Prüfungsteilnehmer	Prüfungstermin	Einzelprüfungsnummer

Kennzahl: _____

Kennwort: _____

Arbeitsplatz-Nr.: _____

Herbst 2020

46116

Erste Staatsprüfung für ein Lehramt an öffentlichen Schulen

— Prüfungsaufgaben —

Fach: **Informatik (Unterrichtsfach)**

Einzelprüfung: **Softwaretechnologie/Datenbanksysteme**

Anzahl der gestellten Themen (Aufgaben): 2

Anzahl der Druckseiten dieser Vorlage: 23

Bitte wenden!

Thema Nr. 1
(Aufgabengruppe)

Es sind alle Aufgaben dieser Aufgabengruppe zu bearbeiten!

Teilaufgabe I: Softwaretechnologie

Aufgabe 1 (Entwicklungsprozesse) [28 PUNKTE]

a) Nennen Sie drei Phasen, die im V-Modell mit steigendem Detailgrad durchlaufen werden sowie das Phasenergebnis und den jeweiligen Test des Phasenergebnisses.

b) Nennen Sie zwei Vorteile und zwei Nachteile der agilen Softwareentwicklung gegenüber dem V-Modell, jeweils mit Begründung.

c) Bestimmen Sie zu folgenden (voneinander unabhängigen) Wartungsvorgängen jeweils die genaue Art der Wartung (adaptiv, korrektiv, perfektionierend) und begründen Sie kurz Ihre Entscheidung:

   i. Das Softwaresystem einer Firma muss nach einem Hardware-Upgrade angepasst werden.

   ii. Ein Softwaresystem zur Berechnung des Firmenumsatzes wird auf ein neues Steuergesetz umgestellt.

   iii. Ein Studentenprojekt soll in ein Startup überführt werden. Dazu wird zwar ein Großteil der Software übernommen, aber mehrere Subroutinen enthalten schwer lesbaren Code und werden daher neu geschrieben.

   iv. Ein Studentenprojekt soll in ein Startup überführt werden. Dazu wird zwar ein Großteil der Software übernommen, aber mehrere Subroutinen werden auf Grund von Patentfragen neu entwickelt.

   v. Ein Chip-Hersteller liefert auf Grund eines fehlerhaften Chip-Designs neue Steuersoftware an alle Kunden.

d) Eine Webseite besteht typischerweise aus drei Komponenten: HTML legt den Aufbau fest, CSS bestimmt das Erscheinungsbild und JavaScript bietet Interaktionsmöglichkeiten. Erklären Sie anhand dieses Beispiels, warum „Separation of Concerns“ eine wichtige Rolle in der Softwareentwicklung einnimmt.

e) Begründen Sie anhand zweier Beispiele, warum die Verwendung einer höheren Programmiersprache einen positiven Einfluss auf die Entwicklung eines größeren Softwaresystems hat!

f) Zeigen Sie anhand zweier Beispiele von Software-Maßen, dass diese ungeeignet zum Messen des Projektfortschritts sind.

Fortsetzung nächste Seite!

Aufgabe 2 (Projektmanagement) [13 PUNKTE]

Betrachten Sie die folgende Tabelle zum Projektmanagement:

Arbeitspaket	**Dauer (Tage)**	**abhängig von**
A1	5	
A2	5	A1
A3	15	A2
A4	30	A1
A5	5	A1, A3
A6	15	
A7	15	A2, A3
A8	15	A4, A5, A6
A9	15	A6
A10	10	A7

a) Erstellen Sie einen Netzplan (CPM-Diagramm) nach der Methode des kritischen Pfades, der die in der Tabelle angegebenen Abhängigkeiten berücksichtigt.
Das Diagramm muss nicht maßstabsgetreu sein, muss aber jede Information aus der gegebenen Tabelle enthalten.

b) Bestimmen Sie die Gesamtdauer des Projektes und geben Sie alle kritischen Pfade an.

c) Bestimmen Sie die Gesamtdauer des Projektes und geben Sie alle kritischen Pfade an, wenn das Arbeitspaket A4 nur halb so lange dauert.

d) Bestimmen Sie die Gesamtdauer des Projektes und geben Sie alle kritischen Pfade an, wenn das Arbeitspaket A5 sieben Tage mehr Zeit benötigt.

e) Beschreiben Sie kurz eine allgemeine Strategie, die während der Projektplanung eingesetzt wird, um Verzögerungen des Projektes zu vermeiden.

Fortsetzung nächste Seite!

Aufgabe 3 (Use-Case-Diagramm) [18 PUNKTE]

Im Folgenden ist eine Systembeschreibung für eine „Schul-Software“ angegeben.

> Für eine Schule soll eine neue Verwaltungssoftware entwickelt werden und Sie sind daran beteiligt. Zur Schulfamilie gehören allgemein mehrere Personengruppen: Lehrer, das Direktorat, der Hausmeister, natürlich Schüler und deren Eltern. Damit alle Beteiligten entsprechend Zugriff haben, soll das System vollständig über den Webbrowser benutzbar sein. Grundsätzlich sind die Hilfe und der aktuellen Speiseplan der Kantine öffentlich verfügbar, um jedem einen schnellen Überblick zu geben. Die Mitglieder der Schulfamilie können sich am System anmelden und dann weitere Aktionen durchführen. Schüler (und deren Eltern) sollen Zugang zum jeweiligen Stundenplan haben. Lehrer können sowohl ihren eigenen als auch die klassenspezifischen Stundenpläne aufrufen. Lehrer können Noten eintragen, welche von den Schülern abgerufen werden können. Es soll die Möglichkeit geben, dass sich beliebige Teilnehmer Nachrichten schicken. Lehrer und das Direktorat können zusätzlich Elternbriefe verschicken, die an eine größere Anzahl an Schülern (bzw. deren Eltern) gerichtet sind. Meldungen über Schäden am Inventar können von den Lehrern an den Hausmeister gemeldet werden. Dieser kann sich die Liste aller Schäden anzeigen lassen. Sollte ein Schaden bereits gemeldet worden sein, wird dem Lehrer ein entsprechender Hinweis angezeigt. Das Direktorat hat die Möglichkeit, den Stundenplan zu erstellen und Zeugnisse drucken zu lassen.

a) Geben Sie die im Text erwähnten Akteure für das beschriebene System an.

b) Identifizieren Sie zwei weitere Stakeholder und nennen Sie dazu je einen unterschiedlichen Anwendungsfall des Systems, in den diese involviert sind.

c) Geben Sie sechs verschiedene Anwendungsfälle für das beschriebene System an.

d) Erstellen Sie aus Ihren vorherigen Antworten ein Use-Case-Diagramm für das beschriebene System, in dem die Akteure und Anwendungsfälle inkl. möglicher Generalisierungen und Beziehungen eingetragen sind. Achten Sie insbesondere auf mögliche ⟨⟨include⟩⟩- und ⟨⟨extends⟩⟩- Beziehungen und Bedingungen für Anwendungsfälle.

Fortsetzung nächste Seite!

Aufgabe 4 (Objektorientierte Analyse) [14 PUNKTE]

Betrachten Sie das folgende Szenario:

> Es ist Freitagabend und Sie überlegen sich, zur Studentenparty im Freundeskreis zu gehen. Nachdem Sie Ihre **Jacke angezogen** haben, **gehen Sie zur Party**. Dort angekommen, wird Ihnen auch schnell klar, warum Sie nicht zu Hause geblieben sind. Es gibt lustige Trinkspiele. Natürlich alles nur mit Mineralwasser, da Alkohol ein echter Stimmungskiller ist. Das Spiel geht so:
>
> - Als Erstes wird per Zufall ein langer oder kurzer **Strohhalm gezogen**. Sofern der kurze Strohhalm gezogen wurde, muss nun so lange mit drei **Würfeln geworfen** werden, bis die Summe der Augen nicht größer als 15 ist.
> - Ist die Summe nicht größer als 15, so wird ein Becher leckeres **Mineralwasser getrunken** und die Runde ist beendet.
> - Wurde ein langer Strohhalm gezogen, so muss die Aufgabe erfüllt werden, gleichzeitig eine vorgegebene **Strecke zu laufen** und dabei einen Becher mit **Mineralwasser zu trinken**. Bevor dies ausgeführt werden kann, muss jedoch dem Veranstalter für die gute Feier ein **Dank ausgesprochen** werden, welcher dies durch ein lautes Lachen bestätigt.
> - Ist nun die eine oder andere Runde beendet, so muss der Teilnehmer noch ein **Lied singen**. Eine Jury **bewertet das Lied** als gut oder schlecht. Gute Lieder führen zum Sieg und damit zum Ende des Spiels. Schlechte Lieder müssen wiederholt und erneut bewertet werden.

a) Erstellen Sie ein UML-Aktivitätsdiagramm für den gegebenen Sachverhalt. Achten Sie dabei auf eine vernünftige Benennung der Aktivitäten und Bedingungen.

b) Sowohl das UML-Zustandsdiagramm wie auch UML-Aktivitätsdiagramm beschreiben Verhaltensaspekte eines Systems. Erklären Sie kurz, wie diese beiden Sichten zusammenhängen.

Fortsetzung nächste Seite!

Aufgabe 5 (UML und Entwurfsmuster) [17 PUNKTE]

Betrachten Sie das folgende Szenario:

Ein *Sandwich* besteht in seiner einfachsten Form lediglich aus einer *Brotscheibe*. Es kann aber auch ein Brot mit Belag (ein sog. *BelegtesBrot*) sein; der Einfachheit halber gehen wir in unserer zu modellierenden Welt davon aus, dass es nur mit Schinken, mit einer Scheibe Gurke und/oder Käse belegte Sandwiches gibt. Die entsprechenden Gerichte heißen dann *Schinkenbrot*, *Gurkenbrot* bzw. *Käsebrot*. Man kann allerdings beliebig viele Beläge kombinieren, also auch mehrfach verwenden. Der oberste Belag bestimmt, ob es sich um ein Schinkenbrot, ein Gurkenbrot oder ein Käsebrot handelt. Mit anderen Worten: Durch Auflegen einer Scheibe Käse auf ein Schinkenbrot wird aus einem vorhandenen Sandwich ein Käsebrot. Für den Verkauf kann man von jedem Sandwich mit *liefere Preis()* den Preis ermitteln (der sich aus der Summe der Preise für die Brotscheibe und die Beläge zusammensetzt).

a) Welches Design-Pattern bietet sich an, um den beschriebenen Sachverhalt zu modellieren?

b) Geben Sie ein UML-Klassendiagramm an, das den beschriebenen Sachverhalt modelliert. Achten Sie auf Multiplizitäten bei Komposition und Aggregation.

c) Gegeben sei ein Sandwich-Objekt *sw*, das aus einer Scheibe Brot, zwei Scheiben Schinken, einer Scheibe Käse und obendrauf einer Scheibe Gurke besteht (in dieser Reihenfolge!). Geben Sie ein UML-Objektdiagramm an, das diese Situation darstellt.

d) Begründen Sie, ob das Objektdiagramm für den gegebenen Term basierend auf Ihrem Klassendiagramm eindeutig definiert ist.

e) Jeder einzelne Bestandteil eines Sandwiches holt sich bei Bedarf den aktuell gültigen Preis aus einer zentralen Datenbank-Instanz *db*. Die Datenbank soll folgenden Datensatz enthalten: {*Brotscheibe: 2€, Scheibe Schinken: 0,50€, Scheibe Käse: 0,40€, Scheibe Gurke: 0,10€*}. Erstellen Sie ein vollständig beschriftetes Sequenzdiagramm für die (erfolgreiche) Preisberechnung des genannten Sandwiches *sw* (aus Teilaufgabe c) und geben Sie den Gesamtpreis Ihrer Berechnung an.

Fortsetzung nächste Seite!

Teilaufgabe II: Datenbanksysteme

Aufgabe 1 (Entwurfstheorie: Entity-Relationship-Modellierung) [26 PUNKTE]

Gegeben sind zwei Entity-Relationship-Diagramme, siehe Abbildungen 1 und 2.

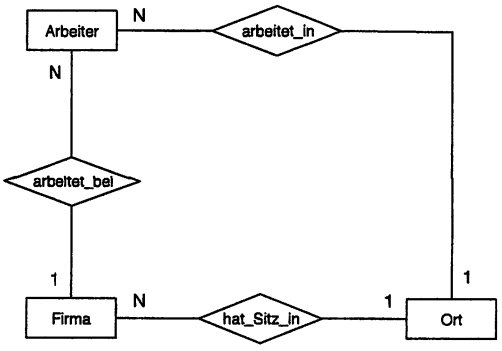


Abbildung 1: Firma

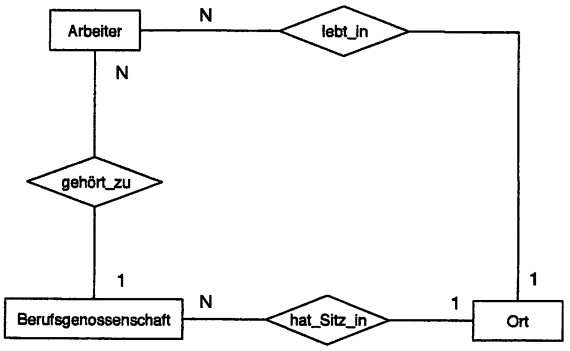


Abbildung 2: Berufsgenossenschaft

a) Beschreiben Sie den Zusammenhang der Entitäts- und Relationshiptypen in Abbildung 1.

b) Beschreiben Sie den Zusammenhang der Entitäts- und Relationshiptypen in Abbildung 2.

c) Untersuchen Sie die beiden Abbildungen auf redundante Relationships. Erläutern Sie, ob und ggf. wo es Redundanz gibt und wie diese aufgelöst werden kann.

d) Überführen Sie das ER-Modell in Abbildung 2 in ein verfeinertes relationales Schema. Wählen Sie geeignete Attribute (mind. zwei pro Entitätstyp) und kennzeichnen Sie Schlüssel durch Unterstreichen. Wählen Sie außerdem für jedes Attribut einen geeigneten Datentyp.

Fortsetzung nächste Seite!

Aufgabe 2 (SQL) [30 PUNKTE]

Gegeben ist das folgende Relationenschema zur Verwaltung der Fußball-Bundesliga:

Bundesliga: {[Mannschaft , Spieldatum, Heimspiel, Heimtore, Gasttore]}

Ferner existiert der folgende Ausschnitt einer Relationeninstanz:

Bundesliga	Mannschaft	Spieldatum	Heimspiel	Heimtore	Gasttore
	FC Augsburg	17.08.2019	0	5	1
	Bayern München	24.08.2019	0	0	3
	Freiburg	31.08.2019	1	1	2
	Werder Bremen	01.09.2019	1	3	2
	Bayern München	14.09.2019	0	1	1
	Freiburg	21.09.2019	1	1	1
	FC Augsburg	28.09.2019	1	0	3
	Werder Bremen	06.10.2019	0	2	2
	...	...	...	...	...

Das Attribut *Heimspiel* hat den Wert 1, falls es sich um ein Heimspiel der jeweiligen Mannschaft handelt, ansonsten 0. *Heimtore* beschreibt die Anzahl erzielter Tore der Heimmannschaft, *Gasttore* die entsprechenden Tore der Gastmannschaft.

Bearbeiten Sie die folgenden Teilaufgaben:

a) Erläutern Sie kurz, warum das Attribut *Spieldatum* Teil des Primärschlüssels ist.

b) Schreiben Sie eine SQL-Anweisung, welche die Tabelle *Bundesliga* anlegt. Verwenden Sie geeignete Datentypen (kein Boolean). Stellen Sie über Constraints sicher, dass das Attribut *Heimspiel* nur die Werte 0 bzw. 1 annehmen kann und dass weder *Heimtore* noch *Gasttore* negativ sein können. Sie brauchen keine Daten "einfügen".

c) Erstellen Sie mittels SQL eine View *Auswaertserfolge*, die alle Einträge von Auswärtsspielen enthält, die mit einem Sieg oder einem Unentschieden für die Gastmannschaft enden.

d) Schreiben Sie eine SQL-Anweisung zum Ändern des Ergebnisses von *Werder Bremen* vom *06.10.2019* in der View *Auswaertserfolge* aus Aufgabe c):
Die Gastmannschaft hat 2:0 verloren.

Fortsetzung nächste Seite!

e) Schreiben Sie eine SQL-Anweisung, welche für jede Mannschaft die Zeit in Tagen, die zwischen dem ersten und dem letzten Spiel liegt, ermittelt.

f) Ermitteln Sie mit einer SQL-Anweisung die *Mannschaft*, *Heimtore* und das *Spieldatum* für solche Heimspiele, an denen der Gastgeber weniger Tore als der Mittelwert aller Heimtore der Mannschaften erzielt hat. Sortieren Sie das Ergebnis bezüglich der Tore absteigend.

Aufgabe 3 (Entwurfstheorie) [14 PUNKTE]

Bearbeiten Sie die folgenden Teilaufgaben:

a) Gegeben sei das folgende Relationenschema **R**:{[A, B, C, D]} (in 1. Normalform) mit folgender Ausprägung:

R	A	B	C	D
	a1	b1	c1	d1
	a1	b1	c2	d1
	a2	b2	c1	d2
	a2	b2	c2	d2
	a3	b2	c2	d3
	a4	b1	c2	d4
	a4	b1	c4	d4
	a4	b1	c5	d4

Welche der folgenden funktionalen Abhängigkeiten sind in obiger Ausprägung der Relation erfüllt? Geben Sie jeweils eine kurze Begründung an.

i. $A \rightarrow B$
ii. $C \rightarrow B$
iii. $AC \rightarrow D$
iv. $A \rightarrow D$
v. $D \rightarrow C$
vi. $BC \rightarrow A$

Fortsetzung nächste Seite!

b) Eine Relation **S** (A, B, C, D, E, F) (in 1. Normalform) habe folgende funktionalen Abhängigkeiten:

$FD = \{$

$$
\begin{aligned}
BC &\rightarrow C \\
C &\rightarrow AD \\
D &\rightarrow CE \\
E &\rightarrow BC \\
F &\rightarrow D
\end{aligned}
$$

$\}$

i. Bestimmen Sie alle Kandidatenschlüssel von **S** mit FD. Begründen Sie Ihre Antwort. Begründen Sie zudem, warum es keine weiteren Kandidatenschlüssel gibt.
Hinweis: Die Angabe von Attributmengen, die keine Kandidatenschlüssel sind, führt zu Punktabzügen.

ii. Prüfen Sie, ob **S** mit FD in 2. Normalform ist. Begründen Sie Ihre Antwort.

iii. Prüfen Sie, ob **S** mit FD in 3. Normalform ist. Begründen Sie Ihre Antwort.

Fortsetzung nächste Seite!

Aufgabe 4 (Transaktionen) [20 PUNKTE]

Bearbeiten Sie die folgenden Teilaufgaben:

a) Beschreiben Sie das *Lost Update* Problem. Geben Sie hierfür ein geeignetes Beispiel in Form von zwei nebenläufigen Transaktionen an, welche bei unkontrolliertem Mehrbenutzerbetrieb zu einem Lost Update führen. Welche ACID-Eigenschaft wird im Lost Update Problem verletzt?

b) Erklären Sie den Unterschied zwischen *Livelock* und *Deadlock* im Zusammenhang mit sperrbasierten Synchronisationsmethoden.

c) Gegeben sind vier Transaktionen T_1, T_2, T_3 und T_4. Jede Transaktion T_i, $i \in \{1,2,3,4\}$, möchte wie folgt ein Objekt X lesen und anschließend schreiben.

$$
\begin{array}{lcllllll}
T_1 & : & l_1(A); & r_1(A); & l_1(B); & w_1(B); & u_1(A); & u_1(B); \\
T_2 & : & l_2(C); & r_2(C); & l_2(A); & w_2(A); & u_2(C); & u_2(A); \\
T_3 & : & l_3(B); & r_3(B); & l_3(C); & w_3(C); & u_3(B); & u_3(C); \\
T_4 & : & l_4(D); & r_4(D); & l_4(A); & w_4(A); & u_4(D); & u_4(A);
\end{array}
$$

Dabei gilt:

- $l_i(X)$ ist ein Lock auf das Element X
- $r_i(X)$ möchte X lesen
- $w_i(X)$ möchte X schreiben
- $u_i(X)$ ist ein Unlock von X

Fortsetzung nächste Seite!

Nachstehend ist der Beginn eines Schedules für obige Transaktionen gegeben.

	T_1	T_2	T_3	T_4
1.	$l_1(A); r_1(A)$			
2.		$l_2(C); r_2(C)$		
3.			$l_3(B); r_3(B)$	
4.				$l_4(D); r_4(D)$
5.		$l_2(A);$		
6.			$l_3(C);$	
7.				$l_4(A);$
8.	$l_1(B);$			

Zeichnen Sie den dazugehörigen Wartegraphen zur Erkennung eines Deadlocks und erläutern Sie anhand des Graphs, ob eine Verklemmung vorhanden ist oder nicht.

Definition Wartegraph: Die Knoten eines Wartegraphen entsprechen den Kennungen der Transaktionen. Wann immer eine Transaktion T_i auf die Freigabe einer Sperre durch eine Transaktion T_j wartet, wird die Kante $T_i \rightarrow T_j$ eingefügt. Die Kanten sind gerichtet. Entsteht ein Zyklus, liegt eine Verklemmung vor.

d) Falls ein Deadlock vorhanden ist, wie kann dieser aus Sicht des Transaktionsmanagers aufgelöst werden?

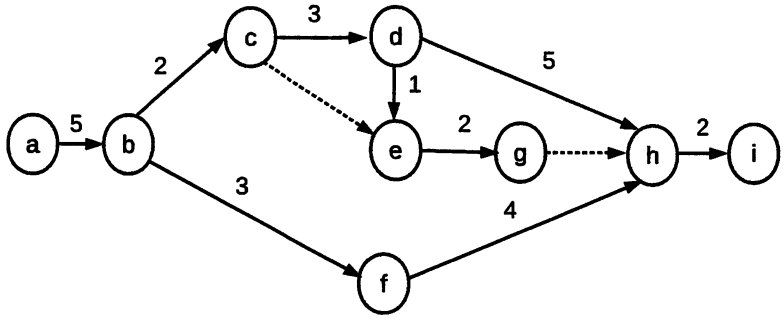
Thema Nr. 2
(Aufgabengruppe)

Es sind alle Aufgaben dieser Aufgabengruppe zu bearbeiten!

Teilaufgabe I: Softwaretechnologie

Aufgabe 1 (Projektplanung) [15 PUNKTE]

Gegeben sei folgendes CPM-Netzwerk ("Critical-Path-Method"):



a) Geben Sie zunächst an, was die Knoten und Kanten in dieser Diagrammart hinsichtlich Aktivitäten im Projekt bedeuten.

b) Welche Pseudoaktivität(en) kann man ohne Informationsverlust aus dem Netzwerk entfernen? Geben Sie auch den Grund dafür an!

c) Welche zwei Knoten liegen – neben Start- und Endknoten – auf jeden Fall auf dem kritischen Pfad (ohne Berechnung der Zeiten)? (Keine weitere Begründung erforderlich!)

d) Geben Sie für jedes Ereignis den frühesten Zeitpunkt unter der Annahme an, dass das Projekt zum Zeitpunkt 0 beginnt.

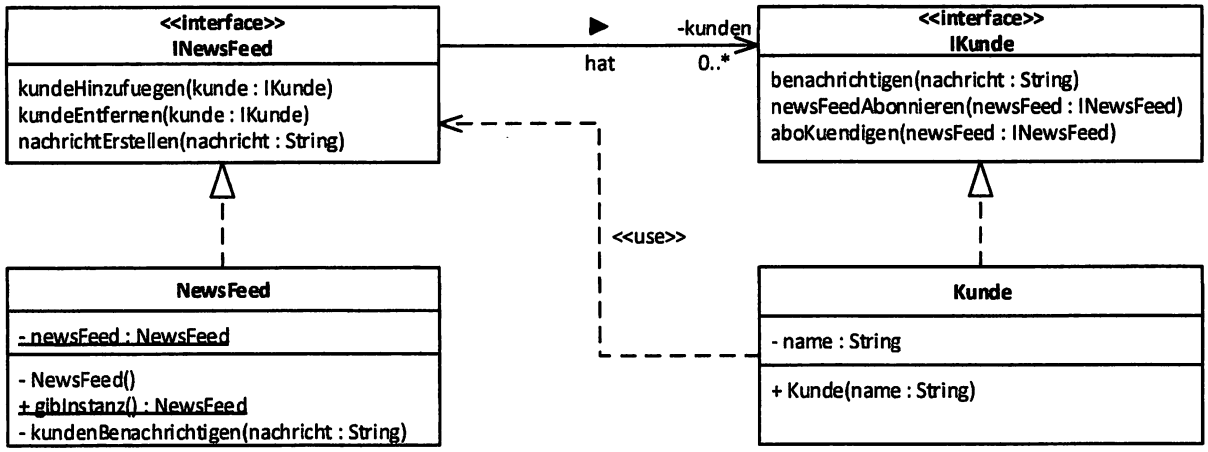
e) Berechnen Sie für jedes Ereignis den spätesten Zeitpunkt. Setzen Sie für das letzte Ereignis den spätesten Zeitpunkt gleich dem frühesten Zeitpunkt und berechnen Sie die übrigen Termine rückwärts.

f) Geben Sie alle kritischen Pfade durch das Netzwerk an.

g) Was heißt es für das Projektende, wenn sich ein Ereignis auf dem kritischen Pfad um 3 Zeiteinheiten verspätet?

Fortsetzung nächste Seite!

Aufgabe 2 (Entwurfsmuster) [30 PUNKTE]



Ihnen sei ein **UML Klassendiagramm** zu folgendem Szenario gegeben. Kunden können einen Newsfeed abonnieren bzw. ein Abo kündigen. Dementsprechend werden sie in dem Newsfeed vermerkt bzw. aus der Kundenliste wieder entfernt. Sobald eine neue Nachricht im Newsfeed erstellt wird, werden alle Abonnenten benachrichtigt.

a) Im angegebenen Klassendiagramm werden **zwei** unterschiedliche Entwurfsmuster verwendet. Um welche Muster handelt es sich? Geben Sie jeweils den Namen des Musters sowie die Elemente des Klassendiagramms an, mit denen diese Muster im Zusammenhang stehen. ACHTUNG: Es handelt sich dabei **nicht** um das *Interface*- oder das Vererbungsmuster.

b) Nennen Sie zwei generelle Vorteile von Entwurfsmustern.

c) Geben Sie eine Implementierung der Klasse `NewsFeed` an. Die Methode `nachrichtErstellen()` soll lediglich die Methode `kundenBenachrichtigen()` mit der entsprechenden Nachricht aufrufen.

d) Geben Sie eine Implementierung der Klasse `Kunde` an. Die Methode `benachrichtigen()` soll den Kundennamen sowie die Nachricht auf der Konsole ausgeben.

Hinweis: Die Implementierungen **müssen** sowohl dem Klassendiagramm, als auch den Konzepten der verwendeten Muster entsprechen. Verwenden Sie eine objektorientierte Programmiersprache, vorzugsweise Java. Sie müssen sich an der Testmethode und ihrer Ausgabe auf der nächsten Seite orientieren. Die Testmethode muss mit Ihrer Implementierung ausführbar sein und sich semantisch korrekt verhalten.

Fortsetzung nächste Seite!

Quelltext der Testmethode:

```
1 public static void main(String[] args) {
2     IKunde alice = new Kunde("Alice");
3     IKunde bob = new Kunde("Bob");
4     alice.newsFeedAbonnieren(NewsFeed.gibInstanz());
5     NewsFeed.gibInstanz().nachrichtErstellen("Bayern hat gewonnen.");
6     bob.newsFeedAbonnieren(NewsFeed.gibInstanz());
7     NewsFeed.gibInstanz().nachrichtErstellen("Spritpreise sinken.");
8     alice.aboKuendigen(NewsFeed.gibInstanz());
9     NewsFeed.gibInstanz().nachrichtErstellen("Wahlergebnisse stehen fest.");
10 }
```

Konsolenausgabe:

```
Nachricht an Alice: Bayern hat gewonnen.
Nachricht an Alice: Spritpreise sinken.
Nachricht an Bob: Spritpreise sinken.
Nachricht an Bob: Wahlergebnisse stehen fest.
```

Fortsetzung nächste Seite!

Aufgabe 3 (White-Box-Testverfahren) [30 PUNKTE]

Die *(einfache) Quersumme* einer natürlichen Zahl ist die Summe aller ihrer Ziffern in ihrer Repräsentation als Dezimalzahl. Die Quersumme der Dezimalzahl 4242 ist beispielsweise $2+4+2+4=12$. Bei der *alternierenden Quersumme* werden die einzelnen Ziffern abwechselnd addiert und subtrahiert (beginnend mit der rechtesten Ziffer). Die alternierende Quersumme der Dezimalzahl 4242 ist beispielsweise $2-4+2-4=-4$.

Folgende **Java-Methode** berechnet für eine übergebene Zahl die (alternierende) Quersumme. Eine Eingabe wird durch ein Paar `(a,b)` beschrieben; dabei beschreibt `a` die Zahl und `b` den Wahrheitswert, ob die alternierende Quersumme berechnet werden soll. Für die Eingabe `(42,true)` wird beispielsweise die alternierende Quersumme der Dezimalzahl 42 berechnet, für die Eingabe `(99,false)` die (einfache) Quersumme der Dezimalzahl 99. Falls `a` keine natürliche Zahl darstellt, wird `0` zurückgegeben.

```
1  public static int berechneQuersumme(int zahl, boolean alternierend) {
2      int resultat = 0;
3      if (zahl >= 0) {
4          if (zahl < 10) {
5              resultat = zahl;
6          } else {
7              int faktor = 1;
8              do {
9                  int ziffer = zahl % 10;
10                 zahl = zahl / 10;
11                 resultat = resultat + faktor * ziffer;
12                 if (alternierend) {
13                     if (faktor == 1) {
14                         faktor = -1;
15                     } else {
16                         faktor = 1;
17                     }
18                 }
19             } while (zahl > 0);
20         }
21     }
22     return resultat;
23 }
```

Beachten Sie bei der Bearbeitung der Aufgabe die Hinweise auf der nächsten Seite!

a) Geben Sie für die Methode einen **Kontrollflussgraphen** an, wobei Sie die Knoten mit den jeweiligen Zeilennummern im Quelltext beschriften.

b) Geben Sie eine **minimale Testmenge** an, die das Kriterium der *Anweisungsüberdeckung* (*Knotenüberdeckung*) erfüllt. Geben Sie für jeden Testfall den Kontrollfluss als Pfad in der Notation $1 \to 2 \to \ldots$ an.

c) Geben Sie eine **minimale Testmenge** an, die das Kriterium der *Zweigüberdeckung* (*Kantenüberdeckung*) erfüllt. Geben Sie für jeden Testfall den Kontrollfluss als Pfad in der Notation $1 \to 2 \to \ldots$ an; falls ein Testfall bereits in Teilaufgabe b) genannt und dessen Pfad beschrieben wurde, genügt es, bei dieser Teilaufgabe nur den Testfall ohne Pfad anzugeben.

d) Beschreiben Sie den konzeptionellen **Unterschied** zwischen *White-Box-Tests* (hier eingesetzt) und *Black-Box-Tests*.

Fortsetzung nächste Seite!

Hinweise:

- Diese Aufgabe betrachtet als *natürliche Zahlen* die Menge $\mathbb{N} = \{0, 1, \dots\}$ (also inklusive der Zahl 0).
- Der Operator % (vgl. Quelltext-Zeile 9) notiert den mathematischen Modulo-Operator, welcher für $a \bmod b$ den Rest der ganzzahligen Division von a durch b berechnet, z. B. $42 \bmod 10 = 2$.
- Eine *Testmenge* ist *minimal*, wenn es keine Testmenge mit einer kleineren Zahl von Testfällen gibt. Die Minimalität muss nicht bewiesen werden.

Aufgabe 4 (Lebenszyklus) [15 PUNKTE]

a) Beschreiben Sie den Entwicklungszyklus bei testgetriebener Entwicklung.

b) Nennen Sie die Phasen des RUP und geben Sie stichpunktartig ein jeweiliges Ziel an.

c) Beschreiben Sie die Grundstuktur des V-Modells.

Fortsetzung nächste Seite!

Teilaufgabe II: Datenbanksysteme

Aufgabe 1 (Modellierung) [19 PUNKTE]

Betrachten Sie folgenden Ausschnitt aus der Prüfungsverwaltung einer Hochschule:

Prüfungen sind durch eine eindeutige Prüfungsnummer (PrüfNr) gekennzeichnet. Sie erstrecken sich über ein bestimmtes Fach und finden an einem bestimmten Datum statt.

Dozenten haben eine eindeutige Personalnummer (PersNr) und einen Namen. Sie können Erstprüfer für beliebig viele Prüfungen sein, aber jede Prüfung hat genau einen Erstprüfer.

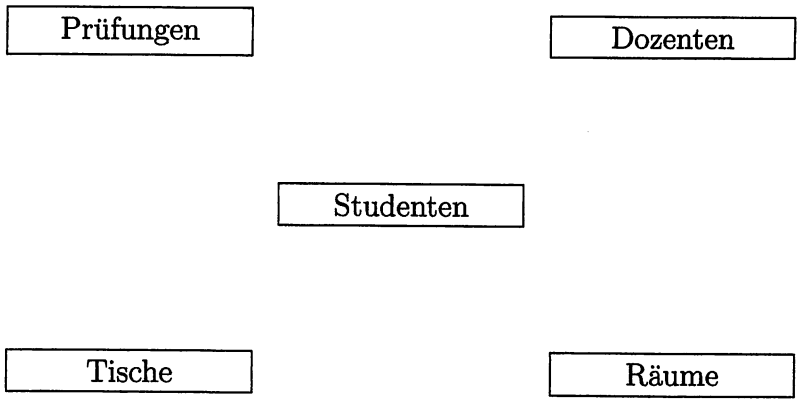
Räume werden durch ein Gebäudekürzel (GebK) und eine Raumnummer (RaumNr) eindeutig beschrieben. Außerdem wird ihre Größe erfasst.

In den Räumen befinden sich *Tische*. Diese werden durch eine Tischnummer (TischNr) gekennzeichnet, die innerhalb des jeweiligen Raumes eindeutig ist.

Studenten haben eine eindeutige Matrikelnummer (MatrNr) und einen Namen. Sie können an beliebig vielen Prüfungen teilnehmen; dabei wird ihnen jeweils ein Tisch zugewiesen, und sie erhalten anschließend eine Note. An jeder Prüfung können beliebig viele Studenten teilnehmen, aber wenn kein Student teilnimmt, wird die Prüfung abgesagt.

a) Modellieren Sie den beschriebenen Sachverhalt als Entity-Relationship-Diagramm. Geben Sie für jeden Gegenstandstyp sämtliche Attribute an und kennzeichnen Sie jeweils einen Primärschlüssel. Charakterisieren Sie die Beziehungstypen mit Funktionalitäten und der (min, max)-Notation entsprechend der Beschreibung.

Übernehmen Sie hierfür folgendes Grundgerüst in Ihre Lösung:



Nicht in die Angabe schreiben!

Fortsetzung nächste Seite!

b) Übersetzen Sie das Entity-Relationship-Diagramm in ein relationales Schema. Fassen Sie dabei Relationen zusammen, soweit dies möglich ist. Typen für die Attribute müssen nicht angegeben werden. Kennzeichnen Sie aber in jeder Relation den Primärschlüssel.

c) Wäre es sinnvoll, zwischen *Prüfungen* und *Räume* noch eine Beziehung einzufügen, die angibt, in welchen Räumen welche Prüfungen stattfinden? Begründen Sie Ihre Antwort.

Aufgabe 2 (Entwurfstheorie) [18 PUNKTE]

Gegeben sei folgendes Relationenschema, das Informationen über Vorlesungen speichert. Beispielsweise liest Professor Codd, der die Personalnummer 2138, den Rang C4 und sein Büro im Raum K123 hat, die Vorlesung Datenbanken mit der Vorlesungsnummer 5432 am Dienstag um 08:15 Uhr im Hörsaal K014 und am Donnerstag um 10:00 Uhr im Hörsaal K003.

Vorlesungsverzeichnis								
PersNr	Name	Rang	Büro	VorlNr	VorlTitel	VorlTag	VorlZeit	Hörsaal
2138	Codd	C4	K123	5432	Datenbanken	Di	08:15	K014
2138	Codd	C4	K123	5432	Datenbanken	Do	10:00	K003
...	...	...	...	...	...	...	...	...

Dabei hat jeder Professor eine eindeutige Personalnummer, einen festgelegten Rang (Besoldungsgruppe) und ein Büro für sich alleine. Jede Vorlesung hat eine eindeutige Vorlesungsnummer, einen Titel (mehrere verschiedene Vorlesungen können den selben Titel tragen) und einen festen Professor, der sie liest. Eine Vorlesung kann an mehreren Terminen, aber nicht mehrfach an einem Tag, und in unterschiedlichen Hörsälen stattfinden. Und natürlich kann zu einem bestimmten Termin in einem Hörsaal nur eine Vorlesung stattfinden.

a) Geben Sie die geltenden funktionalen Abhängigkeiten in kanonischer Überdeckung an.

b) Bestimmen Sie den oder die Kandidatenschlüssel (ohne Begründung).

c) Ist das Schema in zweiter Normalform? Geben Sie eine kurze Begründung an.

Fortsetzung nächste Seite!

d) Überführen Sie das Schema mit Hilfe des Synthesealgorithmus in die dritte Normalform. Benennen Sie dabei jeden Schritt des Algorithmus und geben Sie das Zwischenergebnis nach jedem Schritt an. Kennzeichnen Sie im Endergebnis in jeder Relation einen Primärschlüssel.

e) Gegeben sei nun ein abstraktes Relationenschema $\mathcal{R} = \{A, B, C, D, E, F\}$ mit den funktionalen Abhängigkeiten

$$S = \{A \to DF,\ BF \to CDE,\ C \to ADF,\ E \to BF,\ F \to AB\}.$$

Berechnen Sie die kanonische Überdeckung von S. Benennen Sie dabei jeden Schritt und geben Sie das Zwischenergebnis nach jedem Schritt an.

Aufgabe 3 (SQL-Anfragen) [25 PUNKTE]

Die folgenden Fragen beziehen sich auf das Universitätsschema, welches Sie in einer Beispielausprägung auf Seite 23 finden. Die Anfragen sind in SQL zu formulieren – und zwar so, dass sie unabhängig von der Ausprägung der Datenbank korrekt ausgewertet werden. Die Ergebnisse sind ohne Duplikate zu berechnen, aber unnötige Duplikatelimination ist zu vermeiden.

a) Welche Studenten haben die Vorlesung „Ethik“ prüfen lassen? Geben Sie die Matrikelnummern und Namen dieser Studenten aus.

MatrNr	Name
25403	Jonas

b) Wie viele Studenten haben die Vorlesung „Ethik“ prüfen lassen?

count
1

c) Was sind die direkten Voraussetzungen (“Vorgänger”) der Vorlesung „Ethik“?
Geben Sie Vorlesungsnummer und Titel dieser Vorlesung(en) aus.

VorlNr	Titel
5001	Grundzüge

Fortsetzung nächste Seite!

d) In welchen Fächern ist die Durchschnittsnote schlechter als 1?
Geben Sie die Vorlesungsnummer und den Titel der entsprechenden Vorlesungen aus.

VorlNr	Titel
4630	Die 3 Kritiken
5041	Ethik

e) Welche Professoren lesen mehr als eine Vorlesung?

PersNr	Name
2137	Kant
2126	Russel
2125	Sokrates

f) Welche Studenten haben (noch) **keine** Prüfungen abgelegt?
Geben Sie die Matrikelnummern und Namen dieser Studenten aus.

MatrNr	Name
24002	Xenokrates
26120	Fichte
26830	Aristoxenos
29120	Theophrastos
29555	Feuerbach

g) Welche Studenten hören **alle** Vorlesungen? Also für welche Studenten existiert keine Vorlesung zu der kein passender hören-Eintrag existiert.

MatrNr	Name

(Leeres Ergebnis)

Fortsetzung nächste Seite!

h) Der Student *Fichte* (MatrNr 26120) hat die Prüfung in *Glaube und Wissen* (VorlNr 5022) mit der Note 1 erfolgreich abgelegt. Prüfer war der Assistent *Spinoza* (PersNr 3007). Geben Sie ein entsprechendes SQL-Statement an, um die Prüfung in die Uni-Datenbank einzufügen. Sie können die gegebenen MatrNr, VorlNr, und PersNr direkt in ihr Insert-Statement übernehmen, ohne sie nochmal mit extra SQL-Anfragen zu bestimmen.

i) Gegeben sei folgende SQL-Anfrage auf dem Universitätsschema (siehe Seite 23).

```
WITH VorlesungenKant AS (
  SELECT * FROM Professoren p, Vorlesungen v
   WHERE p.PersNr = v.gelesenVon AND p.Name = 'Kant'
)
SELECT count(*), sum(vk.SWS)
  FROM VorlesungenKant vk
 GROUP BY vk.PersNr
```

Geben Sie das Ergebnis der Anfrage tabellarisch an.

Fortsetzung nächste Seite!

Beispielausprägung

Professoren			
PersNr	Name	Rang	Raum
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	52
2134	Augustinus	C3	309
2136	Curie	C4	36
2137	Kant	C4	7

Studenten		
MatrNr	Name	Semester
24002	Xenokrates	18
25403	Jonas	12
26120	Fichte	10
26830	Aristoxenos	8
27550	Schopenhauer	6
28106	Carnap	3
29120	Theophrastos	2
29555	Feuerbach	2

Vorlesungen			
VorlNr	Titel	SWS	gelesenVon
5001	Grundzüge	4	2137
5041	Ethik	4	2125
5043	Erkenntnistheorie	3	2126
5049	Mäeutik	2	2125
4052	Logik	4	2125
5052	Wissenschaftstheorie	3	2126
5216	Bioethik	2	2126
5259	Der Wiener Kreis	2	2133
5022	Glaube und Wissen	2	2134
4630	Die 3 Kritiken	4	2137

voraussetzen	
Vorgänger	Nachfolger
5001	5041
5001	5043
5001	5049
5041	5216
5043	5052
5041	5052
5052	5259

hören	
MatrNr	VorlNr
26120	5001
27550	5001
27550	4052
28106	5041
28106	5052
28106	5216
28106	5259
29120	5001
29120	5041
29120	5049
29555	5022
25403	5022
29555	5001

Assistenten			
PersNr	Name	Fachgebiet	Boss
3002	Platon	Ideenlehre	2125
3003	Aristoteles	Syllogistik	2125
3004	Wittgenstein	Sprachtheorie	2126
3005	Rhetikus	Planetenbewegung	2127
3006	Newton	Keplersche Gesetze	2127
3007	Spinoza	Gott und Natur	2134

prüfen			
MatrNr	VorlNr	PersNr	Note
28106	5001	2126	1
25403	5041	2125	2
27550	4630	2137	2
25403	4630	2137	5

Abb. 1: Beispielausprägung für eine Universitäts-Datenbank