

Musteraufgaben für die Abschlussprüfung im Ausbildungsberuf Mathematisch-technische Softwareentwickler

Prüfungsbereich "Entwicklung eines Softwaresystems"

Vorbemerkung

Die vorliegende bundeseinheitliche Prüfungsaufgabe wird in drei Phasen bearbeitet.

Phase I:

- Schriftliche Klausur unter Aufsicht des Prüfungsausschusses der IHK
- Sie soll in der Regel montags stattfinden, Dauer 7 Stunden
- Es sind keine Hilfsmittel zugelassen.
- Die Ergebnisse sind handschriftlich auf Papier zu erstellen.
- Das Original wird dem Prüfungsausschuss übergeben, eine Durchschrift behält der Prüfling zur weiteren Bearbeitung.

Phase II:

- Die Bearbeitung erfolgt am betrieblichen Arbeitsplatz.
- Sie findet von dienstags bis freitags statt.
- Verwendete Hilfsmittel und Quellen sind anzugeben.
- Die Ergebnisse sind auf Papier und elektronisch lesbar nach Vorgabe des Prüfungsausschusses abzugeben.
- Hinzuzufügen ist auch eine Eigenständigkeitserklärung.

Phase III:

- Das Fachgespräch findet zeitnah als Einzelprüfung mit dem Prüfungsausschuss statt.
- Der Prüfling soll das Prüfungsprodukt, die Aufgabenanalyse und den Lösungsentwurf in maximal 10 Minuten vorstellen und begründen. Hilfsmittel wie Flip Chart, Folien o.ä. können verwendet werden.
- Im anschließenden etwa 20-minütigen Gespräch sind die Ergebnisse zu verteidigen.

Problemstellung

Ein Servicetechniker betreut Kunden, die an verschiedenen Orten wohnen (diese werden mit Buchstaben, mit B beginnend, bezeichnet) und die er täglich genau einmal besuchen muss. Danach kehrt er wieder an seinen Startort (A) zurück. Er benötigt für die Planung seiner Rundreise ein Programm, das ihm eine möglichst kurze Reiseroute berechnet. Man kann davon ausgehen, dass er von jedem Ort zu jedem Ort gelangen kann. Allerdings kann die Reiselänge zwischen zwei Orten von der durchlaufenen Richtung abhängen (z. B. Einbahnstraßen).

Die Aufgabenstellung des Programms ist es, eine kurze Rundreise zu finden, bei der alle Orte einmal besucht werden und die am Startort endet. Die Zahl möglicher Rundreisen ist schon für kleine n immens hoch, nämlich $n!$. Da das Durchprobieren aller Möglichkeiten wegen der Rechenzeit nicht möglich ist, soll eine Näherung für die optimale Lösung nach dem folgenden heuristischen Verfahren gefunden werden.

Lösungsansatz

Ausgangspunkt des Verfahrens ist die quadratische Entfernungsmatrix der Dimension $(0:n;0:n)$. Sie enthält die Entfernungen zwischen dem Startort und den n Kunden bzw. zwischen den Kunden. Sie ist unsymmetrisch und voll besetzt, die Diagonalelemente sind gleich 0.

In der i . Zeile stehen die Entfernungen von i . Kunden zu den anderen Kunden und in der j . Spalte stehen die Entfernungen von allen anderen Kunden zu Kunde Nr. j . Die 0. Zeile bzw. die 0. Spalte gehören zum Startort.

Grundlegend für den Algorithmus sind folgende Überlegungen:

- Da jeder Kunde (auch der Anfangsort) einmal besucht werden muss, ist die Summe der zu jedem Kunden jeweils kürzesten Hinwege (von einem anderen Kunden her) sicher eine untere Schranke für den Gesamtweg.
- Andererseits muss jeder Kunde mindestens einmal verlassen werden. Daher bildet die Summe der kürzesten Wege von jedem Ort weg (zu einem anderen Kunden) ebenfalls eine untere Schranke.
- Das Maximum der beiden Werte bildet eine untere Schranke für die Weglänge. Dieser Wert stellt unser Auswahlkriterium dar.

Die gesuchte Route wird nun Schritt für Schritt aufgebaut:

Man startet mit dem Anfangsort und kann jeden Kunden als nächstes besuchen. Bei Besuch von z. B. Kunde i ergibt sich folgendermaßen eine eingeschränkte Entfernungsmatrix aus der ursprünglichen Matrix:

Die Entfernungen vom Startort zu den anderen Kunden spielen keine Rolle mehr (0. Zeile, außer Eintrag i), wie auch die Entfernungen von den anderen Kunden zum Kunden i (i . Spalte, außer 0. Eintrag) und vom Kunden i zum Startort (Eintrag i in Spalte 0). Sie werden gleich Null gesetzt.

...

Mit dieser neuen Matrix wird das Auswahlkriterium (Bewertung) berechnet. Dies wiederholt man für jeden Kunden und entscheidet sich dann für denjenigen Kunden, für den das Auswahlkriterium zum niedrigsten Wert führt. Bei Gleichheit nimmt man den zuerst Gefundenen.

Analog verfährt man Schritt für Schritt, wenn man schon einige Kundenbesuche festgelegt hat, mit den noch nicht besuchten Kunden, bis man alle bis auf einen Kunden besucht hat. Dieser ist automatisch der letzte und von dort kehrt man zum Startort zurück.

Die zugrundeliegende Matrix ändert sich von Schritt zu Schritt, denn legt man einen nächsten Kunden fest, so

- werden alle anderen Wege, die vom vorigen Kunden wegführen, gestrichen, da man jeden Kunden nur einmal verlässt.
- werden alle anderen Wege zu dem neuen Kunden gestrichen (gleich 0 gesetzt), da man zu jedem Kunden genau einmal gelangt.
- wird der Weg zum Startort gestrichen, außer bei dem letzten Kunden.
- bleibt nach $n-1$ Schritten eine Matrix mit je einem Eintrag pro Zeile und Spalte übrig.

Schreiben Sie in einer der zugelassenen Programmiersprachen ein Programm, das eine beliebige Entfernungsmatrix einliest, die Rundreise berechnet und das Ergebnis (siehe Beispieloutput, letzte Seite) ausgibt. Zur Verarbeitung mehrerer Beispiele ist ein geeignetes Skript (Shell- oder Batch-) vorzusehen.

Zur näheren Erläuterung des Verfahrens ein konkretes Beispiel:

Eingabe/Startmatrix:

```
0 10 15 20
5  0  9 10
6 13  0 12
8  8  9  0
```

In diesem Beispiel ergibt die Summation der kleinsten positiven (von Null verschiedenen) Einträge pro Zeile (Zeilenminima)

$$Z_{weg} = 10 + 5 + 6 + 8 = 29.$$

Bei der Summation der kleinsten positiven (von Null verschiedenen) Einträge pro Spalte (Spaltenminima) erhält man

$$Z_{hin} = 5 + 8 + 9 + 10 = 32.$$

Es kann keinen kürzeren Rundweg als $\max(Z_{weg}, Z_{hin})$ geben.

Man erhält als Bewertung den Wert 32.

...

Die nächsten Schritte beinhalten die Bewertung der Kunden, die unmittelbar auf den Startort folgen. Da A der Startort ist, werden die Wege AB, AC und AD genannt. Wir beginnen mit AB. Die eingeschränkte Entfernungsmatrix lautet dann:

	A	B	C	D
A	0	10	0	0
B	0	0	9	10
C	6	0	9	12
D	8	0	9	0

Bei der neuen Bewertung (bzw. Tabelle) hat man $Z_{\text{weg}} = 10 + 9 + 6 + 8 = 33$ (Zeilenminima) und $Z_{\text{hin}} = 6 + 10 + 9 + 10 = 35$ (Spaltenminima), also den Wert 35.

Entsprechend geht es weiter. Die zu AC gehörende Matrix lautet:

	A	B	C	D
A	0	0	15	0
B	5	0	0	10
C	0	13	0	12
D	8	8	0	0

Man erhält $Z_{\text{weg}} = 15 + 5 + 12 + 8 = 40$ (Zeilenminima) und $Z_{\text{hin}} = 5 + 8 + 15 + 10 = 38$ (Spaltenminima), also als Wert 40.

Entsprechend geht man vor bei AD. Die zugehörige Matrix lautet:

	A	B	C	D
A	0	0	0	20
B	5	0	9	0
C	6	13	0	0
D	0	8	9	0

Hier sind $Z_{\text{weg}} = 20 + 5 + 6 + 8 = 39$ und $Z_{\text{hin}} = 5 + 8 + 9 + 20 = 42$, also 42 als Wert.

Da der durch AB definierte Weg den kleinsten Wert hat, geht es mit ihm weiter. (Hätte man mehrere Wege mit gleichem Wert, soll man den nach dem Alphabet ersten auswählen.) Die Wege mit höheren Werten werden im Folgenden nicht mehr betrachtet.

Die Matrix für den Weg ABC lautet:

	A	B	C	D
A	0	10	0	0
B	0	0	9	0
C	0	0	0	12
D	8	0	0	0

Man erhält $Z_{\text{weg}} = Z_{\text{hin}} = 10 + 9 + 12 + 8 = 39$ als Wert.

Die Matrix für ABD lautet:

	A	B	C	D
A	0	10	0	0
B	0	0	0	10
C	6	0	0	0
D	0	0	9	0

mit dem Wert $Z_{weg} = Z_{hin} = 10 + 10 + 6 + 9 = 35$.

Als Ergebnis hat man schließlich ABDCA mit der Länge 35.

Aufgabenstellung für die Prüfung

Es ist ein Programm zu entwickeln, das anhand des beschriebenen Algorithmus eine Annäherung an die kürzeste Rundreise bei vorgegebener Entfernungsmatrix berechnet.

Als Maximalzahl soll $n=20$ angenommen werden. Die Entfernung der Orte wird durch positive Ganzzahlen angegeben. Die Hauptdiagonale der Matrix soll Nullen enthalten. Zur Bezeichnung der Orte wird das deutsche Alphabet in Großbuchstaben verwendet. Start- und Zielort ist A. Die Matrix soll zeilenweise eingelesen werden. Für jeden untersuchten Weg ist die untersuchte Matrix und die Bewertung des Weges anzugeben. Als Ergebnis ist die durch den Algorithmus vorgeschlagene Rundreise und die dazugehörige Rundreiselänge anzugeben.

Im Rahmen der schriftlichen Aufgabe sind am ersten Tag abzugeben:

- Aufgabenanalyse
- Verbale Verfahrensbeschreibung (Algorithmus)
- Datenstrukturen in Form von UML-Diagrammen
- Programmkonzeption (Klassen, Methoden)
- Funktionale Trennung nach dem Drei-Schichten-Modell sowie der Definition der Schnittstellen
- Detaillierte Beschreibung der Methoden in Form von Nassi-Shneiderman-Diagrammen

Im Rahmen des Prüfprodukts sind abzugeben:

- Programmsystem (bestehend aus Klassen, Methoden und Skripten) als Quellcode und in ausführbarer Form.
- Quellcode gedruckt und als Datei
- Entwicklerdokumentation und Benutzeranleitung
- Beschreibung, Begründung und Diskussion des angegebenen Beispiels und eine ausreichende Zahl von Beispielen, die sowohl die Normalfälle als auch auftretende Spezialfälle abdecken (Ein- und Ausgabe gedruckt und als Dateien, bei mehr als 2 Seiten Output nur in elektronischer Form)
- Eigenständigkeitserklärung

Im Rahmen des auftragsbezogenen Fachgesprächs sind die Aufgabenanalyse und der Lösungsentwurf zu begründen und das Prüfungsprodukt zu erläutern.

...

Eingabe/Startmatrix in dem Beispiel (Dimension in der i . Zeile)

```

4
0 10 15 20
5  0  9 10
6 13  0 12
8  8  9  0

```

Ausgabe:

Startmatrix:

```

0 10 15 20
5  0  9 10
6 13  0 12
8  8  9  0

```

mit Bewertung: $\max(29, 32) = 32$

naechster Ort: B

```

0 10  0  0
0  0  9 10
6  0  0 12
8  0  9  0

```

mit Bewertung: $\max(33, 35) = 35$

naechster Ort: C

```

0  0 15  0
5  0  0 10
0 13  0 12
8  8  0  0

```

mit Bewertung: $\max(40, 38) = 40$

naechster Ort: D

```

0  0  0 20
5  0  9  0
6 13  0  0
0  8  9  0

```

mit Bewertung: $\max(39, 42) = 42$

also von A nach B

naechster Ort: C

```

0 10  0  0
0  0  9  0
0  0  0 12
8  0  0  0

```

mit Bewertung: $\max(39, 39) = 39$

naechster Ort: D

0 10 0 0

0 0 0 10

6 0 0 0

0 0 9 0

mit Bewertung: $\max(35, 35) = 35$

also von B nach D

Rundreise:

von A

ueber B

ueber D

ueber C

nach A

Entfernung: 35

Bewertungshinweise

Insgesamt 100 Punkte

1. Schriftliche Aufgabe (30 Punkte)

Aufgabenanalyse	6 P.
Verbale Verfahrensbeschreibung	6 P.
Datenstrukturen in form von UML-Diagrammen	6 P.
Programmkonzeption (Klassen, Methoden)	4 P.
Funktionale Trennung nach dem Drei -Schichten- Modell sowie Definition der Schnittstellen	2 P.
Detaillierte Beschreibung der Methoden in Form von Nassi -Shneiderman-Diagrammen	6 P.

**2. Prüfungsprodukt und 3. Auftragsbezogenes Fachgespräch
(70 Punkte)**

Analyse des Auftrags	10 P.
- thematische Durchdringung und klare Herausstellung des Auftragsziels	
Umsetzung des Programmsystems	25 P.
- Abgleich des Konzeptentwurfs mit der Realisierung	
- Realisierung der internen Daten	
- Implementierung von Algorithmen	
- Erstellung einer lauffähigen Version	
Testen und Qualitätssicherungsmaßnahmen	
- Testbeispiele: Beschreibungen, Begründung	15 P.
Entwicklerdokumentation	10 P.
- Moduldefinition, Schnittstellendokumentation, Klassendefinitionen in UML	
Benutzeranleitung	5 P.
durchgehende Verwendung von Fachbegriffen	5 P.