

1

Ganzheitliche Aufgabe I Fachqualifikationen

Allgemeine Korrekturhinweise

Die Lösungs- und Bewertungshinweise zu den einzelnen Handlungsschritten sind als Korrekturhilfen zu verstehen und erheben nicht in jedem Fall Anspruch auf Vollständigkeit und Ausschließlichkeit. Neben hier beispielhaft angeführten Lösungsmöglichkeiten sind auch andere sach- und fachgerechte Lösungsalternativen bzw. Darstellungsformen mit der vorgesehenen Punktzahl zu bewerten. Der Bewertungsspielraum des Korrektors (z. B. hinsichtlich der Berücksichtigung regionaler oder branchenspezifischer Gegebenheiten) bleibt unberührt.

Zu beachten ist die unterschiedliche Dimension der Aufgabenstellung (nennen – erklären – beschreiben – erläutern usw.). Wird eine bestimmte Anzahl verlangt (z. B. „Nennen Sie fünf Merkmale ...“), so ist bei Aufzählung von fünf richtigen Merkmalen die volle vorgesehene Punktzahl zu geben, auch wenn im Lösungshinweis mehr als fünf Merkmale genannt sind. Bei Angabe von Teilpunkten in den Lösungshinweisen sind diese auch für richtig erbrachte Teileleistungen zu geben.

In den Fällen, in denen vom Prüfungsteilnehmer

- keiner der fünf Handlungsschritte ausdrücklich als „nicht bearbeitet“ gekennzeichnet wurde,
- der 5. Handlungsschritt bearbeitet wurde,
- einer der Handlungsschritte 1 bis 4 deutlich erkennbar nicht bearbeitet wurde,

ist der tatsächlich nicht bearbeitete Handlungsschritt von der Bewertung auszuschließen.

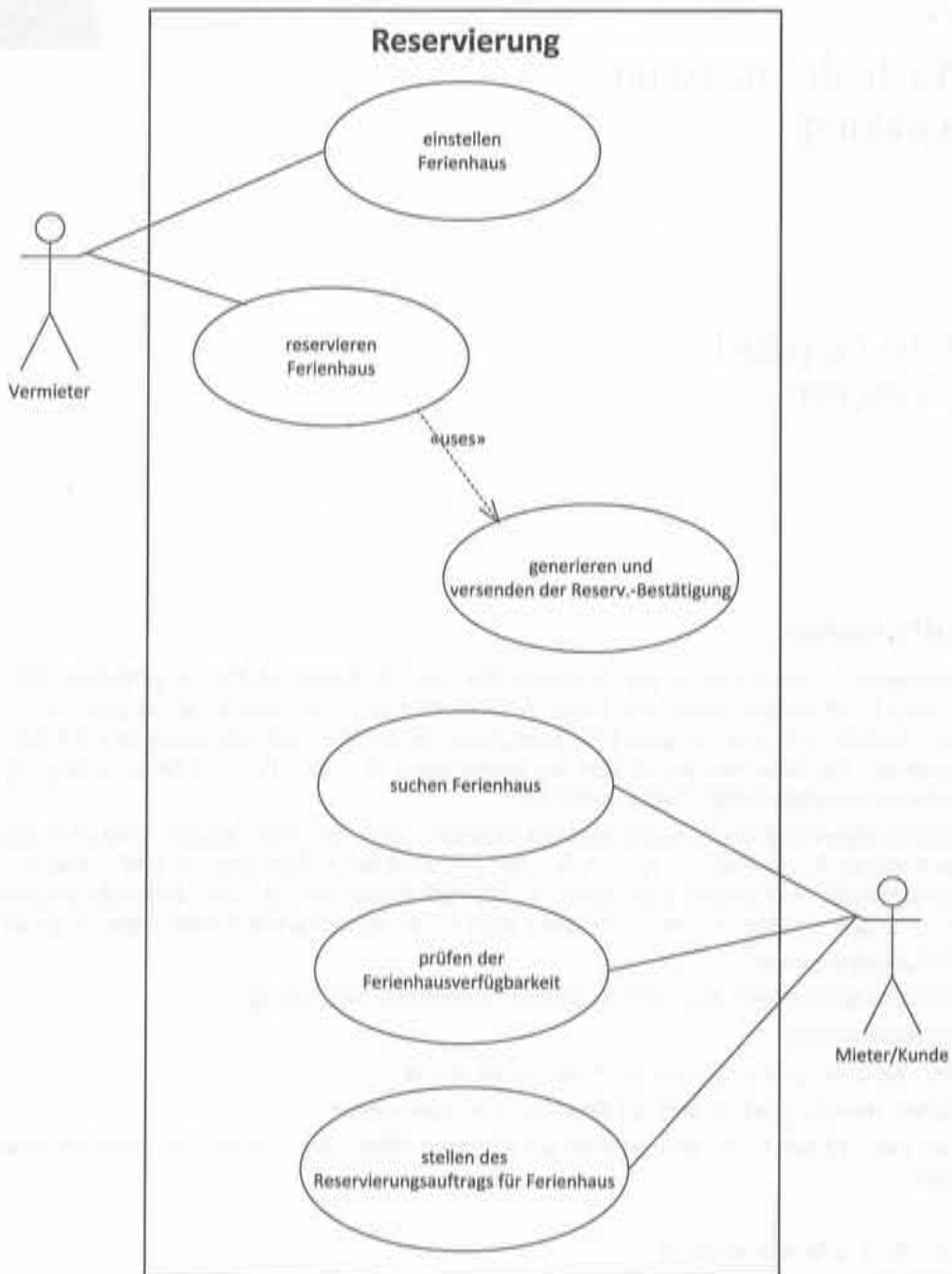
Ein weiterer Punktabzug für den bearbeiteten 5. Handlungsschritt soll in diesen Fällen allein wegen des Verstoßes gegen die Formvorschrift nicht erfolgen!

Für die Bewertung gilt folgender Punkte-Noten-Schlüssel:

Note 1 =	100 – 92 Punkte	Note 2 =	unter	92 – 81 Punkte
Note 3 =	unter 81 – 67 Punkte	Note 4 =	unter	67 – 50 Punkte
Note 5 =	unter 50 – 30 Punkte	Note 6 =	unter	30 – 0 Punkte

1. Handlungsschritt (25 Punkte)

a) 12 Punkte



b) 13 Punkte

getHolidayEstates(String destination, int persons, int rooms, double maxPrice,
date arrival, int duration)

Beginn Methode

residences = createList(); //Erzeugt Objekte residences vom Datentyp List;
vaccancies = createList(); //Erzeugt Objekt vaccancies vom Datentyp List;

residences = getEstates(destination);
numberResidences = Anzahl der Einträge in Liste residences;
k = 1; //Merker, ob Objekte verfügbar sind!

Beginn Zählschleife

von i = 1 bis i = numberResidences erhöhe i um 1

Beginn 1. Verzweigung //Überprüfung, ob Objekt residences[i] an der Stelle i verfügbar
Wenn

persons = getPersons(residences[i]) &
rooms = getRooms(residences[i]) &
maxPrice >= getPrice(residences[i]) &
getVaccancies(arrival, duration, residences[i]) = true

dann
vaccancies.add(residences[i]); //gefundenen Objekt in Liste vaccancies einfügen
k = k + 1;

Ende 1. Verzweigung

Ende Zählschleife

Beginn 2. Verzweigung

Wenn

k = 1

dann
Ausgabe: "Kein passendes Objekt verfügbar";
Ende der Methode getHolidayEstates();

sonst
gebe Objekt vaccancies zurück;

Ende 2. Verzweigung

Ende Methode getEstates()

getHolidayEstates()

Werte destination, persons, rooms, maxPrice, arrival und duration einlesen

Objekte residences und vaccancies vom Datentyp List erzeugen

residences[] = getEstates(destination)

numberResidences = Anzahl der Einträge in residences[]

k = 1

von i = 1 bis i = numberResidences erhöhe i um 1

wahr	persons = getPersons(residences[i]) & rooms = getRooms(residences[i]) & maxPrice >= getPrice(residences[i]) & getVaccancies(arrival, duration, residences[i]) = true	falsch

residences[i] in vaccancies einfügen

k = k + 1

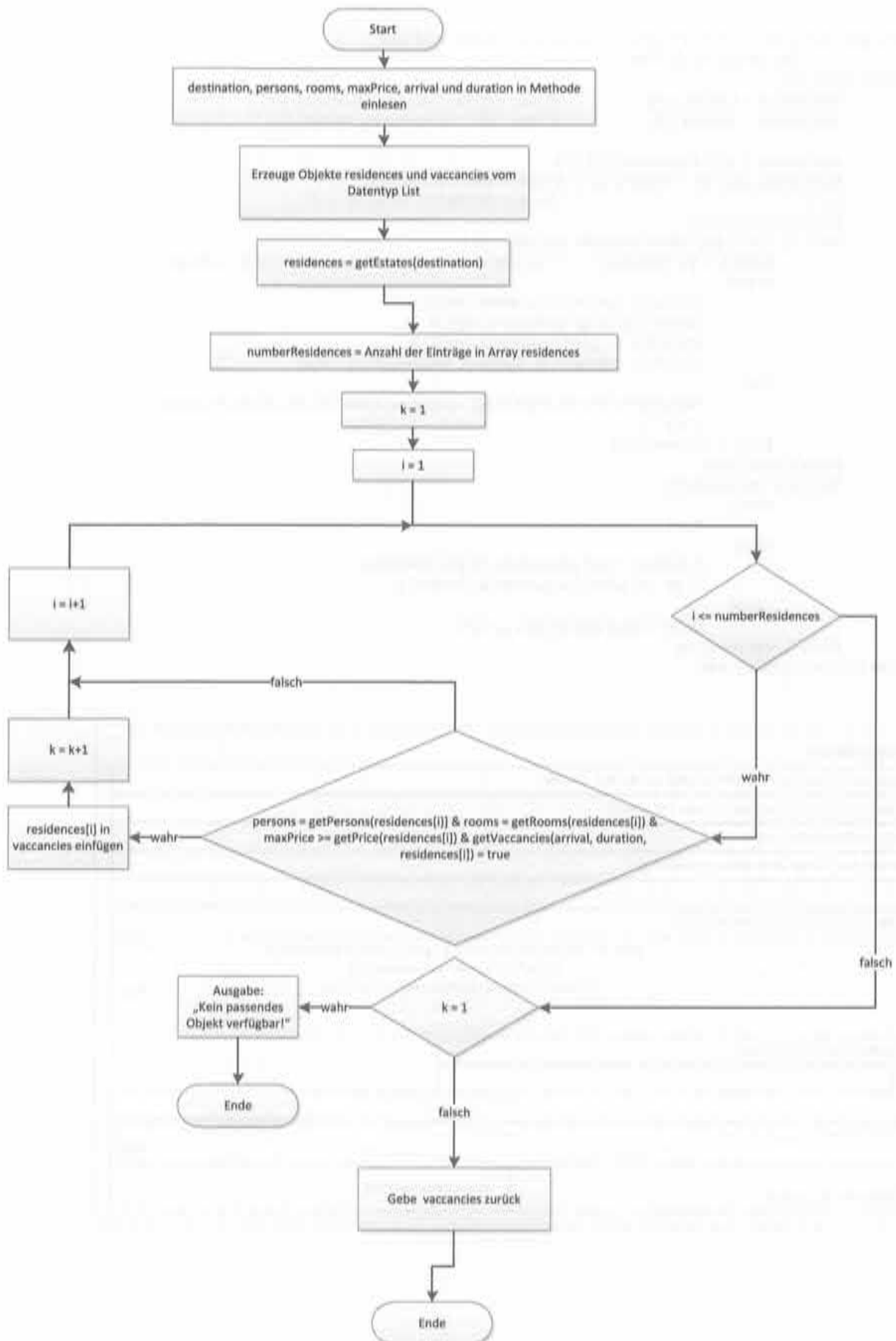
wahr

k = 1

falsch

Ausgabe:
Kein passendes Objekt verfügbar

gebe vaccancies zurück



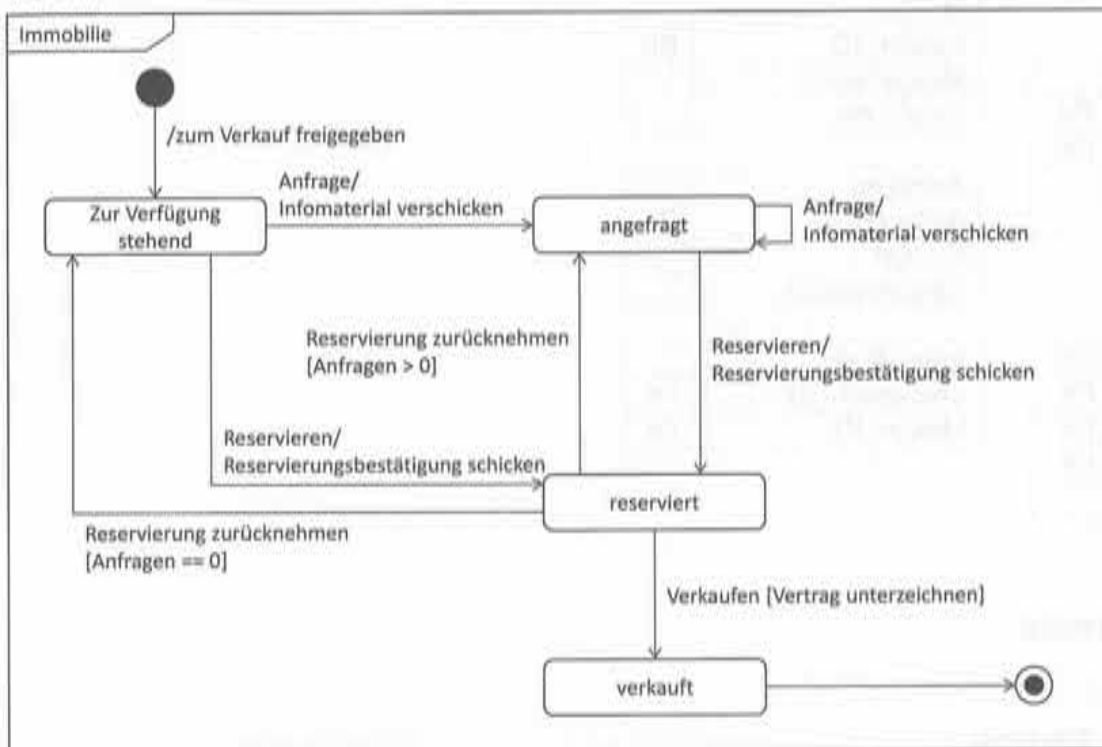
2. Handlungsschritt (25 Punkte)

Alternative Lösung möglich

Hinweis:

Der Prüfling soll zeigen, dass er die Notation eines UML-Zustandsdiagramms anwenden kann und nicht, dass er mit beliebigen Zeichen den beschriebenen Sachverhalt darstellen kann.

a) 13 Punkte



b) 12 Punkte,

3 x 4 Punkte: 1 Punkt je Beziehungstyp
1 Punkt je Klassendiagramm
2 Punkte je Begründung

Beschreibung	Beziehungstyp	Klassendiagramm	Begründung
Eine Immobilie besteht aus mehreren Wohnungen.	Komposition		Eine Immobilie besteht aus Wohnungen und eine Wohnung kann nur mit der Immobilie existieren.
Bewohner können entweder Mieter oder Eigentümer sein.	Vererbung		Mieter und Eigentümer haben gemeinsame Attribute/Methoden, die in einer Basisklasse Bewohner zusammengefasst werden können.
In einer Mietervereinigung gibt es mehrere Mieter.	Aggregation		Eine Mietervereinigung besteht aus Mietern, die aber auch unabhängig von der Vereinigung bestehen können.

3. Handlungsschritt (25 Punkte)

Alternative Lösung möglich.

Immobilie	
Immobilien_ID	PK
Adresse_ID	FK
Eigentuermer_ID	FK
Beschreibung	

Kunde	
Kunde_ID	PK
Adresse_ID	FK
oder	
PLZ-Ort	
Straße-Hausnr.	

Besuchstermin	
BT_ID	PK
Kunde_ID	FK
Immobilien_ID	FK
Makler_ID	FK
DatumUhrzeit	

Eigentuermer	
Eigentuermer_ID	PK
Adresse_ID	FK

Makler	
Makler_ID	PK
MaklerName	
TelefonNr	

Adresse	
Adresse_ID	PK
PLZOrt	
StraßeHausnr.	

ImmoMak	
Immobilien_ID	FK
Makler_ID	FK

4. Handlungsschritt (25 Punkte)

Alternative Lösungen möglich

Nebenkosten(wurzel: Element)

```
hausliste : NodeList  
haus : Node  
kostenhaus : double  
kostenliste : NodeList  
kosten : Node  
kostenwert : double
```

```
hausliste := wurzel.getElementsByTagName("Haus")  
für i = 0 , hausliste.getLength()-1, 1
```

```
    haus = hausliste.item(i)  
    Ausgabe "Haus: ", haus.getAttributes().item(0).getNodeValue()
```

```
    kostenhaus := 0  
    kostenliste = haus.getChildNodes()  
    für j = 0, kostenliste.getLength()-1, 1  
        kosten := kostenliste.item(j)  
        kostenwert := kosten.getFirstChild().getNodeValue()  
        kostenhaus := kostenhaus + kostenwert  
        Ausgabe kosten.getNodeName(), ": ", kostenwert  
    ende für j  
    Ausgabe "Haus-Gesamtkosten: ", kostenhaus
```

```
ende für i
```

5. Handlungsschritt (25 Punkte)

Alternative Lösungen möglich

a) 5 Punkte

```
SELECT TOP 1 Ferienhaus.*,  
      ( SELECT SUM(Tage)  
        FROM Mietvertrag  
        WHERE Ferienhaus.Ferienhaus_ID = Mietvertrag.Ferienhaus_ID ) AS sumTg  
FROM Ferienhaus ORDER BY sumTg DESC
```

b) 5 Punkte

```
SELECT Kunde.*  
FROM Kunde  
WHERE 0 = ( SELECT COUNT(Mietvertrag_ID) FROM Mietvertrag WHERE  
           Mietvertrag.Kunde_ID = Kunde.Kunde_ID)
```

c) 5 Punkte

```
SELECT Ferienhaus.*,  
(  
  ( SELECT COUNT(Mietvertrag. Mietvertrag_ID)  
    FROM Mietvertrag  
    WHERE Ferienhaus.Ferienhaus_ID = Mietvertrag.Ferienhaus_ID ),  
  ( SELECT COUNT(Maengelanzeige.Maengelanzeige_ID)  
    FROM Maengelanzeige  
    WHERE Ferienhaus.Ferienhaus_ID = Maengelanzeige.Ferienhaus_ID )  
) AS Wert  
FROM Ferienhaus ORDER BY Wert
```

d) 5 Punkte

```
SELECT Ferienhaus.Ferienhaus_Id, SUM(Mietvertrag.Tage) AS Tg  
FROM Ferienhaus  
LEFT JOIN Mietvertrag ON Mietvertrag.Ferienhaus_ID = Ferienhaus.Ferienhaus_ID  
GROUP BY Ferienhaus.Ferienhaus_Id ORDER BY Tg DESC
```

e) 5 Punkte

```
SELECT Ferienhaus.*,  
      ( SELECT COUNT(Mietvertrag.Tage)  
        FROM Mietvertrag  
        WHERE Mietvertrag.Ferienhaus_ID = Ferienhaus.Ferienhaus_ID ) AS ANZAHL  
FROM Ferienhaus  
WHERE  
      ( SELECT COUNT(Mietvertrag.Tage)  
        FROM Mietvertrag  
        WHERE Mietvertrag.Ferienhaus_ID = Ferienhaus.Ferienhaus_ID ) / 3.65 < 50  
ORDER BY ANZAHL DESC
```