Modellierung und Programmierung 1

Übung 2

Stefan Preußner

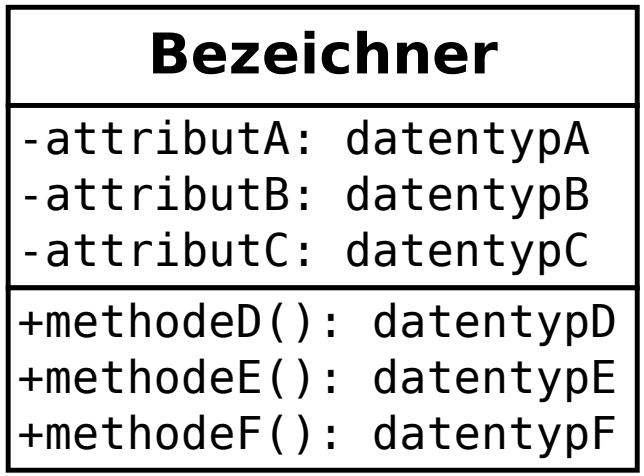
9. / 10. November 2020

Download der Übungsunterlagen

- Im Rahmen dieser Übung gibt es Folien und Codebeispiele
- Homepage des Lehrstuhls für Schwarmintelligenz und Komplexe Systeme:
  `http://pacosy.informatik.uni-leipzig.de`
- Homepage → Mitarbeiter → Stefan Preußner → MuP1
- Benutzer: `mup`
- Passwort: `MuP1WS20/21`

Objekte und Klassen

- Annahme: alle materiellen und immateriellen Dinge sind Objekte
- **Ein Objekt ist eine Instanz (Ausprägung) einer Klasse**
- Klassen haben einen Bezeichner, ggf. mehrere Eigenschaften/Attribute und ggf. mehrere Funktionalitäten/Methoden



Datentypen

Jedes Attribut hat einen bestimmten Datentyp, beispielsweise:

- `boolean`: Boolesche Variablen können zwei mögliche Werte annehmen - wahr oder falsch
- `int` und `long`: Ganzzahlen (*integer*)
- `float` und `double`: Gleitkommazahlen (*floating-point number*)
- `char`: Buchstaben. Ein `char` wird von **einfachen** Anführungszeichen eingeschlossen ('A')
- `String`: Zeichenketten. Ein `String` wird von **doppelten** Anführungszeichen eingeschlossen ("A")

Datentypen

Weitere Datentypen:

- `Klassenname`: Klassen (und Schnittstellen) sind ebenfalls Datentypen. Ist der Wert eines Attributs die Instanz einer Klasse, dann ist die Klasse selbst der Datentyp dieses Attributs.
  - Beispiel: die Klasse Vorlesung besitzt das Attribut `vortragender` mit dem Datentyp `Professor`
    $\rightarrow$ `vortragender: Professor`
- `void`: kennzeichnet eine Methode, die nichts zurückgibt, und steht **anstelle** eines Datentyps

Datentypen

Weitere Datentypen:

- String[]: Ein Array vom Typ String. Arrays sind Listen mit einer festen Größe.
- Collection<String>: Eine `Collection` vom Typ String. In Java gibt es zahlreiche Datentypen, die Listen mit veränderlicher Größe, Mengen und/oder Queues darstellen. Da bei der Modellierung die konkrete Implementierung einer Sammlung von Elementen oftmals unwichtig ist, kann im UML-Klassendiagramm der viele Container umfassende Datentyp `Collection` verwendet werden.

Funktionen

Beispiele:

```
public String getName()
private void erstelleAdresse( String strasse, int hausnummer, int
postleitzahl, String ort)
boolean halteVorlesung( String titel, String inhalt, int dauer )
```

Sichtbarkeit: private, protected, *leer*, public
Rückgabe: ein Datentyp oder void ($\rightarrow$ kein Rückgabewert)
Funktionsname
Datentyp: boolean, int, float, String, Collection<String>...
Parameter

Funktionen in UML

```
public String getName()
private void erstelleAdresse( String strasse, int hausnummer, int
postleitzahl, String ort)
boolean halteVorlesung( String titel, String inhalt, int dauer )
```

↓

Professor
+getName(): String -erstelleAdresse(strasse:String,hausnummer:int,postleitzahl:int,ort:String): void ~halteVorlesung(titel:String,inhalt:String,dauer:int): boolean

Funktionen in UML

Professor
+getName(): String -erstelleAdresse(strasse:String,hausnummer:int,postleitzahl:int,ort:String): void ~halteVorlesung(titel:String,inhalt:String,dauer:int): boolean

Sichtbarkeit: private, protected, *leer*, public
Rückgabe: ein Datentyp oder void ($\rightarrow$ kein Rückgabewert)
Funktionsname
Datentyp: boolean, int, float, String, Collection<Integer>...
Parameter

Eine Beispielklasse

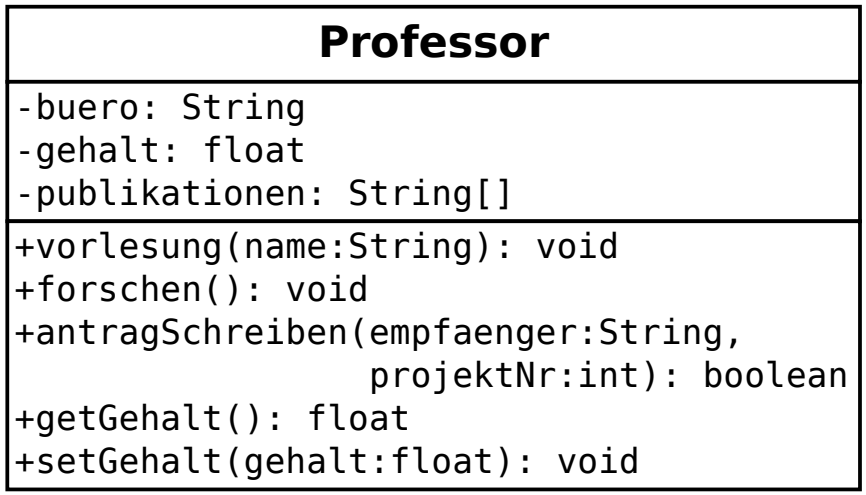
An der Uni Leipzig arbeiten zahlreiche Professoren, welche hier modelliert werden sollen. Ein Professor besitzt ein Büro (bspw. im Raum P3-456) und bekommt ein jährliches Gehalt (bspw. 87654,32 €) gezahlt. Ein Professor kann eine Vorlesung (bspw. "MuP1") halten, forschen oder Anträge schreiben. Bei einem Antrag müssen der Empfänger (bspw. "DFG") und die Projektnummer (bspw. 715299) angegeben werden, das Ergebnis der Antragsstellung kann positiv oder negativ sein. Viele Professoren haben eine lange Liste von Publikationen (bspw. "Java ist auch eine Insel" und "Java in 21 Wochen") vorzuweisen.
Für das Gehalt sollen ein Getter und ein Setter erstellt werden. Auf weitere Getter und Setter sowie Konstruktoren kann verzichtet werden.

Die Beispielklasse in UML

Ein Professor besitzt ein Büro (bspw. im Raum P3-456) und bekommt ein jährliches Gehalt (bspw. 87654,32 €) gezahlt. Ein Professor kann eine Vorlesung (bspw. "MuP1") halten, forschen oder Anträge schreiben. Bei einem Antrag müssen der Empfänger (bspw. "DFG") und die Projektnummer (bspw. 715299) angegeben werden, das Ergebnis der Antragsstellung kann positiv oder negativ sein. Viele Professoren haben eine lange Liste von Publikationen (bspw. "Java ist auch eine Insel" und "Java in 21 Wochen") vorzuweisen.
Für das Gehalt sollen ein Getter und ein Setter erstellt werden. Auf weitere Getter und Setter sowie Konstruktoren kann verzichtet werden.

Die Beispielklasse in UML

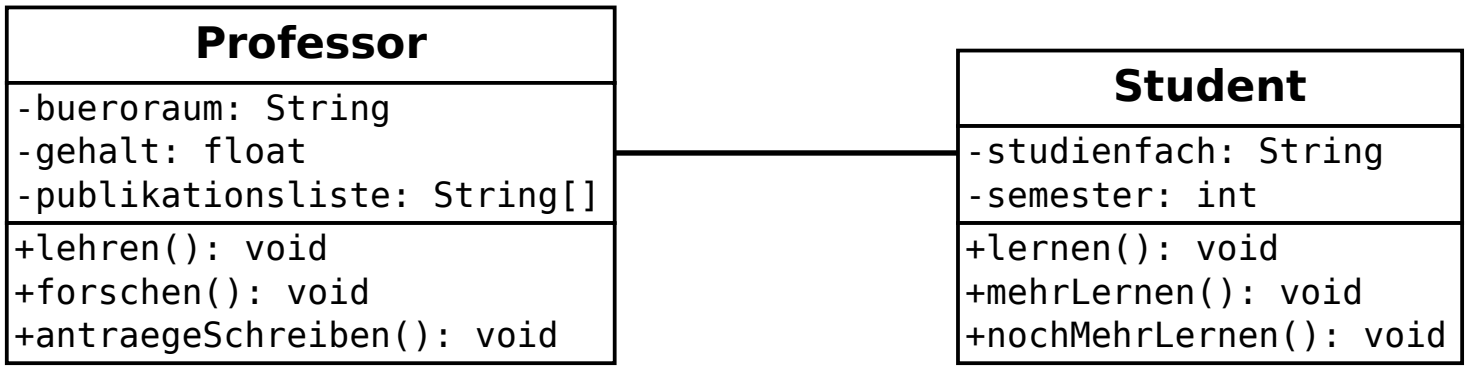
Ein Professor besitzt ein Büro (bspw. im Raum P3-456) und bekommt ein jährliches Gehalt (bspw. 87654,32 €) gezahlt. Ein Professor kann eine Vorlesung (bspw. "MuP1") halten, forschen oder Anträge schreiben. Bei einem Antrag müssen der Empfänger (bspw. "DFG") und die Projektnummer (bspw. 715299) angegeben werden, das Ergebnis der Antragsstellung kann positiv oder negativ sein. Viele Professoren haben eine lange Liste von Publikationen (bspw. "Java ist auch eine Insel" und "Java in 21 Wochen") vorzuweisen.
Für das Gehalt sollen ein Getter und ein Setter erstellt werden. Auf weitere Getter und Setter sowie Konstruktoren kann verzichtet werden.



Klassenbeziehungen

- Klassen können fünf unterschiedliche Arten von Beziehungen zueinander haben:
  - Assoziation
  - Aggregation
  - Komposition
  - Vererbung
  - Realisierung / Interface

Assoziation

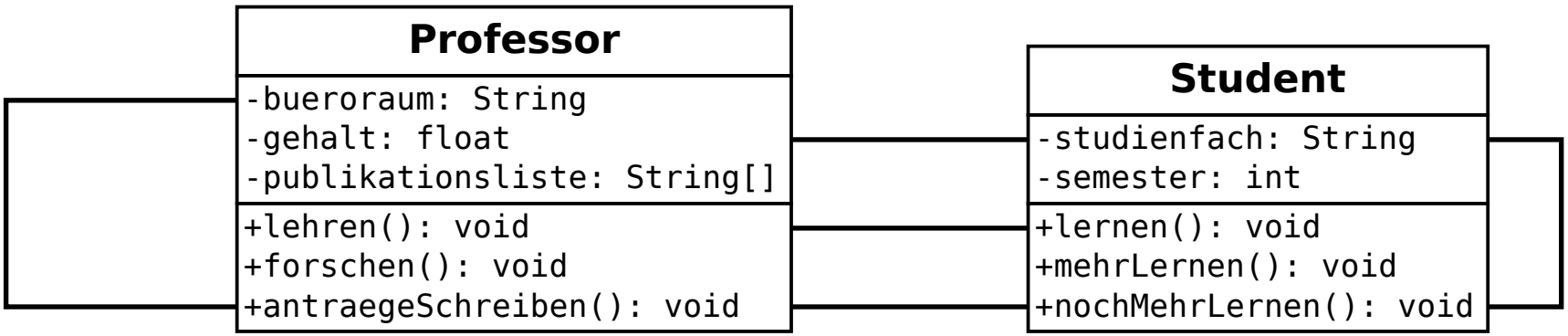


- „X steht in Beziehung zu Y“
- „X kommuniziert mit Y“
- „X nutzt Y“

Assoziation

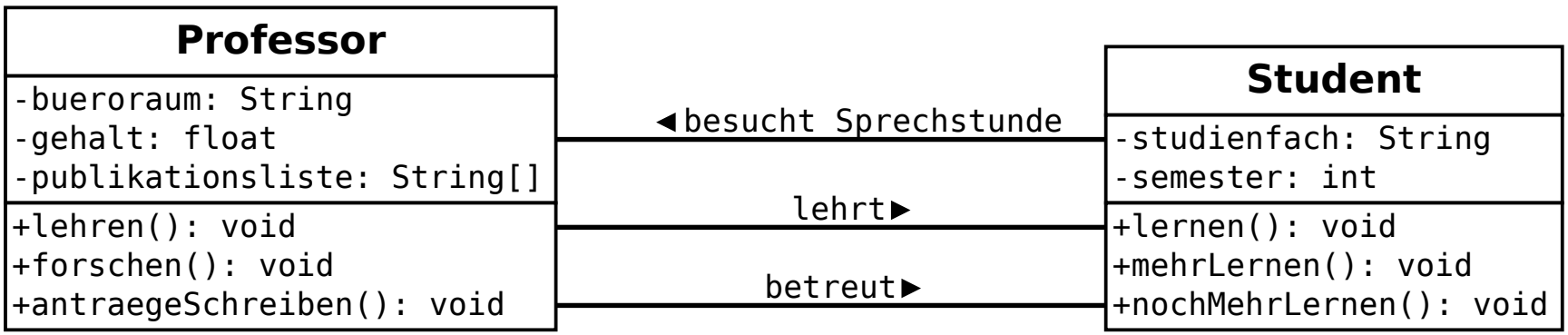
- Selbstassoziativität ist möglich
  - Beziehungen von Professoren untereinander
  - Beziehungen von Studenten untereinander

Assoziation



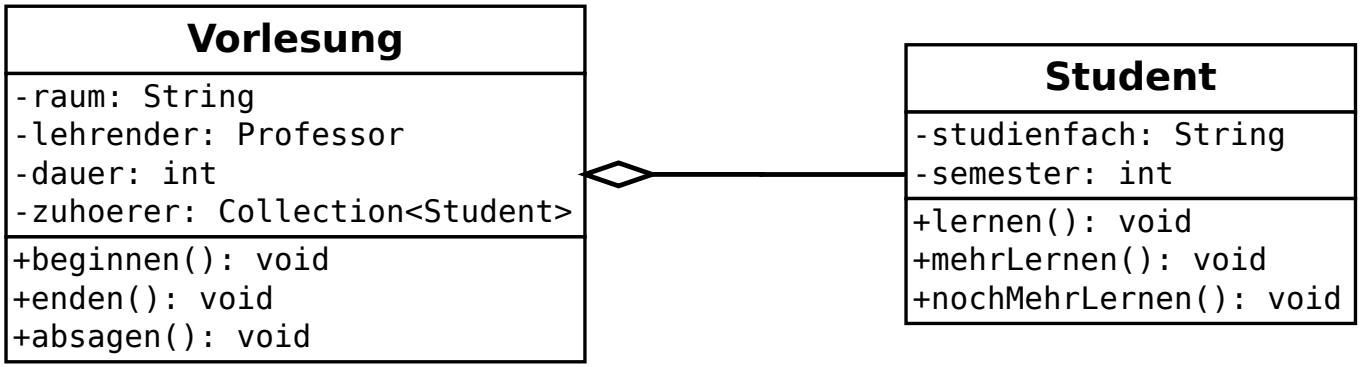
- Mehrere Beziehungen sind möglich
  - Professoren halten vor Studenten Vorlesungen, betreuen Arbeiten, nehmen Prüfungen ab und beraten in Sprechstunden

Bezeichnete Beziehungen



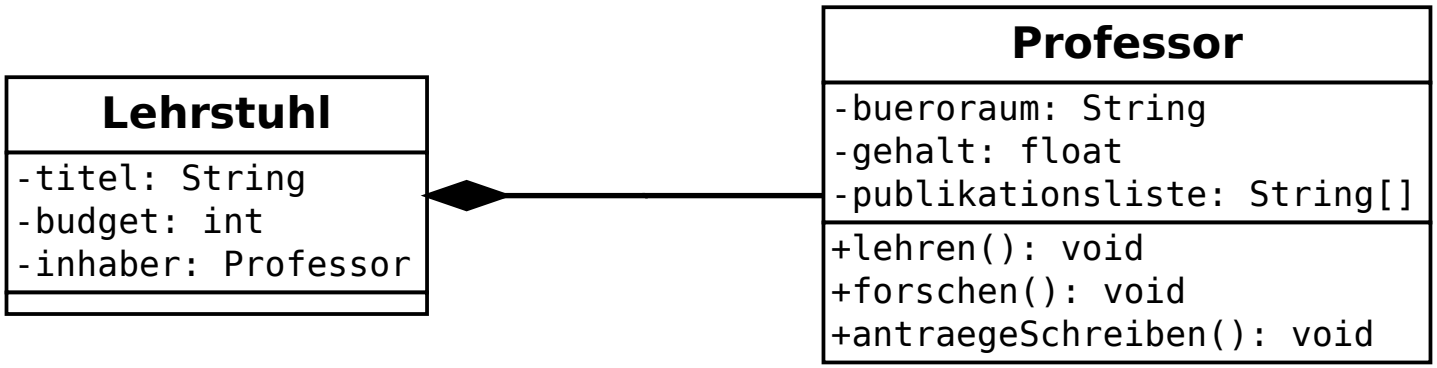
- Beziehungen können bezeichnet werden
- Durch einen Pfeil kann die Richtung der Beziehung angegeben werden

Aggregation



- „besitzt ein(e)“
- „ist ein Teil von“
- Beide Objekte können unabhängig voneinander existieren
- Eine Aggregation impliziert fast immer ein entsprechendes Attribut bei der besitzenden Klasse!

Komposition

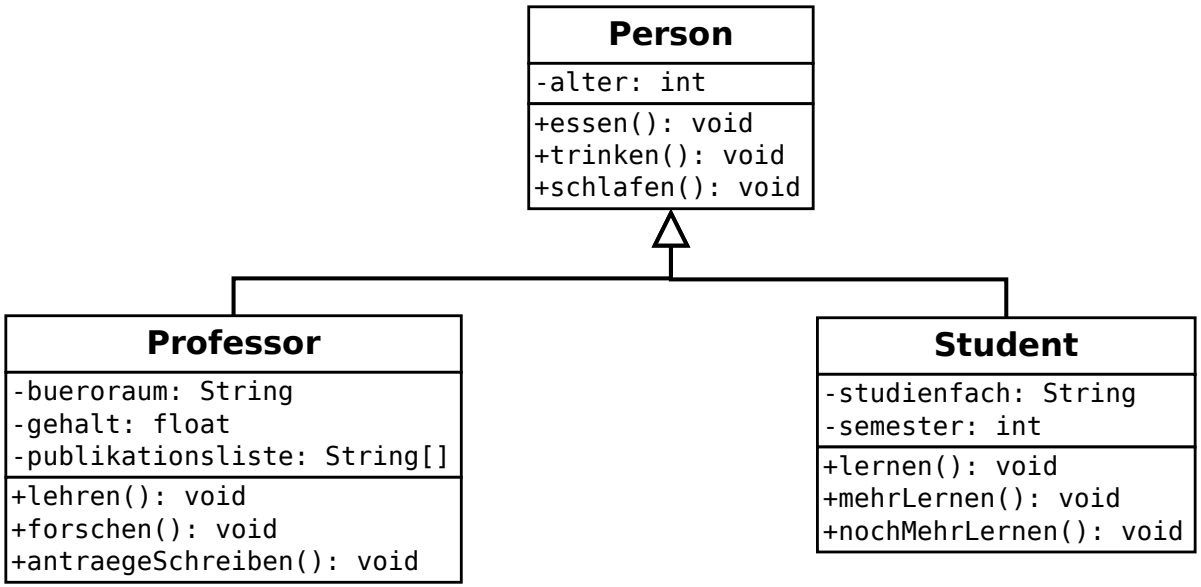


- „besteht aus“
- Die Lebzeit des einen Objekts ist an die Lebzeit des anderen Objekts gekoppelt

Assoziation vs. Aggregation vs. Komposition

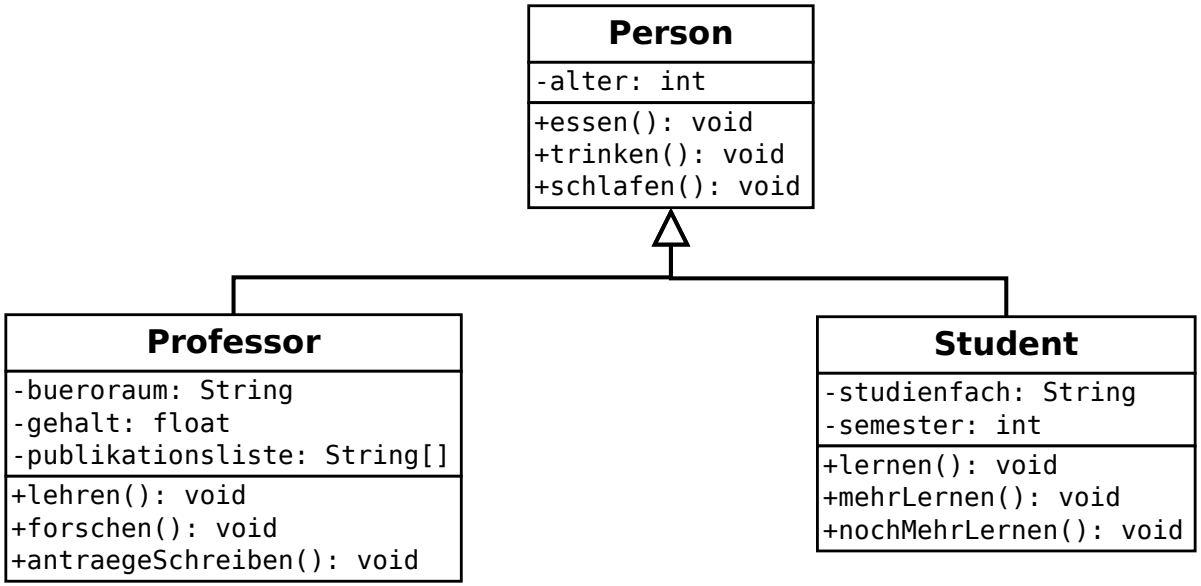
- Die Aggregation ist ein Spezialfall der Assoziation
  - $\rightarrow$ Eine Aggregation kann immer durch eine Assoziation ersetzt werden
- Die Komposition ist ein Spezialfall der Aggregation
  - $\rightarrow$ Eine Komposition kann immer durch eine Aggregation und damit auch durch eine Assoziation ersetzt werden
- Die Verwendung von Kompoisitonen und Aggregationen liegt im Ermessen des Modellierers, im UML-Standard gibt es hierzu kaum Vorgaben
  - Aber: eine Klasse kann nicht über jeweils eine Komposition Teil zwei anderer Klassen sein

Vererbung



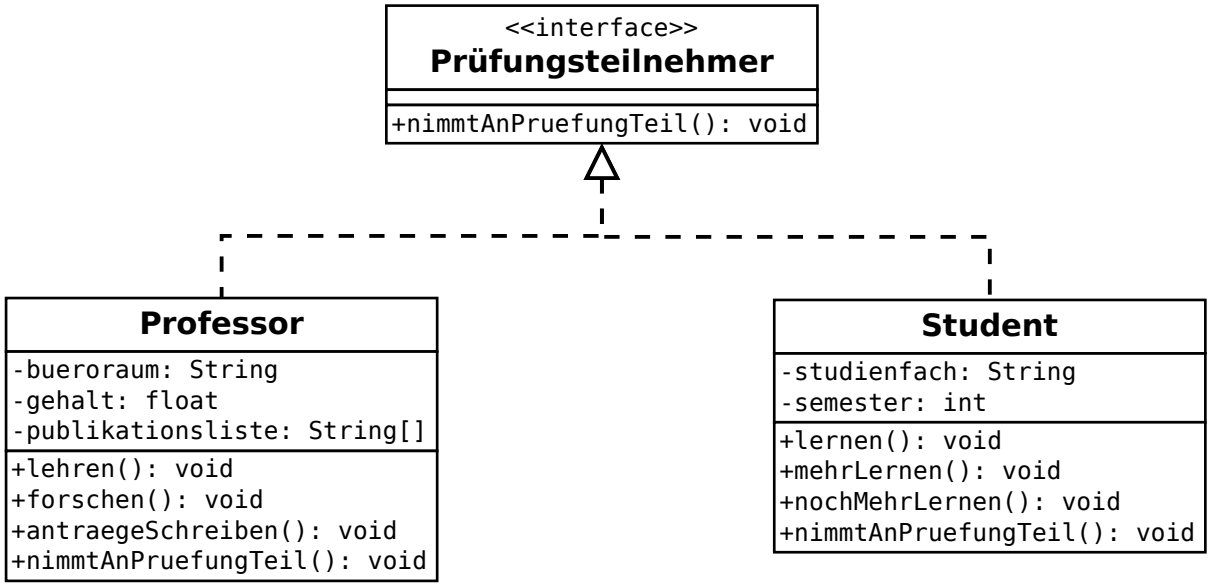
- „ist ein(e)“
- **Person** ist hier die Generalisierung der Klassen **Professor** und **Student**
- **Professor** und **Student** sind Spezialisierungen von **Person**

Vererbung



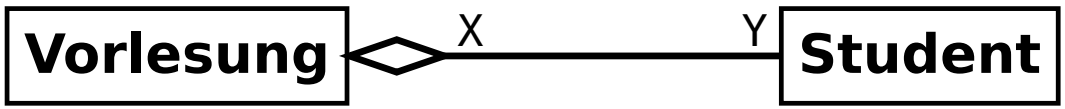
- **Professor** und **Student** erben alle Attribute und Methoden von **Person**, d.h. ein `Professor` besitzt ebenfalls ein Attribut `alter`, auch wenn dieses nicht explizit angegeben wird

Realisierung / Interfaces



- Vom **Prüfungsteilnehmer** selbst kann keine Instanz erzeugt werden
- Die Funktion nimmtAnpruefungTeil() ist bei **Prüfungsteilnehmer** nicht implementiert

Multiplizitäten



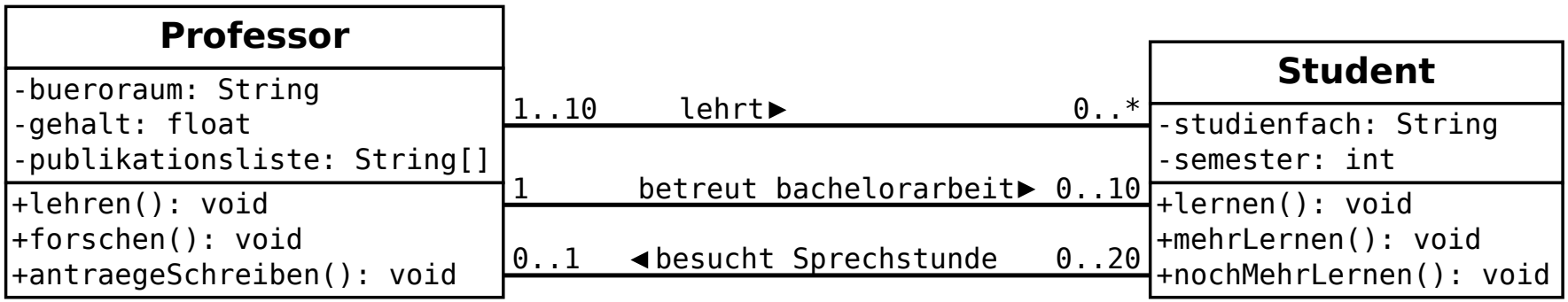
- „Eine Vorlesung besteht aus Y Studenten“
- „Ein Student ist Teil von X Vorlesungen“

Multiplizitäten

Wert	Bedeutung
	genau 1 oder beliebig viele [1]
m	genau m
m..n	mindestens m, höchstens n
*	beliebig viele
m..*	mindestens m
0..*	beliebig viele
m,n	m oder n

[1] Im Rahmen von MuP wird angenommen, dass eine nicht angegebene Multiplizität gleichbedeutend mit "beliebig viele" ist.

Multiplizitäten



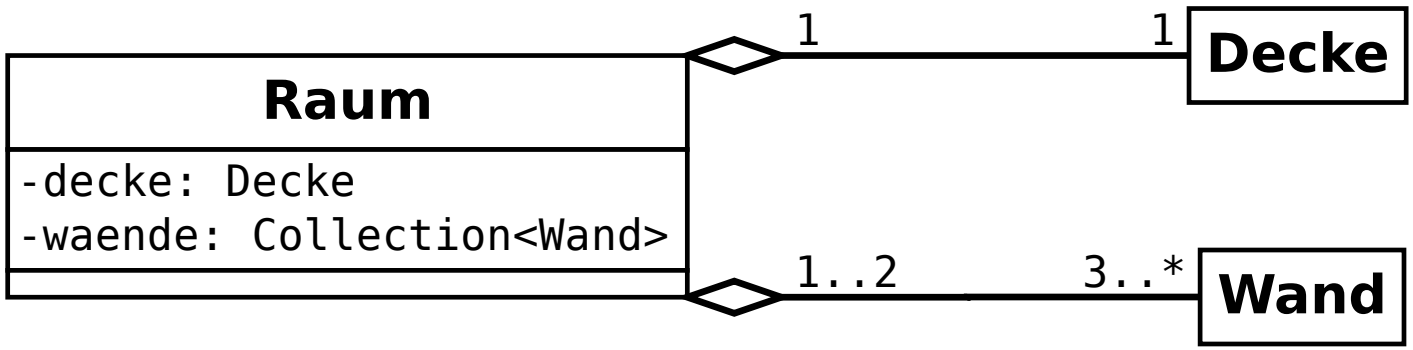
(Die Zahlen sind rein willkürlich gewählt und sollen nur verdeutlichen, dass die Benennung von Assoziationen besonders bei unterschiedlichen Multiplizitäten hilfreich oder notwendig sein kann.)

Beispiele

Ein **Raum** besteht aus mindestens drei **Wänden** und einer **Decke**. Eine Wand gehört zu einem bis zwei Räumen, eine Decke gehört zu genau einem Raum.

Beispiele

Ein **Raum** besteht aus mindestens drei **Wänden** und einer **Decke**. Eine Wand gehört zu einem bis zwei Räumen, eine Decke gehört zu genau einem Raum.

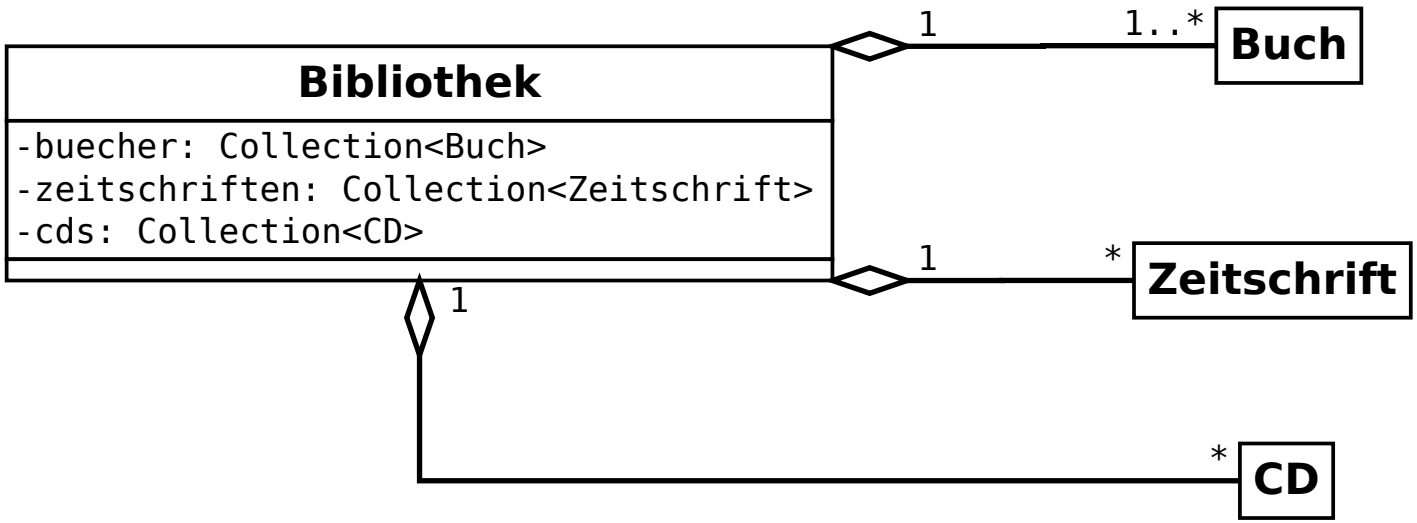


Beispiele

Eine **Bibliothek** besteht aus mindestens einem **Buch**. Eine Bibliothek kann auch **Zeitschriften** und **CDs** umfassen. Jedes Buch, jede Zeitschrift und jede CD gehört in genau eine Bibliothek.

Beispiele

Eine **Bibliothek** besteht aus mindestens einem **Buch**. Eine Bibliothek kann auch **Zeitschriften** und **CDs** umfassen. Jedes Buch, jede Zeitschrift und jede CD gehört in genau eine Bibliothek.



Beispiele

Honorarprofessoren sind sehr spezielle Professoren. Wie alle anderen **Professoren** und **Studenten** sind sie **Personen**.

Beispiele

Honorarprofessoren sind sehr spezielle Professoren. Wie alle anderen **Professoren** und **Studenten** sind sie **Personen**.