Abschlussprüfung Winter 2023/24
Lösungshinweise

Fachinformatiker/Fachinformatikerin
Anwendungsentwicklung (AO 2020)
1201

2 Entwicklung und Umsetzung von Algorithmen

Teil 2 der Abschlussprüfung

Allgemeine Korrekturhinweise

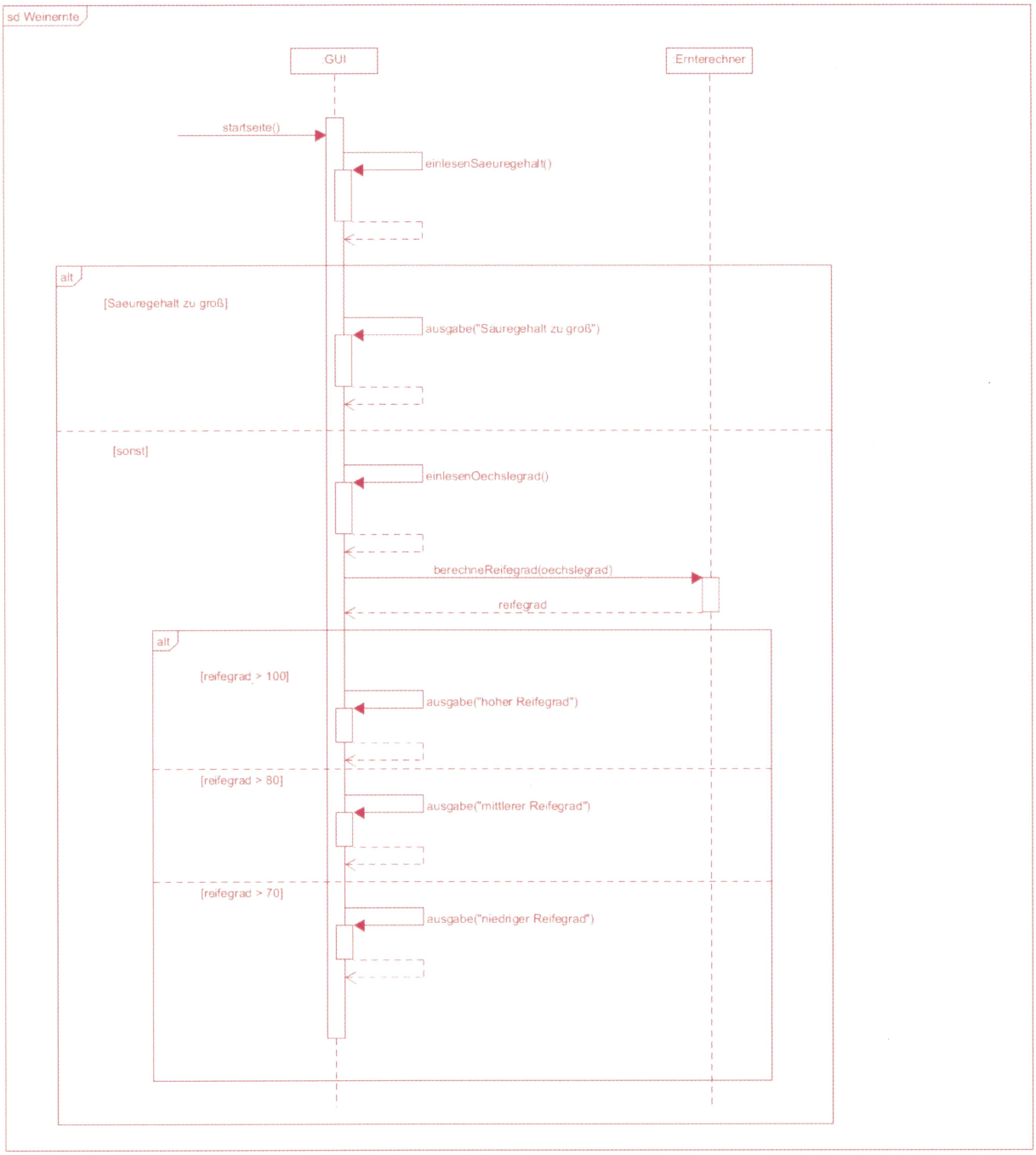
Die Lösungs- und Bewertungshinweise zu den einzelnen Handlungsschritten sind als Korrekturhilfen zu verstehen und erheben nicht in jedem Fall Anspruch auf Vollständigkeit und Ausschließlichkeit. Neben hier beispielhaft angeführten Lösungsmöglichkeiten sind auch andere sach- und fachgerechte Lösungsalternativen 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.).

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. Aufgabe (25 Punkte)

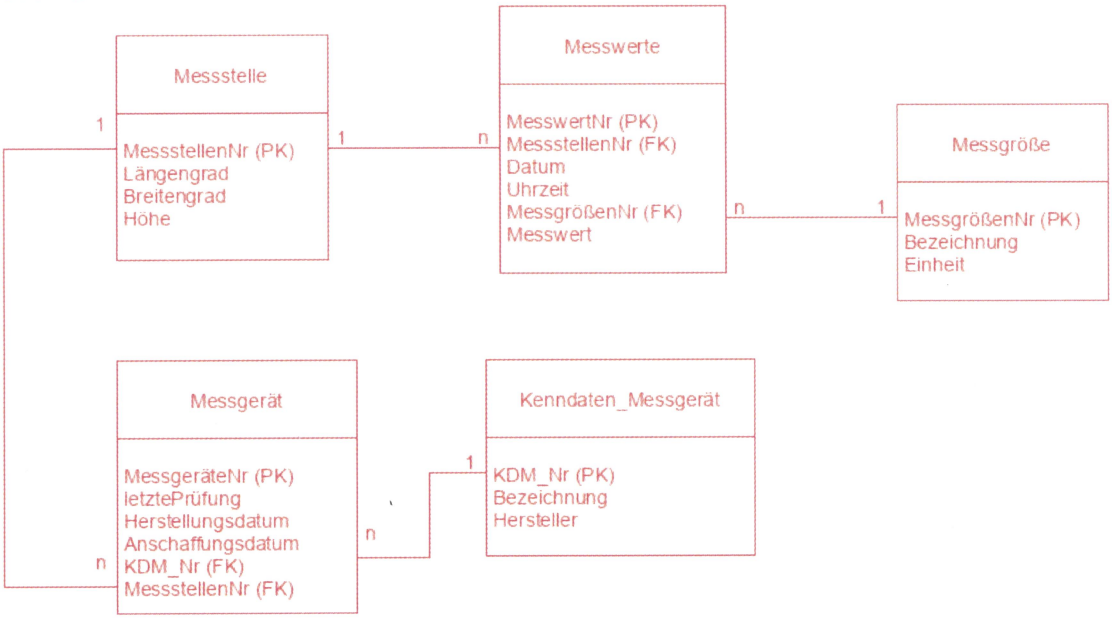


2. Aufgabe (25 Punkte)

```
calculateFlight(geoPositions: GeoPos[] ) : GeoPos[]

flight_positions = new GeoPos[geoPositions.length]  // Array der angeflogenen
                                                    // Positionen
flightPositions[0] = geoPositions[0]                // Position 0 als Startposition
currentPos = geoPositions[0]
geoPositions.remove(0)                              // aus dem Array der noch
                                                    // anzufliegenden Positionen
                                                    // entfernen

next = 1                                        // nächster Index in flightPositions
while (geoPositions.length > 0)                 // Solange noch Positionen anzufliegen sind
       minDistance = Double.MAX_VALUE
       minPos = 0
       for i = 0 to geoPositions.length -1
              distance = geoCalculator.getDistance(currentPos, geoPositions[i])
              if distance < minDistance
                     minDistance = distance
                     minPos = i
              end if
       end for

       flightPositions[next] = geoPositions[minPos]
       next = next + 1
       currentPos = geoPositions[minPos]
       geoPositions.remove(minPos)
end while

return flightPositions

end calculateFlight
```

3. Aufgabe (25 Punkte)

a) 18 Punkte



Je Tabelle mit allen Attributen 1 Punkt (5 Punkte)
Je Primärschlüssel 1 Punkt (5 Punkte)
Je Beziehung mit Kardinalität 1 Punkt (4 Punkte)
Je Fremdschlüssel 1 Punkt (4 Punkte)

b) 7 Punkte

Hinweis: Rundungsdifferenzen können im Ergebnis berücksichtigt werden.
20.000.000 Pixel/Bild * 8 Bit/Farbkanal * 3 Farbkanal/Pixel = 480.000.000 Bit/Bild

480.000.000 Bit/Bild / 8 Bit/Byte = 60.000.000 Byte/Bild / 1.024 / 1.024 /1.024
= 0,05587 GiB/Bild

0,05587 GiB/Bild * 200 Bilder/Woche * 52 Wochen = **581,14 GiB -> 582 GiB**

4. Aufgabe (25 Punkte)

a) 4 Punkte

```
SELECT *
FROM Kunde
WHERE Kd_PLZ LIKE '508%' OR Kd_PLZ LIKE '509%' OR Kd_PLZ LIKE '51%'
```

b) 5 Punkte

```
SELECT Art_Jahrgang AS Jahrgang
      ,MIN(Art_Preis) AS NiedrigsterPreis
      ,MAX(Art_Preis) AS HöchsterPreis
      ,COUNT(Art_IdKey) AS 'Anzahl der Weine je Jahrgang'
FROM Artikel
GROUP BY Art_Jahrgang
ORDER BY Jahrgang DESC;
```

c) 8 Punkte

```
Update      RechnungPosition Set RgPos_RabattProzent = 12
FROM RechnungPosition AS RP
            INNER JOIN Rechnung AS R
                  ON R.Rg_IdKey = RP.RgPos_RgIdKey
            INNER JOIN Artikel AS A
                  ON A.Art_IdKey = RP.RgPos_ArtIdKey
            INNER JOIN WeinArt AS WA
                  ON WA.WA_IdKey = A.Art_WAIdKey
WHERE (Month(R.Rg_Datum) = 5 AND YEAR(R.Rg_Datum) = 2023) AND
      RP.RgPos_RabattProzent = 0 AND
      WA.WA_Weinart = 'Rotwein';
```

d) 8 Punkte

```
SELECT Kd_IdKey, Kd_Firma, Kd_Strasse, Kd_PLZ, Kd_Ort, Kd_Nummer,
      (SELECT AVG(RgPos_Einzelpreis * RgPos_Mg)
            FROM RechnungPosition AS RP
                  INNER JOIN Rechnung AS R ON
                        R.Rg_IdKey = RP.RgPos_RgIdKey
            WHERE Kd_IdKey = R.Rg_KdIdKey) AS Umsatz
FROM Kunde;
```