Prüfungsteilnehmer	Prüfungstermin	Einzelprüfungsnummer

Kennzahl: _____

Kennwort: _____

Arbeitsplatz-Nr.: _____

Frühjahr 2012

66116

Erste Staatsprüfung für ein Lehramt an öffentlichen Schulen

— Prüfungsaufgaben —

Fach: **Informatik (vertieft studiert)**

Einzelprüfung: **Datenbanksysteme, Softwaretechnologie**

Anzahl der gestellten Themen (Aufgaben): **2**

Anzahl der Druckseiten dieser Vorlage: **13**

Bitte wenden!

Thema Nr. 1

Teilaufgabe 1:

1. Entity-Relationship-Modell

Ein Handelsunternehmen möchte seine Struktur verbessern und ein Datenbanksystem zur Verwaltung seiner Filialen, angebotenen Waren und Kunden erstellen.
Die Basis dieses Systems bilden die Filialen des Unternehmens. Jede Filiale ist eindeutig durch ihre Filialnummer gekennzeichnet und befindet sich in einer Stadt. Außerdem hat jede Filiale einen Filialleiter.
Zu jeder Filiale gehört genau ein Lager mit einer eindeutigen Lagernummer und ebenfalls einem Leiter. Jedes Lager verfügt über eine bestimmte Menge an verschiedenen Waren. Jede Ware kann in mehreren Lagern vorrätig sein und ist über eine Nummer, einen Namen und einen Preis gekennzeichnet.
Ein Kunde kann in einer Filiale des Unternehmens Bestellungen aufgeben. Der Kunde hat eine Kundennummer, einen Namen und eine Adresse. Eine Bestellung enthält dabei jeweils einen Warenartikel, dessen gewünschte Menge und das Datum, an dem die Bestellung abgeholt wird.

a) Erstellen Sie ein Entity-Relationship-Diagramm für obige Datenbank.

b) Setzen Sie das in Teilaufgabe a) erstellte Entity-Relationship-Diagramm in ein Relationenschema um. Relationships sollen mit einer möglichst geringen Anzahl von Relationen realisiert werden. Dabei sind unnötige Redundanzen zu vermeiden. Ein Relationenschema ist in folgender Form anzugeben: `Relation (Attribut1, Attribut2, ...)`. Schlüsselattribute sind dabei zu unterstreichen. Achten Sie bei der Wahl des Schlüssels auf Eindeutigkeit und Minimalität.

2. Normalisierung

Gegeben sei folgende Datenbank für Wareneingänge eines Warenlagers. Die Primärschlüssel-Attribute sind unterstrichen.

ZulieferungsNr	ArtikelNr	Datum	Artikelname	Menge
1	1	01.01.2009	Handschuhe	5
1	2	01.01.2009	Mütze	10
2	3	05.01.2009	Schal	2
2	1	05.01.2009	Handschuhe	8
3	4	06.01.2009	Jacke	2

a) Erläutern Sie, inwiefern obiges Schema die 3. Normalform verletzt.

b) Geben Sie für obige Datenbank alle vollen funktionalen Abhängigkeiten (einschließlich der transitiven) an.

c) Überführen Sie das obige Relationenschema in die 3. Normalform. Erläutern Sie die dazu durchzuführenden Schritte jeweils kurz.

Fortsetzung nächste Seite!

3. SQL

Gegeben sei das folgende Relationenschema:

```
Fahrzeug(MNR:int(3), FZGNR:char(12), Baujahr:int(4),
         KMStand:int(5), Preis:int(5))

Modell(MNR:int(3), HNR:int(3), Typ:char(20), Neupreis:int(5),
       ps:int(3))

Hersteller(HNR:int(3), Name:char(20))
```

Dabei sind die Schlüsselattribute jeweils unterstrichen und zusätzlich für alle Attribute die Typen angegeben. Formulieren Sie die folgenden Anfragen bzw. Anweisungen in SQL.

a) Geben Sie die Anweisungen in SQL-DDL an, die notwendig sind, um die Relationen „Fahrzeug", „Modell" und „Hersteller" zu erzeugen. Achten Sie dabei darauf, die Primärschlüssel der Relationen zu kennzeichnen.

b) Bestimmen Sie die Typen aller Modelle des Herstellers mit Namen BMW.

c) Bestimmen Sie den Mindestpreis, bezogen auf das Attribut „Preis", der Fahrzeuge eines jeden Herstellers.

d) Bestimmen Sie die Namen der Hersteller, für die von jedem ihrer Modelle mindestens ein Fahrzeug in der Datenbank gespeichert ist.

e) Bestimmen Sie die Namen aller Hersteller, von denen mindestens fünf Fahrzeuge eines beliebigen Modells in der Datenbank gespeichert sind.

Fortsetzung nächste Seite!

Teilaufgabe 2:

Aufgabe 1

Welche Tätigkeiten sind in der Anforderungsanalyse für ein zu entwickelndes Softwaresystem durchzuführen? Welche Probleme können typischerweise in der Anforderungsanalyse auftreten?

Aufgabe 2

Gegeben sei das Klassendiagramm in Abb. 1.

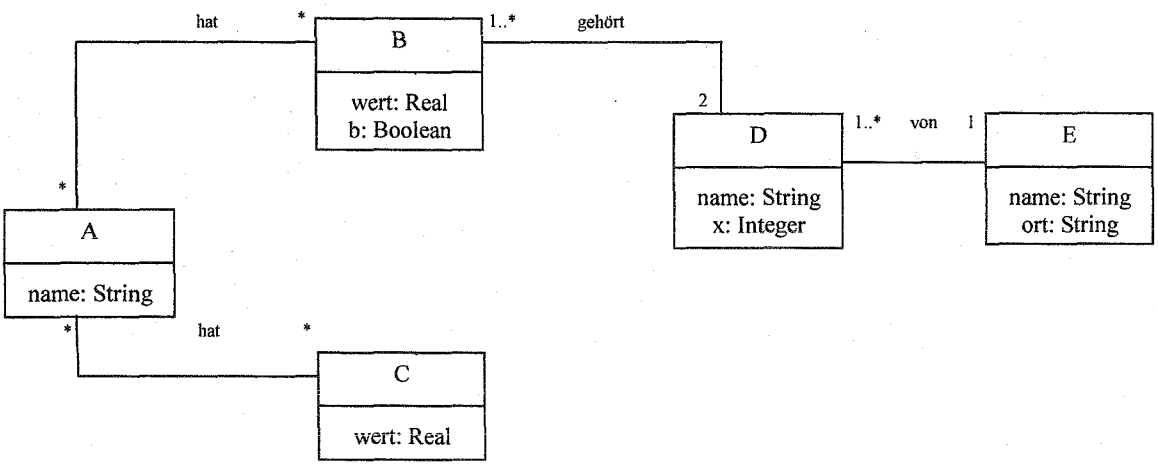


Abbildung 1: Klassendiagramm

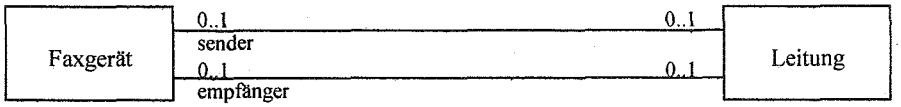
a) Konstruieren Sie ein (gemäß des Klassendiagramms zulässiges) Instanzendiagramm, das möglichst wenige Objekte aber mindestens ein Objekt der Klasse E enthält.

b) Gemeinsame Attribute und Assoziationen können in Oberklassen zusammengefasst werden. Konstruieren Sie auf diese Weise ein Klassendiagramm mit geeigneten Oberklassen und Vererbungsbeziehungen, das genau dieselben Instanzendiagramme zulässt wie das Klassendiagramm in Abb. 1. Falls sich dabei abstrakte Oberklassen ergeben, sind diese zu kennzeichnen.

Fortsetzung nächste Seite!

Aufgabe 3

Die (statistische) Struktur eines einfachen Faxübertragungssystems wird durch das Klassendiagramm in Abb. 2 beschrieben.



Das Sequenzdiagramm in Abb. 3 beschreibt ein Szenario, in dem ein Fax mit zwei Seiten erfolgreich übertragen wird.

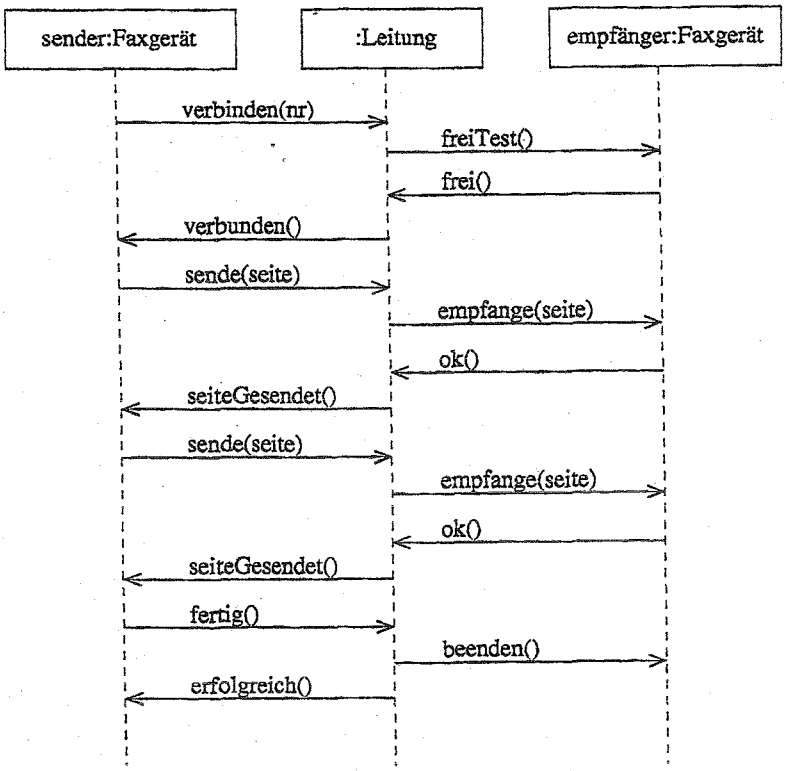


Abbildung 3: Sequenzdiagramm

a) Falls der Empfänger belegt ist oder falls eine Seite nicht richtig beim Empfänger angekommen ist, soll der Vorgang mit entsprechender Rückmeldung an den Sender abgebrochen werden. Erstellen Sie für jedes dieser beiden Szenarien ein Sequenzdiagramm.

b) Konstruieren Sie ein Zustandsdiagramm für die Klasse *Leitung*. Das Zustandsdiagramm soll das in den drei Szenarien (vgl. Abb. 3 und Teil a)) beschriebene Verhalten eines Leitungsobjekts integrieren, so dass beliebig viele Seiten übertragen werden können.

Fortsetzung nächste Seite!

Aufgabe 4

a) Was versteht man unter einem Entwurfsmuster bei der Entwicklung von Softwaresystemen? Erläutern Sie kurz die Vorteile von Entwurfsmustern.

Abb. 4 zeigt, in UML-Notation, das statische Modell des Observer-Patterns (aus dem Design-Pattern Katalog von Gamma et al.).

b) Erläutern Sie die Wirkungsweise des Observer-Patterns. Gehen Sie dabei auch auf die Aufgabe der einzelnen Operationen ein.

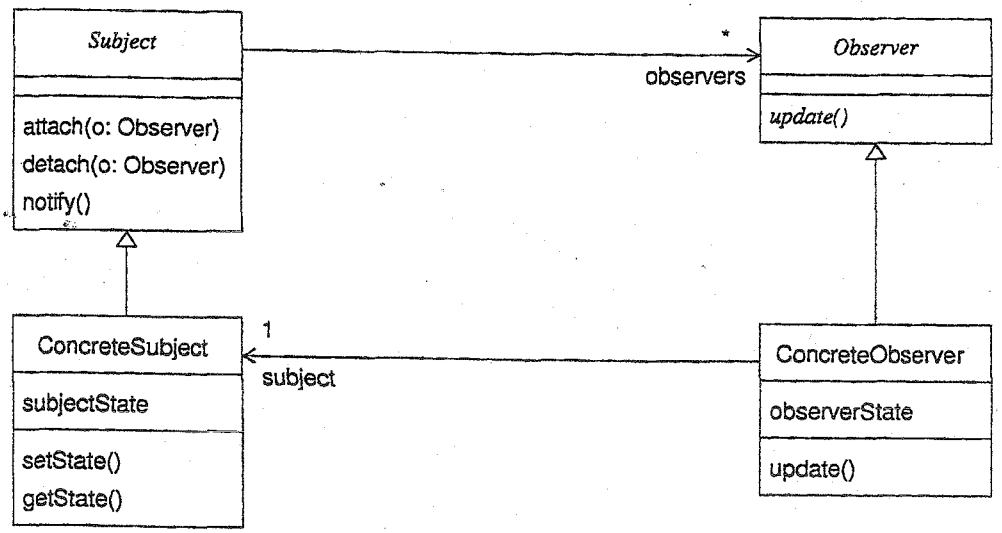


Abbildung 4: Observer-Pattern

c) Schildern Sie eine typische Anwendung des Observer-Patterns und erläutern Sie dessen Nutzen.

Thema Nr. 2

Teilaufgabe 1:

1. Datenbankanfragen in SQL

Wir betrachten die Datenbank FUSSBALL-EUROPAMEISTERSCHAFT mit den folgenden Tabellen:

- Spieler (Rückennummer, Land, Name, Vorname, Länderspiele, Verein, Land-V):
  Spielerinformation, einschließlich des Heimatlandes (Land), der Anzahl der Länderspiele, dem Verein und dem Land des Vereins (Land-V)
- Nationen(Land, Hauptstadt):
  Informationen zu den teilnehmenden Nationen
- nimmt_teil_an(Rückennummer, Land, Spiel_ID):
  Beteiligung eines Spielers an Spielen.
- Spiele(Spiel_ID, Land_Heim, Land_Gast, Datum, Stadion_ID, Tore_Heim, Tore_Gast):
  Informationen zu den Spielen, wobei Land_Heim und Land_Gast die beiden an einem Spiel beteiligten Nationen sind
- Stadion(Stadion_ID, Stadt, Land, Zuschauerkapazität):
  Informationen zu den Stadien

Erstellen Sie in SQL folgende Anfragen:

a) Bestimmen Sie alle teilnehmenden Spieler, die beim "FC Bayern München" (Verein) spielen.

b) Bestimmen Sie alle Spiele, die im Heimatland von einer der beiden beteiligten Nationen stattfinden.

c) An wie vielen Spielen nimmt "Bastian Schweinsteiger" teil?

d) Bestimmen Sie für jede Nation den Spieler mit den meisten Länderspielen. Dazu können Sie eine Unteranfrage oder eine Sicht (View) verwenden, die in der Hauptanfrage aufgerufen wird.

Fortsetzung nächste Seite!

2. Datenmodellierung und Datenbankaufbau

Wir betrachten die Datenbank FUSSBALL-EUROPAMEISTERSCHAFT aus Aufgabe 1.

a) Definieren Sie die Relationenschemata „Spieler" und „nimmt_teil_an" in SQL mittels CREATE TABLE und geben Sie dabei geeignete Schlüssel- und Fremdschlüsselbedingungen an. Nehmen Sie dabei an, dass die anderen Tabellen schon erzeugt sind.

b) Fügen Sie in SQL mittels INSERT in jede der beiden Relationen „Spieler" und „nimmt_teil_an" jeweils mindestens ein Tupel ein, so dass die natürlichen Schlüssel- und Fremdschlüsselbedingungen aus Teilaufgabe a) erfüllt sind. Eventuelle weitere Fremdschlüsselbedingungen auf andere Tabellen müssen nicht berücksichtigt werden.

c) Geben Sie ein ER-Diagramm mit Funktionalitätsbedingungen für das Datenbankschema an, in dem möglichst viele Relationen als Relationships modelliert sind.

3. Funktionale Abhängigkeiten

Gegeben sei das Relationenschema $R = (U, F)$ mit der Attributmenge

$$U = \{A,B,C,D,E,G\}$$

und der folgenden Menge F von funktionalen Abhängigkeiten:

$$F = \{A \rightarrow B,\ AB \rightarrow CD,\ CD \rightarrow E\}.$$

a) Bestimmen Sie die Attributhüllen $\{A\}_F^+$ und $\{C,D\}_F^+$.

b) Erklären Sie, warum die funktionale Abhängigkeit $CD \rightarrow E$ zu Update-Anomalien (d. h. beim Einfügen, Löschen und Ändern) führt.

c) Bestimmen Sie alle Schlüssel von R.

d) Bestimmen Sie eine Basis und eine 3NF-Zerlegung von R.

e) Zerlegen Sie R bezüglich der BCNF-verletzenden funktionalen Abhängigkeit $CD \rightarrow E$.

Fortsetzung nächste Seite!

Teilaufgabe 2:

1. Testen

Gegeben sei folgendes Modul, das einen ganzzahligen Wert *element* in die sortierte, verkettete Liste *list* einfügt. Gleiche Werte werden nur einmal aufgenommen. Der Wert *element* ist bereits im Knoten *newNode* gespeichert.

```
if (list.first == null) {
  list.first = newNode;
} else if (element < list.first.element) {
  newNode.next = list.first;
  list.first = newNode;
} else {
  actualNode = list.first;
  while (actualNode != null) {
    if (actualNode.element < element
        && actualNode.next == null) {
      actualNode.next = newNode;
    } else if (actualNode.element < element
              && element < actualNode.next.element) {
      newNode.next = actualNode.next;
      actualNode.next = newNode;
    }
    actualNode = actualNode.next;
  }
}
```

a) Bestimmen Sie anhand des Kontrollflussgraphen des obigen Code-Fragments die maximale Anzahl linear unabhängiger Programmpfade, also die zyklomatische Komplexität nach McCabe.

b) Kann für dieses Modul eine 100%-ige Pfadüberdeckung erzielt werden? (Grenzen aufgrund physikalischer Gegebenheiten sind dabei nicht zu berücksichtigen). Begründen Sie Ihre Antwort.

c) Geben Sie einen möglichst kleinen Testdatensatz an, der eine 100%-ige Verzweigungsüberdeckung dieses Moduls erzielt.

d) Beschreiben Sie kurz, welche Eigenschaften eine Testfallmenge allgemein haben muss, damit sie das datenflussorientierte Überdeckungskriterium „all-uses“ erfüllt.

e) Welche Anforderungen stellt das kontrollflussorientierte Testkriterium „boundary-interior“ („Schleife-Inneres-Überdeckung“) an den Tester?

f) Skizzieren Sie die drei klassischen Strategien beim Software-Integrationstest: „bottom-up“, „top-down“ und „sandwich“.

Fortsetzung nächste Seite!

2. Prozessmodelle

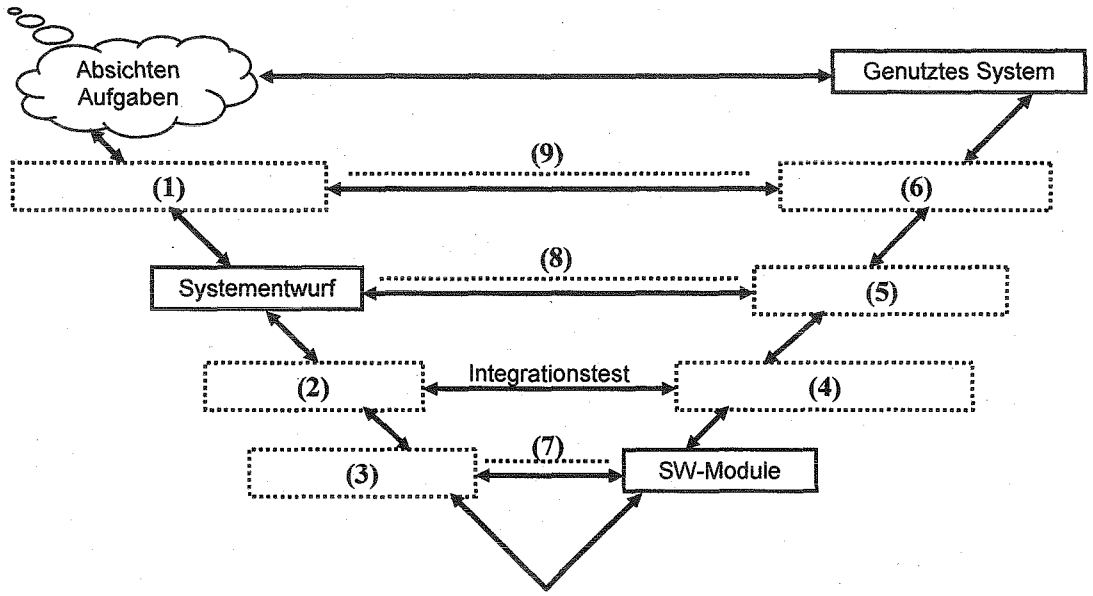
a) Um welche Aspekte erweitert das V-Modell 97 das „ältere“ Wasserfallmodell?

b) Welche grundlegende Phasenart unterscheidet das Spiralmodell von anderen vergleichbaren Modellen wie V-Modell oder Wasserfallmodell?

c) Beschreiben Sie kurz die folgenden vier Kategorien von Anforderungen und geben Sie jeweils ein entsprechendes kurzes Beispiel an: „funktional“, „qualitativ“, „systembezogen“, „prozessbezogen“.

d) Erläutern Sie knapp mit je einem Satz die folgenden vier klassischen Wartungsanlässe und skizzieren Sie dabei jeweils den Unterschied zu den anderen: „corrective“, „adaptive“, „perfective“, „preventive“.

e) Nennen Sie die fehlenden Angaben (Zwischenprodukte und Testarten) - jeweils mit ihrer Nummer - im folgenden, am V-Modell 97 angelehnten Prozessmodell:



Fortsetzung nächste Seite!

3. Formale Verifikation

Gegeben sei das folgende Programmfragment „**MinMax**" mit zwei ganzzahligen Parametern **x** und **y**:

```
int h;
if (x > y) {
    h = x;
    x = y;
    y = h;
}
```

Unter Verwendung des Hoare-Kalküls weisen Sie folgende Aussage nach:

$\{x, y \in N\}$ `MinMax` $\{x \leq y\}$

Hinweise:

Zuweisungsaxiom: {q | l ersetzt durch r} [l := r] {q}

Bedingungsaxiom: $\{p \wedge B\}\ S1\ \{q\};\ \{p \wedge \neg B\}\ S2\ \{q\} \Rightarrow \{p\}$ if B then S1 else S2 {q}

4. UML - OOA

Gegeben sei folgende natürlich-sprachliche Spezifikation eines PKI-Systems (Public-Key-Infrastruktur):

Damit Lehrer Anton seiner Kollegin Berta die Aufgaben der Klausur verschlüsselt per eMail übermitteln kann, bedarf es einer entsprechenden Infrastruktur.
Nachdem beide Lehrer die notwendige Software installiert haben, erstellt Berta zunächst ein sogenanntes Schlüsselpaar, bestehend aus einem „öffentlichen" (ÖS) und einem zugehörigen „privaten" Schlüssel (PS). Anschließend veröffentlicht Berta ihren ÖS durch Hochladen auf einen sogenannten Schlüsselverzeichnisdienst (Keyserver). Damit steht er Anton zur Verfügung, so dass dieser den ÖS jederzeit vom Keyserver herunterladen kann.
Anton kann nun seine Nachricht (z. B. das Aufgabenblatt) mit dem ÖS von Berta verschlüsseln und mit einer eMail versenden. Empfängt Berta eine solche eMail, so kann sie (und nur sie) mit ihrem PS die Nachricht wieder entschlüsseln.
Umgekehrt kann Berta mit ihrem PS beliebige Informationen (z. B. die Notenliste) „digital signieren". Diese unterschriebenen Daten übermittelt Berta dann zusammen mit der digitalen Signatur per eMail an Anton. Anton kann mit dem ÖS von Berta prüfen, ob die übermittelte Nachricht unverändert und tatsächlich von Berta stammt.

Fortsetzung nächste Seite!

a) Modellieren Sie die in der Spezifikation beschriebenen Anwendungsfälle zusammen mit den jeweils beteiligten Aktoren in einem Use-Case-Diagramm. Betrachten Sie den *Keyserver* zunächst ebenfalls als Aktor, welcher gegenüber PKI als „externer Vermittler" auftritt.

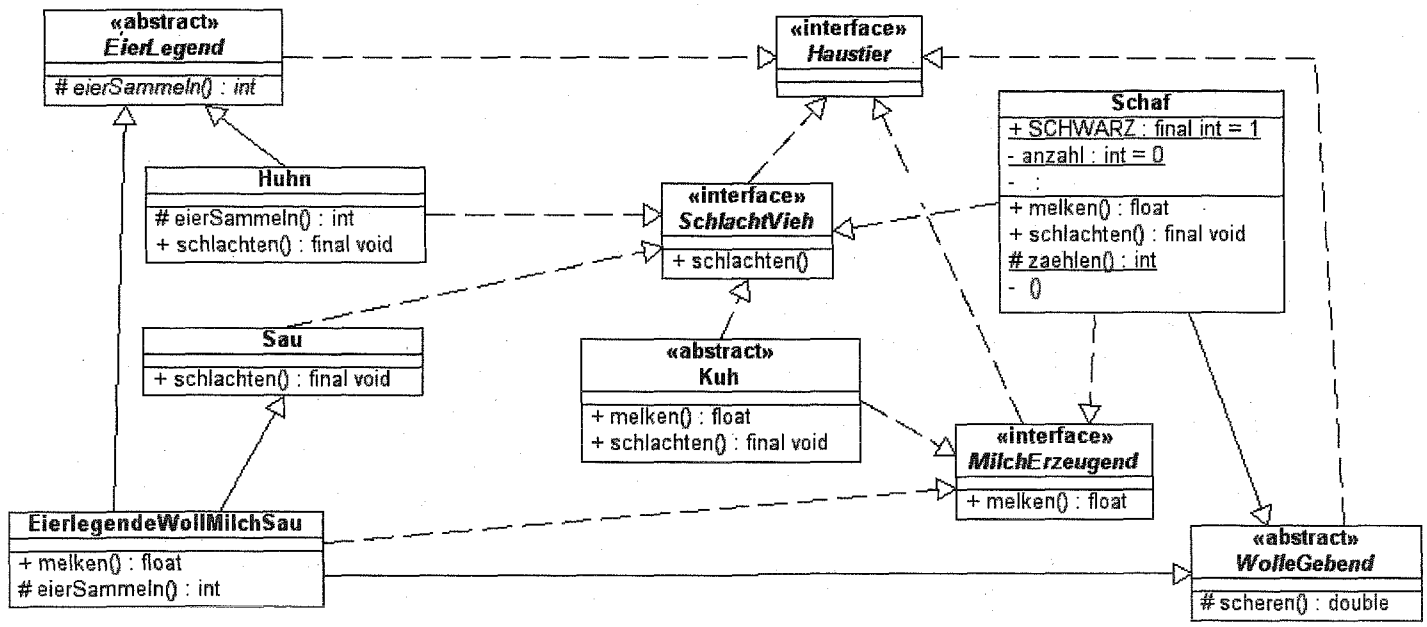
b) Erstellen Sie das Klassendiagramm für die von Ihnen identifizierten Klassen. Berücksichtigen Sie zusätzlich zur verbalen Spezifikation noch folgende Präzisierungen:

1. Es gibt genau einen *Keyserver*; dieser verwaltet aber beliebig viele Schlüssel. Er bietet die entsprechenden Dienste zum Veröffentlichen bzw. Abfragen von Schlüsseln an.
2. Der Schlüssel wird mit der eMail-Adresse von Berta „benannt", damit Anton den richtigen Schlüssel abrufen kann.
3. Jeder Lehrer hat höchstens ein Schlüsselpaar, aber jeder Schlüssel gehört genau einem Lehrer. Anton hingegen kennt außer Berta auch Christian und Dora, denen er auch verschlüsselte Nachrichten schicken möchte.
4. Eine Nachricht kann (muss aber nicht) eine Signatur als „Anhang" zusätzlich zum eigentlichen Inhalt mit sich führen.
5. Für das Signieren bzw. Entschlüsseln ist der PS zuständig. Dafür bekommt er den Inhalt der Nachricht und gibt entsprechend eine Signatur bzw. den entschlüsselten Inhalt zurück.
6. Für das Prüfen der Signatur und das Verschlüsseln ist der ÖS zuständig. Dazu bekommen die Methoden je nach Bedarf den Inhalt der Nachricht und die Signatur und liefern einen Wahrheitswert bzw. den verschlüsselten Inhalt zurück.

Fortsetzung nächste Seite!

5. UML - OOD

Gegeben sei folgendes Klassendiagramm eines Moduls „**bauernhof**“:



a) Geben Sie den Code der Klassen **`EierLegend`** und **`Schaf`** aus dem obigen Klassendiagramm in einer objekt-orientierten Sprache Ihrer Wahl an und annotieren Sie ihn mit verständlichen Kommentaren. Etwaige Methodenrümpfe können Sie leer lassen, auch wenn dies zu einem Compiler-Fehler führen würde. Beachten Sie die Sichtbarkeiten (**`#` = `protected`**) und dass **`SCHWARZ`** sowie **`schlachten()`** *final* sind!

b) Warum kann es die Klasse **`EierlegendeWollMilchSau`** in Java nicht geben? Beschreiben Sie kurz aber vollständig, welche minimalen Änderungen Sie an der obigen Architektur vornehmen würden, damit die Klasse **`EierlegendeWollMilchSau`** in Java doch implementiert werden kann, sie aber weiterhin eine Unterklasse von **`Sau`** bleibt.

c) Stellen Sie sich eine Klasse Bauer mit einer Methode **`alltag()`** vor:
In dieser Methode werden zunächst je zwei Instanzen der Klassen **`Huhn`** bzw. **`Schaf`** erstellt. Danach muss der Bauer bei seinen Hühnern die **`eierSammeln()`**, das erste Schaf **`scheren()`** und das zweite **`melken()`**. Schließlich erstellt er sich noch eine Instanz von **`Sau`**, die er gleich im Anschluss **`schlachten()`** kann.
Mit welcher UML-Diagrammart würden Sie das obige Szenario modellieren? Warum halten Sie diese Diagrammart für besonders geeignet?

d) Modellieren Sie die Methode **`alltag()`** in der vorangehend gewählten Diagrammart Ihrer Wahl.