DataFormatException{
        if(checkPriority(priority)) {
            this.priority = priority;
        } else {
            throw new DataFormatException("Priority ist ungueltig!");
        }
    }

    private static boolean checkName(String name) {
        return !name.isEmpty();
    }

    private static boolean checkPriority(int priority) {
        return priority > 0 && priority < 6;
    }
}
```

**Student)**

```
package aufgabe12;

import java.util.ArrayList;
import java.util.List;
import java.util.zip.DataFormatException;

public class Student {
    private List<String> email = new ArrayList<>();

    public void addEmail(String email) throws DataFormatException {
        if(!checkEmail(email) ) {
            throw new DataFormatException("Email ist unguelteig. email=" + email);
        } else {
            if (!this.email.contains(email)) {
                this.email.add(email);
            } else {
                throw new DataFormatException("Email bereits vorhanden. email=" +email);
            }
        }
    }

    public List<String> getEmail() {
        return this.email;
    }

    public void deleteEmail(String email) {
        this.email.remove(email);
    }

    public int getEmailCount() {
        return this.email.size();
    }

    private static boolean checkEmail(String email) {
        return email.indexOf("@") >= 1 && email.length() > (email.indexOf("@") + 1);
    }
}
```