



**Die Handlungsschritte 1 bis 5 beziehen sich auf die folgende Ausgangssituation:**

Das Stadtkrankenhaus muss seine IT neu strukturieren.

Sie arbeiten in der EProg GmbH, die Softwarelösungen zur Verfügung stellt und verwaltet.

Sie sollen vier der folgenden fünf Aufgaben in diesem Projekt erledigen:

1. Projekt für die Erstellung eines „Wiki“ planen
2. Algorithmus zur Komprimierung von Bilddaten entwickeln
3. Datenansicht nach Entwurfsmuster entwickeln
4. Datenbank für Abrechnungssystem planen
5. SQL-Abfragen zu Zimmer- und Bettenbelegung formulieren

**1. Handlungsschritt (25 Punkte)**

Aufgrund der bestehenden Altersstruktur der Mitarbeiter muss das Stadtkrankenhaus frühzeitig Maßnahmen entwickeln, um den Wissensverlust zu minimieren. Um das Know-how und die Erfahrung der Mitarbeiter allen Abteilungen zugänglich zu machen, hat die Krankenhausleitung beschlossen, ein Wissensmanagementsystem (Wiki), aufzubauen. Das Pflegen des Systems soll durch die Mitarbeiter der Abteilung eigenständig erfolgen.

- a) Nennen Sie vier Vorteile eines firmeninternen Wikis für das Krankenhaus. 4 Punkte

---



---



---



---

- b) Zur Realisierung des Wikis wird ein Content-Management-System ausgewählt.

- Beschreiben Sie vier Funktionen, die ein solches System enthalten soll. 4 Punkte

---



---



---



---

- c) Das Projekt „Wiki“ wurde wie folgt geplant:

Vorgang	Beschreibung	Dauer	Vorgänger	Nachfolger
A	Istanalyse	4	–	B
B	Grobkonzeption	5	A	C, D
C	Vorstellung der Grobkonzeption	2	B	E, G
D	Feinkonzeption	6	B	F
E	Installation	2	C	F
F	Anpassung	4	D, E	H, I
G	Dokumentation	3	C	I
H	Planung Schulungsmaßnahmen	4	F	J
I	Tests	7	F, G	J
J	Übergabe	1	H, I	–

- ca) Erstellen Sie auf der gegenüberliegenden Seite anhand der Vorgangsliste einen Netzplan und kennzeichnen Sie den kritischen Pfad. 15 Punkte

Hinweis: Verwenden Sie folgenden Vorgangsknoten.

Vorgang	Vorgangs-ID (A, B, C ...)		
Dauer	Dauer in Arbeitstagen		
FAZ	Frühest Anfangzeitpunkt		
FEZ	Frühest Endzeitpunkt		
SAZ	Spätester Anfangzeitpunkt		
SEZ	Spätester Endzeitpunkt		
GP	Gesamtbuffer, GP = SAZ - FAZ Oder GP = SEZ - FEZ		
FP	Freier Puffer, FP = FAZ des Nachfolgers - FEZ des Vorgangs		

Vorgang		FEZ	FAZ
Dauer	GP	FP	
SAZ	FAZ		

Fortsetzung 1. Handlungsschritt →

## **Fortsetzung 1. Handlungsschritt**

Korrekturrand

cb) Vorgang E verschiebt sich um drei Tage.

Erläutern Sie, wie sich das auf das Projektende auswirkt.

2 Punkte

## **2. Handlungsschritt (25 Punkte)**

Die bildgebende Diagnostik liefert täglich viele Dateien, die gespeichert werden müssen. Um Speicherplatz einzusparen, soll ein Komprimierungsalgorithmus entwickelt werden. Für einen ersten Prototypen wurde folgende Vorgabe erstellt.

Vorgabe:

Die Bilddaten sollen mit einer Lauflängenkodierung komprimiert werden. Dabei werden sich direkt wiederholende Zeichen zusammengefasst, und nur die Anzahl und das entsprechende Zeichen erfasst. Eine Zusammenfassung soll erst bei mehr als vier Zeichen erfolgen. Zur Erkennung der Lauflängenkodierung wird das „%“-Zeichen verwendet, das in den unkomprimierten Bilddaten nicht vorkommt. Die Bilddaten liegen als String-Arrays vor.

Beispiel:

String[ ] „unkomprimiert“

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]	[16]	[17]	[18]	[19]	[20]	[21]
Z	Z	Z	Z	7	7	7	7	7	7	7	7	7	M	P	P	P	P	H	H		

String[ ] „komprimiert“

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]
Z	Z	Z	Z	%	10	7	M	%	5	P	H	H

ZZZZ7777777777MPPPPP(HH (unkomprimiert)

ZZZZ%107M%5PHH (komprimiert)

Folgende Funktionen stehen zu Verfügung:

laenge(String[ ]) : Integer	Gibt die Länge des übergebenen Zeichenkettenarrays als ganze Zahl zurück
add(String[], String) : String[ ]	Verlängert das übergebene Zeichenkettenarray um den übergebenen String und gibt es zurück
toString(Integer) : String	Gibt die übergebene ganze Zahl als String zurück

a) Entwickeln Sie auf der gegenüberliegenden Seite nach Vorgabe als Struktogramm, PAP oder Pseudocode eine Funktion „erstelleKomprimierung“, die aus einem übergebenen unkomprimierten String-Array ein komprimiertes String-Array erstellt und zurückgibt.  
20 Punkte

b) Bei der Dekomprimierung von „%53V“ tritt ein Problem auf.

ba) Beschreiben Sie kurz das Problem.

2 Punkte

bb) Beschreiben Sie kurz eine mögliche Lösung.  
3 Punkte

erstelleKomprimierung(umkomprimiert : String[ ]) : String[ ]

Korrekturrand



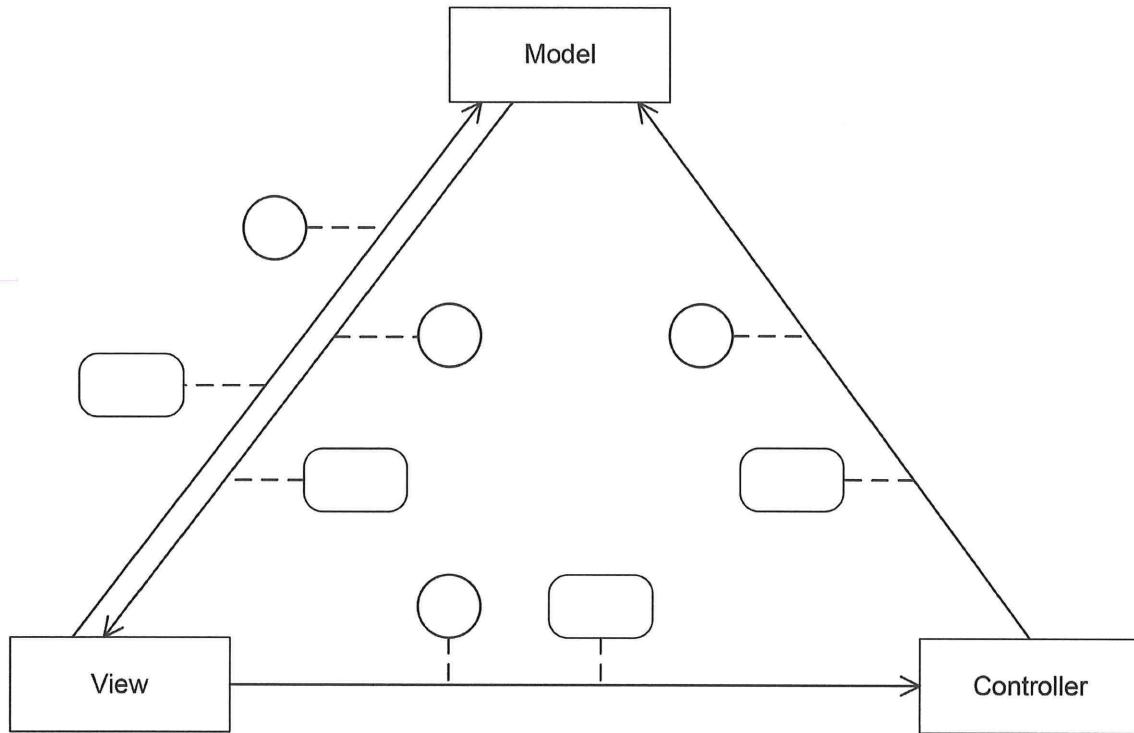
### 3. Handlungsschritt (25 Punkte)

Korrekturrand

Die Patientendaten (z. B. Blutdruck, Körpertemperatur) sollen im zeitlichen Verlauf in verschiedenen Ansichten (z. B. Tabelle, Säulendiagramm) dargestellt werden. Damit die Implementierung für zukünftige Erweiterungen offenbleibt, schlägt ein Teamkollege die Realisierung mit dem Model-View-Controller-Pattern (MVC-Muster) vor.

- a) Für das Verständnis des MVC-Musters soll eine Reihenfolge der Benachrichtigungen angegeben werden.

Ergänzen Sie im folgenden Diagramm die entsprechenden Ziffern (Reihenfolge) in den Kreisen und die Aktivitäten durch Zuweisung der entsprechenden Buchstaben. 4 Punkte

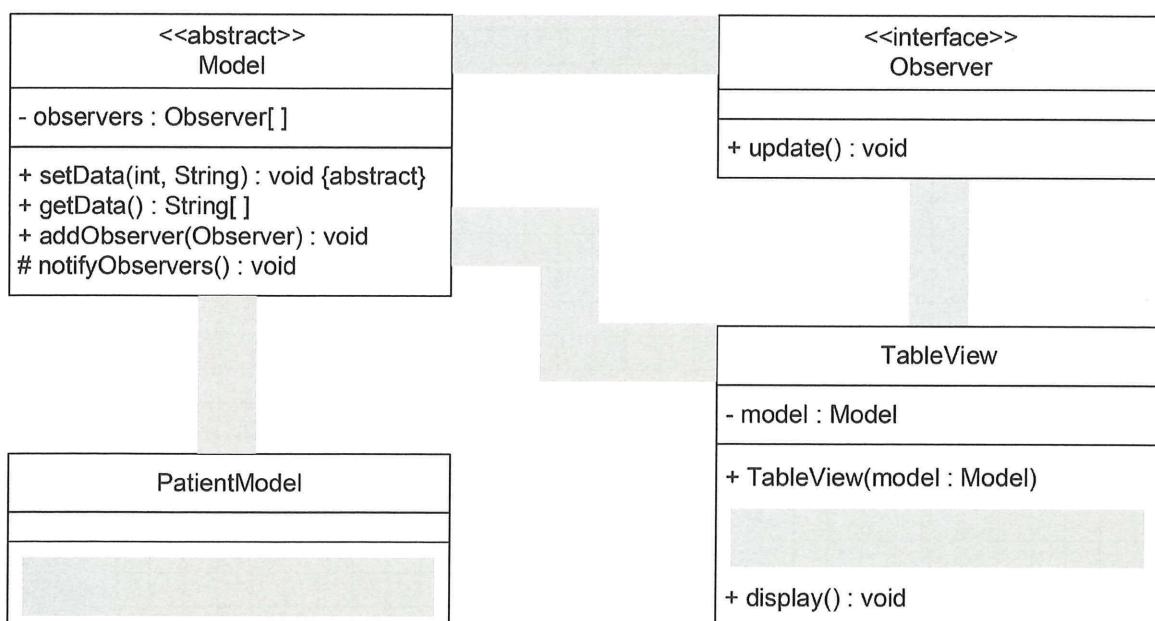


Aktivitäten

- |   |   |
|---|---|
| A | Controller fordert Model zu Zustandsänderung auf                            |
| B | Model informiert View über Zustandsänderung                                 |
| C | View fordert die geänderten Daten vom Model zur Ansicht für den Benutzer an |
| D | View informiert Controller über Benutzereingabe                             |

- ba) Model und View werden häufig über das Observer-Pattern realisiert. Dabei erbt die konkrete Klasse „PatientModel“ von der abstrakten Klasse „Model“. Die Klasse „TableView“ implementiert das Interface „Observer“.

Ergänzen Sie im vorliegenden UML-Klassendiagramm Methoden und Klassenbeziehungen. 6 Punkte



- bb) Der Konstruktor von „TableView“ initialisiert seine Modelreferenz mit dem übergebenen Modelobjekt und registriert sich mit der Methode „addObserver“ als Observer.

Korrekturrand

Geben Sie den Konstruktor in Pseudocode an.

3 Punkte

---

---

---

- bc) Die Methode „notifyObservers“ sorgt dafür, dass alle registrierten Observer die Methode „update“ ausführen.

Geben Sie die Methode in Pseudocode an.

3 Punkte

---

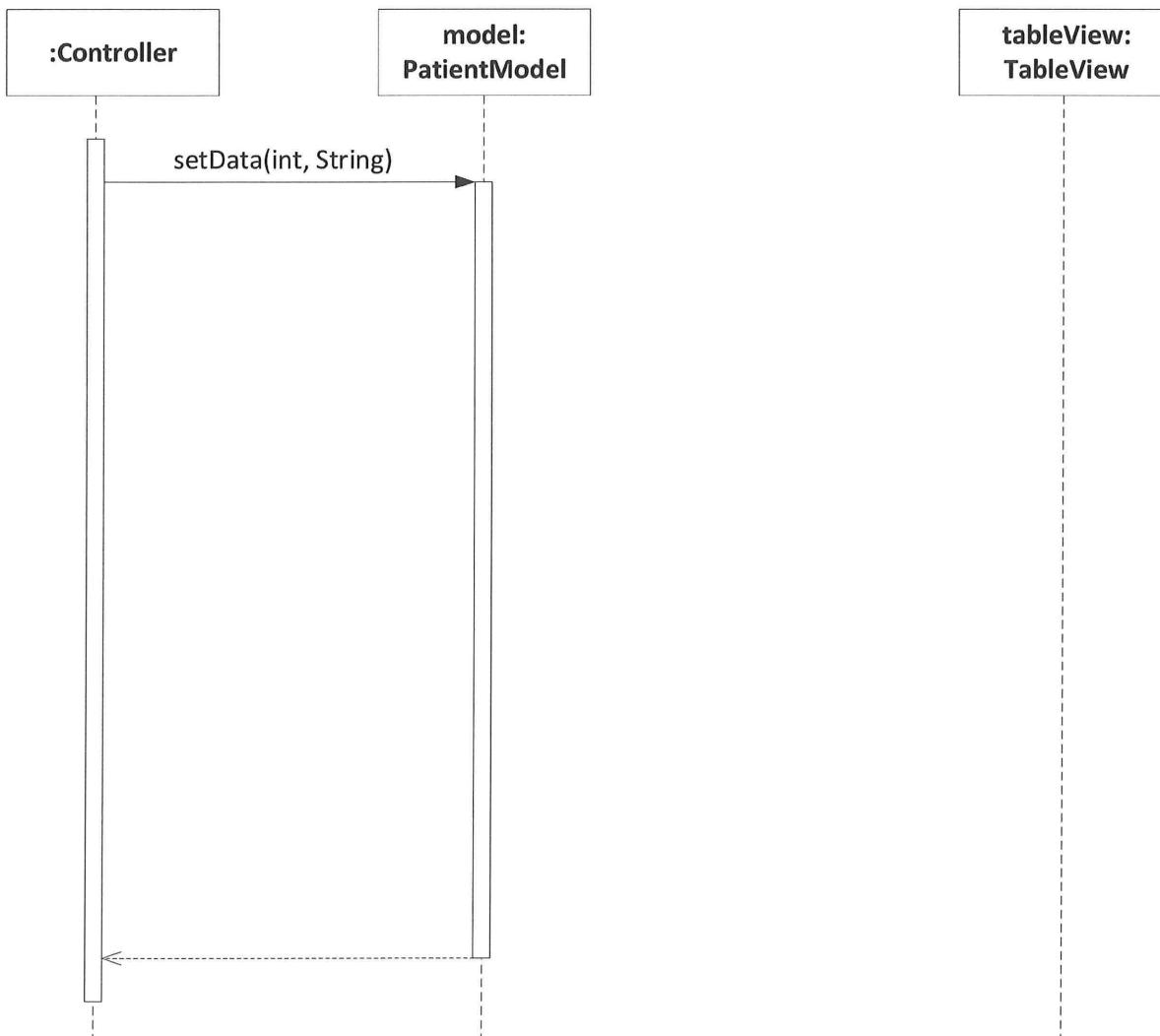
---

---

- bd) Sobald ein Benutzer mit der View interagiert, ruft der entsprechende Controller „setData“ auf dem Model auf. Die Methode „setData“ aktualisiert die Daten und startet anschließend „notifyObservers“. Die Methode „update“ ruft „getData“ auf und sorgt abschließend durch Aufruf von „display“ dafür, dass die geänderten Daten des zurückgegebenen String-Arrays auf „tableView“ dargestellt werden.

Ergänzen Sie das gegebene Sequenzdiagramm entsprechend der Vorgaben.

7 Punkte



Fortsetzung 3. Handlungsschritt →

### **Fortsetzung 3. Handlungsschritt**

Korrekturrand

Ein Kollege schlägt vor, anstatt der Observer-Musters zur Aktualisierung des Views Datenbindung (Data Binding) einzusetzen.

- c) Erläutern Sie den Begriff. 2 Punkte
- 
- 
- 

### **4. Handlungsschritt (25 Punkte)**

Das Stadtkrankenhaus benötigt ein neues Abrechnungssystem für seine Patienten.

Die medizinischen Leistungen wurden bislang in folgender Tabelle erfasst:

Patient-Nr	Patient Name	Patient Anschrift	Leistung Datum	Leistung Nr	Bezeichnung	Preis	Arzt Nr	Arzt Name	Arzt Faktor
56843	Müller, Klaus	Südstr. 24 54321 Burg	20.04.2020	1234	Untersuchung	53,20	101	Sauer	1,5
				4889	Infektion	19,80	52	Helmig	1,0
				4932	Verband	17,79			
4569	Schulz, Britta	Nordstr. 9 57912 Hagen	20.04.2020	4889	Infektion	19,80	35	Birkeler	2,0
				8963	Visite	21,56	101	Sauer	1,5
56843	Müller, Klaus	Südstr. 24 54321 Burg	21.04.2020	8963	Visite	21,56	52	Helmig	1,0
6897	Rose, Bernd	Weststr. 5 55691 Schnurz	21.04.2020	4932	Verband	17,79	35	Birkeler	2,0
4569	Schulz, Britta	Nordstr. 9 57912 Hagen	22.04.2020	4889	Infektion	19,80	101	Sauer	1,5
				4711	MRT	800,00			
				8963	Visite	21,56			

...

Die nichtmedizinischen Zusatzleistungen wurden in der folgenden Tabelle erfasst.

Patient-Nr	Patient Name	Leistung von	Leistung bis	Leistung Nr	Bezeichnung	Tagespreis
56843	Müller, Klaus	20.04.	24.04.	Z12	Einzelzimmer	130,00
				Z13	Fernseher	8,50
				Z14	WLAN	2,00
4569	Schulz, Britta	19.04.	23.04.	Z12	Einzelzimmer	130,00
				Z13	Fernseher	8,50
56843	Müller, Klaus	21.04.	24.04.	Z18	Wahlessen	25,00
4569	Schulz, Britta	19.04.	23.04.	Z12	Einzelzimmer	130,00
				Z13	Fernseher	8,50
				Z14	WLAN	2,00

...

- a) Überführen Sie auf der gegenüberliegenden Seite den oben dargestellten Datenbestand in ein relationales Tabellenmodell, das der dritten Normalform genügt. Geben Sie alle Beziehungen mit Kardinalitäten an. Kennzeichnen Sie Primärschlüssel mit (PK) und Fremdschlüssel mit (FK).

21 Punkte

Fortsetzung 4. Handlungsschritt →

## **Fortsetzung 4. Handlungsschritt**

Korrekturrand

- b) Eine nicht mehr benötigte medizinische Leistung soll gelöscht werden. Nach Ausführung des korrekten DELETE-Statements erscheint folgende Fehlermeldung:

Cannot delete or update a parent row: a foreign key constraint fails

Erläutern Sie die Ursache für diese Fehlermeldung.

4 Punkte

---

---

---

---

## 5. Handlungsschritt (25 Punkte)

Die Belegung der Zimmer und Betten mit Patienten ist in einer Datenbank erfasst (siehe perforierte Anlage).

Erstellen Sie für die gegebene Datenbank nachfolgende Abfragen.

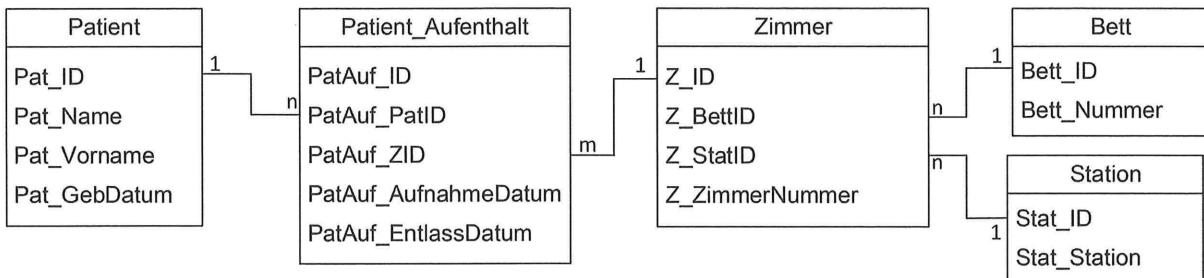
- a) Erstellen Sie eine SQL-Anweisung, mit der Sie alle Patienten aus der Tabelle Patienten dem Alter nach aufsteigend sortiert ausgeben. Sollte es mehrere Patienten mit gleichem Geburtsdatum geben, soll eine weitere Sortierung zuerst nach Namen aufsteigend und dann nach Vornamen absteigend sortiert werden. 5 Punkte

## Ergebnisbeispiel:

Zimmer Abfrage		
Pat_Name	Pat_Vorname	Pat_GebDatum
Grenzfeld	Thorsten	04.06.1990
Neuhaus	Anne	01.06.1988
Sardon	Sandra	31.03.1988
Müller	Peter	06.02.1966
Trostan	Jannick	15.02.1957

**Fortsetzung 5. Handlungsschritt auf Seite 13 →**

**Dieses Blatt kann an der Perforation aus dem Aufgabensatz herausgetrennt werden!**



Patient			
Pat_ID	Pat_Name	Pat_Vorname	Pat_GebDatum
1	Müller	Peter	06.02.1966
2	Trostan	Jannick	15.02.1957
3	Sardon	Sandra	31.03.1988
4	Grenzfeld	Thorsten	04.06.1990
5	Neuhaus	Anne	01.06.1988

Zimmer			
Z_ID	Z_BettID	Z_StatID	Z_ZimmerNummer
1	2	1	212
2	3	1	212
3	4	1	214

Patient_Aufenthalt				
PatAuf_ID	PatAuf_PatID	PatAuf_ZID	PatAuf_AufnahmeDatum	PatAuf_EntlassDatum
1	2	2	07.02.2020	24.02.2020
2	1	2	01.02.2020	26.02.2020
3	3	2	26.02.2020	28.02.2020
4	2	3	11.04.2020	30.04.2020
5	4	3	01.05.2020	08.05.2020
6	2	1	02.05.2020	18.05.2020

Bett	
Bett_ID	Bett_Number
1	00347783
2	00448637
3	00358999
4	07785688
5	55800987

Station	
Stat_ID	Stat_Station
1	Innere
2	Kardiologie
3	Onkologie



## **Fortsetzung 5. Handlungsschritt**

## Korrekturrand

- b) Erstellen Sie eine SQL-Anweisung, mit der Sie die Zimmerbelegungen für den Zeitraum Februar 2020 nach folgender Ergebnistabelle auflisten:  
10 Punkte

PatAuf_AufnahmeDatum	PatAuf_EntlassDatum	Dauer	Z_ZimmerNummer	Stat_Station	Bett_Nummer
07.02.2020	24.02.2020	17	212	Innere	00358999
01.02.2020	26.02.2020	25	212	Innere	00358999
26.02.2020	28.02.2020	2	212	Innere	00358999

#### **Fortsetzung 5. Handlungsschritt →**

## **Fortsetzung 5. Handlungsschritt**

Korrekturrand

- c) Erstellen Sie eine SQL-Anweisung, mit der alle freien Betten am 21.04.2020 wie folgt aufgelistet werden: 10 Punkte

<b>Abfrage1</b>
<b>Bett_Nummer</b>
00347783
00448637
07785688
55800987

**PRÜFUNGSZEIT – NICHT BESTANDTEIL DER PRÜFUNG!**

Wie beurteilen Sie nach der Bearbeitung der Aufgaben die zur Verfügung stehende Prüfungszeit?

- 1 Sie hätte kürzer sein können.       2 Sie war angemessen.       3 Sie hätte länger sein müssen.







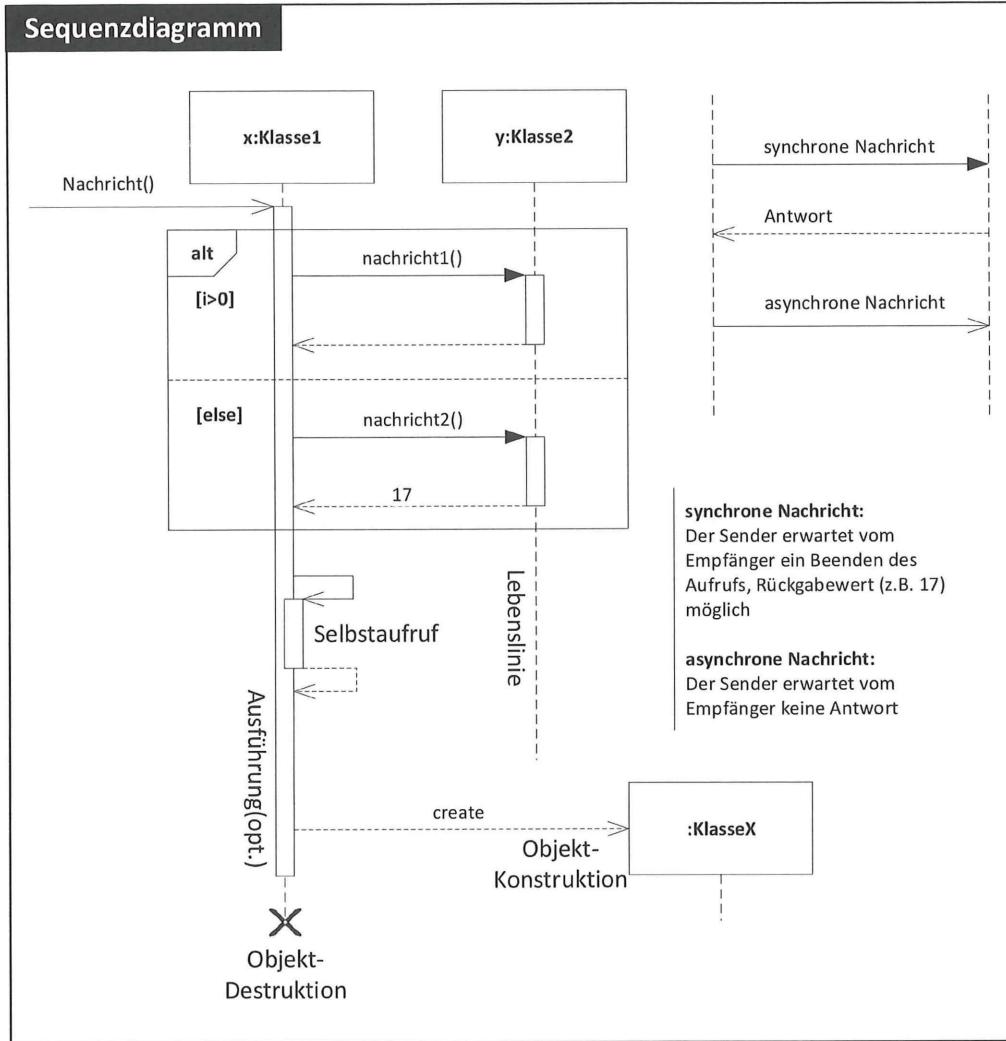
## Belegsatz

Fachinformatiker Anwendungsentwicklung  
Fachinformatikerin Anwendungsentwicklung  
1196

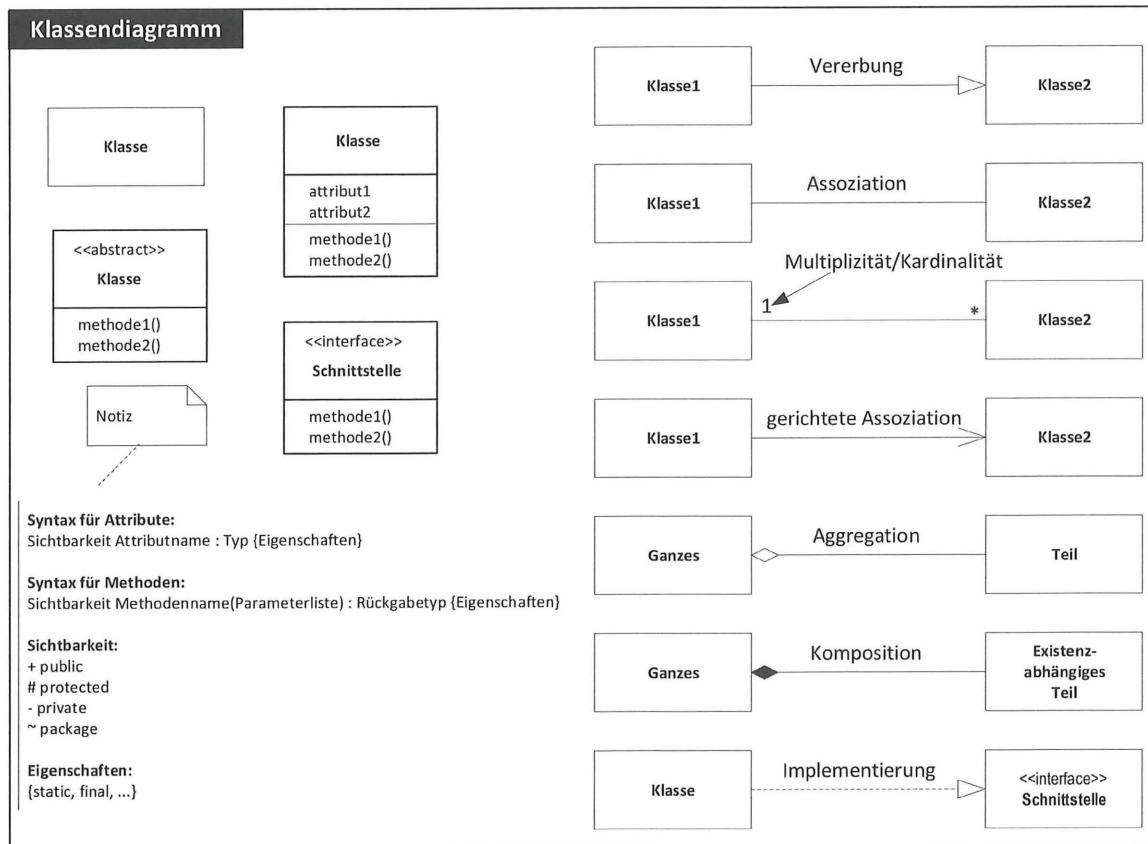
## 1 Ganzheitliche Aufgabe I Fachqualifikationen

UML-Sequenzdiagramm	Seite 2
UML-Klassendiagramm	Seite 2
SQL-Syntax (Auszug)	Seite 3 – 4

## UML-Sequenzdiagramm



## UML-Klassendiagramm



## SQL-Syntax (Auszug)

Syntax	Beschreibung
<b>Tabelle</b>	
<b>CREATE TABLE</b> Tabellenname( Spaltenname < DATENTYP >, Primärschlüssel, Fremdschlüssel)	Erzeugt eine neue leere Tabelle mit der beschriebenen Struktur
<b>ALTER TABLE</b> Tabellenname <b>ADD COLUMN</b> Spaltenname Datentyp <b>DROP COLUMN</b> Spaltenname Datentyp  <b>ADD FOREIGN KEY</b> (Spaltenname) <b>REFERENCES</b> Tabellenname( Primärschlüsselspaltenname )	Änderungen an einer Tabelle: Hinzufügen einer Spalte Entfernen einer Spalte  Definiert eine Spalte als Fremdschlüssel
<b>CHARACTER</b>	Textdatentyp
<b>DECIMAL</b>	Numerischer Datentyp (Festkommazahl)
<b>DOUBLE</b>	Numerischer Datentyp (Doppelte Präzision)
<b>INTEGER</b>	Numerischer Datentyp (Ganzzahl)
<b>DATE</b>	Datum (Format DD.MM.YYYY)
<b>PRIMARY KEY</b> (Spaltenname)	Erstellung eines Primärschlüssels
<b>FOREIGN KEY</b> (Spaltenname) <b>REFERENCES</b> Tabellenname( Primärschlüsselspaltenname )	Erstellung einer Fremdschlüssel-Beziehung
<b>DROP TABLE</b> Tabellenname	Löscht eine Tabelle
<i>Befehle, Klauseln, Attribute</i>	
<b>SELECT</b> *   Spaltenname1 [, Spaltenname2, ...]	Wählt die Spalten einer oder mehrerer Tabellen, deren Inhalte in die Liste aufgenommen werden sollen; alle Spalten (*) oder die namentlich aufgeführten
<b>FROM</b>	Name der Tabelle oder Namen der Tabellen, aus denen die Daten der Ausgabe stammen sollen
<b>SELECT ...</b> <b>(SELECT ...</b> <b>FROM ...</b> <b>WHERE ...) AS xyz</b> <b>FROM ...</b> <b>WHERE ...</b>	Unterabfrage, die in eine äußere SELECT-Anweisung geschachtelt ist. Das Ergebnis der Unterabfrage wird im Spaltenausdruck (z. B. hier: xyz) ausgegeben.
<b>SELECT DISTINCT</b>	Eliminiert Redundanzen, die in einer Tabelle auftreten können, Werte werden jeweils nur einmal angezeigt.
<b>INNER JOIN</b>	Liefert nur die Datensätze zweier Tabellen, die gleiche Datenwerte enthalten
<b>LEFT JOIN / LEFT OUTER JOIN</b>	Liefert von der erstgenannten (linken) Tabelle alle Datensätze und von der zweiten Tabelle jene, deren Datenwerte mit denen der ersten Tabelle übereinstimmen
<b>RIGHT JOIN / RIGHT OUTER JOIN</b>	Liefert von der zweiten (rechten) Tabelle alle Datensätze und von der ersten Tabelle jene, deren Datenwerte mit denen der zweiten Tabelle übereinstimmen
<b>FULL JOIN</b>	Liefert aus beiden Tabellen jeweils alle Datensätze
<b>WHERE</b>	Bedingung, nach der Datensätze ausgewählt werden sollen
<b>WHERE EXISTS ( subquery )</b> <b>WHERE NOT EXISTS ( subquery )</b>	Die Bedingungen EXISTS prüft, ob die Suchbedingung einer Unterabfrage mindestens eine Zeile zurückliefert. NOT EXIST negiert die Bedingung.
<b>GROUP BY</b> Spaltenname1 [, Spaltenname2, ...]	Gruppierung (Aggregation) nach Inhalt des genannten Feldes
<b>ORDER BY</b> Spaltenname1 [, Spaltenname2, ...] <b>ASC   DESC</b>	Sortierung nach Inhalt des genannten Feldes oder der genannten Felder ASC: aufsteigend; DESC: absteigend
<b>Syntax</b>	<b>Beschreibung</b>
<i>Datenmanipulation</i>	
<b>DELETE FROM</b> Tabellenname	Löschen von Datensätzen in der genannten Tabelle
<b>UPDATE</b> Tabellenname <b>SET</b>	Aktualisiert Daten in Feldern einer Tabelle
<b>INSERT INTO</b> Tabellenname <b>VALUES</b> (Wert für Spalte 1 [, Wert für Spalte 2, ...]) oder <b>SELECT ... FROM ... WHERE</b>	Fügt Datensätze in die genannte Tabelle, die entweder mit festen Werten belegt oder Ergebnis eines SELECT-Befehls sind

Fortsetzung →

## SQL-Syntax (Auszug) – Fortsetzung

<b>Aggregatfunktionen</b>	
<b>AVG(Spaltenname)</b>	Ermittelt das arithmetische Mittel aller Werte im angegebenen Feld
<b>COUNT(Spaltenname   * )</b>	Ermittelt die Anzahl der Datensätze mit Nicht-NULL-Werten im angegebenen Feld oder alle Datensätze der Tabelle (dann mit Operator *)
<b>SUM(Spaltenname   Formel)</b>	Ermittelt die Summe aller Werte im angegebenen Feld oder der Formelergebnisse
<b>MIN(Spaltenname   Formel)</b>	Ermittelt den kleinsten aller Werte im angegebenen Feld
<b>MAX (Spaltenname   Formel)</b>	Ermittelt den größten aller Werte im angegebenen Feld
<b>Funktionen</b>	
<b>LEFT(Zeichenkette, Anzahlzeichen)</b>	Liefert Anzahlzeichen der Zeichenkette von links.
<b>RIGHT(Zeichenkette, Anzahlzeichen)</b>	Liefert Anzahlzeichen der Zeichenkette von rechts.
<b>CURRENT</b>	Liefert das aktuelle Datum mit der aktuellen Uhrzeit
<b>CONVERT(time,[DatumZeit])</b>	Liefert die Uhrzeit aus einer DatumZeit-Angabe
<b>DATE(Wert)</b>	Wandelt einen Wert in ein Datum um
<b>DAY(Datum)</b>	Liefert den Tag des Monats aus dem angegebenen Datum
<b>MONTH(Datum)</b>	Liefert den Monat aus dem angegebenen Datum
<b>TODAY</b>	Liefert das aktuelle Datum
<b>WEEKDAY(Datum)</b>	Liefert den Tag der Woche aus dem angegebenen Datum
<b>YEAR(Datum)</b>	Liefert das Jahr aus dem angegebenen Datum
<b>DATEADD(Datumsteil, Intervall, Datum)</b>	Fügt einem Datum ein Intervall (ausgedrückt in den unter Datumsteil angegebenen Einheiten) hinzu
<b>DATEDIFF(Datumsteil, Anfangsdatum, Enddatum)</b> Datumsteile: <b>DAY, MONTH, YEAR</b>	Liefert Enddatum-Startdatum (ausgedrückt in den unter Datumsteil angegebenen Einheiten)
<b>Operatoren</b>	
<b>AND</b>	Logisches UND
<b>LIKE</b>	Überprüfung von Textattributen auf Gleichheit, Verwendung von Platzhaltern möglich.
<b>NOT</b>	Logische Negation
<b>OR</b>	Logisches ODER
<b>=</b>	Test auf Gleichheit
<b>&gt;, &gt;=, &lt;, &lt;=, &lt; &gt;</b>	Test auf Ungleichheit
<b>*</b>	Multiplikation
<b>/</b>	Division
<b>+</b>	Addition, positives Vorzeichen
<b>-</b>	Subtraktion, negatives Vorzeichen

Stand 2018-03-29