

Hinweis:

Erläutern Sie die Theoriefragen sämtlicher Übungen anhand eines selbst gewählten Beispiels. Dieses Beispiel sollte **nicht** aus der Vorlesung oder aus den Lösungen anderer Übungsaufgaben stammen. Des Weiteren sollte das Beispiel konkret (z.B. Klasse „Person“ anstelle abstrakt Klasse „A“) modelliert werden.

Aufgabe 1: Strukturmodellierung mittels Klassendiagramm - Theoriefragen 1

Wiederholen Sie das Kapitel aus der Vorlesung, das sich mit dem Klassendiagramm beschäftigt.

- Erläutern Sie die Notation einer Klasse. Gehen Sie dabei auf alle Details ein. Zur Erläuterung dieser Details geben Sie ein Beispiel für eine konkrete Klasse an. (Es ist nicht das Ziel, möglichst viele Attribute und Operationen anzugeben, sondern mit Hilfe weniger Attribute und Operationen sämtliche Notationsmöglichkeiten aufzuzeigen.)
- Erläutern Sie die Notation eines Objekts. (Sie sollten in der Lage sein, ein Beispiel-Objekt für die Klasse aus Aufgabe a) zu modellieren, auch wenn a) von jemand anderem gelöst wurde.)
- Erläutern Sie alle Notationsmöglichkeiten für eine Assoziation anhand eines konkreten Beispiels.
- Illustrieren Sie den Unterschied zwischen 1:1, 1:n und n:m Assoziationen anhand eines **Objektdiagramms**.

Aufgabe 2: Strukturmodellierung mittels Klassendiagramm - Theoriefragen 2

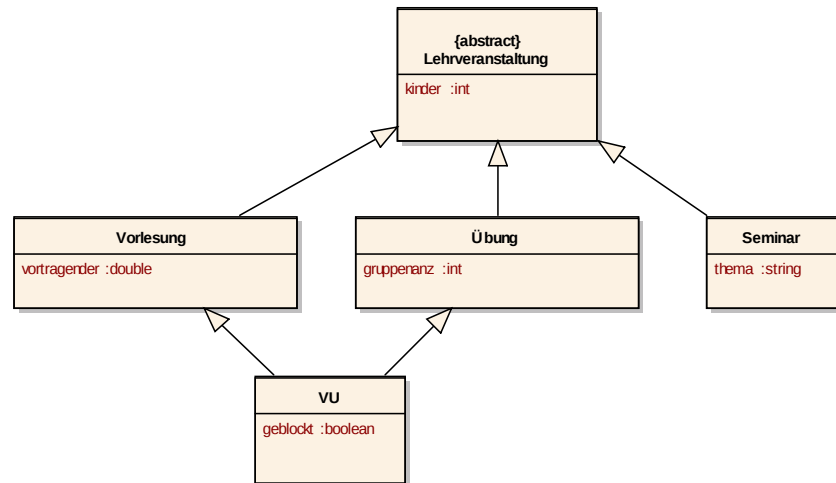
Wiederholen Sie das Kapitel aus der Vorlesung, das sich mit dem Klassendiagramm beschäftigt.

- Was ist eine Assoziationsklasse? Erklären Sie die Notation anhand eines konkreten Beispiels. Wann ist es sinnvoll, eine solche einzusetzen?
- Was ist eine n-äre Assoziation? Erklären Sie die Notation anhand eines konkreten Beispiels.
- Beschreiben Sie den Unterschied zwischen starker und schwacher Aggregation. Erklären Sie die Notation jeweils anhand eines konkreten Beispiels.
- Was ist eine Generalisierung, was versteht man unter Mehrfachvererbung und was ist eine abstrakte Klasse? Erläutern Sie die Notation dieser Konzepte anhand eines oder mehrerer Beispiele.

Aufgabe 3: Assoziation, Aggregation, Komposition und Generalisierung

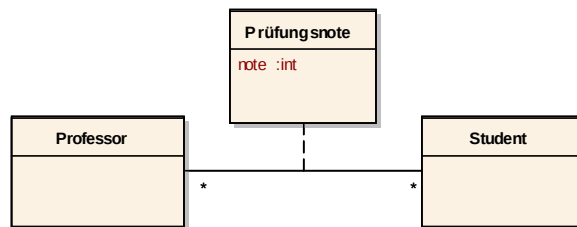
Erläutern Sie die Eigenschaften, die besonderen Varianten und Unterschiede von Assoziation, Aggregation, Komposition und Generalisierung anhand folgender Beispiele. Wählen Sie für die einzelnen Teilaufgaben jeweils das aus Ihrer Sicht am besten geeignete Modellierungskonstrukt.

- Es gibt genau vier verschiedene Lehrveranstaltungen: Vorlesungen, Übungen, Seminare und Vorlesungen mit Übungen. Von jeder Lehrveranstaltung werden die Anzahl der Studierenden sowie die Semesterwochenstunden gespeichert. Von einer Vorlesung wird zusätzlich der Vortragende und von einer Übung die Gruppenanzahl gespeichert. Bei einem Seminar wird zusätzlich das Thema gespeichert. Vorlesungen mit Übungen sind eine spezielle Kombination von Vorlesung und Übung für welche zusätzlich noch gespeichert wird, ob die Lehrveranstaltung geblockt stattfindet oder nicht.

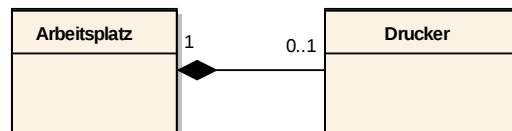


*

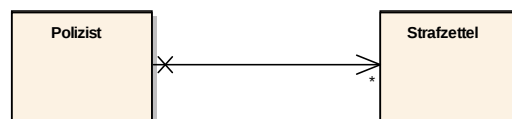
- b) Ein Professor prüft mehrere Studenten und ein Student kann von mehreren Professoren geprüft werden. Bei jeder Prüfung wird die Note gespeichert.



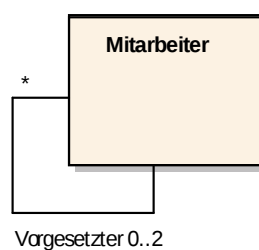
- c) Ein Drucker wird fix einem Arbeitsplatz zugewiesen und kann nicht mehr entfernt werden. Jeder Arbeitsplatz hat höchstens einen Drucker.



- d) Ein Polizist stellt mehrere Strafzettel aus. In unserem Modell kann von einem Strafzettel jedoch nicht auf den Polizisten Rückschluss gezogen werden.

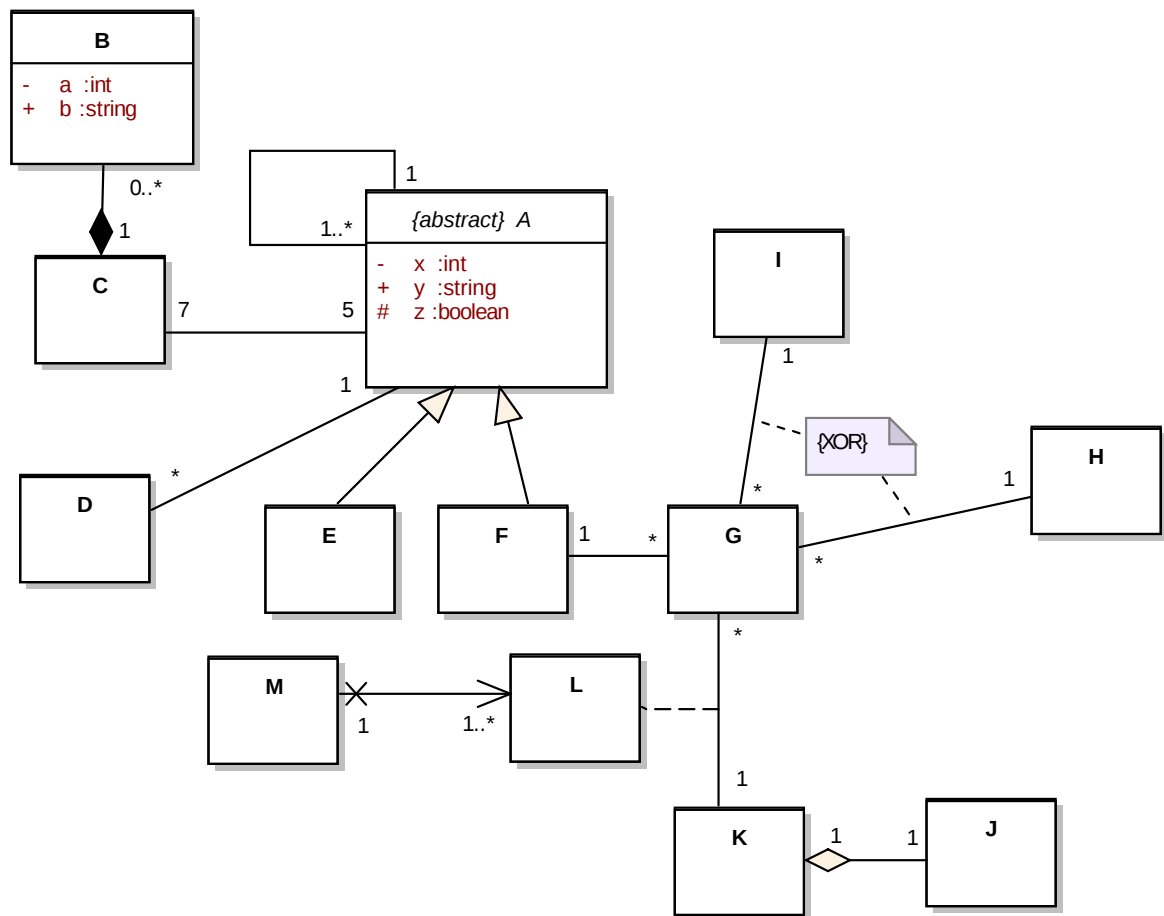


- e) Ein Mitarbeiter kann bis zu zwei Mitarbeiter als Vorgesetzte haben. Es gibt aber auch Mitarbeiter die keinen Vorgesetzten haben.



Aufgabe 4: Wahr oder falsch?

Es ist folgendes UML-Modell gegeben:



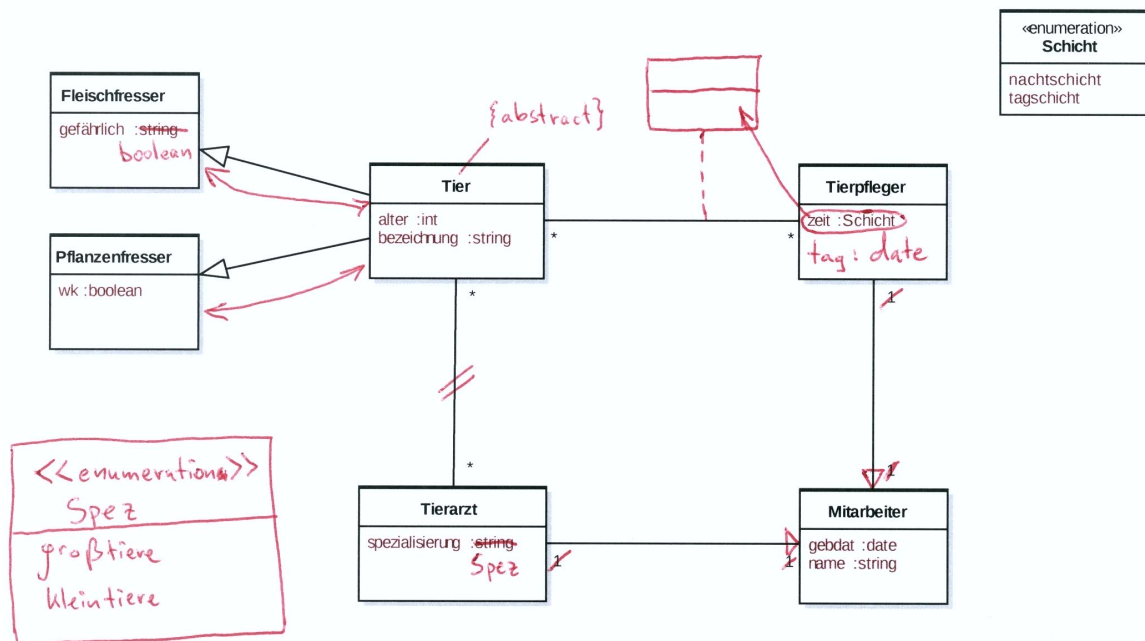
Welche Aussagen treffen zu? Begründen Sie Ihre Antwort!

Ein Objekt von C ist in genau einem Objekt von B enthalten.	<input checked="" type="checkbox"/> ja <input type="checkbox"/> nein
Wenn eine Instanz von K gelöscht wird, bleiben die enthaltenen Instanzen von J erhalten.	<input checked="" type="checkbox"/> ja <input type="checkbox"/> nein
Ein Objekt von G steht entweder in Beziehung zu genau einem Objekt von H oder zu genau einem Objekt von I.	<input checked="" type="checkbox"/> ja <input type="checkbox"/> nein
Ein Objekt von D steht in Beziehung zu einer direkten Instanz von A.	<input checked="" type="checkbox"/> ja <input type="checkbox"/> nein
Eine Instanz von C kann auf die Variable <i>x</i> zugreifen.	<input checked="" type="checkbox"/> ja <input type="checkbox"/> nein
Ein Objekt von A steht in Beziehung zu mindestens einem Objekt von D.	<input checked="" type="checkbox"/> ja <input type="checkbox"/> nein
Ein Objekt von C steht mit genau 5 Objekten von A in Beziehung, ein Objekt von A steht mit genau 7 Objekten von C in Beziehung.	<input checked="" type="checkbox"/> ja <input type="checkbox"/> nein
Ein Objekt vom E kann in Beziehung zu Objekten von F stehen.	<input checked="" type="checkbox"/> ja <input type="checkbox"/> nein
Ein Objekt von K kann in einem Objekt von J enthalten sein, muss aber nicht.	<input checked="" type="checkbox"/> ja <input type="checkbox"/> nein
E erbt alle Attribute von C.	<input checked="" type="checkbox"/> ja <input type="checkbox"/> nein
Ein Objekt von L steht in Beziehung zu genau einem Objekt von M und die Beziehung kann von M aus navigiert werden.	<input checked="" type="checkbox"/> ja <input type="checkbox"/> nein
Es gibt genau 7 Objekte von C die zu genau 5 Objekten von A in Beziehung stehen.	<input checked="" type="checkbox"/> ja <input type="checkbox"/> nein
Objekte von D können direkt auf <i>y</i> zugreifen.	<input checked="" type="checkbox"/> ja <input type="checkbox"/> nein
Die Raute bei C wird als <i>starke Aggregation</i> bezeichnet.	<input checked="" type="checkbox"/> ja <input type="checkbox"/> nein
Ein Objekt von E kann in Beziehung zu mehreren Objekten von F stehen.	<input checked="" type="checkbox"/> ja <input type="checkbox"/> nein

Aufgabe 5: Fehler finden

Gegeben ist folgendes UML Klassendiagramm. Bei der Modellierung sind leider einige Fehler passiert. Finden Sie die Fehler und korrigieren Sie diese im Diagramm.

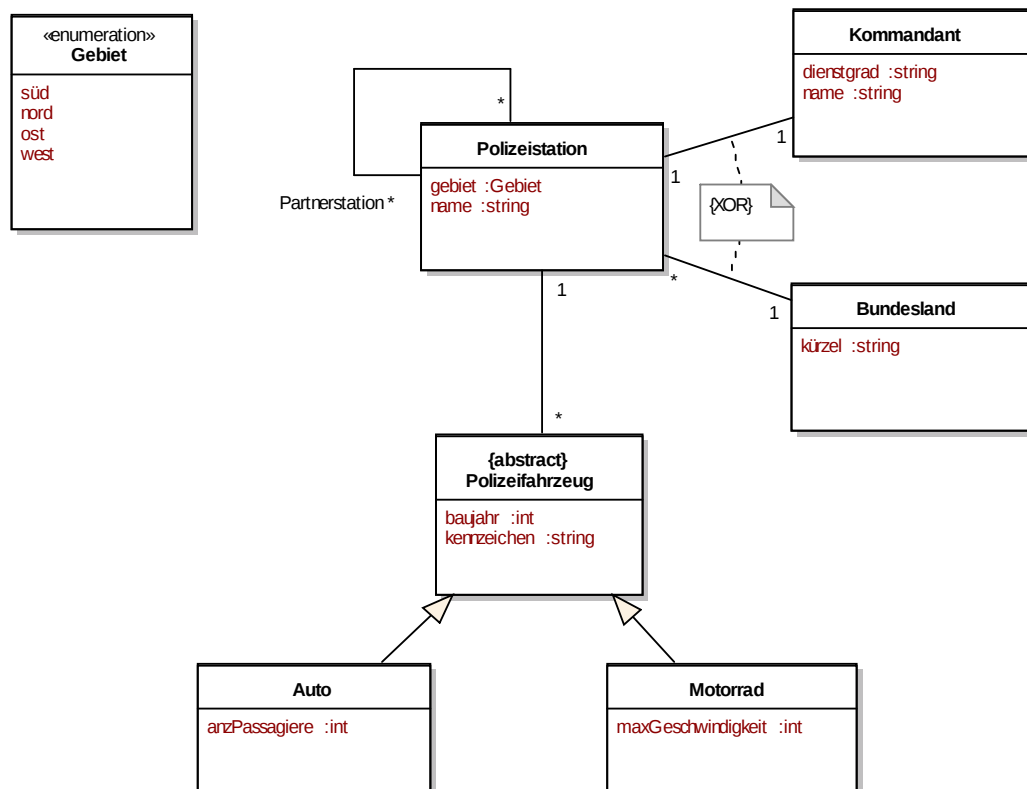
Von allen Tieren wird deren Bezeichnung und Alter gespeichert. Von Tieren gibt es genau zwei Arten, Fleischfresser und Pflanzenfresser. Für Fleischfresser speichern wir ob sie gefährlich für den Menschen sind, für Pflanzenfresser speichern wir, ob diese Widerkauer sind oder nicht. Es gibt unter anderem zwei Arten von Mitarbeitern, Tierpfleger und Tierärzte. Für jeden Mitarbeiter speichern wir Name und Geburtsdatum. Für Tierärzte speichern wir zusätzlich deren Spezialisierung, wobei es hier genau zwei Arten gibt, Großtiere und Kleintiere. Jeder Tierpfleger kann mehrere Tiere betreuen und jedes Tier kann von mehreren Tierpflegern betreut werden. Dabei wird für den Tierpfleger für jedes betreute Tier gespeichert, an welchem Tag und in welcher Schicht, das Tier betreut wurde. Hier wird entweder die Nachtschicht oder die Tagschicht eingetragen.



Aufgabe 6: Klassendiagramm modellieren

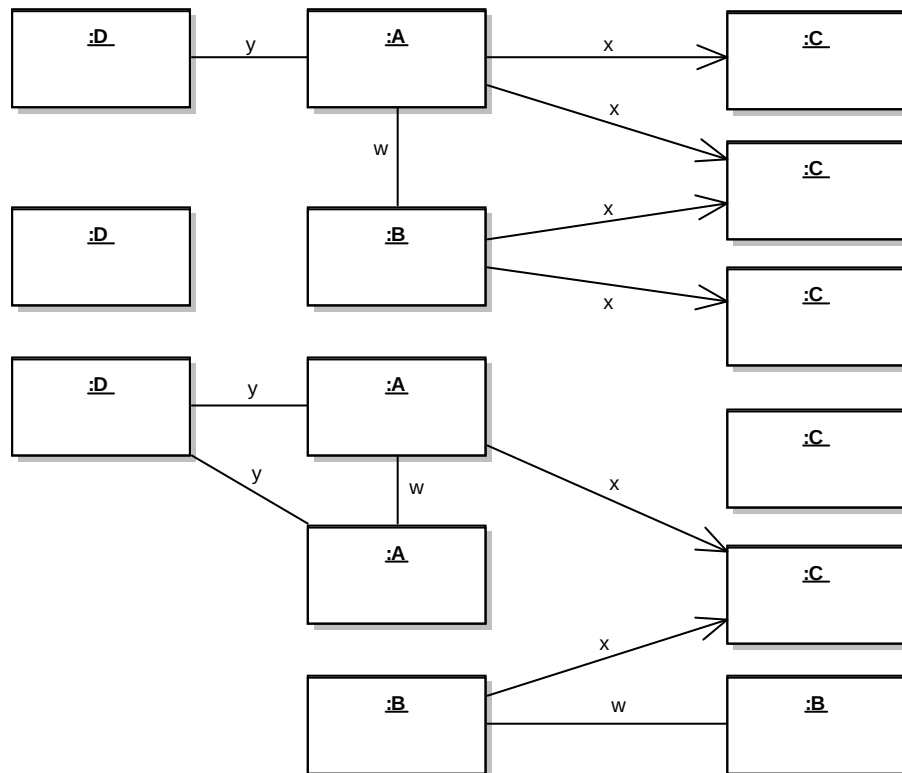
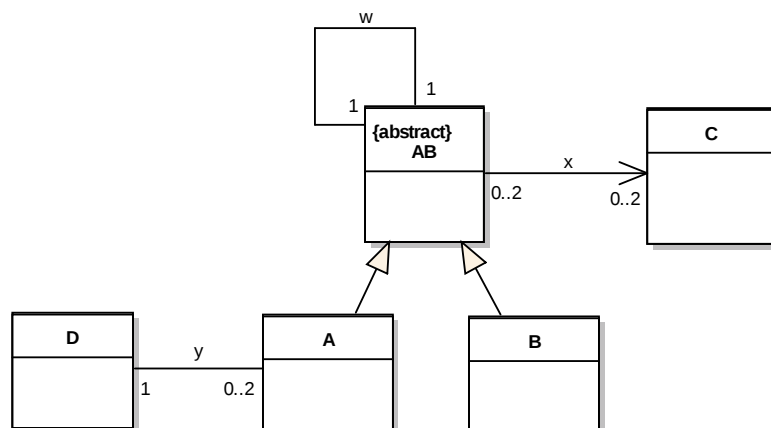
Modellieren Sie folgenden Sachverhalt als UML-Klassendiagramm. Geben Sie in Ihrem Diagramm die beschriebenen (abstrakten) Klassen und ihre Attribute (inklusive Typen), sowie die Assoziationen (inklusive Multiplizitäten, Aggregationen, etc.) an.

Von jedem Polizeifahrzeug wird das Kennzeichen und das Baujahr gespeichert. Es gibt genau zwei Arten von Polizeifahrzeugen, Autos und Motorräder. Von Autos wird zusätzlich die Anzahl der möglichen Passagiere gespeichert, von Motorrädern deren Maximalgeschwindigkeit. Jedes Polizeifahrzeug wird von genau einer Polizeistation verwaltet. Eine Polizeistation hat einen Namen sowie ein Einsatzgebiet—Nord, Süd, West oder Ost. Jede Polizeistation kann mehrere Partnerstationen haben, welche unterstützend bei Einsätzen helfen. Eine Polizeistation ist entweder einem Kommandanten unterstellt (von dem wir den Namen und Dienstgrad speichern) oder einem Bundesland, das mit seinem Länderkürzel erfasst wird. Ein Kommandant leitet dabei genau eine Station.

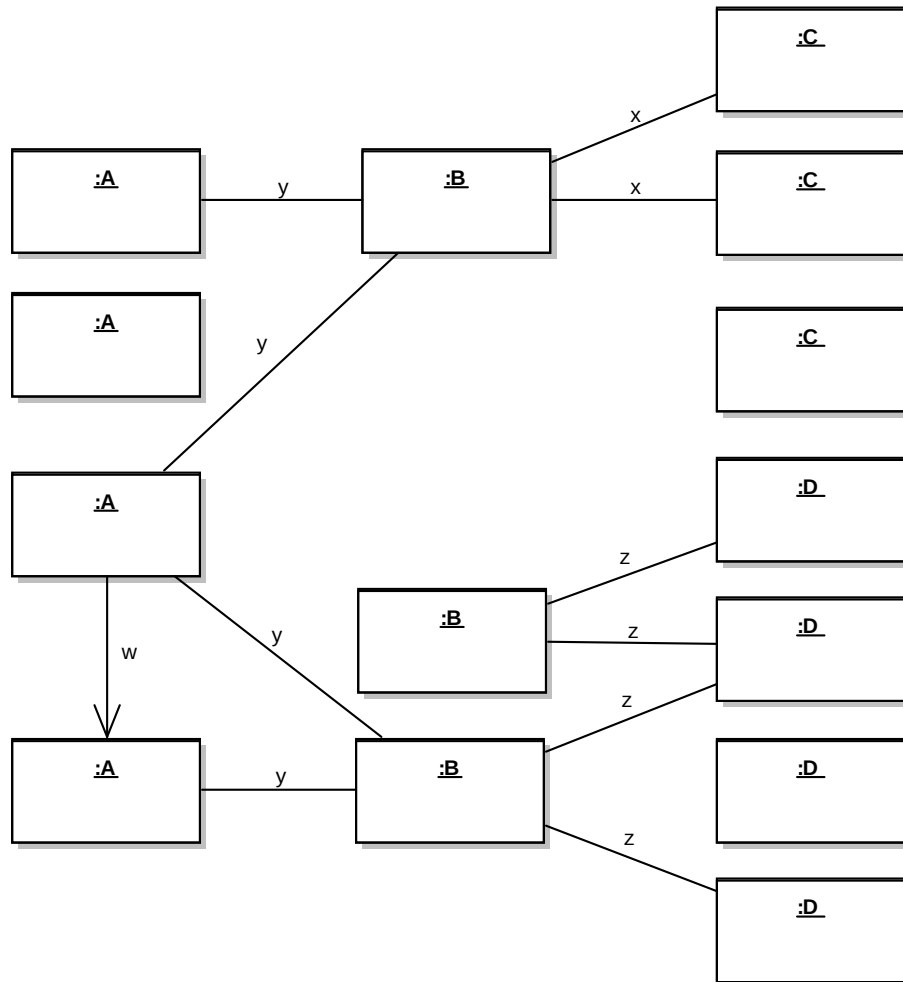


Aufgabe 1: Objektdiagramm

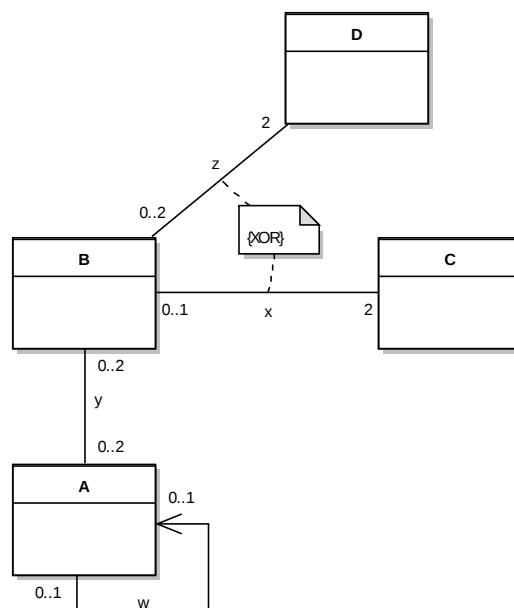
Entwerfen Sie zwei Klassendiagramme, zu denen nachfolgende Objektdiagramme konform sind. Wählen Sie die Kardinalitäten an den Assoziationsenden möglichst genau. Sie können davon ausgehen, dass diese Objektdiagramme die minimal und die maximal mögliche Anzahl an Beziehungen mit Objekten einer anderen Klasse darstellen. Der Name jeder Beziehung ist im Klassendiagramm eindeutig (es sollen also keine Beziehungen denselben Namen haben). Weiters sollen mögliche Generalisierungen bzw. XOR-Beziehungen erkannt werden.

• **Objektdiagramm 1:****Klassendiagramm 1:**

- Objektdiagramm 2:



Klassendiagramm 2:



Aufgabe 2: Vergleich von Klassendiagrammausschnitten

Erklären Sie den Unterschied zwischen folgenden Klassendiagrammausschnitten:

a)



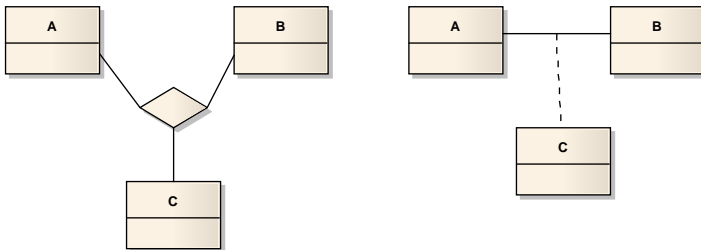
b)



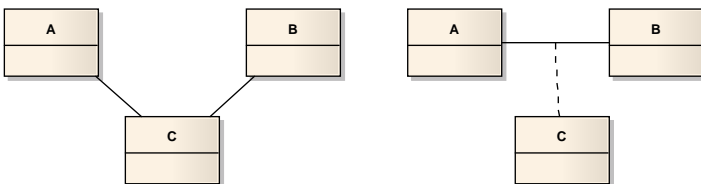
c)



d)



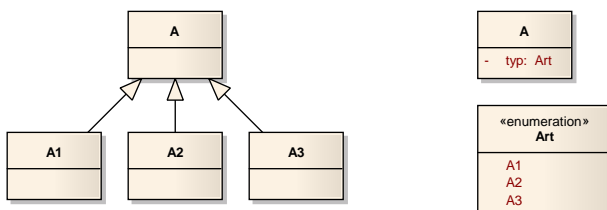
e)



f)



g)



Aufgabe 3: Reverse Engineering

Gegeben sei der unten angeführte Java ähnliche Code. Führen Sie ein Reverse Engineering des Codes in ein UML Klassendiagramm durch. Das heißt, Sie müssen ein UML Klassendiagramm entwerfen, das semantisch dem Java Code entspricht. Bilden Sie Referenzen möglichst durch Assoziationen ab.

```
abstract class Person {
    public String name;
}

class Mieter extends Person {
    private Date gebDat;
    private Mietobjekt mo;
    public Mietobjekt [] interessierteObjekte;
}

class Vermieter extends Person {
    private String wohnsitz;
    private Mietobjekt [] objekte;
}

class Mitarbeiter {
    public String name;
    public String telNr;
    private Anstellung anst;

    public Anstellung getAnst() {
        return anst;
    }
    public void setAnst(Anstellung anst) {
        this.anst = anst;
    }
}

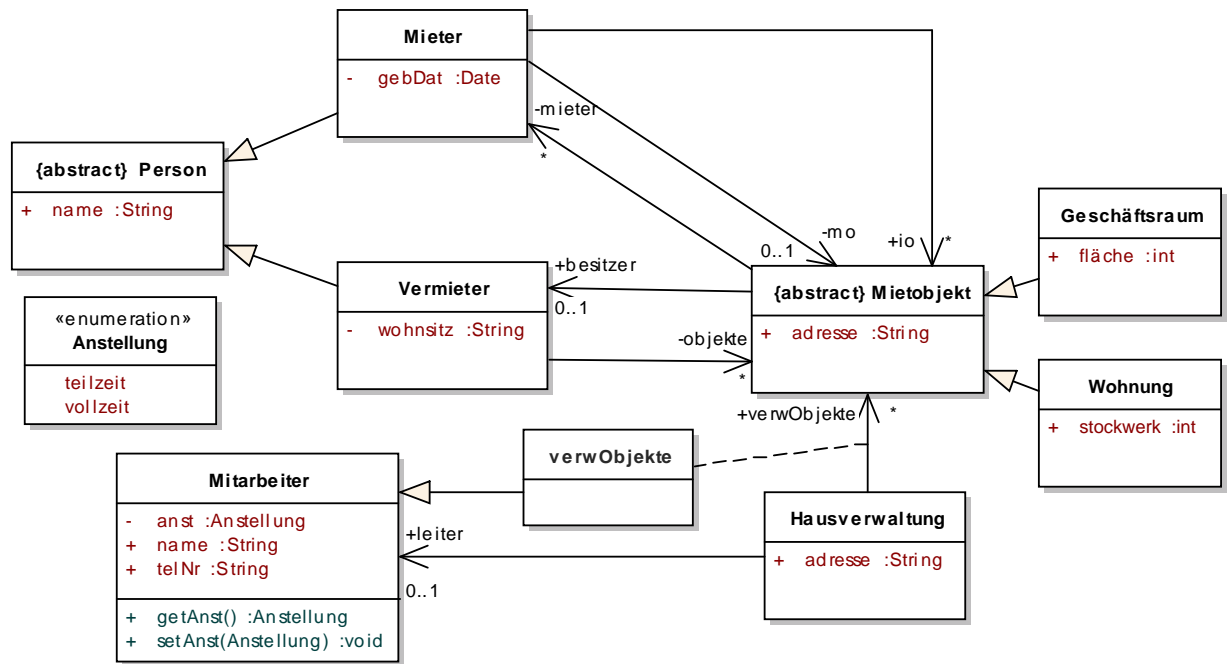
abstract class Mietobjekt {
    public String adresse;
    private Vermieter besitzer;
    private Mieter [] mieter;
}

class Wohnung extends Mietobjekt {
    public int stockwerk;
}

class Geschäftsraum extends Mietobjekt {
    public int fläche;
}

class Hausverwaltung {
    public String adresse;
    public Mitarbeiter leiter;
    public Hashtable verwObjekte;
        // Key: Mietobjekt
        // Value: verantwortlicher (Typ: Mitarbeiter)
}

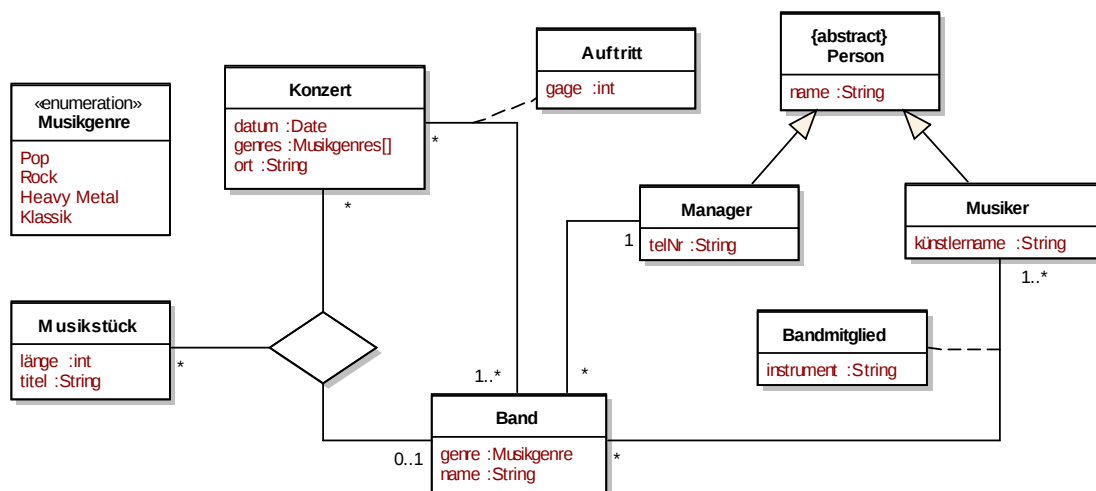
enum Anstellung { vollzeit , teilzeit }
```



Aufgabe 4: Konzert

Modellieren Sie folgenden Sachverhalt mittels UML Klassendiagramm gemäß nachstehenden Informationen. Verwenden Sie bei Bedarf abstrakte Klassen, Enumerationen und Aggregationen. Datentypen müssen Sie nicht darstellen, sofern diese nicht explizit im Text spezifiziert wurden.

Von jedem Konzert werden das Datum sowie der Veranstaltungsort gespeichert. Weiters wird vermerkt, um welche Musikgenres bei dem Konzert vertreten sind. Dabei gibt es Klassik, Rock, Pop und Heavy Metal. Auf einem Konzert können mehrere Bands spielen, die wiederum auf mehreren Konzerten spielen können. Dabei wird die Höhe der Gage, die eine Band für einen Auftritt erhält, gespeichert. Für eine Band speichern wir deren Namen sowie deren Musikgenre. Von Personen speichern wir deren Namen. Es gibt zwei Arten von Personen, Musiker und Manager. Von Musikern vermerken wir zusätzlich deren Künstlernamen und von Managern deren Telefonnummer. Ein Musiker kann in mehreren Bands spielen welche wiederum aus mehreren Musikern bestehen kann. Für die Zugehörigkeit eines Musikers zu einer Band speichern wir das von ihm gespielte Instrument. Eine Band wird von genau einem Manager betreut und ein Manager kann mehrere Bands betreuen. Für ein Musikstück speichern wir dessen Titel und Länge in Sekunden. Auf einem Konzert kann ein Musikstück von höchstens einer Band gespielt werden. Jedoch kann eine Band auf einem Konzert mehrere Musikstücke spielen. Weiters kann ein Musikstück von einer Band auf mehreren Konzerten gespielt werden.



Aufgabe 5: Regierung I

Bilden Sie die folgenden Sachverhalte mit einem Klassendiagramm ab:

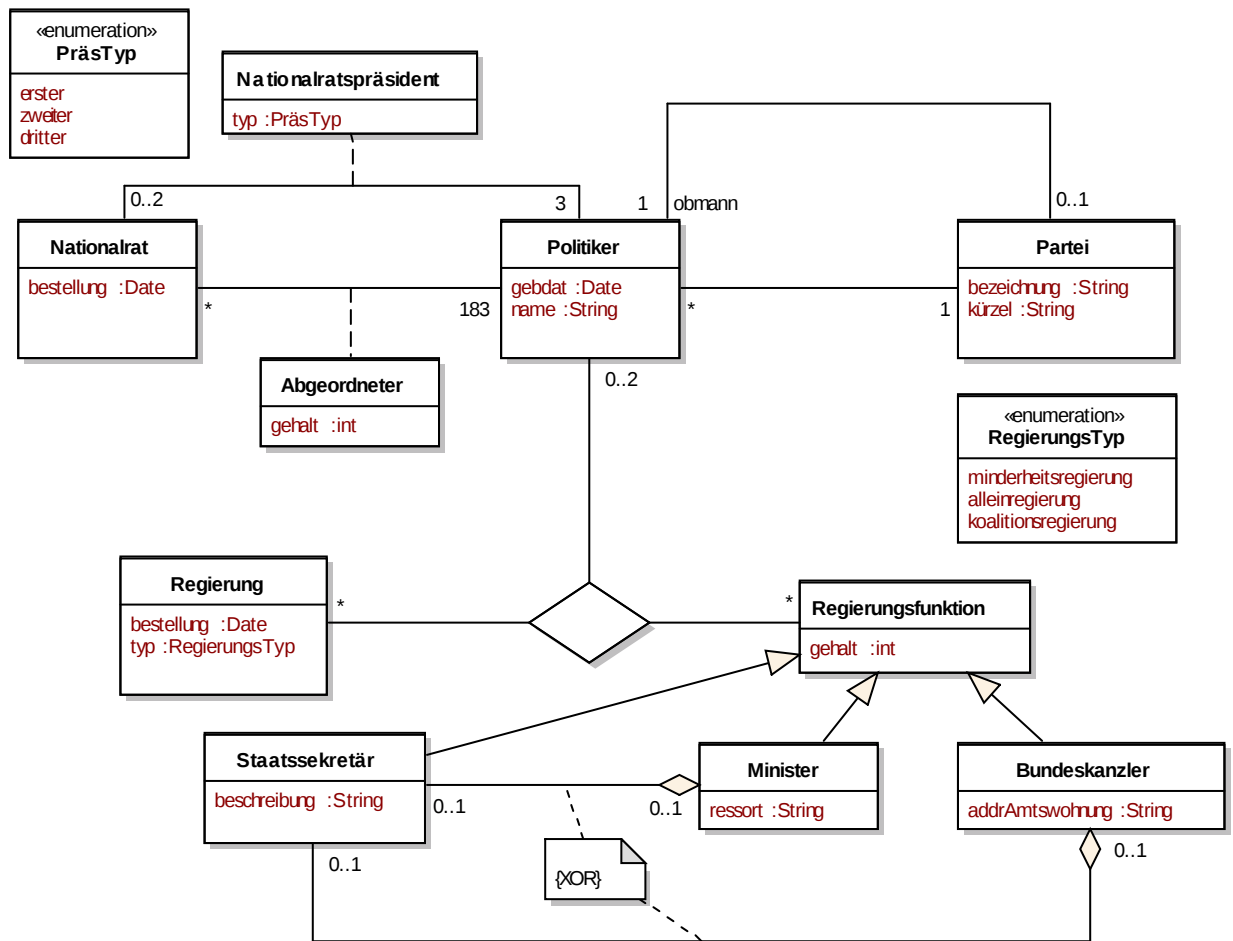
Von Politikern speichern wir Name und Geburtsdatum. Jeder Politiker ist Mitglied von genau einer Partei. Eine Partei kann wiederum mehrere Politiker als Mitglieder haben. Von Parteien speichern wir deren Bezeichnung sowie deren Kürzel. Jede Partei hat einen Obmann der Politiker ist. Von einem Nationalrat speichern wir den Zeitpunkt dessen Bestellung. Ein Nationalrat hat genau 183 Abgeordnete die Politiker sind. Von jedem dieser Abgeordneten erfassen wir dessen Gehalt. Ein Politiker kann in mehreren Nationalräten bestellt sein. Weiters hat ein Nationalrat genau drei Politiker die Nationalratspräsidenten sind. Ein Politiker kann in höchstens zwei Nationalräten als Präsident fungieren. Für jeden dieser Präsidenten speichern wir ob der Präsident erster, zweiter oder dritter Nationalratspräsident ist.

Aufgabe 6: Regierung II

Erweitern Sie das Klassendiagramm aus Aufgabe 5 wie folgt:

Von einer Regierungsfunktion speichern wir deren Gehalt. Es gibt zwei verschiedene solche Funktionen, Minister und Bundeskanzler. Von der Funktion Minister erfassen wir zusätzlich das zugewiesene Ressort und für die Funktion Bundeskanzler erfassen wir die Adresse der Amtswohnung. Staatssekretäre sind formell nicht eine Regierungsfunktion, jedoch wollen wir auch diese Positionen als solche modellieren. Wir speichern von dieser Funktion zusätzlich eine Beschreibung der Aufgaben. Ein Minister und der Bundeskanzler haben höchstens einen Staatssekretär zugeordnet. Ferner ist ein Staatssekretär entweder einem Minister oder dem Bundeskanzler zugeordnet. Von einer Regierung speichern wir den Zeitpunkt der Bestellung sowie den Typ der Regierung. Hierbei gibt es Minderheitsregierungen, Alleinregierungen und Koalitionsregierungen. Ein Politiker kann eine Regierungsfunktion in mehreren Regierungen übernehmen. Weiters kann in einer Regierung eine Regierungsfunktion von bis zu zwei Politikern übernommen werden. Umgekehrt kann ein Politiker in einer Regierung mehrere Funktionen übernehmen.

Lösung für I und II:



Aufgabe 1: Verhaltensmodellierung mittels Sequenzdiagramm

Wiederholen Sie das Kapitel aus der Vorlesung, das sich mit Sequenzdiagrammen beschäftigt.

- Welche vier Arten von Interaktionsdiagrammen gibt es? Beschreiben Sie diese kurz. Wofür werden Interaktionsdiagramme eingesetzt?
- Wie ist ein Sequenzdiagramm prinzipiell aufgebaut? Welche Elemente kann es enthalten?
- Beschreiben Sie die Unterschiede zwischen synchronen und asynchronen Nachrichten.
- Was ist ein aktives Objekt, was ist ein passives Objekt? Wie unterscheiden sich diese?

Aufgabe 2: Verhaltensmodellierung mittels Sequenzdiagramm

Wiederholen Sie das Kapitel aus der Vorlesung, das sich mit Sequenzdiagrammen beschäftigt.

- Was ist eine Zustandsinvariante im Kontext des Sequenzdiagramms? Wie können Zeiteinschränkungen angegeben werden?
- Welche Arten von Verzweigungen und Schleifen können in Sequenzdiagrammen auftreten? Beschreiben Sie die entsprechenden Operatoren.
- Welche Operatoren stehen im Sequenzdiagramm zur Verfügung, um parallele Abläufe zu realisieren bzw. um Ordnungen im Ablauf festzulegen?
- Erklären Sie die kombinierten Fragmente aus der Gruppe "Filterungen und Zusicherungen".

Aufgabe 3: Kommunikation im Sequenzdiagramm

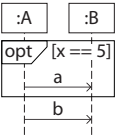
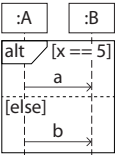
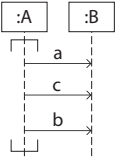
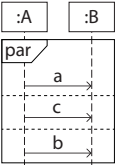
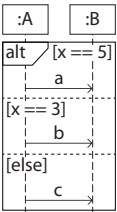
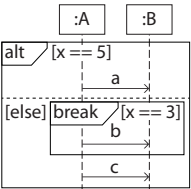
Handelt es sich in den folgenden Kommunikationsszenarien um synchrone oder asynchrone Kommunikation? Identifizieren Sie die involvierten Interaktionspartner!

Eine Mitarbeiterin informiert einen Gast über die Zimmerpreise.	<input checked="" type="checkbox"/> synchron
Auf einem Infoscreen im Wellnessbereich werden die neuesten Massagen und Behandlungen eingeblendet.	<input checked="" type="checkbox"/> asynchron
Eine Mitarbeiterin begleitet die Hotelgäste auf ihr Zimmer.	<input checked="" type="checkbox"/> synchron
Über das Zimmerradio ertönt beruhigende Entspannungsmusik.	<input checked="" type="checkbox"/> asynchron
Der Gast schaltet sämtliche Lichtquellen des Hotelzimmers per Touch-Screen ein.	<input checked="" type="checkbox"/> synchron
Ein Gast nutzt den Computer in der Hotellobby, um einem Freund eine E-Mail zu senden.	<input checked="" type="checkbox"/> asynchron
Ein Gast beschwert sich per Telefon bei der Rezeption, dass ein Bademantel fehlt.	<input checked="" type="checkbox"/> synchron
Ein Gast füllt einen Fragebogen zur Qualität des Hotels aus.	<input checked="" type="checkbox"/> asynchron
Ein Ehepaar diskutiert, ob es zuerst die Nachmittagsjause oder den Fitnessraum in Anspruch nehmen soll.	<input checked="" type="checkbox"/> synchron
Vor der Abreise bezahlt ein Gast die Hotelrechnung mit Kreditkarte.	<input checked="" type="checkbox"/> synchron

Aufgabe 4: Kombinierte Fragmente

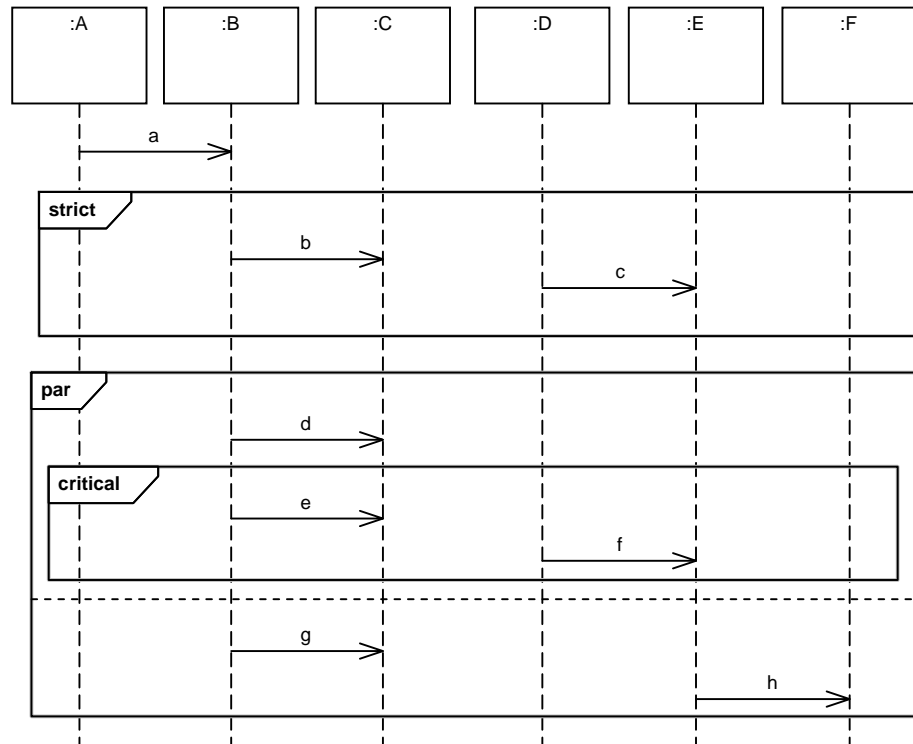
a) Äquivalenzen

Gegeben sind jeweils zwei Ausschnitte eines Sequenzdiagramms. Kreuzen Sie an, ob die beiden Ausschnitte jeweils „äquivalent“ oder „nicht äquivalent“ sind. Begründen Sie warum.

- a)   ☐ äquivalent
☒ nicht äquivalent
- a)   ☒ äquivalent
☐ nicht äquivalent
- a)   ☒ äquivalent
☐ nicht äquivalent

b) Berechnung von Traces

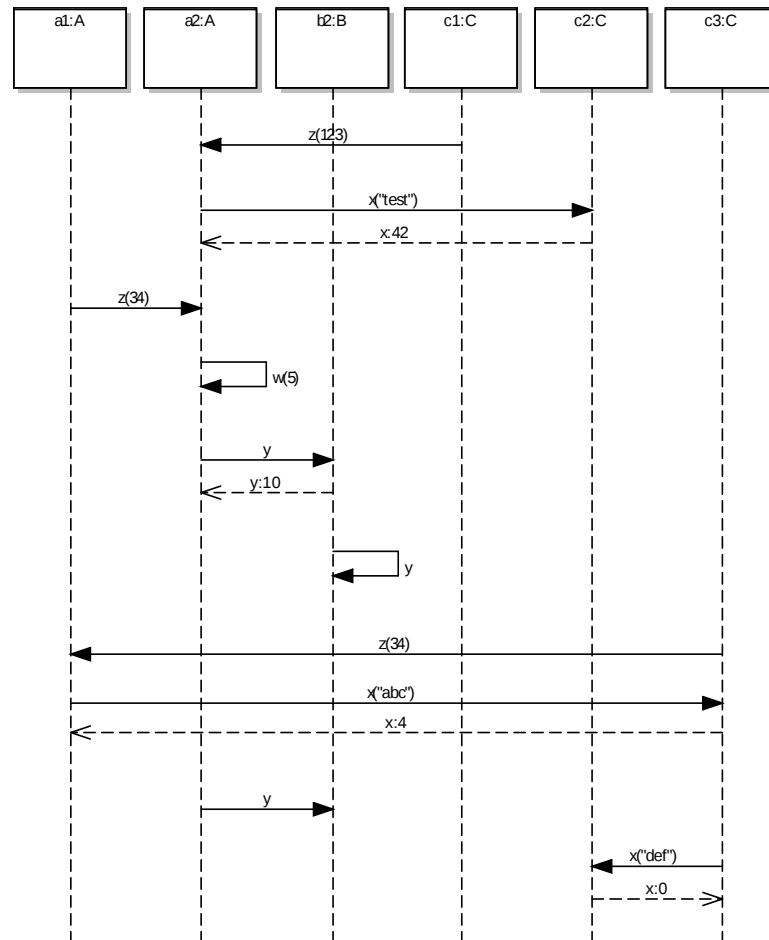
Beschreiben Sie alle möglichen Ereignisfolgen des folgenden Diagramms.



- Default-Reihenfolge für a;
- strict: Die Reihenfolge ist strikt einzuhalten, d.h. b wird vor c ausgeführt.
- par: Die Nachrichten von verschiedenen par-Abschnitten können beliebig kombiniert werden, solange die folgenden Einschränkungen berücksichtigt werden:
- Erster par-Abschnitt: critical; die Operationen dürfen nicht unterbrochen werden.
- Zweiter par-Abschnitt: innerhalb des Abschnittes gilt default Reihenfolge. Sofern Nachrichten nicht kausal voneinander abhängen, ist jede Reihenfolge gestattet.

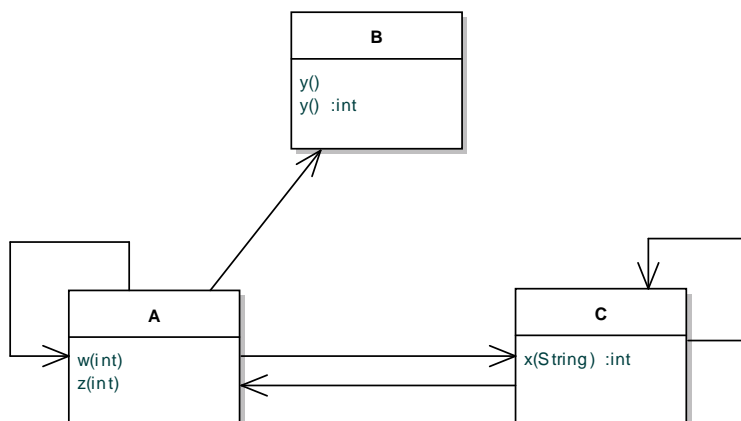
Aufgabe 5: Klassendiagramm aus Sequenzdiagramm

Gegeben ist folgendes Sequenzdiagramm:



Vervollständigen Sie nachfolgendes Klassendiagramm.

- Operationsdefinitionen mit Typangaben, soweit ersichtlich
- Beziehungen zwischen Klassen in Form von navigierbaren Assoziationen: Zeichnen Sie nur Navigationsrichtungen ein, die aus dem gegebenen Sequenzdiagramm ersichtlich sind.



Aufgabe 6: Darstellung von Programmabläufen mittels Sequenzdiagramm

Stellen Sie die Abläufe von folgendem Programm mittels Sequenzdiagramm dar. Modellieren Sie auch allfällige Antwortnachrichten.

Sie können davon ausgehen, dass alle nicht explizit deklarierten Variablen bereits deklariert und initialisiert sind. Der Ausdruck „...“ markiert vernachlässigte Codeteile, die nicht modelliert werden müssen.

```
class Main {
    ...
    Worker w1 = new Worker();
    w1.run();

    print("We will buy the cheaper one!");

    q = w1.getQuantity();

    if(q < 1) {
        print("Invalid quantity");
        exit; // Programm wird beendet
    }

    Article a1 = new Article();
    pa1 = a1.getPrice(w1);

    Article a2 = new Article();
    pa2 = a2.getPrice(w1);

    if (pa1 < pa2) {
        for ( int i = 1; i <= q; i++ ) {
            resNr = w1.reserve(a1);
            res[i] = resNr;
        }
        else
            ...
    }

    private void print(String m) {...}
}

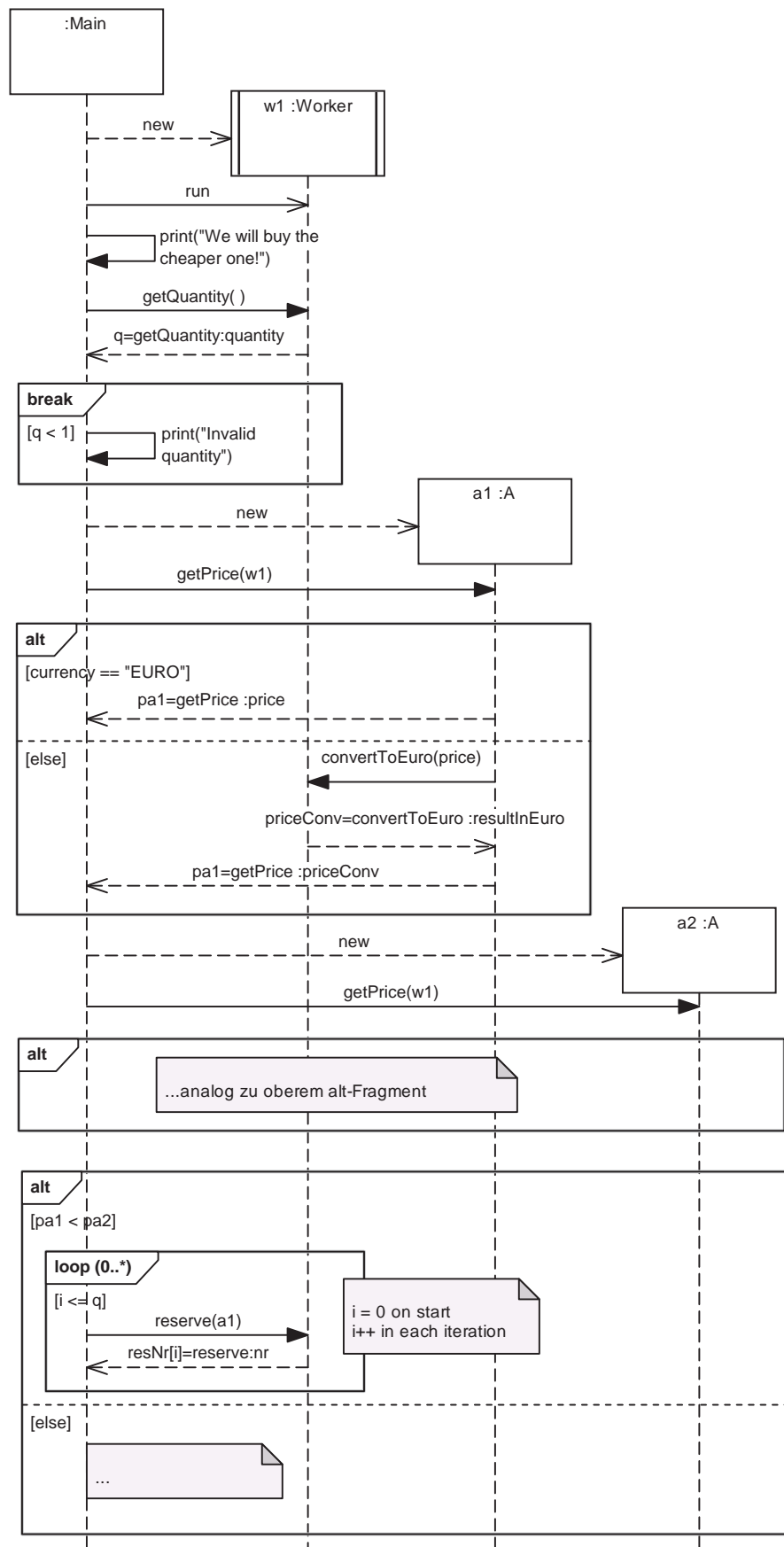
class Worker extends Thread {
    public void run() { }

    public int getQuantity {
        return quantity;
    }

    public double convertToEuro (double amount) {
        double resultInEuro = amount*0.814929509;
        return resultInEuro;
    }

    public int reserve(Article art) {
        ...
        return nr;
    }
}
```

```
class Article {  
    public double getPrice(Worker w1) {  
        if (currency.equals("EURO")) {  
            return price;  
        } else {  
            double priceConv = w1.convertToEuro(price);  
            return priceConv;  
        }  
    }  
}
```



Aufgabe 1: Theoriefragen 1

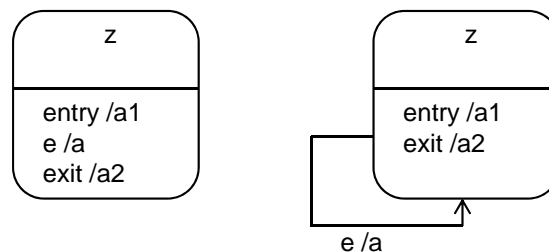
Beantworten Sie folgende Fragen:

- Erklären Sie die Konzepte *Ereignis*, *Bedingung* und *Aktivität*.
- Welche vordefinierten Aktivitäten gibt es innerhalb eines Zustands?
- Wann erfolgt eine Transition (von einem Zustand in einen anderen)?
- Was versteht man unter einem Historischen Zustand? Wann, warum und wie wird er eingesetzt?

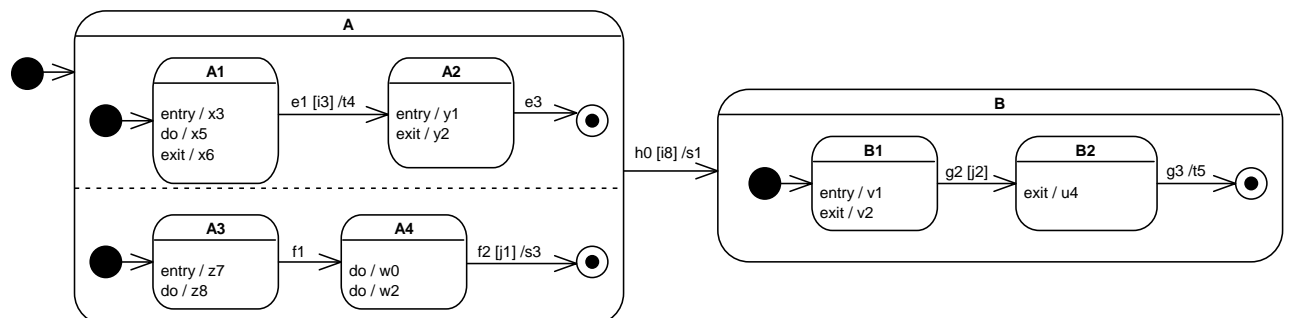
Aufgabe 2: Theoriefragen 2

Beantworten Sie folgende Fragen:

- Erklären Sie das Konzept der UND- sowie der ODER-Verfeinerung.
- Gegeben sind folgende zwei Ausschnitte eines Zustandsdiagramms. Sind die beiden Ausschnitte äquivalent? Begründen Sie Ihre Antwort!



- Gegeben ist das nachfolgende Zustandsdiagramm.



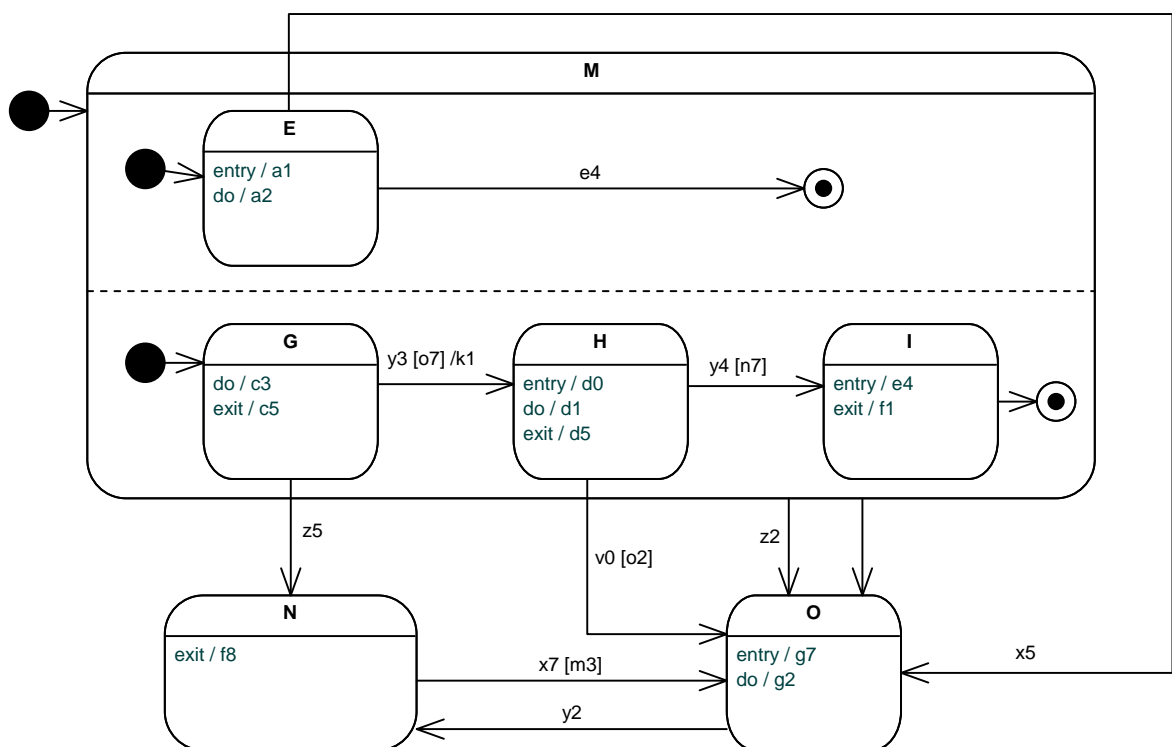
Beantworten Sie folgende Fragen:

- Welche Zustände gibt es in diesem Diagramm?
A, A1, A2, A3, A4, B, B1, B2 sowie die Startzustände (Pseudozustand) und Endzustände
- Welche Ereignisse gibt es in diesem Diagramm?
e1, e3, f1, f2, g2, g3, h0 sowie die vordefinierten Ereignisse entry, do und exit
- Welche Bedingungen gibt es in diesem Diagramm?
i3, i8, j1, j2
- Welche Aktivitäten gibt es in diesem Diagramm?
s1, s3, t4, t5, u4, v1, v2, w0, w2, x3, x5, x6, y1, y2, z7, z8

- In welchem Zustand/welchen Zuständen befindet sich der Automat unmittelbar nach dem Start?
A1 und A3
- In welchem Zustand/welchen Zuständen muss sich der Automat befinden, damit er nach dem Eintritt von h0 in den Zustand B übergeht?
Er geht in den Zustand B über, egal in welchen Zuständen von A er sich vorher befunden hat.
- Gibt es in diesem Diagramm Pseudozustände? Wenn ja, welche?
Ja, die Startzustände sind Pseudozustände.

Aufgabe 3: Allgemeines Verständnis

Gegeben ist das nachfolgende Zustandsdiagramm:



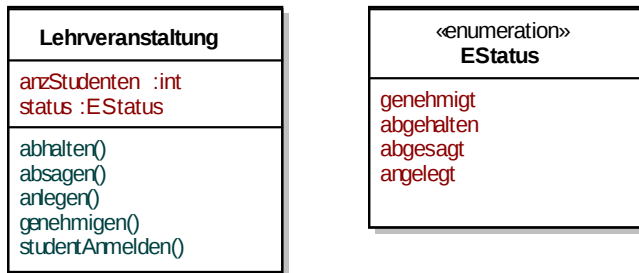
Beantworten Sie folgende Fragen:

- In welchen Kombinationen von Zuständen kann sich das System zu einem Zeitpunkt gleichzeitig befinden?
 - N
 - O
 - E und G
 - E und H
 - E und im Endzustand des Unteren Bereichs
 - Im Endzustand des oberen Bereichs und in G
 - Im Endzustand des oberen Bereichs und in H
 - Im Endzustand des oberen Bereichs und in I

- Welche Möglichkeit(en) gibt es, dass das System vom Zustand M direkt in den Zustand O übergeht?
System befindet sich in beiden Subzustandsfolgen von M am Ende.
oder
Ereignis z2 tritt ein.
oder
System befindet sich (unter anderem) im Zustand H, das Ereignis v0 tritt ein und die Bedingung o2 ist erfüllt.
oder
System befindet sich (unter anderem) im Zustand E und das Ereignis x5 tritt ein.

Aufgabe 4: Lehrveranstaltung

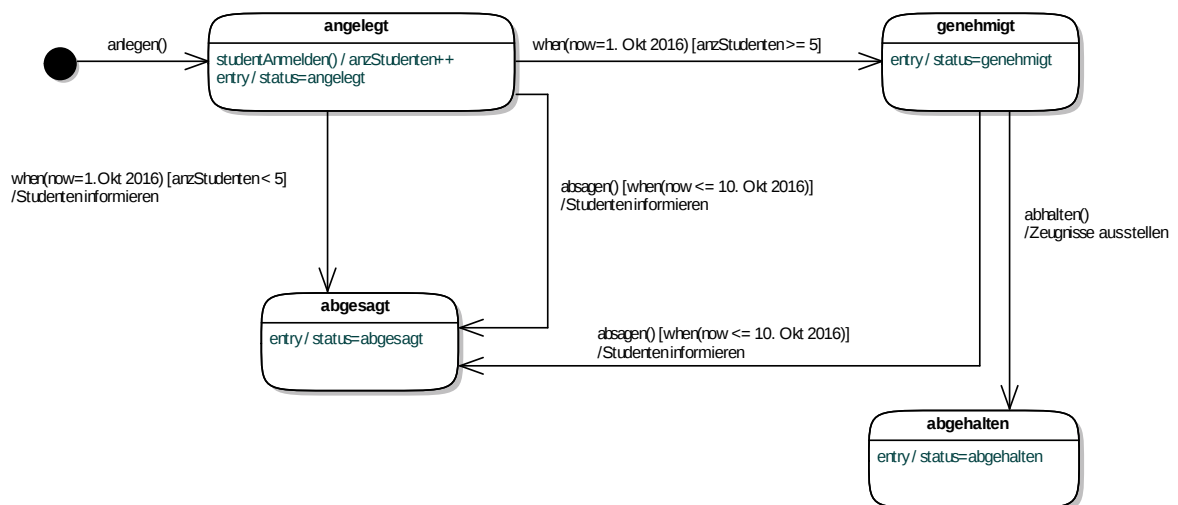
Sie haben über eine Lehrveranstaltung in einem Verwaltungssystem folgende Informationen:



Jede Lehrveranstaltung muss zu Beginn angelegt werden. Sobald die Lehrveranstaltung angelegt ist, können Studenten für diese registriert werden. Dabei erhöht sich bei einer Registrierung die Studentenzahl um eins. Am 1. Oktober 2016 werden die Lehrveranstaltungen genehmigt. Dabei wird zuerst geprüft, ob mindestens fünf Studenten angemeldet sind. Wenn dies nicht der Fall ist, so wird die Lehrveranstaltung abgesagt. Weiters kann der Lehrveranstaltungsleiter manuell die Lehrveranstaltung bis 10. Oktober 2016 absagen, unabhängig von der Anzahl der Studenten. Wie auch immer die Lehrveranstaltung abgesagt wird, es werden sicherlich die Studenten informiert. Wird eine Lehrveranstaltung genehmigt, so kann der Lehrende diese als abgehalten melden. Dabei werden die Zeugnisse der Studenten ausgestellt.

Modellieren Sie ein UML Zustandsdiagramm, das die Zustände einer Lehrveranstaltung abbildet.

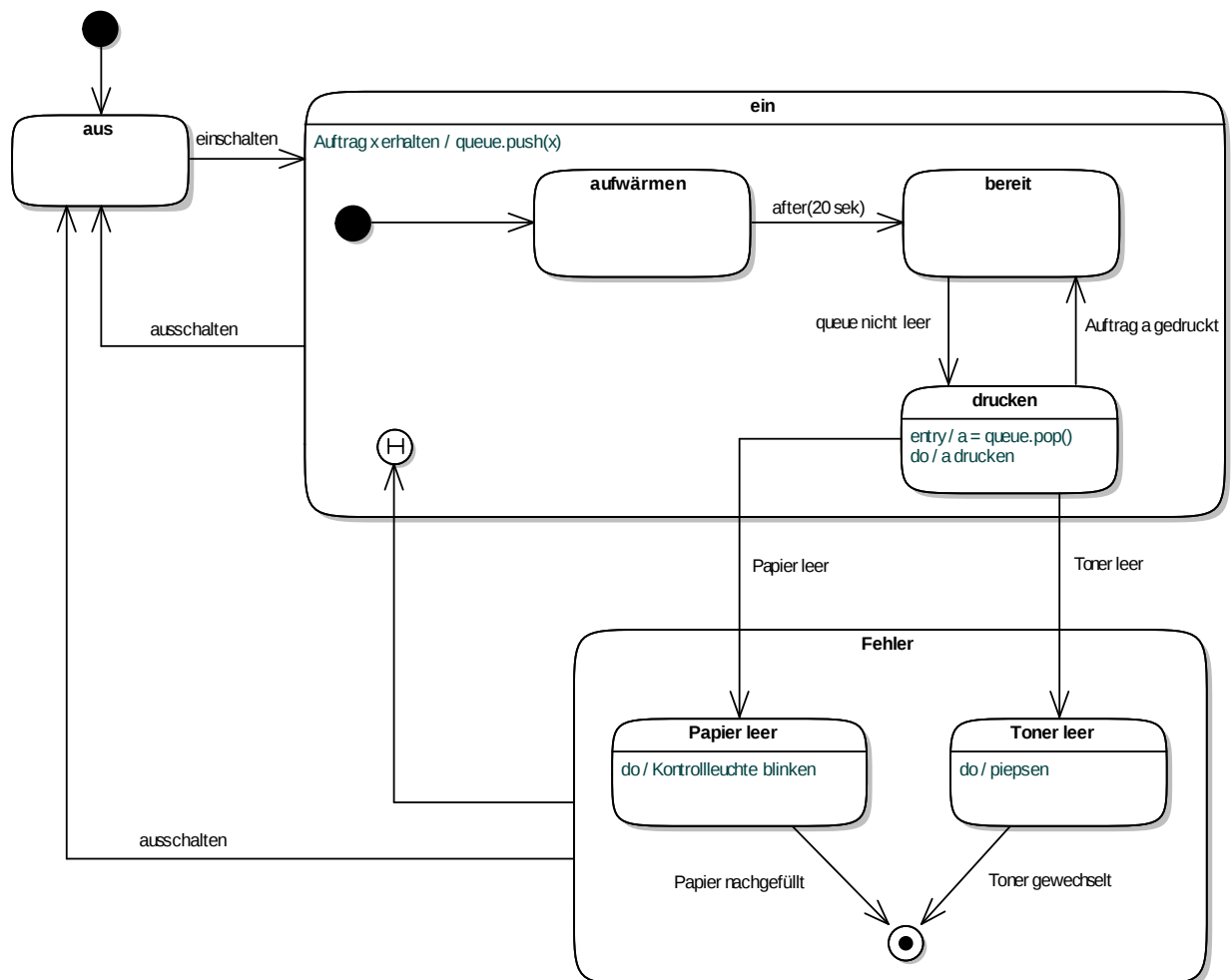
Vergessen Sie nicht, die Methodenaufrufe soweit bekannt jeweils explizit zu modellieren. Treffen Sie sinnvolle Annahmen falls Sie weitere Informationen benötigen.



Aufgabe 5: Drucker

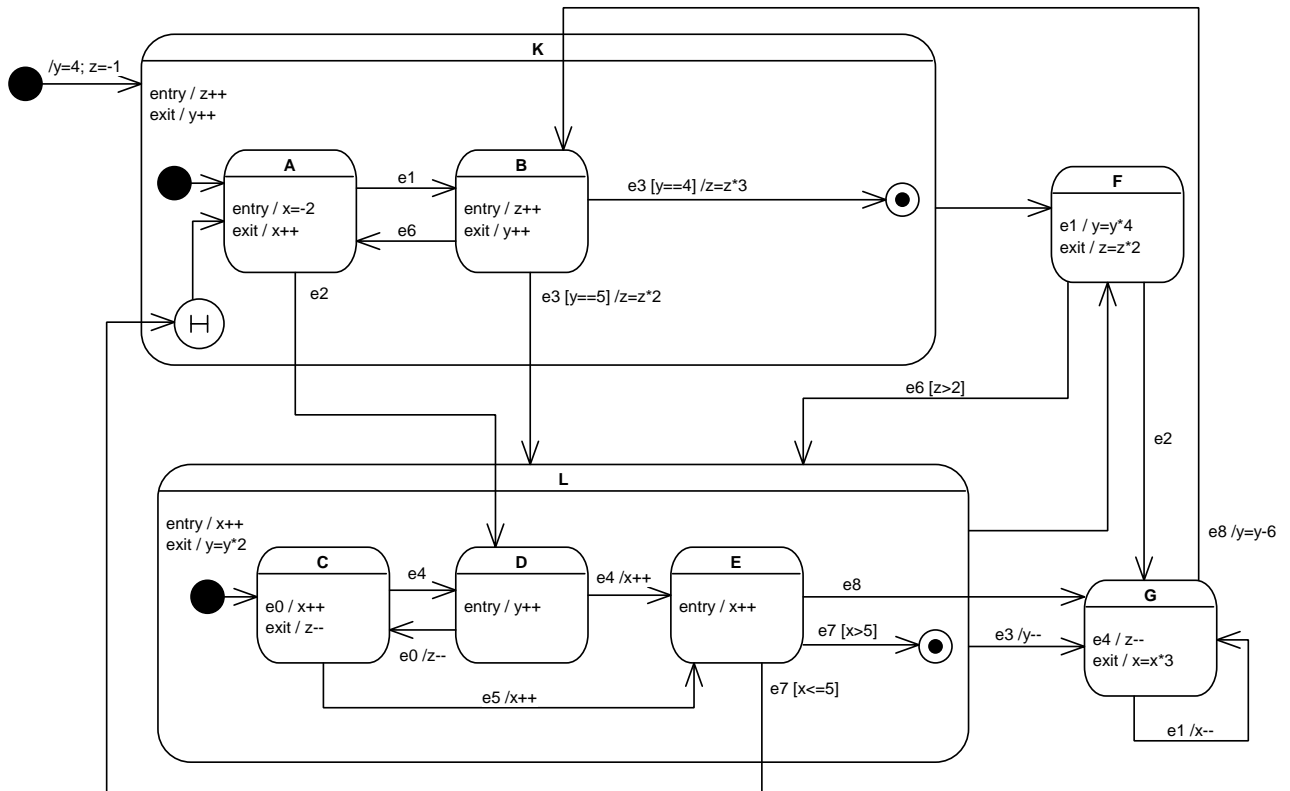
Modellieren Sie ein UML Zustandsdiagramm das die Funktionsweise eines Druckers wiedergibt.

Ein Drucker ist entweder in den Zuständen an, aus oder in einem Fehlerzustand. Der Drucker kann durch drücken des On/Off-Schalters ein- bzw. ausgeschaltet werden. Zu Beginn ist der Drucker ausgeschaltet. Wurde der Drucker eingeschaltet, so muss dieser zunächst 20 Sekunden lang aufwärmen. Danach ist der Drucker bereit für Druckaufträge. Erhält der Drucker einen Druckauftrag, so fügt er diesen zunächst in die Warteschlange. Aufträge können immer dann eintreffen, wenn der Drucker eingeschaltet ist außer der Drucker ist in einem der Fehlerzustände (siehe unten). Immer wenn die Warteschlange nicht leer ist, so druckt der Drucker die Aufträge in der Warteschlange. Ist ein Auftrag abgearbeitet, so wird dieser aus der Warteschlange entfernt. Beim Drucken kann es passieren, dass der Papierschacht leer ist. In diesem Fall wartet der Drucker auf die Eingabe von neuem Papier und währenddessen blinkt eine Kontrollleuchte. Weiters kann es passieren, dass der Toner unter dem Drucken leer wird. In diesem Fall piepst der Drucker solange kein neuer Toner eingesetzt wurde. In beiden Fehlerfällen kehrt der Drucker nach der Behebung des Fehlers (Papier bzw. Toner nachfüllen) in seinen ursprünglichen Zustand (d.h. der Zustand vor Eintreten des Fehlers) zurück.



Aufgabe 6: Ereignisfolge

Gegeben ist das folgende Zustandsdiagramm:



- a) Vervollständigen Sie die folgende Tabelle, um zu veranschaulichen, welche Zustände und Aktionen bei der folgenden Ereignisfolge vorkommen.

Belegung der Variablen				
Ereignis	Eingetr. Zustand	x	y	z
<i>Beginn</i>	K/A	-2	4	-1/0
e1	K/B	-1		1
e3	F		5/6	3
e6	L/C	0		6
e5	L/E	1/2		5
e7	K/A	-2	12	6
e2	L/D	-1/0	13/14	
e4	L/E	1/2		
e8	G		28	
e1	G	6/5		

- b) Vervollständigen Sie die folgende Tabelle, um zu veranschaulichen, welche Zustände und Aktionen bei der folgenden Ereignisfolge vorkommen.

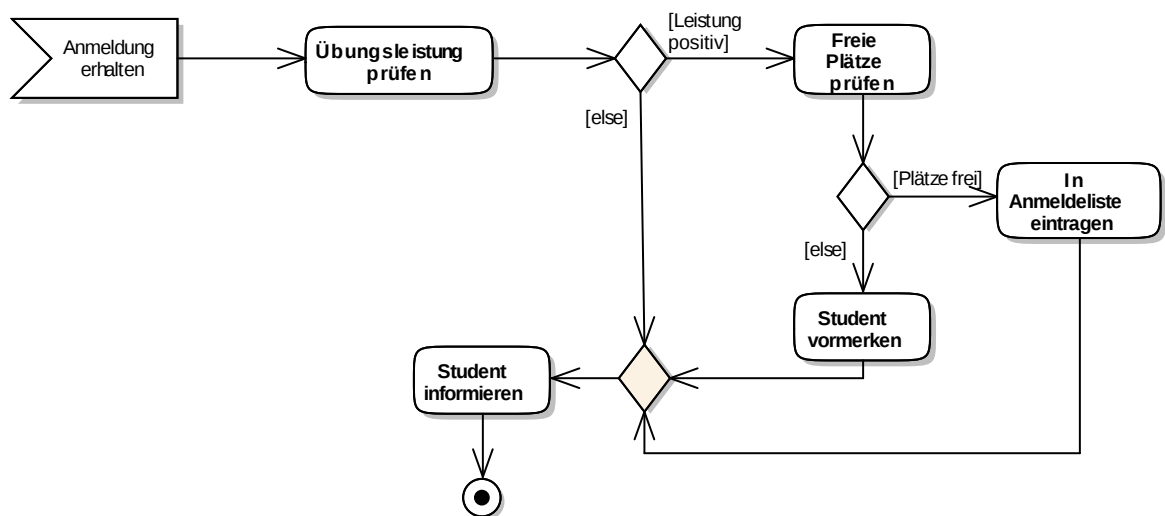
Belegung der Variablen

Ereignis	Eingetr. Zustand	x	y	z
<i>Beginn</i>	K/A	-2	4	-1/0
e2	L/D	-1/0	5/6	
e0	L/C			-1
e5	L/E	1/2		-2
e3	G		12/11	
e4	G			-3
e8	K/B	6	5	-2/-1
e3	L/C	7	6/7	-2
e4	L/D		8	-3
e1	L/D			

Aufgabe 1: Activity/Action – Split/Merge – Fork/Join – Partitions

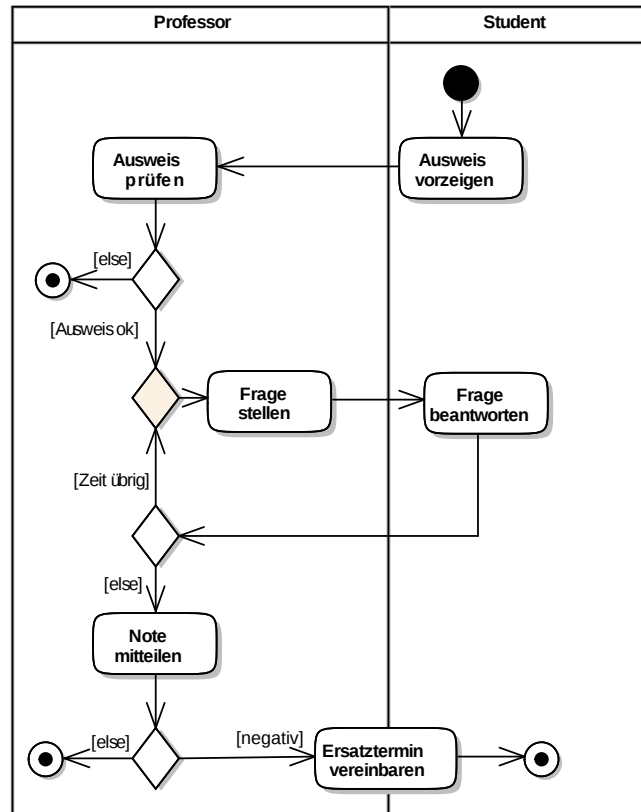
- a) Wodurch unterscheiden sich Aktivität und Aktion? Geben Sie ein Beispiel, das den Unterschied verdeutlicht.
- b) Modellieren Sie folgenden Ablauf (Kontrollfluss) mittels Aktivitätsdiagramm:

Ein Professor erhält eine Anmeldung zu einer mündlichen Prüfung. Zunächst prüft er, ob der Student eine positive Übungsleistung hat. Falls nicht, so informiert er den Studenten darüber, dass er nicht zur Prüfung antreten darf. Falls der Student positiv in der Übung ist, so prüft der Professor, ob noch Plätze für die Prüfung frei sind. Falls nein, so merkt er den Studenten für den nächsten Termin vor. Falls Plätze frei sind, so trägt er diesen in die Anmeldeliste ein. In beiden Fällen informiert der Professor den Studenten.



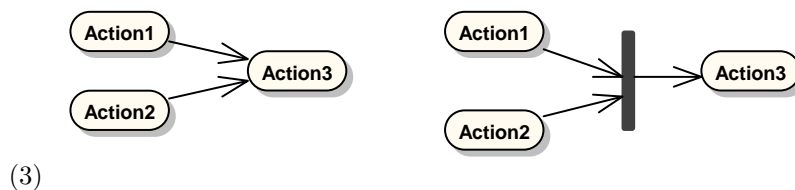
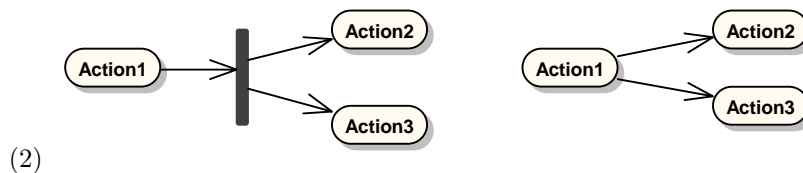
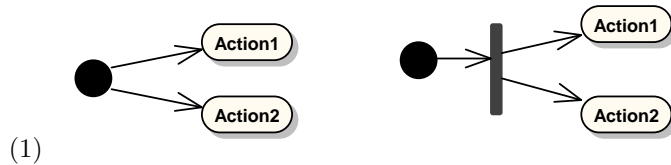
- c) Was versteht man unter Partitionen? Wozu und wie werden sie eingesetzt?
- d) Modellieren Sie folgenden Ablauf (Kontrollfluss) mittels Aktivitätsdiagramm:

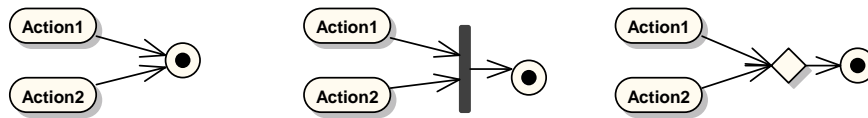
Bei einer Prüfung zeigt ein Student dem Professor zunächst seinen Ausweis, den der Professor daraufhin prüft. Ist der Ausweis nicht gültig, so wird die Prüfung beendet. Anderenfalls stellt der Professor den Studenten Fragen, welche der Student beantwortet. Dabei wird dieser Prozess solange wiederholt, bis keine Prüfungszeit mehr übrig ist. Nach der letzten Frage teilt der Professor dem Student die Note mit. Ist die Note negativ, so vereinbart der Student mit dem Professor einen Ersatztermin.



Aufgabe 2: Tokenkonzept

- Was versteht man unter einem Token? Welchen Zweck hat das Tokenkonzept?
- Wie funktioniert die Tokenverarbeitung bei Parallelisierungsknoten und Synchronisierungsknoten bzw. bei Entscheidungsknoten und Vereinigungsknoten?
- Sind folgende Konstrukte äquivalent?



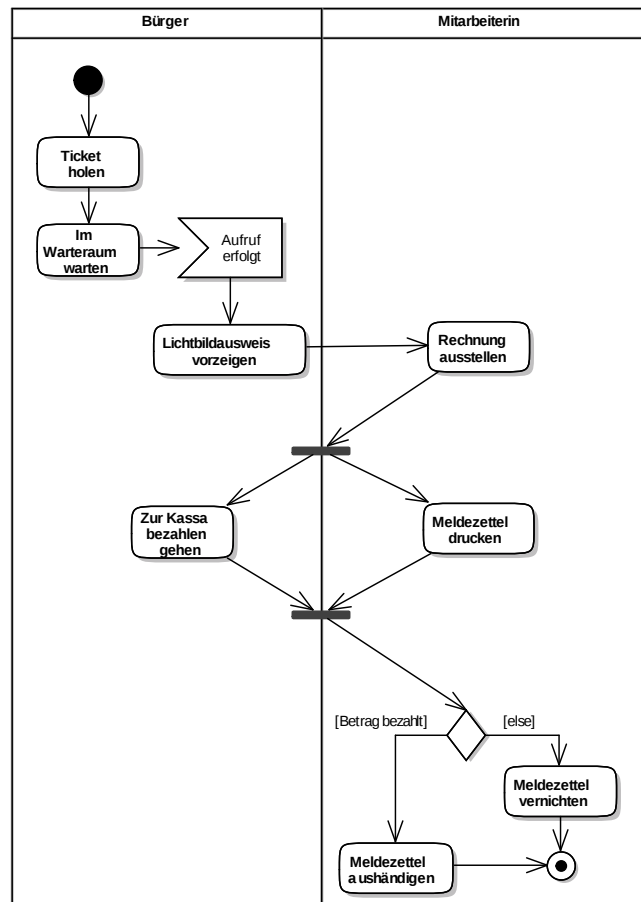


(4)

Aufgabe 3: Activity Final/Flow Final – Ausnahmebehandlung

- Wodurch unterscheiden sich Aktivitätssendknoten und Ablaufendknoten?
- Modellieren Sie folgenden Ablauf (Kontrollfluss) mittels Aktivitätsdiagramm:

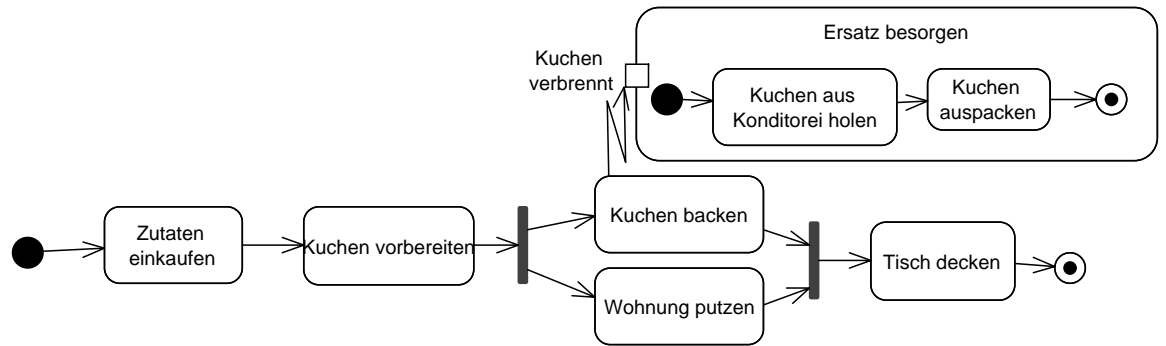
Ein Bürger möchte sich am Meldeamt einen Meldezettel besorgen. Zunächst benützt er den Ticketautomat im Warteraum, um ein Ticket für die Bearbeitung seines Anliegens zu erhalten. Danach wartet er zunächst im Warteraum bis er aufgerufen wird. Wird der Bürger aufgerufen, so zeigt er der Mitarbeiterin des Meldeamtes zunächst einen Lichtbildausweis. Die Mitarbeiterin gibt dem Bürger eine Rechnung, welche der Bürger bei der Kassa bezahlen soll. Parallel dazu druckt die Mitarbeiterin des Meldeamtes den Meldezettel aus und wartet auf die Rückkehr des Bürgers von der Kassa. Wurde der Betrag bezahlt, so händigt die Mitarbeiterin den Meldezettel aus, anderenfalls wird dieser vernichtet.



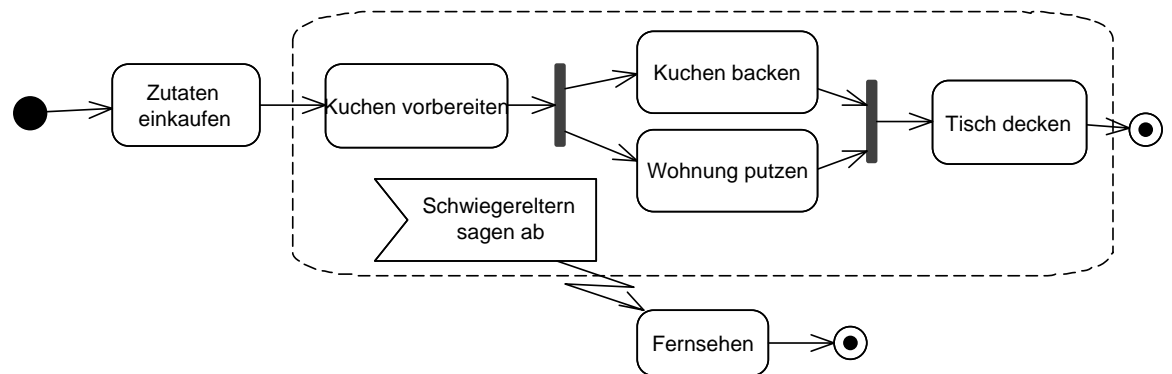
- Es steht ein Besuch der Schwiegereltern bevor, weshalb einige Dinge zu erledigen sind. Gegeben ist folgendes Aktivitätsdiagramm das den Ablauf der Vorbereitungen zeigt:

Erweitern/Ändern Sie das Aktivitätsdiagramm so, dass folgende **Fehlersituationen** entsprechend behandelt werden:

- Der Kuchen verbrennt. Daher wird stattdessen ein Kuchen aus der Konditorei geholt und ausgepackt. Danach geht der Prozess regulär weiter.



- (2) Irgendwann nach dem Einkauf der Zutaten sagen die Schwiegereltern spontan ab. Daraufhin wird der Prozess abgebrochen und stattdessen die Lieblingsfernsehserie angeschaut.



Aufgabe 4: Algorithmus

Gegeben sei der folgende Code für einen Selectionsort:

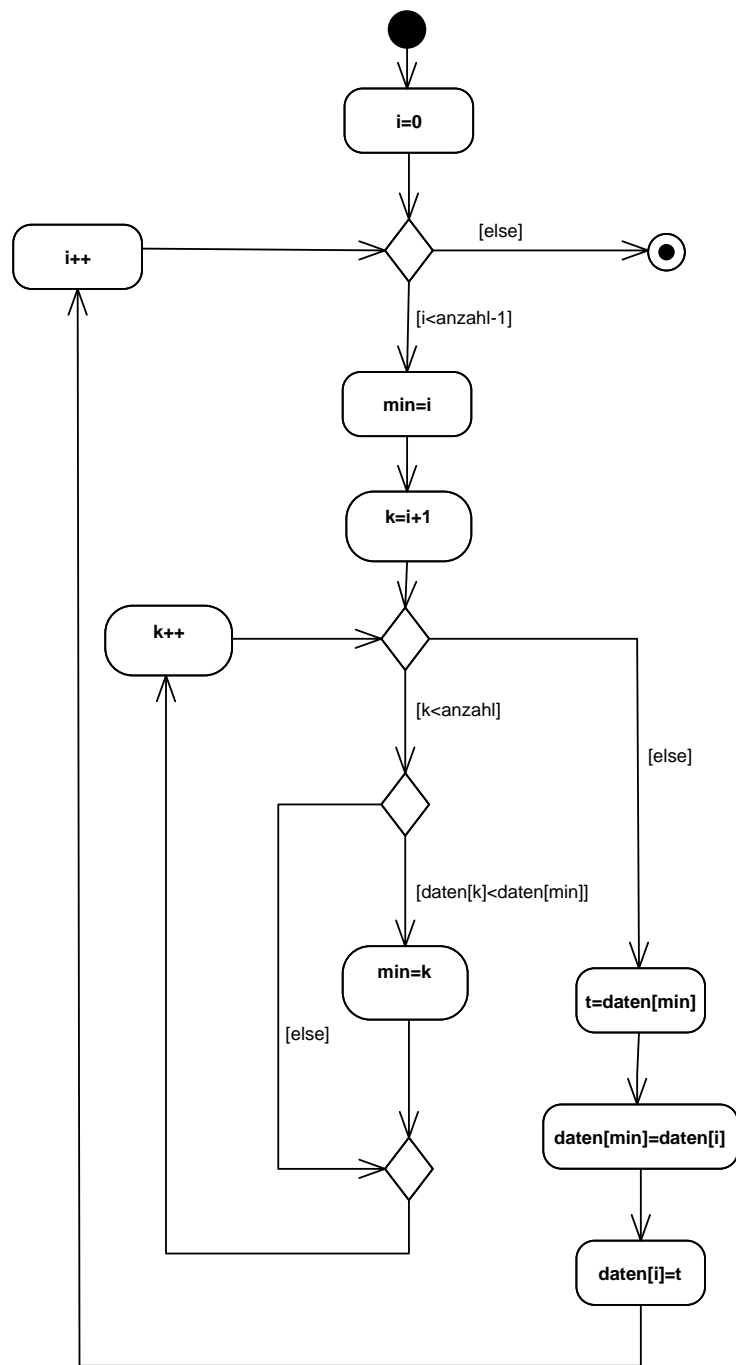
```
void selectionsort(int anzahl, int daten[]) // Anzahl und Array mit Daten
{
    int i, k, t, min;

    // i ist Kontrollvariable für die innere, k für die äußere Schleife.
    // t ist Zwischenspeicher beim Tauschen. min merkt sich den kleinsten Wert
    // im Teilarray.

    for( i = 0; i < anzahl-1; i++)
    {
        min = i;
        for( k = i+1; k < anzahl; k++)
        {
            if(daten[k] < daten[min])
                min = k;
        }

        t = daten[min]; // Tauschen
        daten[min] = daten[i];
        daten[i] = t;
    }
}
```

Modellieren Sie den Selectionsort Algorithmus als Aktivitätsdiagramm.

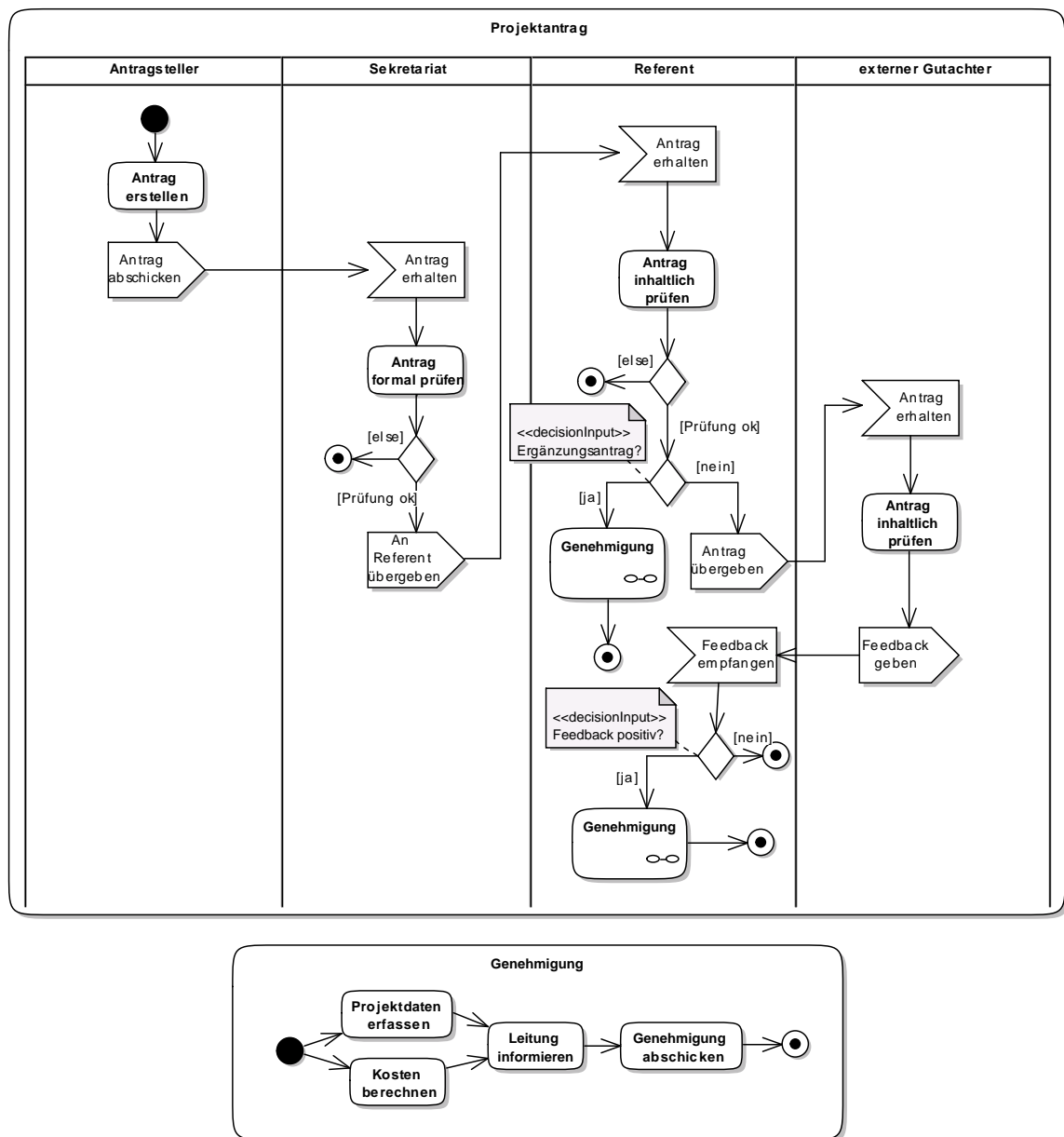


Aufgabe 5: Send/Receive – Subprozesse

Sie haben folgende Information über den Ablauf einer Antragstellung eines Projektes.

Der Prozess beginnt damit, dass der Antragsteller den Projektantrag erstellt und abschickt. Das Sekretariat der Antragsstelle prüft den Antrag zunächst auf formale Vollständigkeit. Ist der Antrag unvollständig, so wird der Prozess beendet. Ist der Antrag vollständig, so übergibt das Sekretariat den Antrag an einen Referenten. Dieser prüft den Antrag zunächst inhaltlich. Ist der Antrag inhaltlich nicht in Ordnung, so wird der Prozess beendet. Anderenfalls wird geprüft, ob es sich bei dem Antrag über einen Ergänzungsantrag handelt. Falls ja, so obliegt die Genehmigungsentscheidung beim Referenten. Anderenfalls übergibt der Referent den Antrag an einen externen Begutachter. Dieser prüft den Antrag inhaltlich und gibt Feedback an den Referenten. Ist das Feedback positiv, so leitet der Referent den Genehmigungsprozess ein. Ist es negativ, so wird der Prozess beendet. Der Genehmigungsprozess verläuft bei Ergänzungsanträgen und anderen Anträgen ident. Zunächst werden die Projektdaten in der Datenbank erfasst und die voraussichtlichen Kosten des Projektes berechnet. Danach wird die Leitung der Antragsstelle informiert und die Genehmigung abgeschickt.

Modellieren Sie den Ablauf (Kontrollfluss) mittels Aktivitätsdiagramm. Lagern Sie die Genehmigung des Projekts in einen separaten Prozess aus, damit Sie den Vorgang nicht doppelt modellieren müssen.

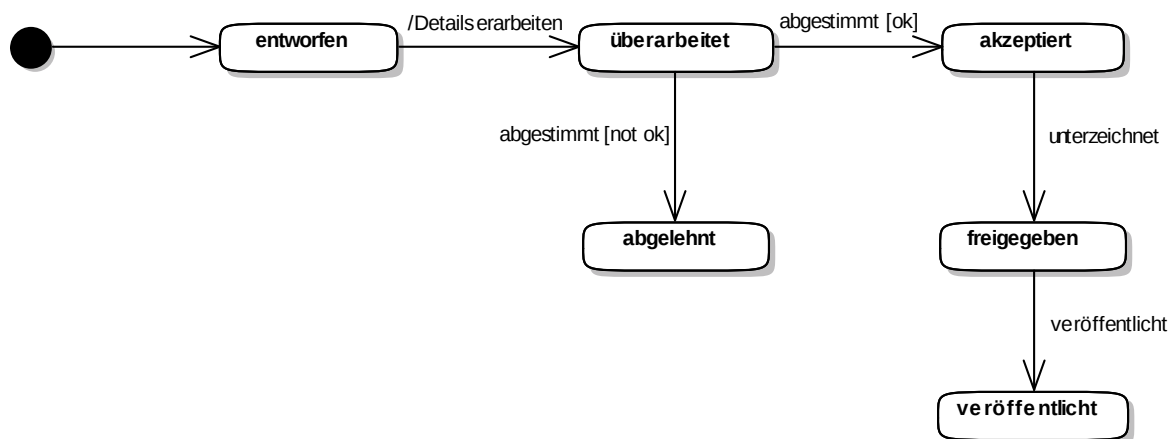


Aufgabe 6: Kontroll- und Datenfluss

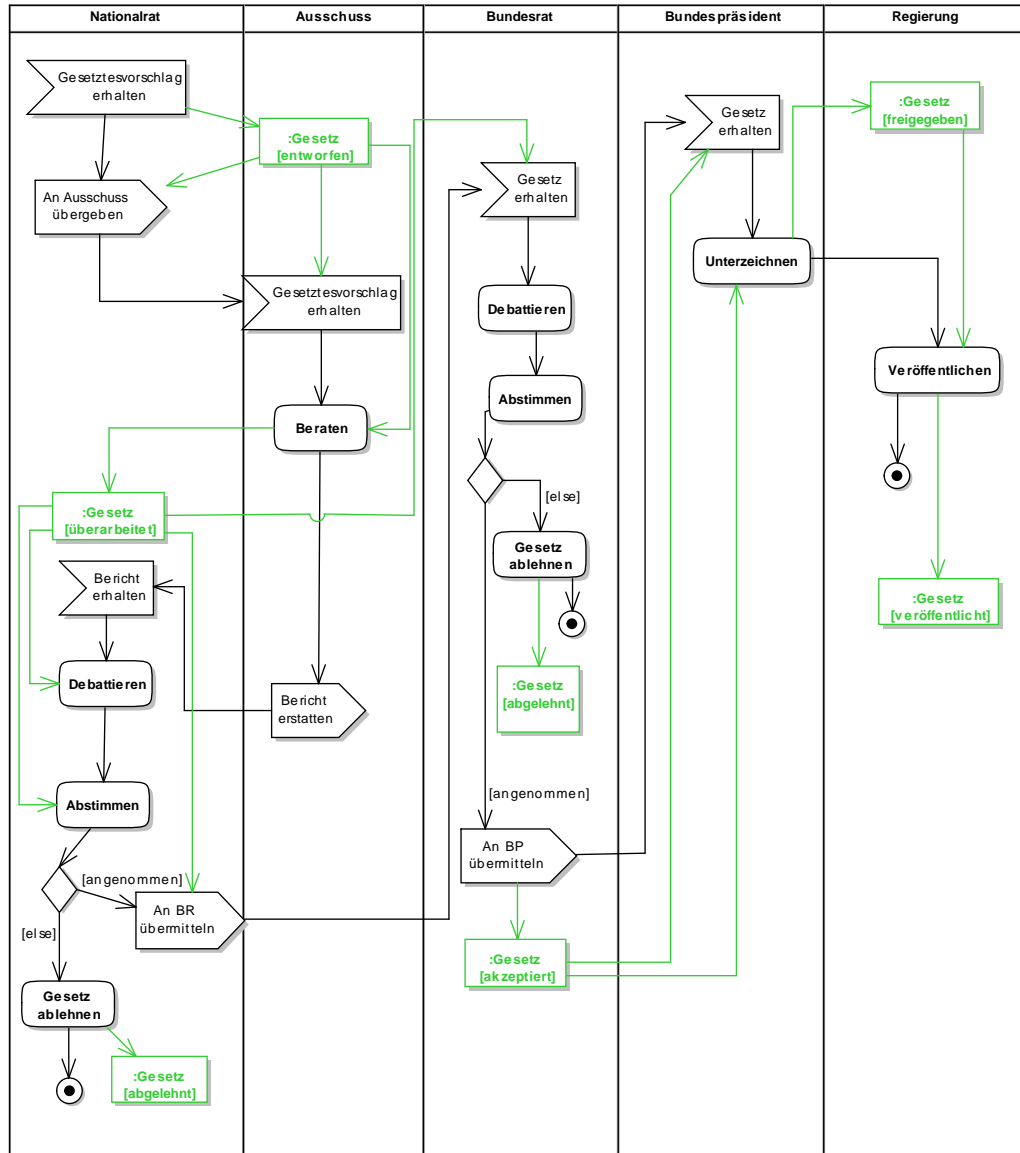
Nachfolgend wird der Weg eines Gesetzes (stark vereinfacht) beschrieben:

Zunächst gelangt ein Gesetzesvorschlag im Nationalrat ein. Dieser weist den Vorschlag einem Ausschuss zu. In dem Ausschuss wird über das Gesetz zunächst beraten und eine überarbeitete Version des Gesetzes erstellt. In weiterer Folge erstattet der Ausschuss dem Nationalrat Bericht. Dem folgt eine Debatte im Nationalrat. Danach wird über das Gesetz abgestimmt. Bei negativem Abstimmungsergebnis wird der Prozess beendet und das Gesetz ist abgelehnt. Bei positiver Abstimmung wird das Gesetz dem Bundesrat übermittelt, welcher auch über das Gesetz debattiert und daraufhin abstimmt. Fällt hier die Abstimmung negativ aus, so wird, falls dies erwünscht ist, das Gesetz wieder dem Nationalrat zugewiesen und der Prozess beginnt von vorne. Fällt die Entscheidung positiv aus, so wird das Gesetz dem Bundespräsidenten übermittelt, welcher es daraufhin unterzeichnet. Daraufhin wird das Gesetz von der Regierung veröffentlicht.

Der Lebenszyklus des Objekts „Gesetz“ ist in folgendem Zustandsdiagramm dargestellt:



Modellieren Sie den Prozess der Entstehung eines Gesetzes, mittels UML2-Aktivitätsdiagramm. Modellieren Sie mittels Objektfluss die durch die Aktionen/Aktivitäten bedingten Änderungen am Objekt „Gesetz“. (Andere Objektflüsse sind für diese Aufgabe nicht relevant!). Illustrieren Sie die involvierten Rollen mit Hilfe von Swimlanes (Partitionen).



Hinweis:

Erläutern Sie alle Theoriefragen anhand eines selbst gewählten Beispiels. Dieses Beispiel sollte **nicht** aus der Vorlesung oder aus den Lösungen anderer Übungsaufgaben stammen. Des Weiteren sollte das Beispiel konkret modelliert werden (z.B. einen Akteur „Person“ anstelle eines abstrakten Akteurs „X“).

Aufgabe 1: Anwendungsfallmodellierung – Theoriefragen I

Wiederholen Sie das Kapitel aus der Vorlesung, das sich mit dem UML2-Anwendungsfalldiagramm beschäftigt.

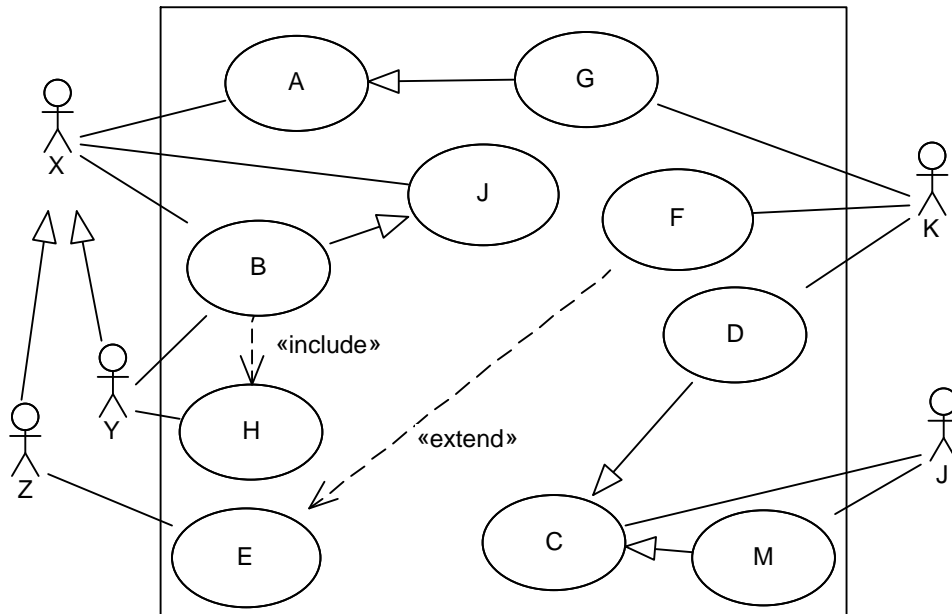
- a) Beschreiben Sie die Bestandteile eines Anwendungsfalldiagramms anhand eines einfachen Beispiels.
- b) Was versteht man unter einem Anwendungsfall?
- c) Grenzen Sie eine anwendungsfall-zentrierte Vorgehensweise von der funktionalen Zerlegung ab.
- d) Wie gehen Sie bei der Identifikation von Akteuren und bei der Identifikation von Anwendungsfällen vor?

Aufgabe 2: Anwendungsfallmodellierung – Theoriefragen I

- a) Wie können Akteure klassifiziert werden? Wie spiegelt sich die Klassifikation in der Notation von Akteuren wider?
- b) Auf welche Bestandteile des Anwendungsfalldiagramms kann das Konzept der Generalisierung angewendet werden? Geben Sie entsprechende Beispiele.
- c) Diskutieren Sie die Erweiterungsbeziehung „include“ in Anwendungsfalldiagrammen. Finden Sie ein Beispiel.
- d) Diskutieren Sie die Erweiterungsbeziehung „extend“ in Anwendungsfalldiagrammen inklusive der Erweiterungsstellen (Extension Points). Finden Sie ein Beispiel.

Aufgabe 3: Include, Extend und Generalisierung

Gegeben sei folgendes Anwendungsfalldiagramm, das streng nach UML Standard modelliert wurde:
Im Folgenden werden nur die direkt beteiligten Akteure angeführt.



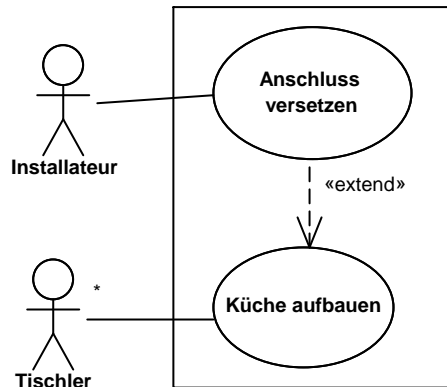
Diskutieren Sie folgende Fragen:

- Welche Akteure sind jeweils an den einzelnen Use Cases beteiligt?
 $A : (X \vee Y \vee Z)$
 $B : (X \vee Y \vee Z) \wedge Y \wedge (X \vee Y \vee Z)$
 $C : J$
 $D : J \wedge K$
 $E : Z$
 $F : K$
 $G : (X \vee Y \vee Z) \wedge K$
 $H : Y$
 $J : (X \vee Y \vee Z)$
 $M : J \wedge J$ (zweimal J)
- Muss E ausgeführt werden, wenn auch F ausgeführt wird? Muss H ausgeführt werden, wenn B ausgeführt wird?
 Nein; Ja.
- Ist F oder E der Basis Use Case? Ist B oder H der Basis Use Case?
 E ; B .
- Muss bei J auch H ausgeführt werden?
 Nein.

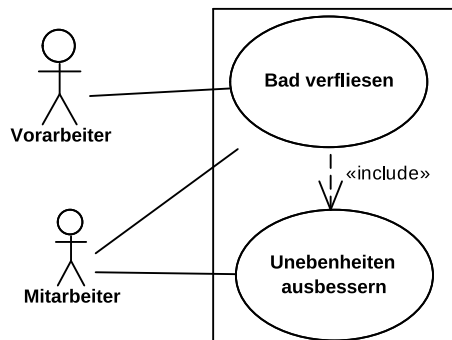
Aufgabe 4: Modellierung verschiedener Sachverhalte

Modellieren Sie die folgenden Sachverhalte in jeweils eigenen Anwendungsfalldiagrammen streng nach UML 2.0 Standard:

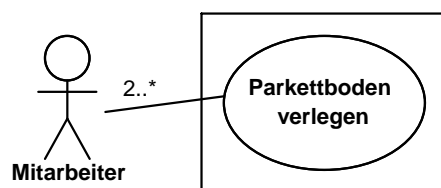
- a) Mehrere Tischler bauen eine Küche auf. Im Zuge dessen kann es notwendig sein, dass ein Installateur den Wasseranschluss versetzt.



- b) Ein Vorarbeiter verfließt gemeinsam mit seinem Mitarbeiter das Bad. Im Zuge dessen muss der Mitarbeiter immer auch die Unebenheiten in den Wänden ausbessern.



- c) Um einen Parkettboden zu verlegen sind immer mindestens zwei Mitarbeiter notwendig.



Aufgabe 5: Anwendungsfalldiagramm

Erstellen Sie ein Anwendungsfalldiagramm, welches alle Anwendungsfälle von Microsoft Word bzw. OpenOffice Writer enthält, die in die Kategorie „Überprüfen“ fallen – dazu zählen z.B. Rechtschreibung und Grammatik, Änderungen, Nachverfolgung, etc.

Aufgabe 6: Anwendungsfallbeschreibung

Erstellen Sie ein Anwendungsfalldiagramm, welches 2 Anwendungsfälle beinhaltet, die in dieselbe Kategorie wie die Anwendungsfälle aus Aufgabe 5 fallen, die aber durch die Software nicht abgedeckt sind und die Ihrer Ansicht nach sinnvoll wären.

Führen Sie anschließend eine Anwendungsfalldiagrammbeschreibung von diesen 2 Anwendungsfällen durch. Nehmen Sie die Anwendungsfallbeschreibungen vorgefertigt (leserlich) auf Papier mit. Kreuzen Sie dieses Beispiel nur, wenn Sie auch die entsprechenden Ausarbeitungen ausgedruckt mitnehmen.

Führen Sie die Anwendungsfallbeschreibung eigenständig durch und bedenken Sie, dass es sehr unwahrscheinlich ist, dass bei einer Anwendungsfallbeschreibung zwei Studierende auf dieselbe Lösung kommen.