VU Objektorientierte Modellierung		Übung 1: Klassendiagramm
188.391, SS2018 ,	MUSTERLÖSUNG	Übungsgruppen: 12.03.2018 - 16.03.2018

Hinweise:

- Theoriefragen: Erläutern Sie die Fragen anhand eines **selbst gewählten, konkreten Beispiels**.
  - selbst gewählt ... **nicht** aus der Vorlesung oder aus den Lösungen anderer Übungsaufgaben
  - konkret ... z.B. Klasse „Person“ anstelle abstrakt Klasse „A“
- Modellierungsbeispiele: Bilden Sie den Sachverhalt, der in der Angabe geschildert wird, möglichst genau ab. Sollte etwas in der Angabe nicht erwähnt sein, treffen Sie sinnvolle Annahmen.

Aufgabe 1: Klassendiagramm - Theoriefragen 1

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

a) 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.)

b) 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.)

c) Erläutern Sie alle Notationsmöglichkeiten für eine Assoziation anhand eines konkreten Beispiels.

d) Illustrieren Sie den Unterschied zwischen 1:1, 1:n und n:m Assoziationen anhand eines **Objektdiagramms**.

Aufgabe 2: Klassendiagramm - Theoriefragen 2

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

a) Was ist eine Assoziationsklasse? Erklären Sie die Notation anhand eines konkreten Beispiels. Wann ist es sinnvoll, eine solche einzusetzen?

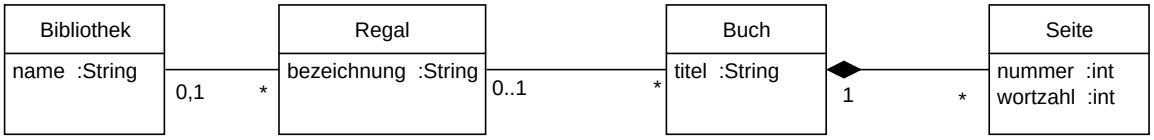
b) Beschreiben Sie den Unterschied zwischen starker und schwacher Aggregation. Erklären Sie die Notation jeweils anhand eines konkreten Beispiels.

c) 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.

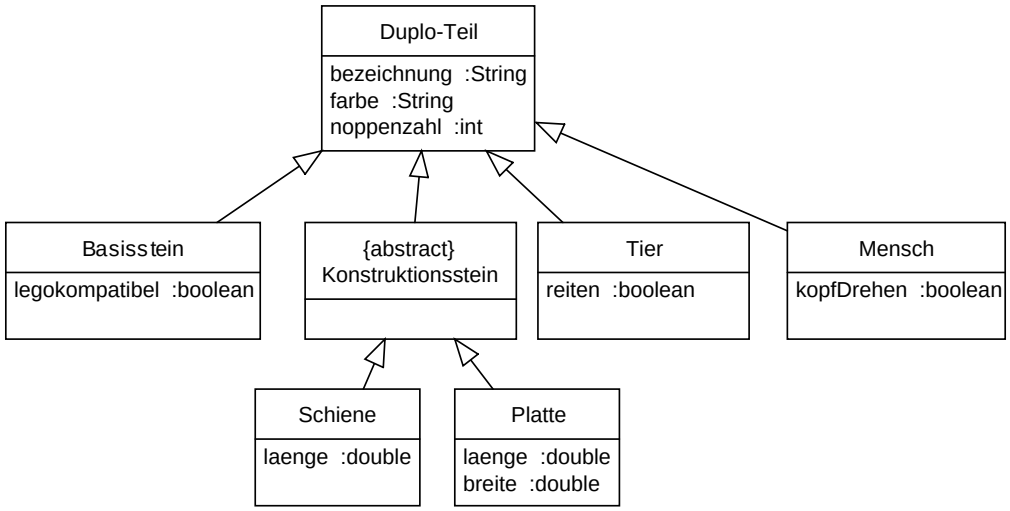
Aufgabe 3: Einleitende Beispiele, Teil 1

Modellieren Sie die geschilderten Sachverhalte und treffen Sie realistische Annahmen wo nötig:

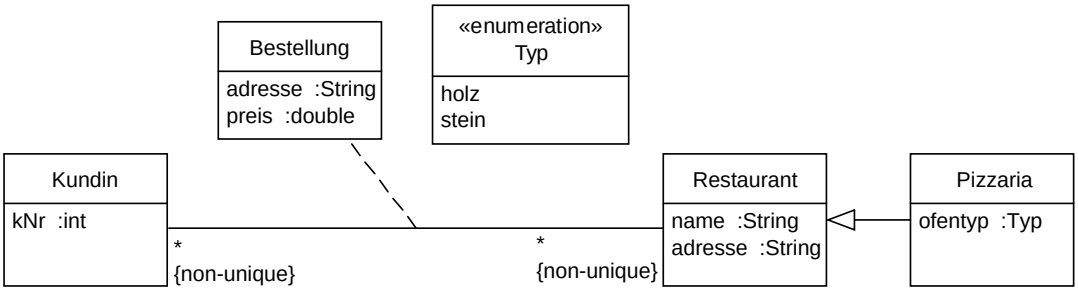
a) Eine Bibliothek hat einen Namen und besteht aus mehreren Bücherregalen. Jedes Regal hat eine Bezeichnung und beinhaltet mehrere Bücher, deren Titel gespeichert werden. Ein Buch besteht aus mehreren Seiten, deren Nummer und Wortanzahl gespeichert werden.



b) Von jedem Duplo-Teil wird die Bezeichnung, die Farbe und die Anzahl der Noppen gespeichert. Es gibt unter anderem folgende Arten von Duplo-Teilen: Basissteine, Konstruktionssteine, Tiere und Menschen. Bei einem Basissteinen wird gespeichert, ob dieser zum (klassischen) Lego kompatibel ist, bei jedem Mensch wird gespeichert, ob er/sie den Kopf drehen kann oder nicht. Bei den Tieren wird gespeichert, ob obendrauf ein Mensch oder ein weiteres Tier reiten kann. Es gibt genau zwei Arten von Konstruktionssteinen, Schienen – von ihnen wird die Länge gespeichert – und Platten, von denen Länge und Breite gespeichert werden.



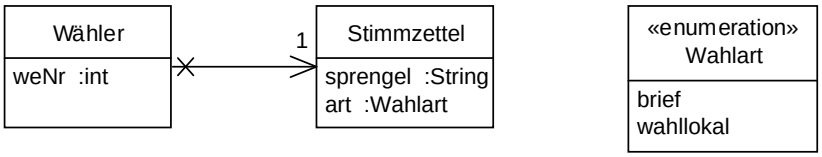
c) Ein Restaurant hat mehrere Kundinnen die wiederum in mehreren Restaurants Essen bestellen können. Von jedem Restaurant werden Name und Adresse gespeichert. Zusätzlich wird für jede Bestellung, die eine Kundin in einem Restaurant macht, die Lieferadresse und der Preis gespeichert. Von jeder Kundin wird die Kundinnen-Nummer gespeichert. Es gibt eine spezielle Art Restaurant, nämlich die Pizzaria. Von dieser wird zusätzlich der Ofentyp gespeichert, den sie zum Herstellen der Pizzen verwendet. Es gibt genau zwei verschiedene Öfen, nämlich Holzofen und Steinofen.



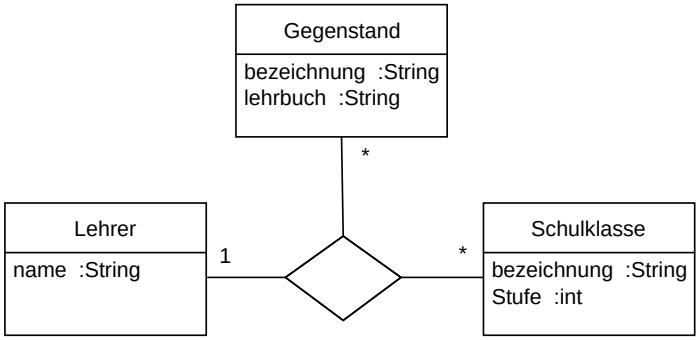
Aufgabe 4: Einleitende Beispiele, Teil 2

Modellieren Sie die geschilderten Sachverhalte und treffen Sie realistische Annahmen wo nötig:

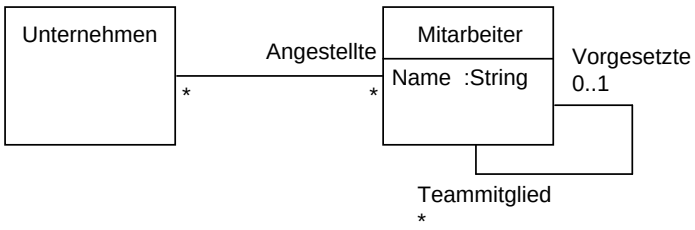
a) Bei einer Wahl bekommt jeder Wähler, von dem die Wählerevidenznummer gespeichert wird, entweder einen Stimmzettel für die Briefwahl oder einen Stimmzettel für die Wahl im Wahllokal. Von beiden wird lediglich der Wahlsprengel gespeichert, es lassen sich keine Rückschlüsse ziehen, welcher Wähler den Stimmzettel ausgefüllt hat.



b) In einer Schule gibt es mehrere Schulklassen. Eine Schulklasse wird von mehreren Lehrern unterrichtet. Es soll gespeichert werden, welcher Lehrer welchen Gegenstand in welcher Klasse unterrichtet. Von jeder Klasse werden die Bezeichnung und Schulstufe gespeichert, vom Lehrer der Name und vom Gegenstand die Bezeichnung und welches Lehrbuch benutzt wird.

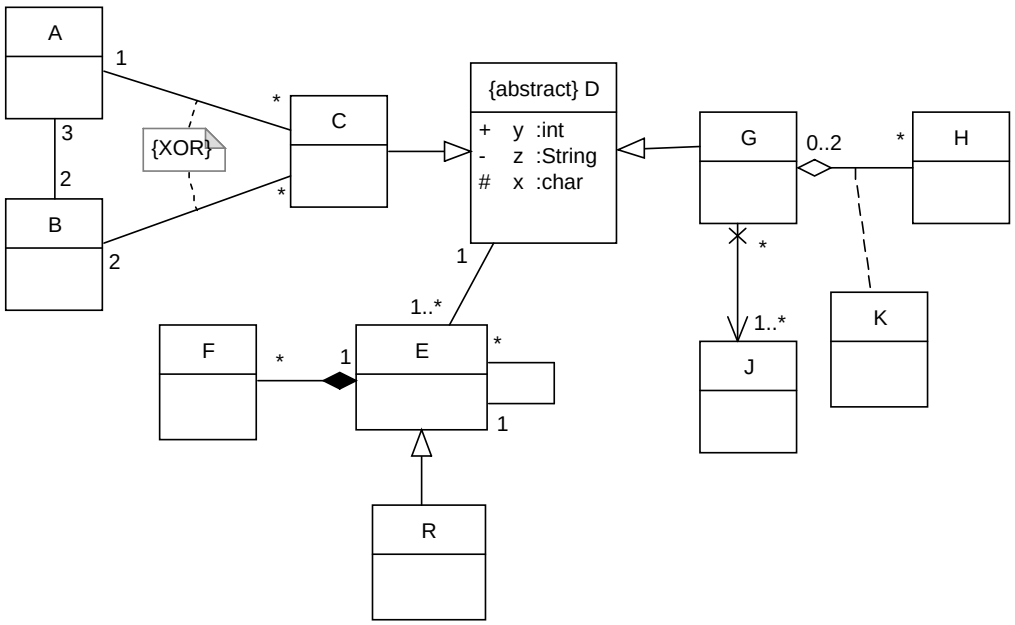


c) In einem Unternehmen gibt es mehrere Mitarbeiterinnen, die als Angestellte bezeichnet werden. Von ihnen wird der Name gespeichert. Jede Mitarbeiterin hat eine andere Mitarbeiterin als Vorgesetzte. Eine Vorgesetzte ist für mehrere Mitarbeiterinnen zuständig, die als Teammitglieder bezeichnet werden.



Aufgabe 5: Klassendiagramm lesen - Wahr oder falsch?

Es ist folgendes UML-Modell gegeben:



Welche Aussagen treffen zu? Begründen Sie Ihre Antwort!

Ein Objekt von E kann direkt auf die Variable x zugreifen.		☒ nein
Ein Objekt von C steht in Beziehung zu genau einem Objekt von A und genau zwei Objekten von B.		☒ nein
Ein Objekt von E steht in Beziehung zu genau einer direkten Instanz von D.		☒ nein
Ein Objekt von R kann mit sich selbst in Beziehung stehen.	☒ ja	
Die Raute bei E wird als starke Aggregation bezeichnet.	☒ ja	

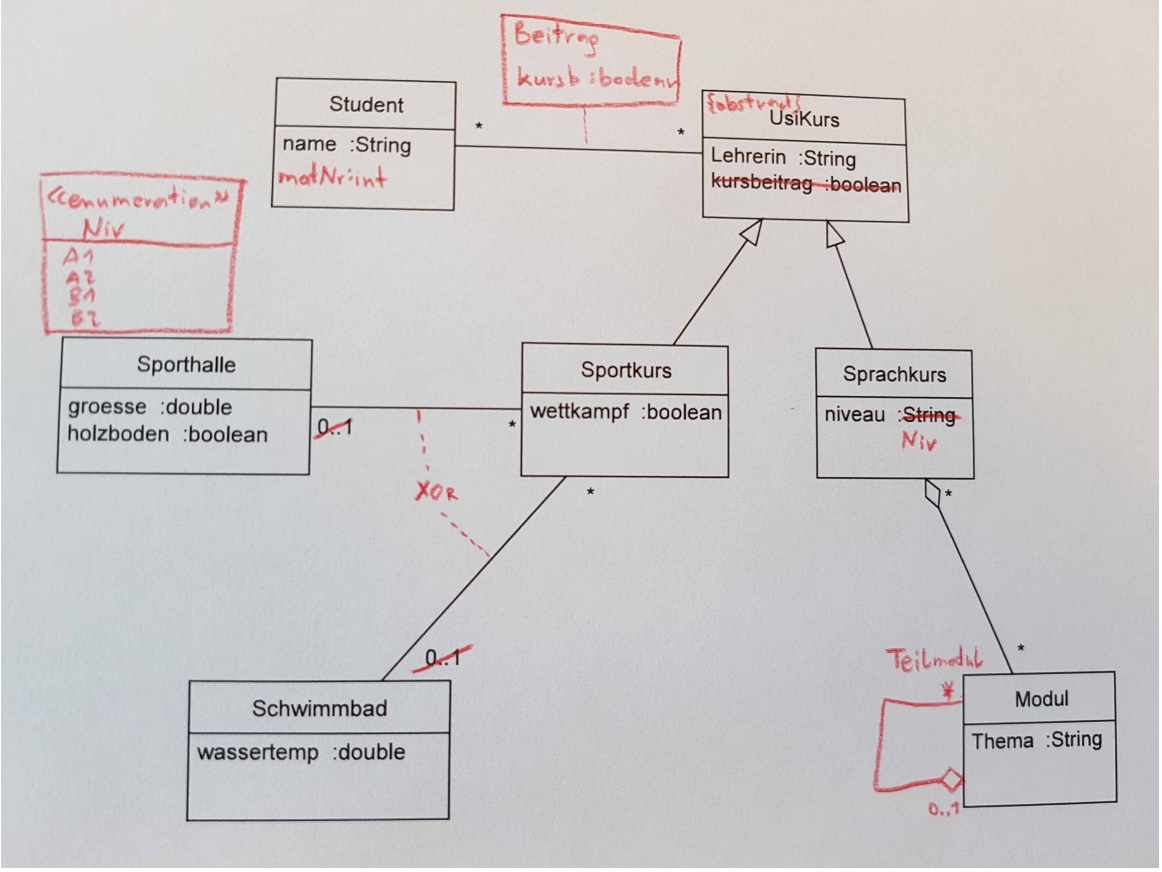
Ein Objekt von G muss in Beziehung zu mindestens einem Objekt von E stehen.	☒ ja	
Ein Objekt von D kann auf die Variable x zugreifen.	☒ ja	
Ein Objekt von E steht in Beziehung zu mindestens einem Objekt von F.		☒ nein
Im System können mehr K als H enthalten sein.	☒ ja	
Ein Objekt von C muss in Beziehung zu einem Objekt von A stehen.		☒ nein

Eine Instanz von E kann auf die Variable y zugreifen.	☒ ja	
Ein Objekt von F ist in genau einem Objekt von E enthalten.	☒ ja	
Ein Objekt von A steht mit genau zwei Objekten von B in Beziehung, ein Objekt von B steht mit genau drei Objekten von A in Beziehung.	☒ ja	
Wenn eine Instanz von G gelöscht wird, werden alle enthaltenen Instanzen von H gelöscht.		☒ nein
Ein Objekt von G steht in Beziehung zu mindestens einem Objekt von J und die Beziehung kann von G aus navigiert werden.	☒ ja	

Aufgabe 6: 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.

Es gibt genau zwei verschiedene Arten von USI-Kursen, nämlich Sportkurse und Sprachkurse. Ein USI-Kurs wird von mehreren Studenten besucht, ein Student kann wiederum mehrere Kurse besuchen. Von jedem Studenten werden Name und Matrikelnummer gespeichert, von jedem Kurs der Name der Lehrerin, die ihn unterrichtet. Außerdem wird für jeden Kurs gespeichert, ob der jeweilige Student den Kursbeitrag bereits bezahlt hat oder nicht. Von Sportkursen wird zusätzlich gespeichert, ob Wettkämpfe stattfinden. Ein Sportkurs findet entweder in einer Sporthalle, von der die Größe gespeichert wird und ob sie einen Holzboden hat, oder in einem Schwimmbad statt. Jedes Schwimmbad hat eine bestimmte Wassertemperatur. Von Sprachkursen wird das Kursniveau gespeichert. Es gibt genau vier verschiedene Niveaus: A1, A2, B1 und B2. Jeder Sprachkurs beinhaltet mehrere Module, ein Modul kann in mehreren Sprachkursen vorkommen. Von jedem Modul wird das Thema gespeichert. Ein Modul kann sich aus mehreren Teilmodulen zusammensetzen, die jeweils genauso aufgebaut sind und die gleichen Attribute aufweisen wie ein Modul.



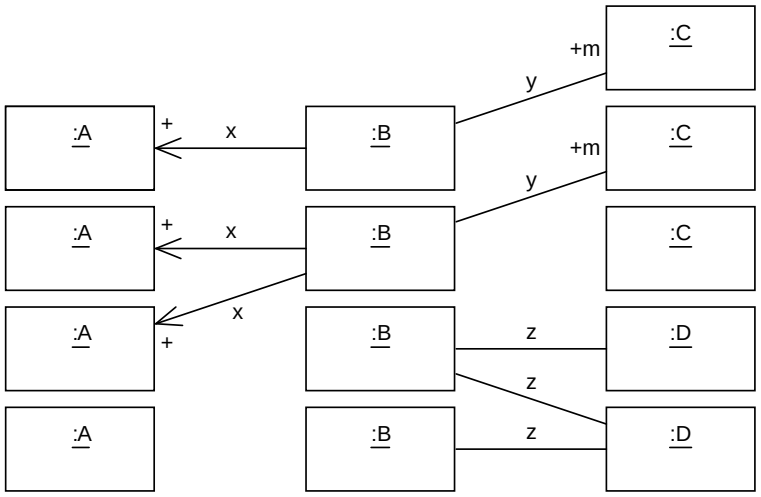
Hinweise zu den Modellierungsbeispielen:

- Bilden Sie den Sachverhalt, der in der Angabe geschildert wird, moeglichst genau ab.
- Datentypen muessen Sie nur darstellen, wenn diese explizit im Text/Code spezifiziert wurden.
- Sollte etwas in der Angabe nicht erwaehnt sein, treffen Sie sinnvolle Annahmen.

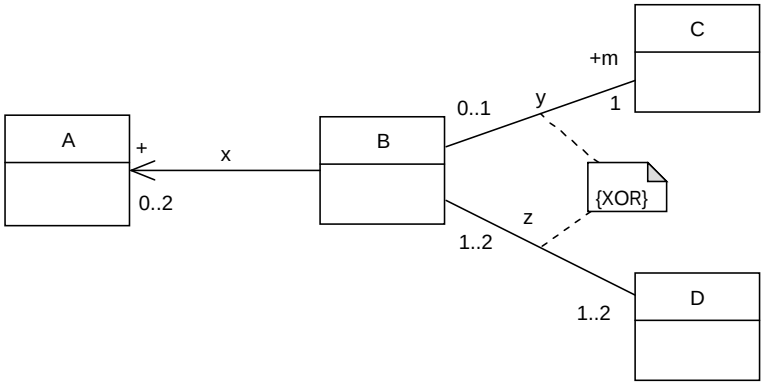
Aufgabe 1: Objektdiagramm

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.

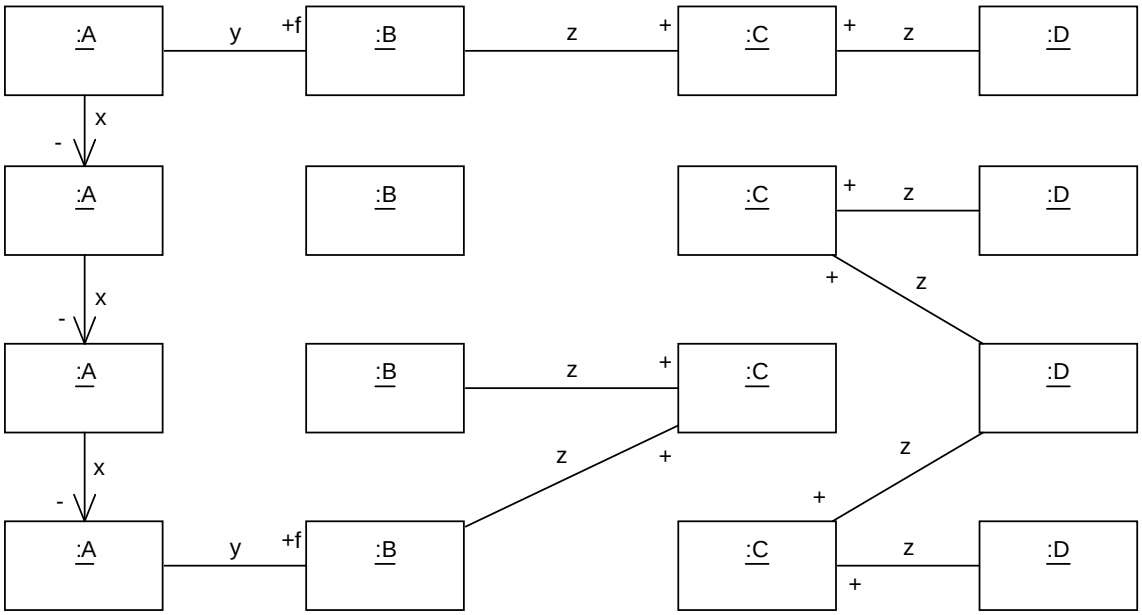
- **Objektdiagramm 1:**



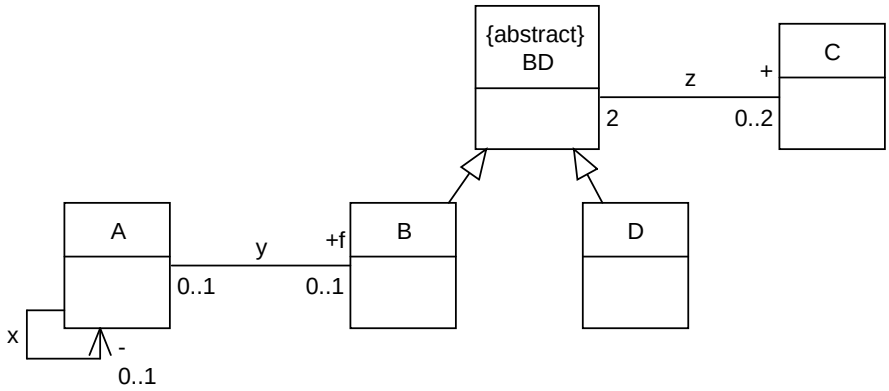
Klassendiagramm 1:



- **Objektdiagramm 2:**



Klassendiagramm 2:



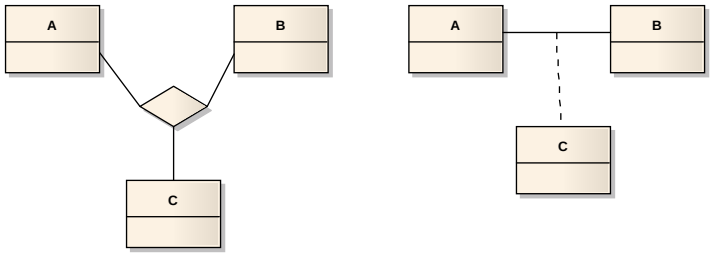
Aufgabe 2: Vergleich von Klassendiagrammausschnitten

Erklären Sie den Unterschied zwischen folgenden Klassendiagrammausschnitten:
a)

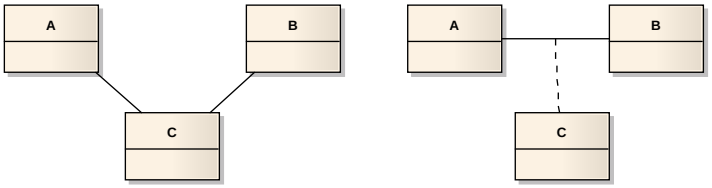
b)

c)

d)

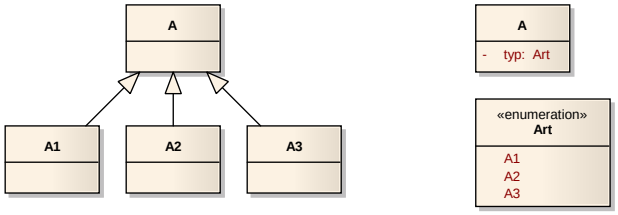


e)



f)

g)

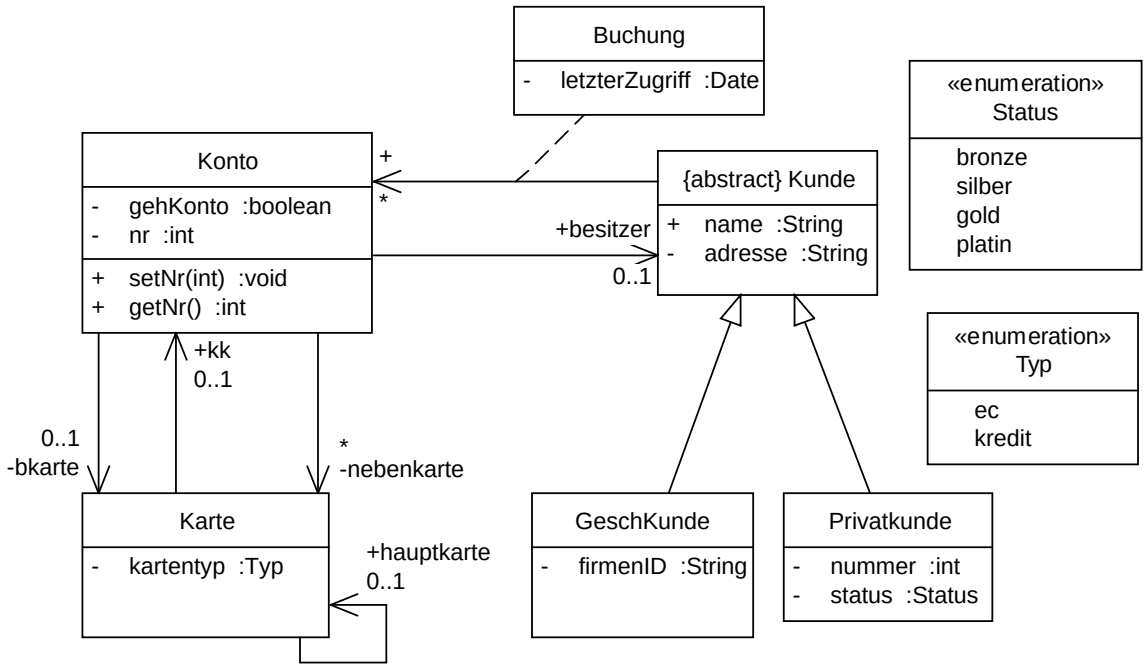


Aufgabe 3: Reverse Engineering

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.

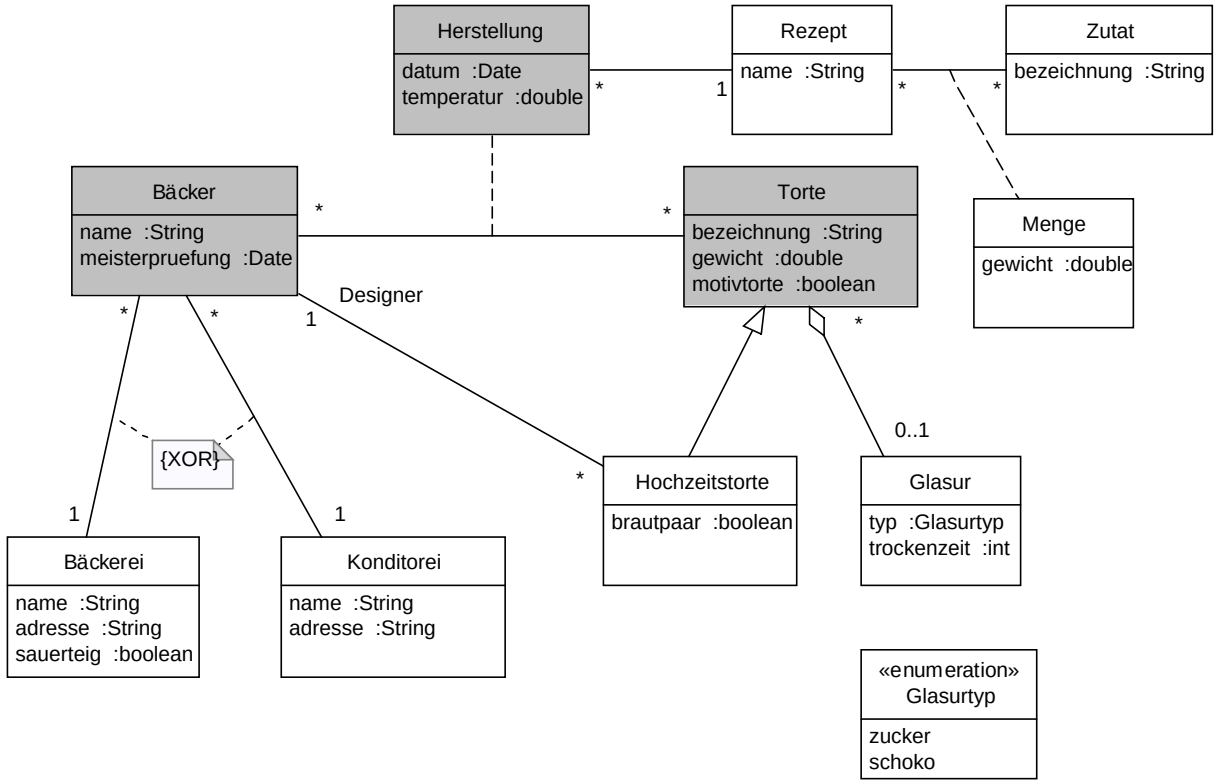
```
class Karte {
  private Typ kartentyp;
  public Karte hauptkarte;
  public Konto kk;
}

enum Typ {
  ec,
  kredit;
}

class Konto {
  private boolean gehKonto;
  private int nr;
  private Karte bkarte;
  private Karte[] nebenkarte;
  public Kunde beistzer;

  public int getNr() {
    return nr;
  }

  public void setNr(nr) {
    this.nr = nr;
  }
}
```

```
abstract class Kunde {
  public String name;
  private String adresse;
  public Hashtable buchung;
    // Key: Konto
    //   (Typ: Konto)
    // Value: letzterZugriff
    //   (Typ: Date)
}

class Privatkunde extends Kunde{
  private int nummer;
  private Status status;
}

class GeschKunde extends Kunde{
  private String firmenID;
}

enum Status {
  bronze,
  silber,
  gold,
  platin;
}
```

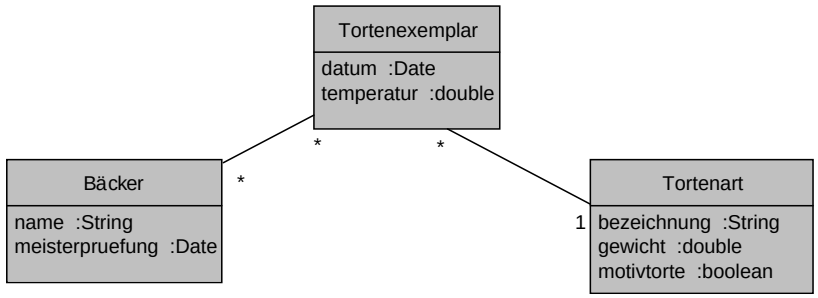


Aufgabe 4: Torte

Bilden Sie die folgenden Sachverhalte mit einem Klassendiagramm ab.
Ein Bäcker bäckt mehrere Torten, eine Torte kann von mehreren Bäckern gebacken werden. Vom Bäcker werden der Name und das Datum, an dem er die Meisterprüfung bestanden hat, gespeichert. Von jeder Torte werden die Bezeichnung und das Gewicht gespeichert und ob es sich um eine Motivtorte handelt oder nicht. Von jeder Torte die von einem oder mehreren Bäckern gebacken wird, wir das Datum der Herstellung und die Backtemperatur gespeichert. Dabei wird außerdem ein bestimmtes Rezept verwendet, das mehrere Zutaten in einer bestimmten Menge (Gewicht) benötigt. Eine Torte kann eine Glasur haben, von der die Trockenzeit und die Art der Glasur – Zucker oder Schokolade – gespeichert werden. Die Hochzeitstorte ist eine spezielle Torte, von der zusätzlich gespeichert wird, ob ein Miniatur-Brautpaar an der Torte angebracht werden soll. Eine Hochzeitstorte wird von einem Bäcker erfunden, dem Designer. Jeder Bäcker ist entweder in einer Bäckerei angestellt oder in einer Konditorei. Von beiden werden Name und Adresse gespeichert. Von der Bäckerei wird zusätzlich gespeichert, ob sie ihren eigenen Sauerteig herstellt.



Ausschnitt einer alternativen Lösung, sodass ein Tortenexemplar von mehreren Bäckern hergestellt werden kann:



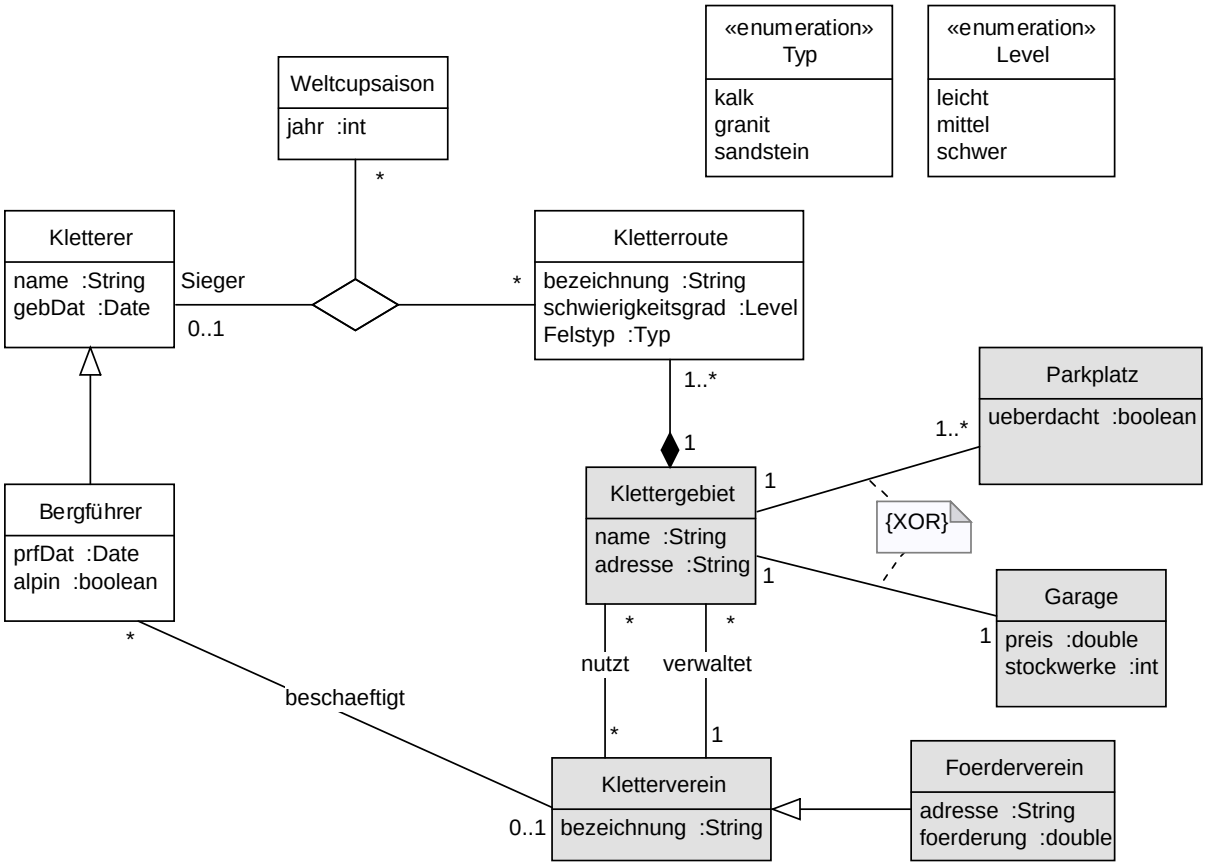
Aufgabe 5: Klettern I

Bilden Sie die folgenden Sachverhalte mit einem Klassendiagramm ab.
Von jedem Kletterer werden Name und Geburtsdatum gespeichert. Es gibt spezielle Kletterer, die Bergführer, von denen zusätzlich das Datum der Bergführerprüfung gespeichert wird und ob diese auch im alpinen Gelände tätig sein dürfen. Beim Kletterweltcup werden mehrere Kletterrouten benutzt. Von jeder Kletterroute werden die Bezeichnung, der Schwierigkeitsgrad (leicht, mittel, schwer) und der Felstyp (kalk, granit, sandstein) gespeichert. In einer Weltcupsaison kann eine Kletterroute genau ein Kletterer gewinnen, dieser wird dann als Sieger bezeichnet.

Aufgabe 6: Klettern II

Erweitern Sie das Klassendiagramm aus Aufgabe 5 wie folgt:
Eine Kletterroute ist Teil von einem Klettergebiet, ein Klettergebiet beinhaltet mindestens eine Kletterroute. Von jedem Klettergebiet werden Name und Adresse gespeichert. Ein Klettergebiet hat entweder eine Garage oder einen oder mehrere Parkplätze. Von der Garage wird der Preis gespeichert und wie viele Stockwerke sie hat, vom Parkplatz wird gespeichert, ob dieser überdacht ist. Jedes Klettergebiet wird von mehreren Klettervereinen genutzt, aber nur von einem Kletterverein verwaltet. Vom Kletterverein wird die Bezeichnung gespeichert. Jeder Kletterverein beschäftigt mehrere Bergführer, ein Bergführer kann in einem Kletterverein tätig sein. Der Förderverein ist ein spezieller Kletterverein, der vom Land gefördert wird. Von ihm wird zusätzlich die Adresse und die Höhe der Förderung gespeichert.

Lösung für I und II:



VU Objektorientierte Modellierung		Übung 3: Sequenzdiagramm
188.391, SS2018	MUSTERLÖSUNG	Übungsgruppen: 23.04.2018 - 26.04.2018

Hinweise:

- Theoriefragen: Erläutern Sie die Fragen anhand eines **selbst gewählten, konkreten Beispiels**.
  - selbst gewählt ... **nicht** aus der Vorlesung oder aus den Lösungen anderer Übungsaufgaben
  - konkret ... z.B. Nachricht „check(x)“ anstelle abstrakt Nachricht „n“
- Modellierungsbeispiele: Bilden Sie den Sachverhalt, der in der Angabe geschildert wird, möglichst genau ab. Sollte etwas in der Angabe nicht erwähnt sein, treffen Sie sinnvolle Annahmen.

Aufgabe 1: Verhaltensmodellierung mittels Sequenzdiagramm

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

- Welche 4 Arten von Interaktionsdigrammen 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: Synchrone/Asynchrone Kommunikation

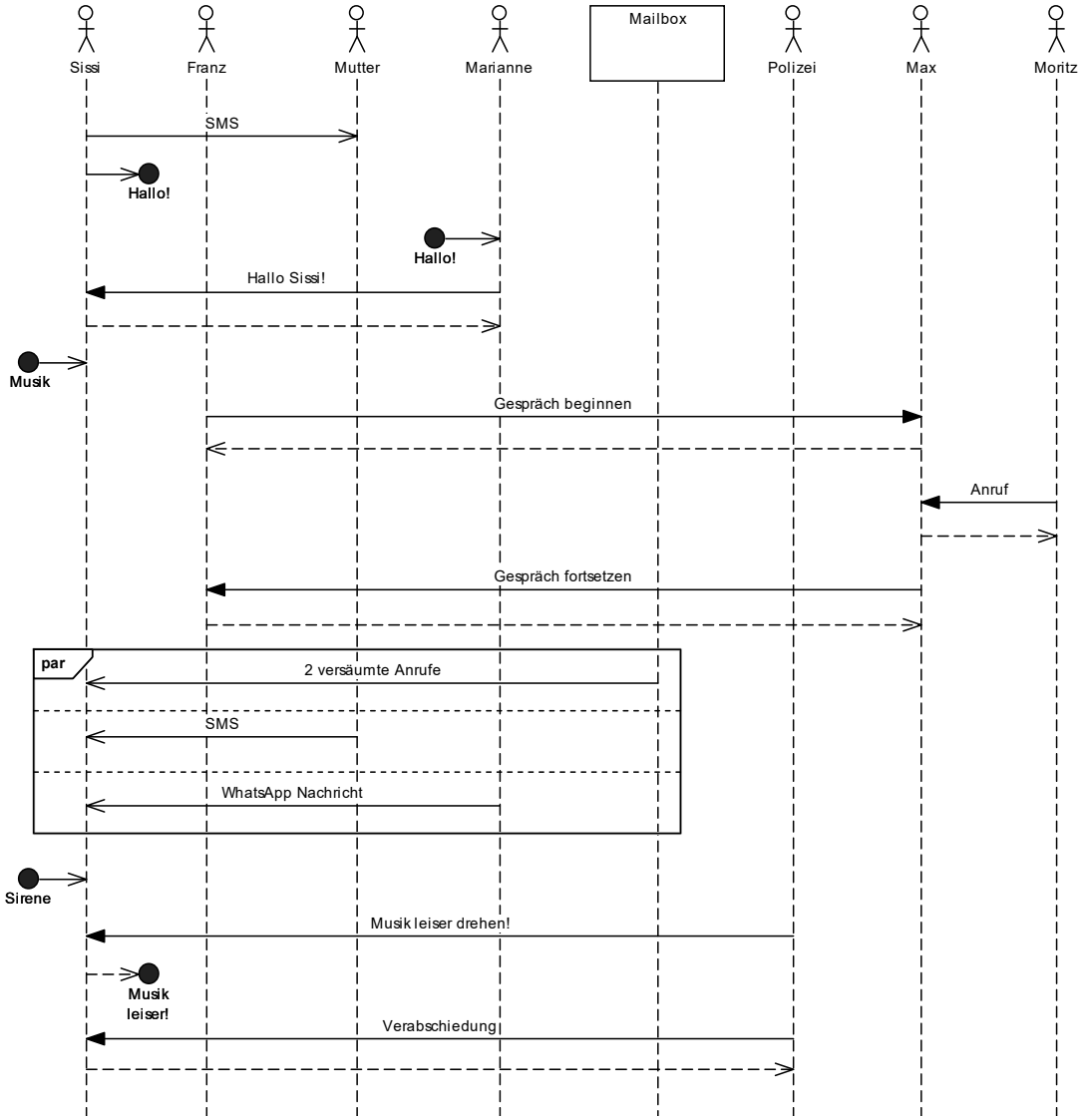
Beschreiben Sie die im folgenden Text vorkommenden Kommunikationsabläufe mittels Sequenzdiagramm.

Sissi und Franz sind am Weg zur Party von Norbert, einem Freund von Franz. Im Auto schreibt Sissi ihrer Mutter noch ein SMS, um sie zu fragen, was sie sich zum Geburtstag wünscht. Als sie vor dem Haus von Norbert eintreffen, sehen sie schon Marianne, die mit einer Freundin vor der Tür steht und raucht. Sissi schreit Marianne freudig „Hallo!“ zu, als sie sich mit Franz dem Haus nähert. Es dauert einen Moment, bis Marianne aus ihrem Gespräch aufhorcht und schaut, wer da ihren Namen gerufen hat. Als sie Sissi erblickt, schreit sie erfreut „Hallo Sissi!“ zurück! Nach einem kurzen Gespräch mit Marianne, betreten Franz und Sissi das Haus. Sie hören die laute Musik, die durch das ganze Haus schallt. Sissi weiß gar nicht, wo genau sie eigentlich herkommt.

Franz trifft seinen alten Freund Max und sie suchen sich ein stilles Plätzchen, um sich über Ihr aktuelles Leben auszutauschen. Franz erzählt Max von seinem Unfall mit dem Fahrrad und dieser wiederum berichtet von seiner neuen, großen Liebe, Moritz. Plötzlich läutet Maxs Telefon. Es ist Moritz, der etwas verzweifelt wirkt, weil er Max in der Menge nicht mehr finden kann. Max teilt ihm mit, wo er zu finden ist, legt auf und schwärmt Franz weiter von Max vor.

Sissi schaut auf ihr Handy und stellt fest, die Mailbox meldet 2 versäumte Anrufe, ein SMS von ihrer Mutter und sie findet eine WhatsApp Nachricht von Marianne, die fragt, wo im Haus sie sie finden kann. Plötzlich hört sie laute Sirenen von außerhalb des Hauses und weiß im ersten Moment gar nicht, wo diese herkommen. Im nächsten Moment steht die Polizei vor der offenen Haustür und verwarnt Sissi, die gerade bei der Tür steht, dass sie die Musik leiser drehen soll, da die Nachbarn sich bereits beschwert haben. Sissi schreit „Musik leiser“ durchs ganze Haus und hofft, dass jemand sie hört und die Musikanlage leiser dreht. Plötzlich ist es still im Haus, jemand hat die Musikanlage ausgeschaltet. Die Polizei ermahnt Sissi noch einmal, dass sie Rücksicht auf ihre Nachbarschaft nehmen sollen und verabschiedet sich. Sissi verspricht das und verabschiedet sich ebenfalls von der Polizei. Alles nochmal gut gegangen!

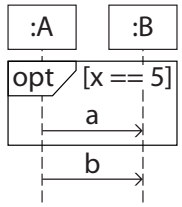
Achten Sie bei diesem Beispiel besonders darauf, ob die beschriebenen Kommunikationsabläufe synchron oder asynchron sind.


Sissi
Franz
Mutter
Marianne
Mailbox
Polizei
Max
Moritz
SMS
Hallo!
Hallo!
Hallo Sissi!
Musik
Gespräch beginnen
Anruf
Gespräch fortsetzen
par
2 versäumte Anrufe
SMS
WhatsApp Nachricht
Sirene
Musik leiser drehen!
Musik leiser!
Verabschiedung

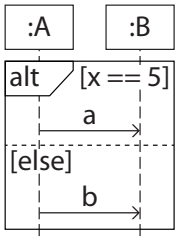
Aufgabe 4: Kombinierte Fragmente

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)



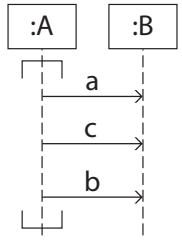
b)



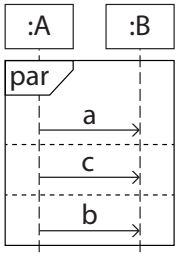
☐ äquivalent

☒ nicht äquivalent

a)



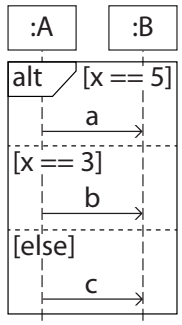
b)



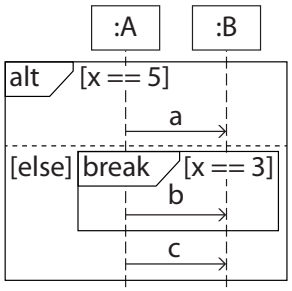
☒ äquivalent

☐ nicht äquivalent

a)



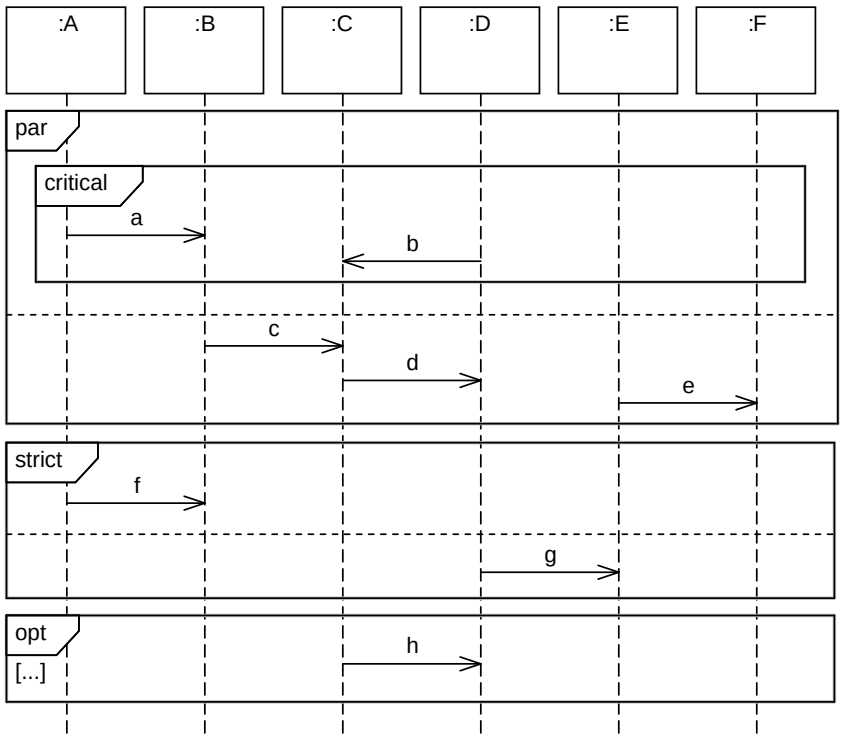
b)



☒ äquivalent

☐ nicht äquivalent

b) **Berechnung von Traces**
Beschreiben Sie alle möglichen Ereignisfolgen des folgenden Diagramms.



- `par`: Die Nachrichten der zwei `par`-Abschnitte können beliebig kombiniert werden, solange die folgenden Einschränkungen berücksichtigt werden:
  - Erster `par`-Abschnitt: `critical`; die Reihenfolge von `a` und `b` ist egal (da sie sich keine Lebenslinie teilen), dazwischen dürfen aber keine anderen Nachrichten kommen.
  - Zweiter `par`-Abschnitt: `seq` (default-Ordnung); `c` muss jedenfalls vor `d` kommen, da sich die beiden eine Lebenslinie teilen; `e` kann davor, dazwischen oder danach folgen.
    Mögliche Sequenzen: `c` → `d` → `e` oder `e` → `c` → `d` oder `c` → `e` → `d`

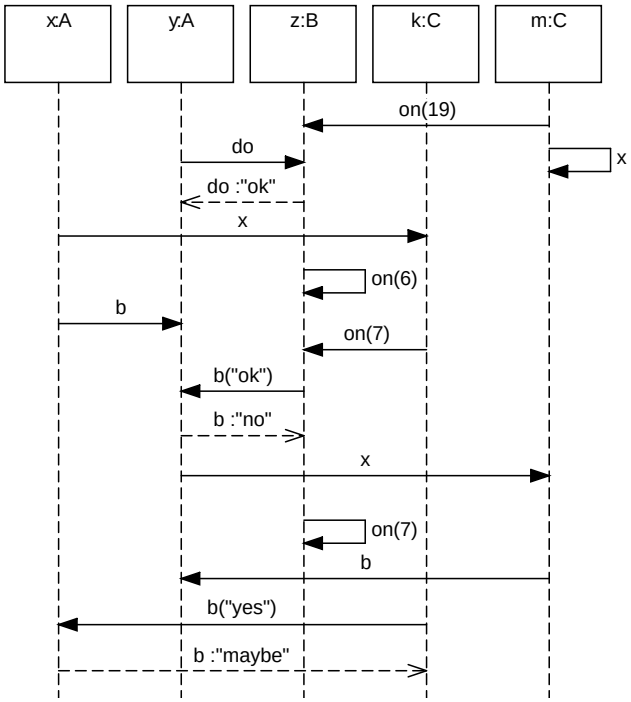
  Mögliche Sequenzen zum Beispiel:
  `b` → `a` → `c` → `d` → `e`
  `c` → `a` → `b` → `d` → `e`
  NICHT möglich zum Beispiel:
  `a` → `c` → `b` → `d` → `e`

- `strict`: genau diese Reihenfolge muss eingehalten werden
  `f` → `g`

- `opt`: abhängig von der Bedingung wird `h` ausgeführt oder nicht.

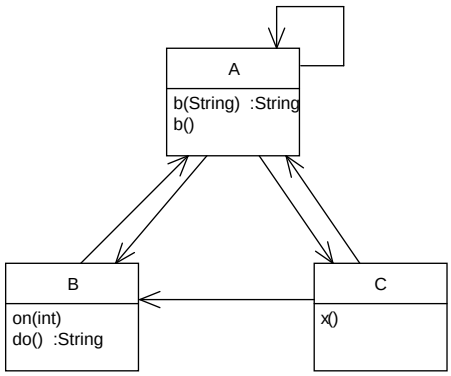
Aufgabe 5: Klassendiagramm aus Sequenzdiagramm

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.

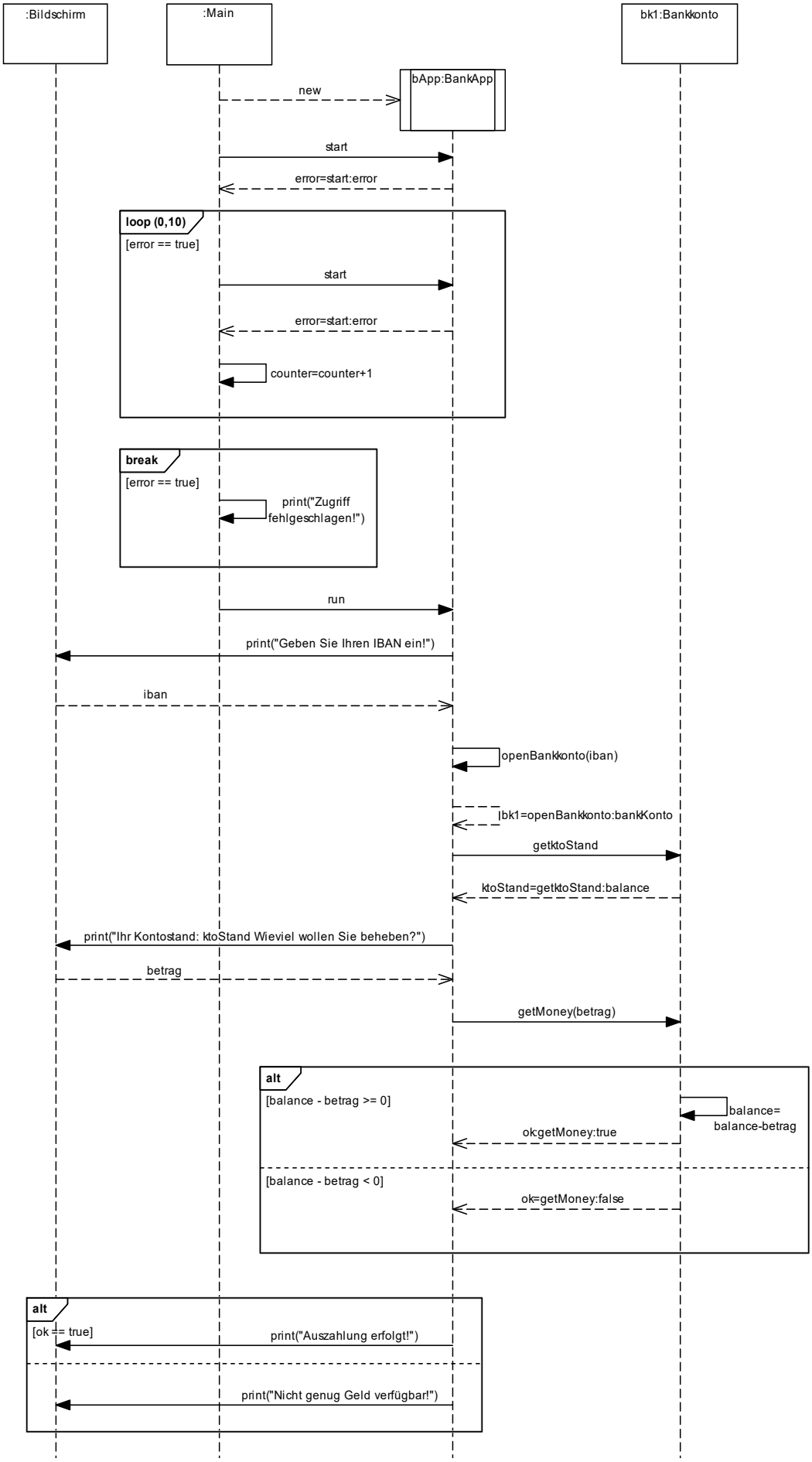


Aufgabe 6: Darstellung von Programmabläufen mittels Sequenzdiagramm

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. „...“ markiert vernachlässigte Codeteile, die nicht modelliert werden müssen.

```
class Main {
  ...
  BankApp bApp = new BankApp();
  boolean error = bApp.start();
  int counter = 0;

  while (error == true && counter < 10) {
    error = bApp.start();
    counter = counter + 1; }

  if (error == true) {
    print("Zugriff fehlgeschlagen!");
    exit; }  // Programm wird beendet

  bApp.run();
  ...

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

class BankApp extends Thread {

  public boolean start() {
    boolean error = true;
    ...
    return error;
  }

  public Bankkonto openBankkonto(String iban) {
    ...
    return bankKonto;
  }

  public boolean run() {

    print("Geben Sie Ihren IBAN ein!"); // Sichtbar am Bildschirm

    // Der User gibt den IBAN ein: AT081547110815
    ...

    Bankkonto bk1 = bApp.openBankkonto("AT081547110815");
    double ktoStand = bk1.getktoStand();
    ...
    // Sichtbar am Bildschirm:
    print("Ihr Kontostand:" + ktoStand + "Wieviel wollen Sie beheben?")
                ...

    double betrag = 200; // Usereingabe
    boolean ok = bk1.getMoney(betrag);
```

```
    if (ok == true)
      print("Auszahlung erfolgt!"); // Sichtbar am Bildschirm
    else
      print("Nicht genug Geld verfügbar!"); // Sichtbar am Bildschirm
    ...
  }

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

class Bankkonto() {
  private double balance;

  public double getktoStand() {
    return balance;
  }

  public boolean getMoney(double betrag) {
    if (balance - betrag >= 0) {
      balance = balance - betrag;
      return true;
    }
    else (balance - betrag < 0)
      return false;
  }
}
```


:Bildschirm
:Main
bk1:Bankkonto
bApp:BankApp
new
start
error=start:error
loop (0,10)
[error == true]
start
error=start:error
counter=counter+1
break
[error == true]
print("Zugriff fehlgeschlagen!")
run
print("Geben Sie Ihren IBAN ein!")
iban
openBankkonto(iban)
bk1=openBankkonto:bankKonto
getktoStand
ktoStand=getktoStand:balance
print("Ihr Kontostand: ktoStand Wieviel wollen Sie beheben?")
betrag
getMoney(betrag)
alt
[balance - betrag >= 0]
balance= balance-betrag
ok:getMoney:true
[balance - betrag < 0]
ok=getMoney:false
alt
[ok == true]
print("Auszahlung erfolgt!")
print("Nicht genug Geld verfügbar!")

VU Objektorientierte Modellierung		Übung 4: Zustandsdiagramm
188.391, SS2018	MUSTERLÖSUNG	Übungsgruppen: 02.05.2018 - 08.05.2018

Hinweise:

- Theoriefragen: Erläutern Sie die Fragen anhand eines **selbst gewählten, konkreten Beispiels**.
  - selbst gewählt ... **nicht** aus der Vorlesung oder aus den Lösungen anderer Übungsaufgaben
  - konkret ... z.B. Zustand „bereit“ anstelle abstrakt Zustand „Z“
- Modellierungsbeispiele: Bilden Sie den Sachverhalt, der in der Angabe geschildert wird, möglichst genau ab. Sollte etwas in der Angabe nicht erwähnt sein, treffen Sie sinnvolle Annahmen.

Aufgabe 1: Theoriefragen 1

Beantworten Sie folgende Fragen:

a) Erklären Sie die Konzepte *Ereignis*, *Bedingung* und *Aktivität*.

b) Welche vordefinierten Aktivitäten gibt es innerhalb eines Zustands?

c) Wann erfolgt eine Transition (von einem Zustand in einen anderen)?

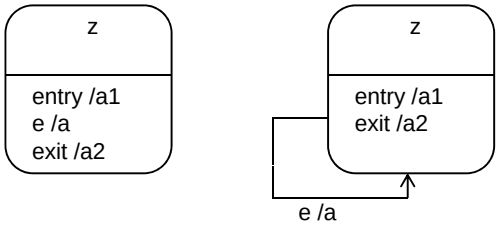
d) Was versteht man unter einem Historischen Zustand? Wann, warum und wie wird er eingesetzt?

Aufgabe 2: Theoriefragen 2

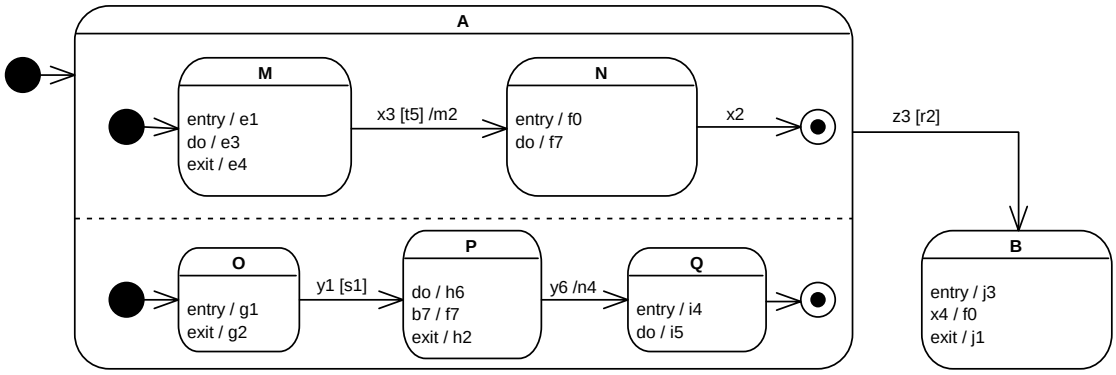
Beantworten Sie folgende Fragen:

a) Erklären Sie das Konzept der UND- sowie der ODER-Verfeinerung (Folien 20/21).

b) Gegeben sind folgende zwei Ausschnitte eines Zustandsdiagramms. Sind die beiden Ausschnitte äquivalent? Begründen Sie Ihre Antwort!



c) **Gegeben ist das nachfolgende Zustandsdiagramm.**



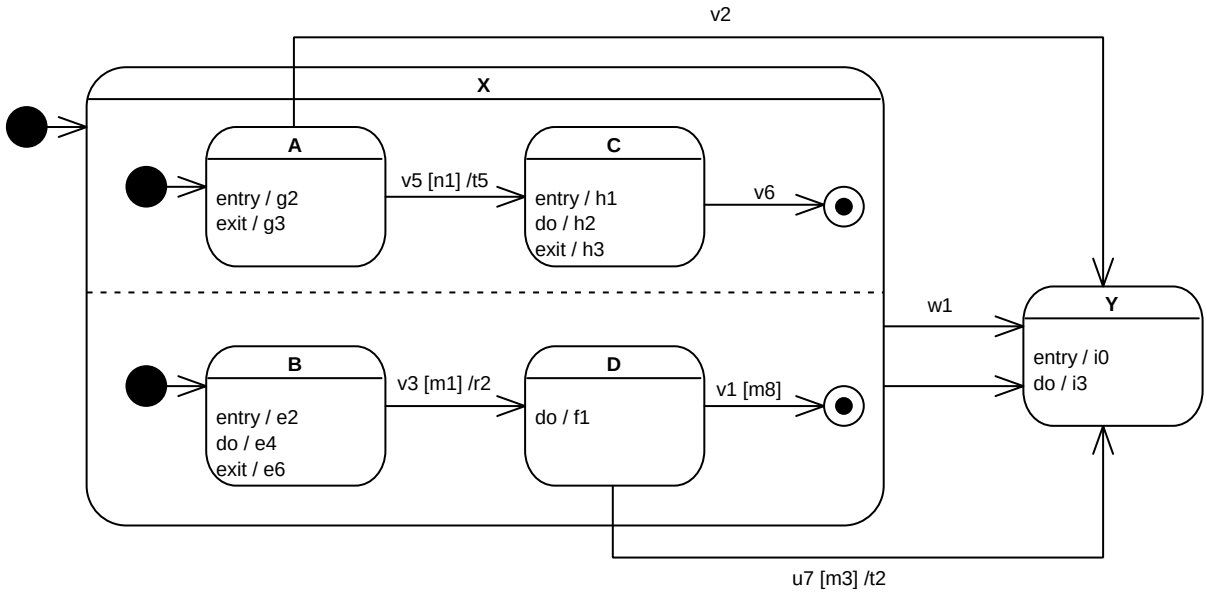
Beantworten Sie folgende Fragen:

- Welche Zustände gibt es in diesem Diagramm?
  A, B, M, N, O, P, Q sowie die Startzustände (Pseudozustand) und Endzustände

- Welche Ereignisse gibt es in diesem Diagramm?
  b7, x2, x3, x4, y1, y6, z3 sowie die vordefinierten (entry, do, exit);
- Welche Bedingungen gibt es in diesem Diagramm?
  r2, s1, t5
- Welche Aktivitäten gibt es in diesem Diagramm?
  e1, e3, e4, f0, f7, g1, g2, h2, h6, i4, i5, j1, j3, m2, n4
- In welchem Zustand/welchen Zuständen befindet sich der Automat unmittelbar nach dem Start?
  M und O
- In welchem Zustand/welchen Zuständen muss sich der Automat befinden, damit er nach dem Eintritt von z3 in den Zustand B übergeht (vorausgesetzt die Bedingung r2 ist erfüllt)?
  Er geht in den Zustand B über, egal in welchem Zustand/welchen Zuständen von A er sich vorher befunden hat. (M xor N xor Ende oberer Abschnitt) and (O xor P xor Q xor Ende unterer Abschnitt
- 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 der folgenden Kombinationen von Zuständen kann sich das System zu einem Zeitpunkt gleichzeitig befinden?
  - A und C **nein**
  - D und B **nein**
  - A und B **ja**
  - A und Y **nein**
  - X und Y **nein**
  - C und B **ja**
  - C und D **ja**
  - A und D und Y **nein**

- Welche Möglichkeit(en) gibt es, dass das System vom Zustand X in den Zustand Y übergeht?

  System befindet sich in beiden Subzustandsfolgen von X am Ende.
  oder
  Ereignis w1 tritt ein.
  oder
  System befindet sich (unter anderem) im Zustand D, das Ereignis u7 tritt ein und die Bedingung m3 ist erfüllt.
  oder
  System befindet sich (unter anderem) im Zustand A und das Ereignis v2 tritt ein.

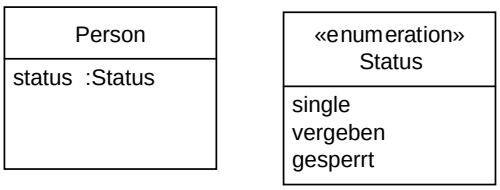
Aufgabe 4: Familienstand

Ziel dieses Beispiels ist die Abbildung des Familienstands einer Person aus Sicht einer Partnervermittlungsagentur.

Dazu gehen Sie bitte wie folgt vor:

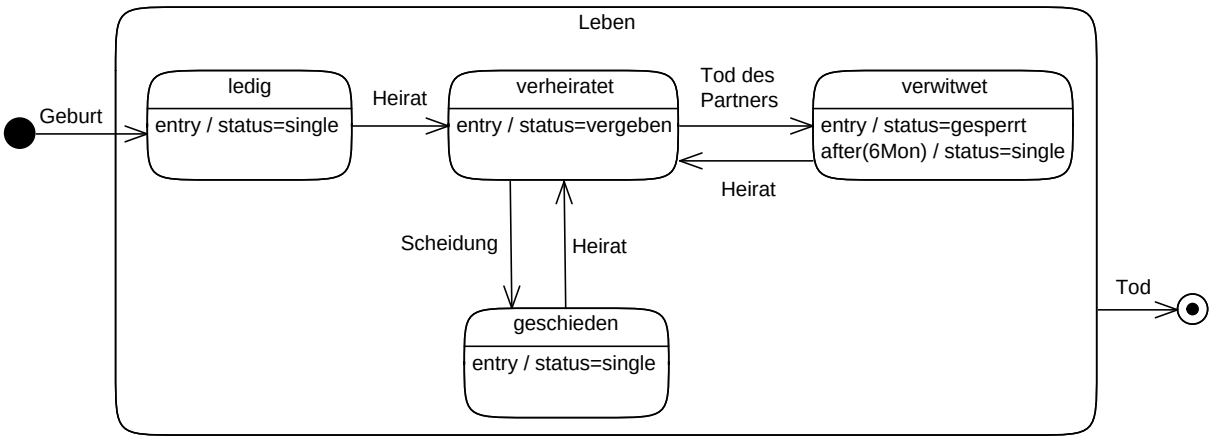
- Eine Person kann ledig, verheiratet, verwitwet oder geschieden sein. Modellieren Sie ein Zustandsdiagramm, das diesen Wechsel zwischen den Familienständen von der Geburt einer Person bis zu deren Tod abbildet. Überlegen Sie, welche Ereignisse jeweils zu einem Zustandswechsel führen.

- Eine Partnervermittlungsagentur interessiert sich nur dafür, ob ihre Klienten „single“, „vergeben“, oder „gesperrt“ sind – verstirbt der Partner einer Person, so gilt diese für 6 Monate als „gesperrt“ (außer er bzw. sie heiratet früher wieder). Ein Auszug aus dem zugehörigen Klassendiagramm sieht wie folgt aus:



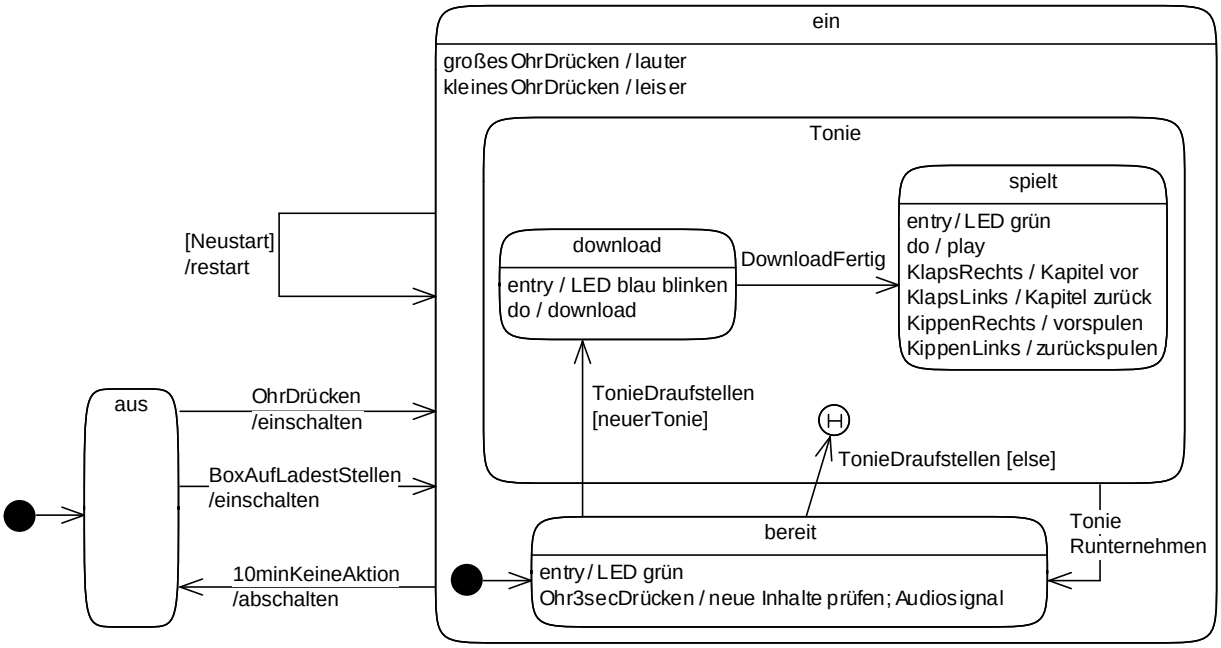
  Erweitern bzw. modifizieren Sie ihr Zustandsdiagramm so, dass die Änderungen am „Status“ einer Person explizit abgebildet werden.

- Betrachten Sie Ihr fertiges Zustandsdiagramm. Ist es übersichtlich und gut lesbar? Lässt es sich eventuell noch vereinfachen? Können Zustände zu komplexen (also zusammengesetzten) Zuständen zusammengefasst werden um die Lesbarkeit zu erhöhen?
  Überarbeiten Sie falls nötig Ihr Zustandsdiagramm um es möglichst übersichtlich und lesbar zu gestalten.



Aufgabe 5: Toniebox

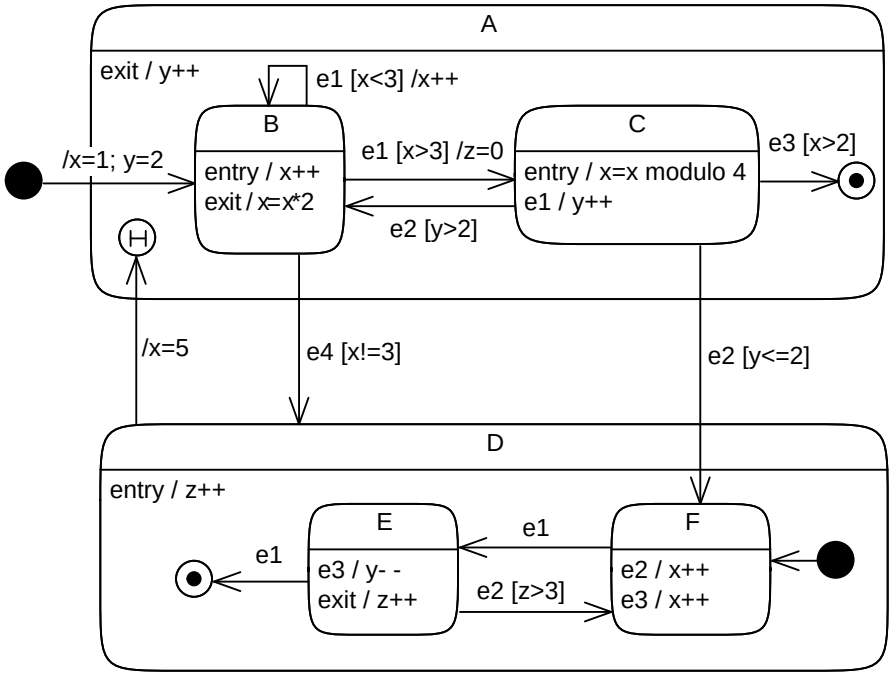
Im Anschluss an das Übungsblatt finden Sie einen Ausschnitt aus der Bedienungsanleitung der „Toniebox“[1]. Modellieren Sie ein UML Zustandsdiagramm, das die Zustände und Zustandsübergänge aus der Sicht der Toniebox abbildet.
Punkt 5., „Toniebox aufladen“, können Sie ignorieren,
Treffen Sie sinnvolle Annahmen wenn Informationen fehlen.



[1] https://tonies.de/

Aufgabe 6: Ereignisfolge

Gegeben ist das folgende Zustandsdiagramm:



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
Beginn	A/B	1/2	2	
e1	A/B	4/5/6		
e1	A/C	12/0		0
e2	D/F		3	1
e1	D/E			
e2	D/E			
e3	D/E		2	
e1	A/C	5/1		2

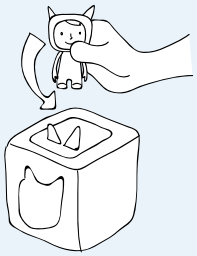
1 Toniebox ein- und ausschalten

Anschalten Mit Daumen und Zeigefinger eines der Ohren drücken oder die Box auf die Ladestation stellen.

Ausschalten Noch einfacher: Erfolgt 10 Minuten keine Aktion, schläft sie automatisch ein.

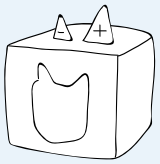
Neustarten Toniebox umdrehen und beide Ohren 10 Sek. drücken.
Achtung: Dabei darf die Toniebox nicht auf der Ladestation stehen.

2 Loshören und anhalten



Leuchtet die LED an deiner Toniebox grün, kannst du einen Tonie aufstellen. Blaues Blinken zeigt an, dass Inhalte aus der Toniecloud geladen werden. Nach vollständigem Download kann das Hörabenteuer beliebig oft abgespielt werden, auch ohne Internetverbindung. Geschichte anhalten? Einfach den Tonie runternehmen. Stellst du ihn wieder drauf, gehts an gleicher Stelle weiter.

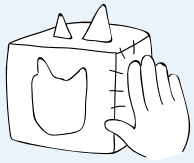
3 Lauter und leiser machen



Drückst du das große Ohr, wird die Toniebox lauter, und mit dem kleinen Ohr wird sie leiser.

Achtung: Kinderohren können sehr empfindlich sein. Über die Toniecloud kannst du die maximale Lautstärke einstellen.

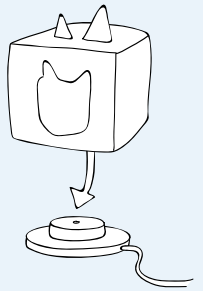
4 Vor- und zurückspringen



Um zwischen den Kapiteln vor- und zurückzuspringen, gibst du der Toniebox einen Klaps auf die Seite. Zum Vor- und Zurückspulen kippst du sie einfach zur Seite.

Hinweis: Rechts vor, links zurück? Oder lieber umgekehrt? Über die Toniecloud legst du selber fest, mit welcher Seite du vor- und zurücksteuerst. So, wie es sich für dich am besten anfühlt.

5 Toniebox aufladen



Auch die Toniebox muss sich mal ausruhen und Kraft tanken. Beim Herunternehmen von der Ladestation zeigt dir die LED den aktuellen Ladezustand durch Blinken an.

Bitte weiterladen

Akku ausreichend geladen

6 Auf neue Inhalte prüfen

Dein Tonie spielt veraltete Inhalte? Einfach eines der beiden Ohren drei Sekunden drücken und die Toniebox prüft, ob neue Inhalte vorliegen. Hierbei darf kein Tonie auf der Box stehen. Ein Audiosignal meldet, ob die Prüfung erfolgreich war.

tonies®

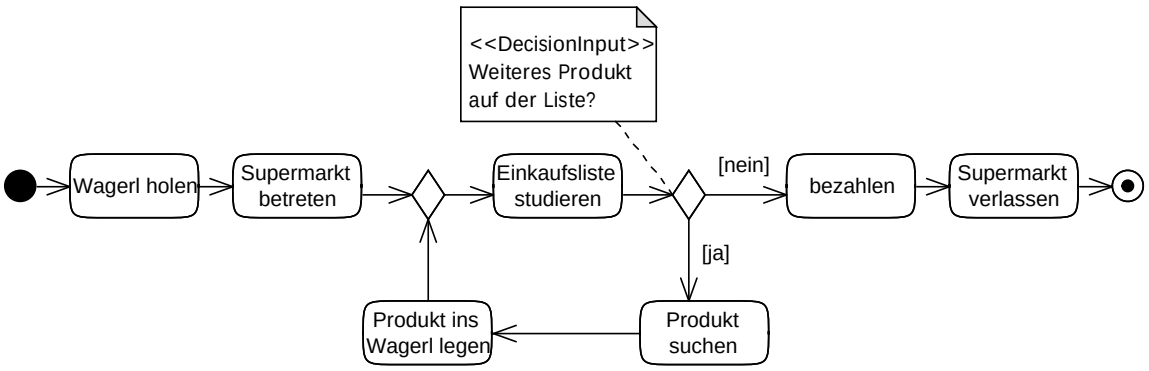
Hinweise:

- Theoriefragen: Erläutern Sie die Fragen anhand eines **selbst gewählten, konkreten Beispiels**.
  - selbst gewählt ... **nicht** aus der Vorlesung oder aus den Lösungen anderer Übungsaufgaben
  - konkret ... z.B. Aktivität „Prüfung schreiben“ anstelle abstrakt Aktivität „A“
- Modellierungsbeispiele: Bilden Sie den Sachverhalt, der in der Angabe geschildert wird, möglichst genau ab. Sollte etwas in der Angabe nicht erwähnt sein, treffen Sie sinnvolle Annahmen.

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

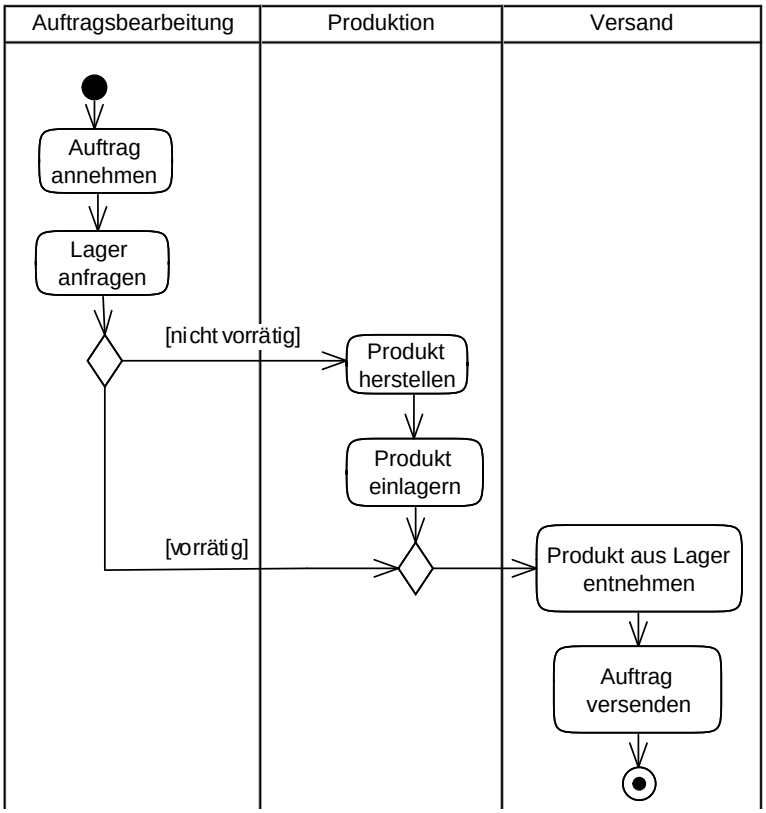
a) Wodurch unterscheiden sich Aktivität und Aktion? Geben Sie ein Beispiel, das den Unterschied verdeutlicht.

b) Modellieren Sie folgenden Ablauf (Kontrollfluss) beim Einkaufen mittels Aktivitätsdiagramm:
Zunächst holt man sich ein Einkaufswagerl und betritt anschließend den Supermarkt. Dann wird die Einkaufsliste angeschaut. Befindet sich ein Produkt auf der Liste so wird dieses gesucht und ins Wagerl gelegt. Dies wird so lange wiederholt, bis sich kein weiteres Produkt auf der Liste befindet, das noch nicht geholt wurde. Anschließend wird der Einkauf bezahlt und der Supermarkt wieder verlassen.



c) Was versteht man unter Partitionen? Wozu und wie werden sie eingesetzt?

d) Modellieren Sie folgenden Ablauf (Kontrollfluss) einer Bestellung mittels Aktivitätsdiagramm:
Die Auftragsbearbeitung nimmt den Auftrag an und fragt beim Lager an, ob das Produkt vorrätig ist. Falls es vorrätig ist, holt der Versand das Produkt aus dem Lager und versendet anschließend den Auftrag. Falls das Produkt nicht vorrätig ist, stellt die Produktion es her und lagert es ein, damit es der Versand anschlißend aus dem Lager entnehmen und versenden kann.



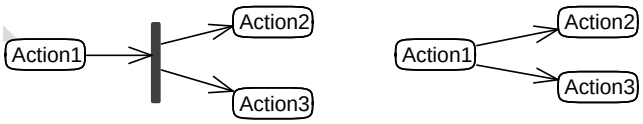
Aufgabe 2: Tokenkonzept

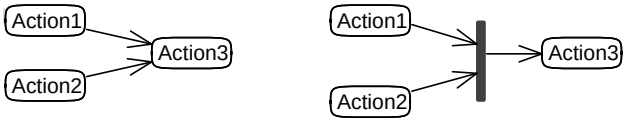
a) Was versteht man unter einem Token? Welchen Zweck hat das Tokenkonzept?

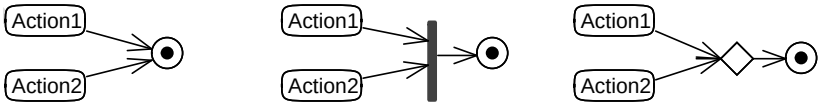
b) Wie funktioniert die Tokenverarbeitung bei Parallelisierungsknoten und Synchronisierungsknoten bzw. bei Entscheidungsknoten und Vereinigungsknoten?

c) Sind folgende Konstrukte äquivalent?

1)


2)


3)


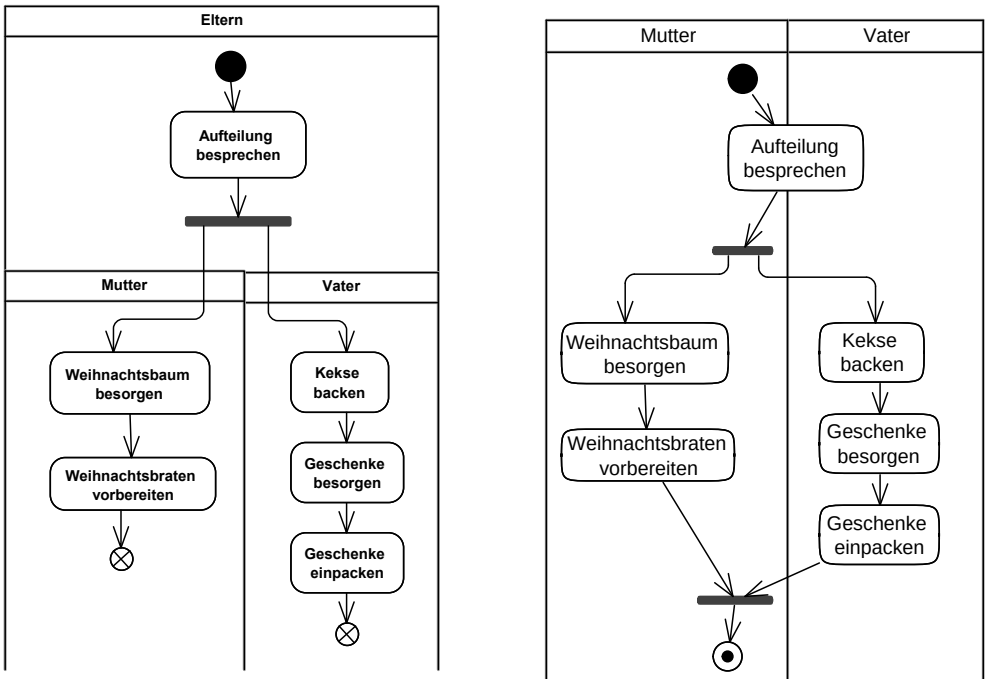
4)


Aufgabe 3: Activity Final/Flow Final – Ausnahmebehandlung

a) Wodurch unterscheiden sich Aktivitätsendknoten und Ablaufendknoten?

b) Modellieren Sie folgenden Ablauf (Kontrollfluss) mittels Aktivitätsdiagramm:
Weihnachten steht vor der Tür. Mutter und Vater besprechen zunächst, wer was erledigt. Anschließend besorgt die Mutter den Weihnachtsbaum und bereitet den Braten vor. Der Vater backt unabhängig davon Kekse, besorgt die Geschenke und packt diese ein. Dann sind die Vorbereitungen beendet.

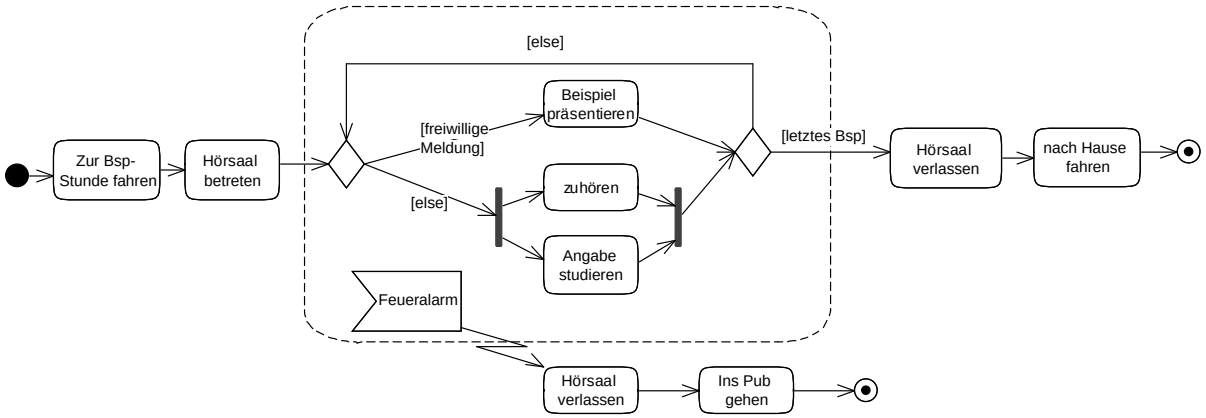
Zwei Lösungsvarianten:



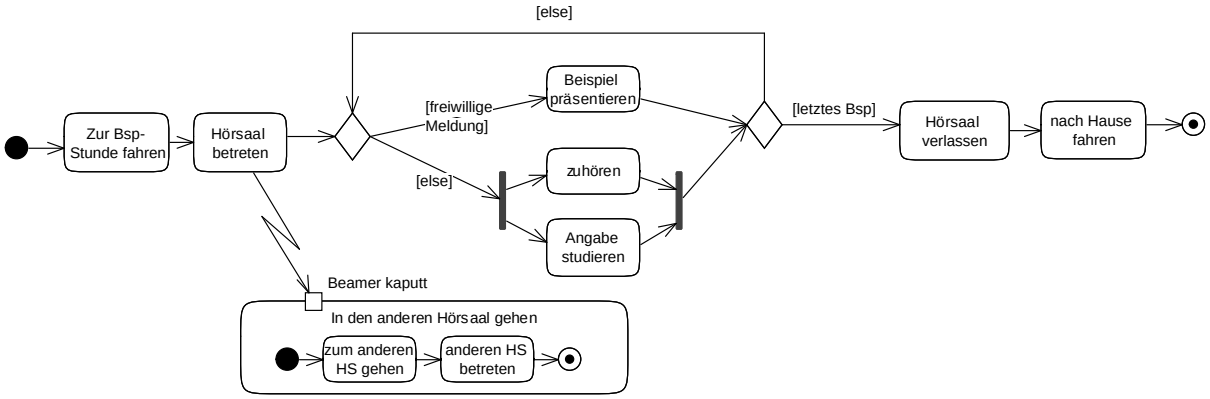
c) Gegeben ist folgendes Aktivitätsdiagramm, das den Ablauf beim Besuch einer Beispielstunde aus Studierendensicht (stark vereinfacht) zeigt:

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

(1) Während der Beispielstunde ertönt ein Feueralarm. Daraufhin verlässt der/die Studierende den Hörsaal und geht stattdessen ins Pub.



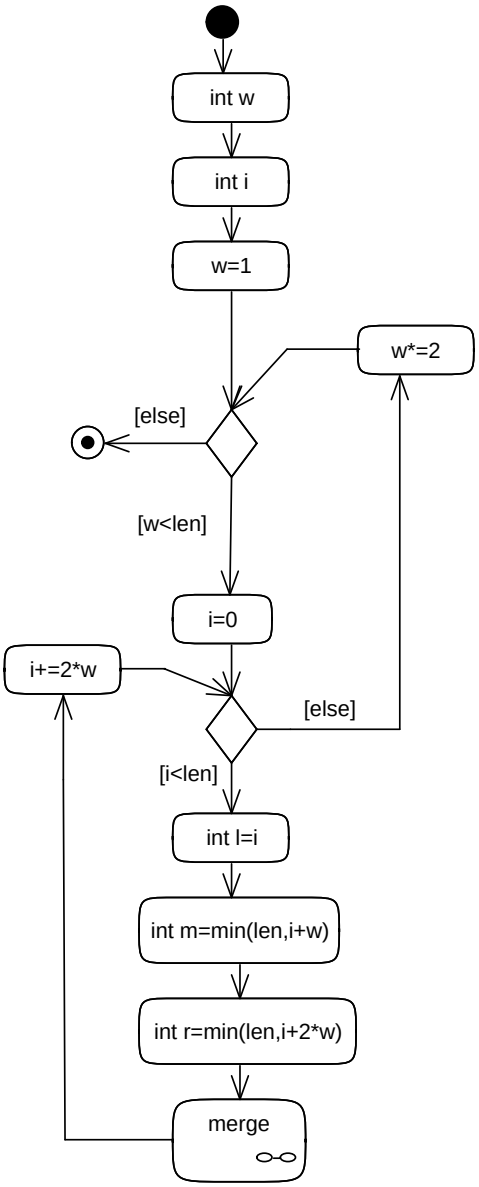
(2) Beim Betreten des Hörsaals entdeckt der/die Studierende eine Notiz an der Tafel, dass der Beamer kaputt ist und die Beispielstunde in einem anderen Hörsaal stattfindet. Daraufhin geht der/die Studierende in den anderen Hörsaal. Danach geht der Prozess regulär weiter.

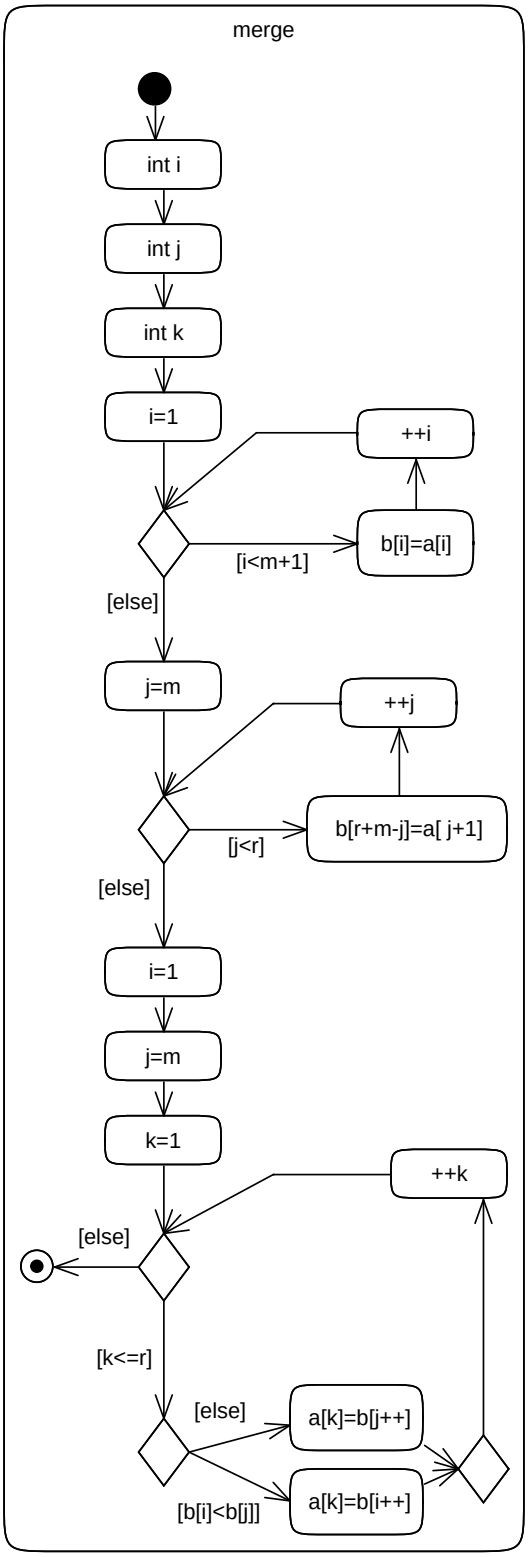


Aufgabe 4: Algorithmus

Gegeben sei eine Pseudocode-Variante des Mergesort. Modellieren Sie den Algorithmus als Aktivitätsdiagramm (nur den Kontrollfluss).

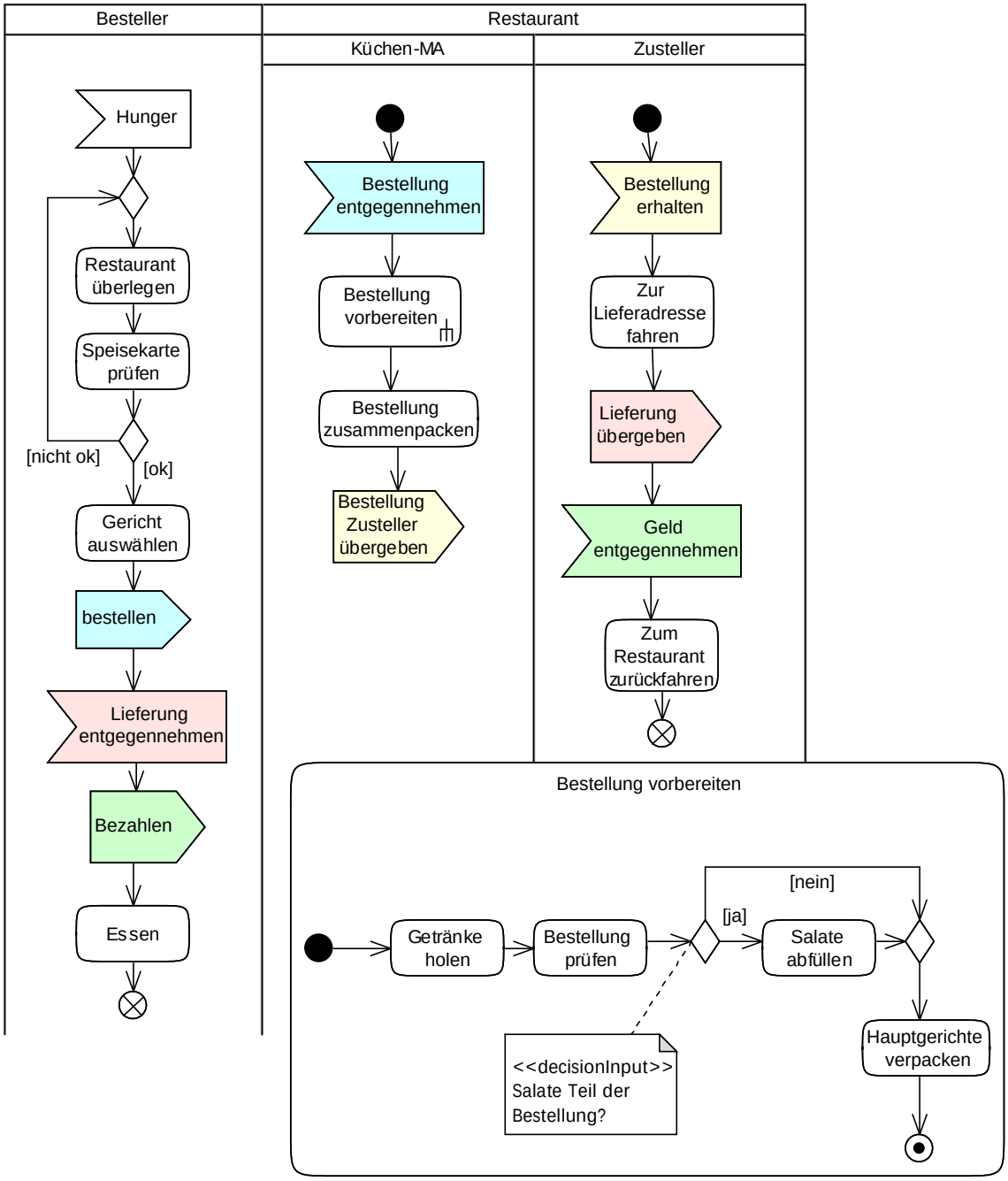
```
1  void mergesort(int a[], int b[], int len) {
2    int w, i;
3    for(w=1; w<len; w*=2) {
4      for(i=0; i<len; i+=2*w) {
5        int l = i;
6        int m = min(len, i + w);
7        int r = min(len, i + 2*w);
8        merge(a, b, l, m-1, r-1);
9      }
10   }
11 }
12
13 void merge(int a[], int b[], int l, int m, int r) {
14   int i, j, k;
15   for(i=1; i<m+1; ++i) b[i] = a[i]; // in Hilfsarray
16   for(j=m; j<r; ++j) b[r+m-j] = a[j+1];
17   i = 1;
18   j = m;
19   for(k=1; k<=r; ++k) { // eigentliches Mergen
20     if(b[i] < b[j])
21       a[k] = b[i++];
22     else
23       a[k] = b[j++];
24   }
25 }
```


int w
int i
w=1
w*=2
[else]
[w<len]
i=0
i+=2*w
[else]
[i<len]
int l=i
int m=min(len,i+w)
int r=min(len,i+2*w)
merge


merge
int i
int j
int k
i=1
++i
b[i]=a[i]
[i<m+1]
[else]
j=m
++j
b[r+m-j]=a[j+1]
[j<r]
[else]
i=1
j=m
k=1
++k
[else]
[k<=r]
[else]
a[k]=b[j++]
a[k]=b[i++]
[b[i]<b[j]]

Aufgabe 5: Send/Receive – Subprozesse

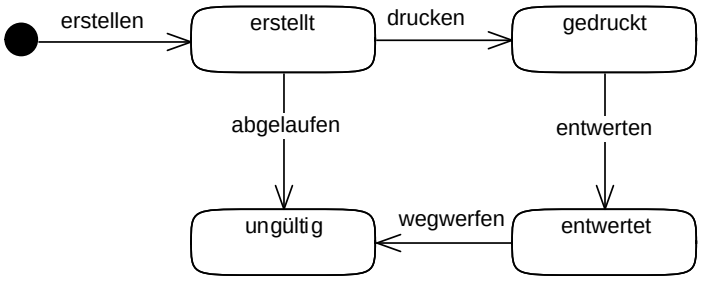
Sie haben folgende Informationen über den Ablauf einer Essensbestellung. Der Prozess beginnt damit, dass eine Person, der Besteller, Hunger hat. Daraufhin überlegt er sich ein Restaurant das in Frage kommt und studiert im Internet die Speisekarte. Ist er mit der Auswahl zufrieden, wählt er Gerichte aus und bestellt bei diesem Restaurant. Ist er nicht zufrieden, überlegt er sich ein weiteres Restaurant, prüft dessen Speisekarte usw.
Die Bestellung wird von einem Mitarbeiter der Küche des Restaurants entgegengenommen. Dieser beginnt sofort, alle Teile der Bestellung vorzubereiten. Er holt die Getränke aus dem Kühlraum, füllt die Salate in Behälter – sofern Salate Teil der Bestellung sind – und verpackt die Hauptgerichte.
Anschließend wird die Bestellung zusammengepackt und dem Zusteller übergeben. Dieser fährt zur Lieferadresse und übergibt die Lieferung. Der Besteller bezahlt die Lieferung beim Zusteller, der daraufhin zum Restaurant zurückfährt. Nun kann der Besteller essen.
Modellieren Sie den beschriebenen Ablauf als Aktivitätsdiagramm. Lagern Sie alle Schritte, die Teil des Vorbereitens der Bestellung in der Küche sind, in einen separaten Prozess aus, um die Lesbarkeit zu erhöhen.



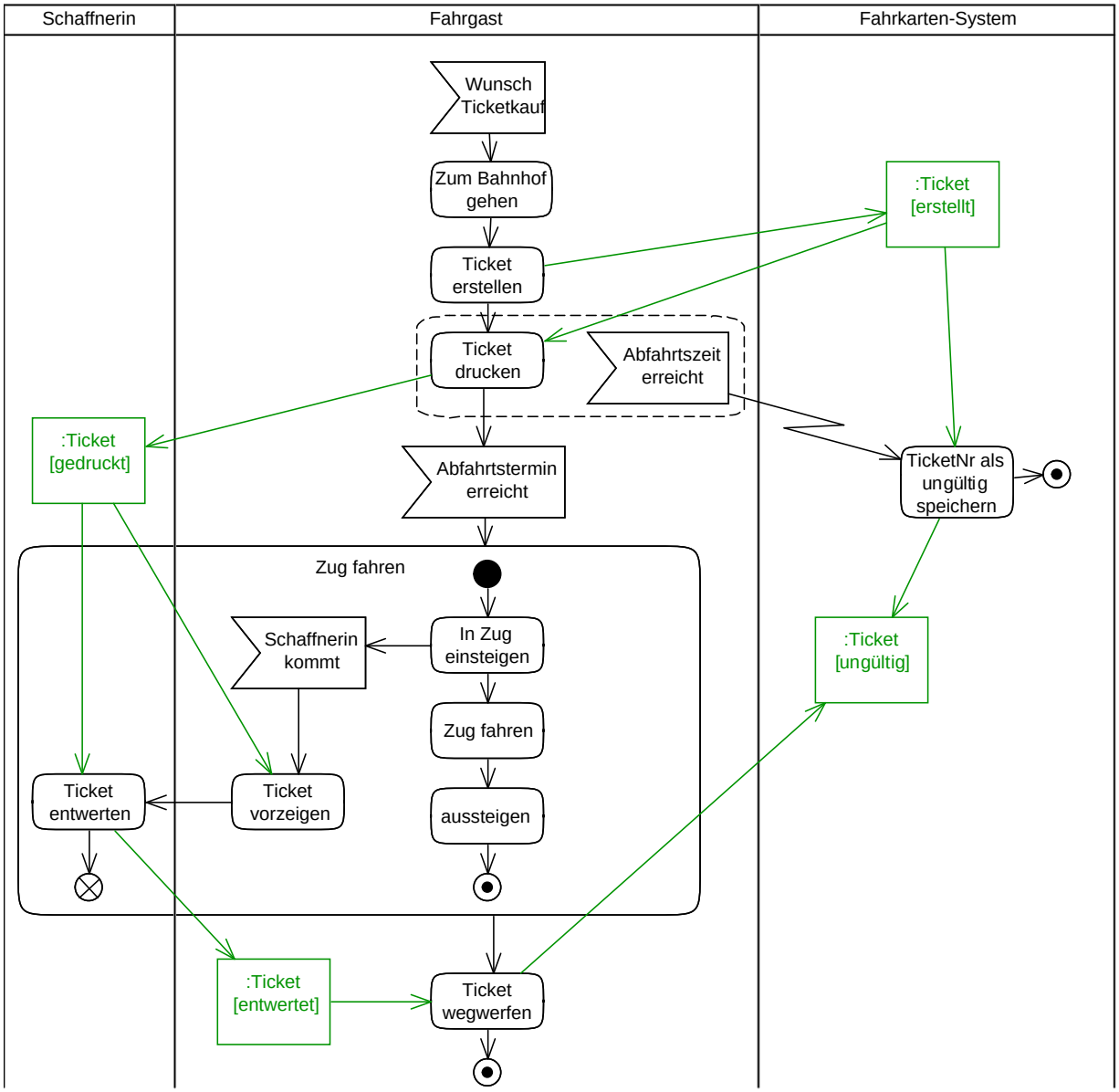
Aufgabe 6: Zugticket

Nachfolgend wird der Lebenszyklus eines Zugtickets (stark vereinfacht) beschrieben:
Der Prozess beginnt damit, dass ein Fahrgast den Wunsch hat, ein Ticket zu kaufen. Daraufhin geht er zum Bahnhof, erstellt und bezahlt beim Automaten das Ticket (beides in einem Schritt) und druckt dieses aus. Ist der Abfahrtstermin erreicht, so steigt er/sie in den Zug ein und fährt mit diesem. Ist die Fahrt beendet, steigt er/sie aus und wirft anschließend das Ticket weg. Irgendwann während der Zugfahrt kommt die Schaffnerin, der Gast zeigt der Schaffnerin das Ticket vor und diese entwertet es. Falls die Abfahrtszeit erreicht ist, ohne dass das Ticket gedruckt (bzw. fertig gedruckt) wurde, so speichert das Fahrkartensystem das Ticket als ungültig ab.

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



Modellieren Sie den Prozess „Zug fahren“, mittels UML2-Aktivitätsdiagramm. Modellieren Sie mittels Objektfluss die durch die Aktionen/Aktivitäten bedingten Änderungen am Objekt „Zugticket“. (Andere Objektflüsse sind für diese Aufgabe nicht relevant!).
Illustrieren Sie die involvierten Rollen mit Hilfe von Swimlanes (Partitionen), Treffen Sie Annahmen wo nötig.


Schaffnerin
Fahrgast
Fahrkarten-System
Wunsch Ticketkauf
Zum Bahnhof gehen
Ticket erstellen
Ticket drucken
Abfahrtszeit erreicht
Abfahrtstermin erreicht
:Ticket [erstellt]
:Ticket [gedruckt]
TicketNr als ungültig speichern
:Ticket [ungültig]
Zug fahren
In Zug einsteigen
Schaffnerin kommt
Zug fahren
aussteigen
Ticket entwerten
Ticket vorzeigen
:Ticket [entwertet]
Ticket wegwerfen

VU Objektorientierte Modellierung		Übung 6: Anwendungsfalldiagramm
188.391, SS2018	MUSTERLÖSUNG	Übungsgruppen: 11.06.2018 - 14.06.2018

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?

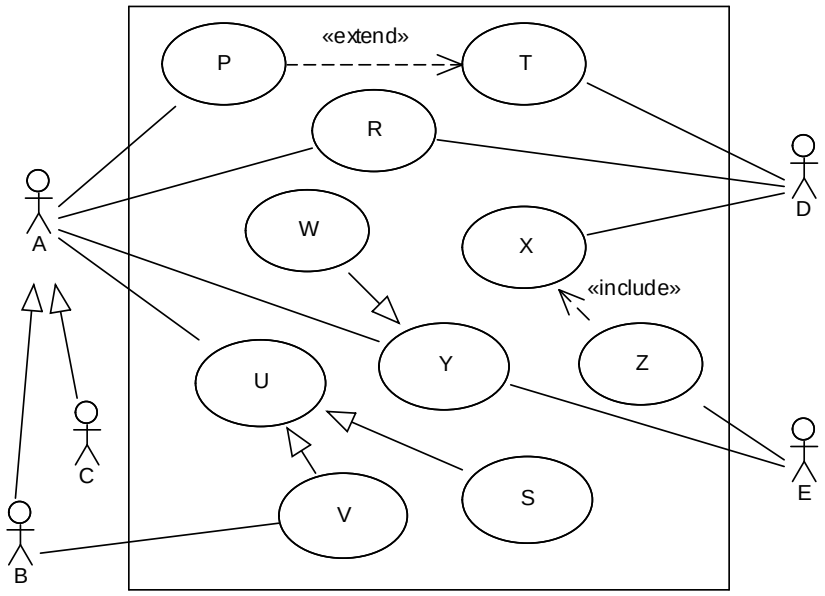
b) Auf welche Bestandteile des Anwendungsfalldiagramms kann das Konzept der Generalisierung angewendet werden? Geben Sie entsprechende Beispiele.

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:

a) Welche Akteure sind jeweils an den einzelnen Use Cases beteiligt?
$P : A \vee B \vee C$
$T : D$
$R : (A \vee B \vee C) \wedge D$
$W : (A \vee B \vee C) \wedge E$
$X : D$
$Y : (A \vee B \vee C) \wedge E$
$Z : E$
$U : (A \vee B \vee C)$
$V : (A \vee B \vee C) \wedge B$
$S : (A \vee B \vee C)$
Anmerkung: das „oder“ ist immer als „exklusives oder“ zu verstehen

b) Muss P ausgeführt werden, wenn auch T ausgeführt wird? Muss X ausgeführt werden, wenn Z ausgeführt wird?
Nein; Ja.

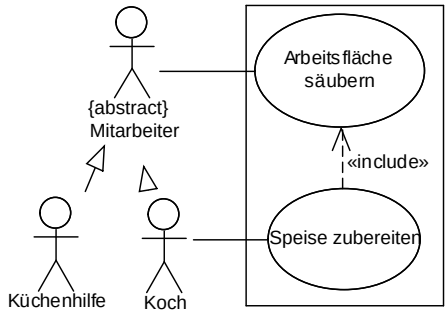
c) Ist P oder T der Basis Use Case? Ist Z oder X der Basis Use Case?
T; Z

d) Kann P auch R erweitern?
Nein. Dies wäre nur möglich wenn R von T erbt (oder es eine explizite extends-Beziehung zu R gibt).

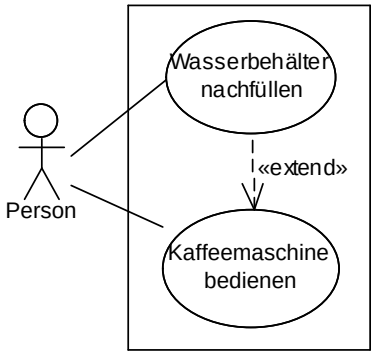
Aufgabe 4: Modellierung verschiedener Sachverhalte

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

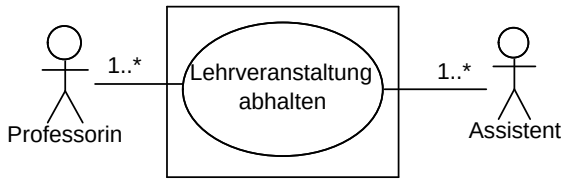
a) Ein Koch bereitet eine Speise zu. Im Zuge dessen ist es notwendig, dass der Koch oder die Küchenhilfe die Arbeitsfläche säubert.



b) Eine Person bedient die Kaffeemaschine um eine Tasse Kaffee zu erhalten. Im Zuge dessen kann es sein, dass die Person den Wasserbehälter nachfüllen muss.



c) Um eine Lehrveranstaltung zu halten sind immer mindestens eine Professorin und mindestens ein Assistent notwendig.



Aufgabe 5: Anwendungsfalldiagramm

Erstellen Sie ein Anwendungsfalldiagramm, welches alle Anwendungsfälle von Microsoft Excel bzw. OpenOffice Calc enthält, die in die Kategorie „Daten“ fallen, also zum Beispiel sortieren, gruppieren etc.
Anmerkung: Das Ziel ist, dass das Anwendungsfalldiagramm einen guten (und korrekten) Überblick über die verfügbaren Funktionen liefert, NICHT dass jeder noch so klitzekleine mögliche Anwendungsfall enthalten ist.

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 bisher 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.