

Abiturprüfung 2024

INFORMATIK

Arbeitszeit: 180 Minuten

Der Fachausschuss wählt je eine Aufgabe aus den Gebieten
Inf1 und Inf2 zur Bearbeitung aus.

Der Fachausschuss ergänzt im folgenden Feld die erlaubten
objektorientierten Programmiersprachen:

--

Die Angabe ist vom Prüfling mit dem Namen zu versehen und abzugeben.

Name des Prüflings
--

BE

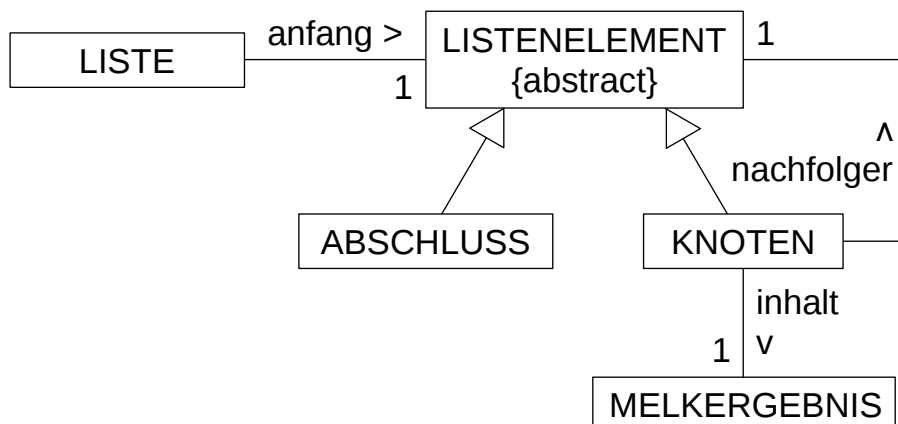
Es soll eine Verwaltungssoftware für Milchviehbetriebe entwickelt werden. Damit soll jede Kuh mit ihrer Chipnummer und ihrem Geburtsdatum gespeichert werden.

1. Die Software soll eine Klasse SYSTEM besitzen, die die Daten der Milchproduktion und die oben genannten Daten der Kühe verwaltet. Die Kühe werden an Melkstationen automatisch gemolken. Dabei wird vom System für jeden Melkvorgang ein Melkergebnis-Objekt erzeugt. In einem solchen Melkergebnis-Objekt werden Datum und Uhrzeit, die Milchmenge (in Liter), die Anzahl an Keimen (in Tausend pro Milliliter) sowie die Nummer der zugehörigen Melkstation gespeichert. Zudem referenziert ein Melkergebnis-Objekt das zur gemolkenen Kuh gehörende Objekt.

5

- a) Zeichnen Sie für die beschriebene Situation ein Klassendiagramm, das als Grundlage für die Verwaltungssoftware eines Milchviehbetriebs verwendet werden kann. Nutzen Sie dazu die Klassen SYSTEM, KUH und MELKERGEBNIS. Auf die Angabe von Methoden kann verzichtet werden.

Bei der Softwareentwicklung wird entschieden, die Melkergebnisse chronologisch in einer einfach verketteten Liste gemäß dem folgenden Klassendiagramm zu speichern:



(Fortsetzung nächste Seite)

BE	
2	b) Nennen Sie einen Vorteil und einen Nachteil der Implementierung als Liste in der konkreten Anwendungssituation.
6	c) Die Kuh mit der Chipnummer 237 wird um 8:43 Uhr an der Melkstation 4 gemolken (12,3 Liter, 72 Tausend Keime pro ml). Am selben Tag um 7:25 Uhr wird die Kuh mit der Chipnummer 645 an Melkstation 2 (8,3 Liter, 95 Tausend Keime pro ml) und um 15:32 Uhr an Melkstation 4 (7,1 Liter, 165 Tausend Keime pro ml) gemolken. Stellen Sie die beschriebene Situation auf Grundlage des gegebenen Klassendiagramms in einem Objektdiagramm dar. Gehen Sie davon aus, dass sich in der Liste keine weiteren Melkergebnisse außer den genannten befinden. Beschränken Sie sich bei den Attributen auf diejenigen, zu denen ein Wert angegeben ist.
16	d) Ist die durchschnittliche Keimanzahl einer Melkstation zu hoch, kann dies auf eine mögliche Verunreinigung dieser Melkstation hinweisen. Deshalb soll eine Methode <i>durchschnittlicheKeimanzahlBerechnen(nr)</i> zur Verfügung stehen, die die durchschnittliche Keimanzahl (in Tausend pro Milliliter) für die Melkstation mit der übergebenen Nummer <i>nr</i> zurückgibt. Außerdem sollen sämtliche Melkergebnisse einer bestimmten Kuh aus der Liste gelöscht werden können. Dazu soll es eine Methode <i>melkergebnisseLöschen(chipnr)</i> geben, die die Melkergebnisse der Kuh mit der übergebenen Chipnummer <i>chipnr</i> aus der Liste entfernt. Notieren Sie in einer auf dem Deckblatt angegebenen Programmiersprache eine mögliche Implementierung der beiden oben genannten Methoden in der Klasse LISTE und aller dafür benötigten Methoden in den Klassen der Listenstruktur. Wenden Sie so weit wie möglich das Prinzip der Rekursion an. Gehen Sie davon aus, dass die Methoden zum Geben und Setzen von Attributwerten bzw. Referenzen in den Klassen MELKERGEBNIS und KUH bereits implementiert sind und zu jeder Melkstation mindestens ein Melkergebnis in der Liste gespeichert ist.

(Fortsetzung nächste Seite)

- e) Die folgende Tabelle repräsentiert eine Liste mit Melkergebnissen, die zu Kühen mit der angegebenen Chipnummer gehören:

Chipnummer	Milchmenge	Anzahl Keime
972	8,1	87
241	6,9	207
972	9,2	147
241	7,3	86
241	6,3	181
241	7,1	133
972	7,9	96
241	8,1	101
972	7,7	112
241	6,3	71

Gegeben sind folgende Methoden *m1* und *m2*:

In der Klasse LISTE:

```
Methode m1(chipnr)
    gib anfang.m2(chipnr, 0, 0) zurück
endeMethode
```

In der Klasse KNOTEN:

```
Methode m2(chipnr, p1, p2)
    wenn chipnr gleich inhalt.kuh.chipnr dann
        wenn inhalt.anzahlKeime > 100 dann
            p2 = p2 + 1
            wenn p1 < p2 dann
                p1 = p2
            endeWenn
        sonst
            p2 = 0
        endeWenn
    endeWenn
    gib nachfolger.m2(chipnr, p1, p2) zurück
endeMethode
```

(Fortsetzung nächste Seite)

In der Klasse ABSCHLUSS:

```
Methode m2(chipnr, p1, p2)
    gib p1 zurück
endeMethode
```

Geben Sie für die Methodenaufrufe $m1(972)$ und $m1(241)$ jeweils sämtliche darauffolgende Aufrufe der Methode $m2$ der Reihe nach an. Geben Sie darüber hinaus die Rückgabewerte von $m1(972)$ und $m1(241)$ an.

Interpretieren Sie den Rückgabewert der Methode $m1$ im Sachzusammenhang und gehen Sie dabei auf die Bedeutung der Parameter $p1$ und $p2$ ein.

2. Zur Verwaltung der Objekte der Klasse KUH wird ein geordneter Binärbaum verwendet. Als Schlüssel dient dabei die für jede Kuh eindeutige Chipnummer.

3 a) Es werden Kuh-Objekte mit den folgenden Chipnummern der Reihe nach in einen zunächst leeren Binärbaum eingefügt:

405 – 502 – 104 – 324 – 426 – 410 – 299 – 345 – 501

Zeichnen Sie den Baum, der dabei entsteht. Bezeichnen Sie die Knoten des Baums mit den gegebenen Chipnummern.

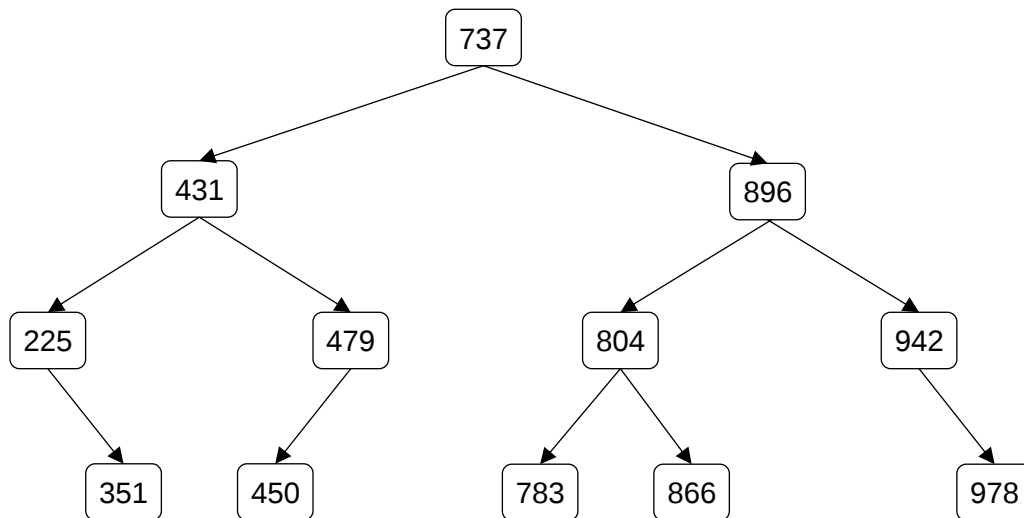
6 b) Begründen Sie, warum der in Teilaufgabe 2a entstandene Baum hinsichtlich der Suche nach einer bestimmten Chipnummer nicht optimal ist.

Zeichnen Sie einen für die Suche optimal aufgebauten Baum für dieselben Kuh-Objekte wie in Teilaufgabe 2a. Geben Sie begründet an, wie viele verschiedene Möglichkeiten es für einen solchen Baum gibt. Geben Sie außerdem an, welche Chipnummern für die Wurzel dafür infrage kommen.

2 c) Ein Großbetrieb besitzt 1200 Kühe. Ermitteln Sie, wie viele Ebenen der Baum für diese Anzahl an Kühen mindestens haben muss.

(Fortsetzung nächste Seite)

Im Folgenden ist ein Baum mit zwölf Knoten gegeben. Dabei werden die Knoten nur mit den entsprechenden Chipnummern der Kühe dargestellt.

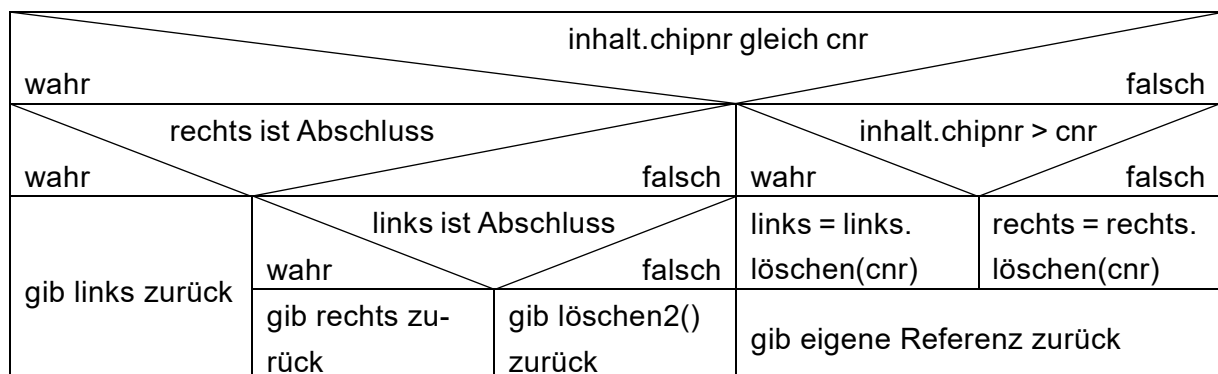


4

d) Geben Sie die Ausgabe einer Preorder-Traversierung an und beschreiben Sie eine mögliche Anwendung dieser Traversierungsart.

Bei der Implementierung des geordneten Binärbaums wurde das Entwurfsmuster Kompositum mit den Klassen BAUM, BAUMELEMENT, ABSCHLUSS und KNOTEN verwendet. Im Attribut *links* der Klasse KNOTEN wird die Referenz auf den linken Nachfolger und im Attribut *rechts* die Referenz auf den rechten Nachfolger gespeichert. Die Trennung von Struktur und Daten ist durch ein Referenzattribut *inhalt* der Klasse KNOTEN realisiert.

Die nachfolgende Methode *löschen(cnr)* der Klasse KNOTEN arbeitet nach dem im Struktogramm dargestellten Algorithmus:



(Fortsetzung nächste Seite)

Die Methode *löschen2* ist dabei wie folgt realisiert:

```
Methode löschen2()  
    vorgänger = eigene Referenz  
    aktueller = rechts  
    wiederhole solange aktueller.links ist nicht Abschluss  
        vorgänger = aktueller  
        aktueller = aktueller.links  
    endeWiederhole  
    vorgänger.links = aktueller.links  
    aktueller.links = links  
    aktueller.rechts = rechts  
    gib aktueller zurück  
endeMethode
```

8

e) Für den oben abgebildeten Baum mit Wurzelknoten *wurzel* werden nacheinander folgende Anweisungen ausgeführt:

- i. *wurzel* = *wurzel.löschen*(866)
- ii. *wurzel* = *wurzel.löschen*(942)
- iii. *wurzel* = *wurzel.löschen*(431)

Zeichnen Sie den nach Ablauf der drei Anweisungen resultierenden Baum und erläutern Sie anhand dieser drei Anweisungen anschaulich die grundsätzliche Vorgehensweise der Methode *löschen(cnr)*.

(Fortsetzung nächste Seite)

3. Die Milchbauern eines Dorfes helfen sich gegenseitig. Man kann die gegenseitige Hilfeleistung für einen bestimmten Zeitraum als Graph modellieren, in dem die Knoten die Milchbauern und die Kantengewichte die Summe der geleisteten Arbeitsstunden darstellen. Zum Beispiel ergibt sich für das Jahr 2023 ein Graph mit folgender Adjazenzmatrix:

	A	B	C	D	E	F
A	-	15	-	-	-	-
B	-	-	8	-	3	-
C	-	-	-	2,5	6	1,5
D	3	-	-	-	-	-
E	5	0,5	-	-	-	-
F	-	-	-	-	-	-

Milchbauer A hat beispielsweise Milchbauer B insgesamt 15 Stunden geholfen.

- 5 a) Zeichnen Sie den durch die Adjazenzmatrix repräsentierten Graphen und geben Sie zwei Eigenschaften des Graphen an.

Der Graph wird mithilfe einer Klasse GRAPH implementiert, in der die Knoten in einem Feld *knoten*, die Anzahl der Knoten im Attribut *anzahl* und die Adjazenzmatrix in einem zweidimensionalen Feld *matrix* gespeichert sind.

In einem Knoten soll im Attribut *name* der Name des Milchbauern und in einem Attribut *bilanz* die Differenz aus der Zeit, die er geholfen hat, und der Zeit, in der er Hilfe in Anspruch genommen hat, gespeichert werden. Der jeweilige Wert des Attributs *bilanz* wird für alle Milchbauern mithilfe einer Methode *bilanzBerechnen()* in der Klasse GRAPH aktualisiert.

Gehen Sie davon aus, dass die Methoden zum Geben und Setzen von Attributwerten in der Klasse KNOTEN bereits implementiert sind. Wenn zwischen zwei Knoten keine Kante vorhanden ist, ist an der entsprechenden Stelle in der Adjazenzmatrix der Wert 0 eingetragen.

- 7 b) Geben Sie in einer auf dem Deckblatt angegebenen Programmiersprache eine mögliche Implementierung der Methode *bilanzBerechnen* an.

(Fortsetzung nächste Seite)

BE
7
80

- c) Im Dorf ist es Brauch, dass am Ende des Jahres der Milchbauer mit der höchsten Bilanz von demjenigen mit der niedrigsten Bilanz zum Essen eingeladen wird.

Implementieren Sie in einer auf dem Deckblatt angegebenen Programmiersprache eine Methode *jahresabschlussAusgeben*, die auf der Konsole ausgibt, welcher Milchbauer welchen zum Essen einlädt. Vereinfachend können Sie davon ausgehen, dass es mindestens zwei Milchbauern gibt und am Ende des Jahres keine zwei Milchbauern dieselbe Bilanz haben.

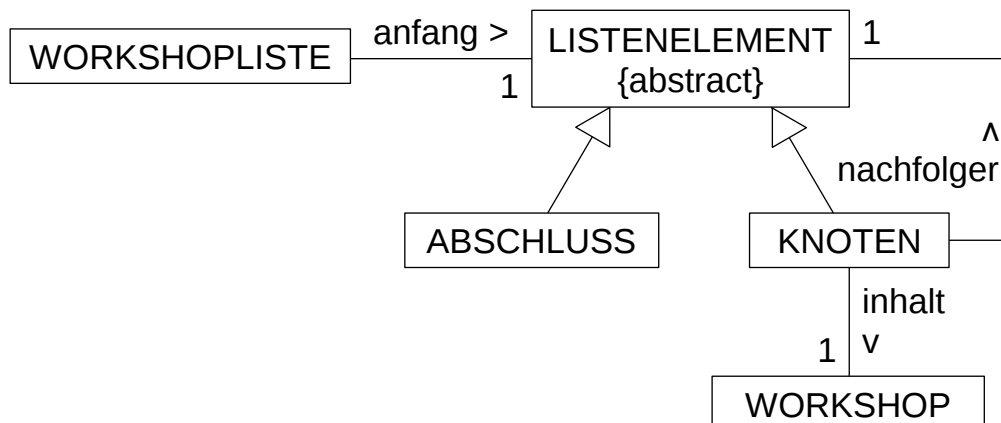
II.

BE

Jährlich im Oktober findet europaweit die Code Week statt, die mit praxisorientierten, eintägigen Workshops das Verständnis für die digitale Welt fördern und Menschen aus ganz Europa zusammenbringen soll.

Im Folgenden wird davon ausgegangen, dass die Code Week mithilfe eines Softwaresystems organisiert wird, bei dem sich die Teilnehmenden mit ihrem Namen und ihrer E-Mail-Adresse registrieren. Die vom System verwalteten Workshops der Code Week haben jeweils einen Titel und eine maximale Teilnehmeranzahl. Da alle Workshops im Oktober stattfinden, wird außerdem vom Datum nur der Tag des Workshops gespeichert. Jeder Workshop findet an genau einem Ort statt, von dem jeweils der Name und die Adresse bekannt sind. Jeder Veranstaltungsort gehört zu genau einer regionalen Organisationseinheit, dem sogenannten Regio-Hub. Jeder Regio-Hub hat einen Namen und eine eindeutige Nummer. Registrierte Teilnehmende können sich für beliebig viele Workshops anmelden.

- 8
1. Entwickeln Sie für die beschriebene Situation ein Klassendiagramm, das unter anderem eine Klasse SYSTEM enthält. Auf die Angabe von Methoden kann verzichtet werden.
 2. Die Workshops der Code Week werden in einer einfach verketteten, nach dem Veranstaltungstag aufsteigend sortierten Liste gemäß folgendem Klassendiagramm verwaltet:



- 2
- a) Nennen Sie zwei Kriterien, die in diesem Kontext für die Datenstruktur Liste gegenüber der Datenstruktur Feld sprechen.

(Fortsetzung nächste Seite)

BE	
3	b) Beschreiben Sie, wie die Trennung von Struktur und Daten im Klassendiagramm umgesetzt wird. Geben Sie zwei mit diesem Konzept verbundene Vorteile an.
8	c) Die Klasse WORKSHOPLISTE soll die Methode <i>sortiertEinfügen(workshopNeu)</i> besitzen, die einen Workshop sortiert in die Liste einfügt. Notieren Sie in einer auf dem Deckblatt angegebenen Programmiersprache eine Implementierung der Methode <i>sortiertEinfügen</i> sowie der dafür nötigen Methoden aller weiteren Klassen der Listenstruktur. Wenden Sie so weit wie möglich das Prinzip der Rekursion an. Die Klasse WORKSHOP kann einschließlich der Methoden zum Geben und Setzen von Attributwerten als vollständig implementiert vorausgesetzt werden.
5	d) Das Pascal-Gymnasium in Fortraningen ist ein Veranstaltungsort der Code Week. Dort findet am 10. Oktober ein Workshop mit dem Titel Netzwerke und am 11. Oktober ein Workshop mit dem Titel Löten statt. Stellen Sie die beschriebene Situation in einem Objektdiagramm dar. Gehen Sie dabei davon aus, dass sich in der Workshopliste außer den beiden genannten keine weiteren Workshops befinden. Beschränken Sie sich bei den Attributen auf diejenigen, zu denen ein Wert angegeben ist.
	3. Das Softwaresystem für die Code Week ist im Rahmen eines Softwareprojekts entstanden.
4	a) Softwareprojekte sind in der Regel in Phasen strukturiert. Nennen und beschreiben Sie mindestens vier typische Phasen eines geeigneten Vorgehensmodells.

(Fortsetzung nächste Seite)

6

- b) Die Software beinhaltet auch eine Webseite, auf der man sich für die Workshops anmelden kann. Die Auftraggebenden beschreiben die gewünschte Funktionalität u. a. wie folgt:

Auf der Startseite gelangt man nach Betätigung des Buttons „Zum Programm“ zu einer Seite mit einer Liste aller Workshops. Wählt man einen Workshop aus dieser Liste aus, erscheint eine Seite mit weiteren Informationen wie Veranstaltungsort und Beginn. Drückt man nun auf „Teilnehmen“, so erscheint auf dem Bildschirm ein Formular zur Eingabe der benötigten Daten. Über den Button „Weiter“ öffnet sich, falls alle Pflichtfelder ausgefüllt wurden, eine Bestätigungsseite, auf der eine Übersicht über die eingegebenen Daten angezeigt wird. Klickt man nun auf den Button „Bestätigen“, werden die Daten übermittelt und man gelangt zur Startseite. Darüber hinaus steht auf jeder Seite mit Ausnahme der Startseite ein Button „Zurück“ zur Verfügung, der jeweils zur vorherigen Seite führt.

Erstellen Sie ein dazu passendes Zustandsübergangsdiagramm.

4. Jeder Regio-Hub speichert mit einer eigenen Software die E-Mail-Adresse und die Anzahl an Workshop-Anmeldungen von denjenigen Teilnehmenden, die diesem Regio-Hub zugeordnete Workshops besuchen. Dazu wird ein lexikografisch nach der E-Mail-Adresse geordneter Binärbaum verwendet.

5

- a) Folgende E-Mail-Adressen zusammen mit der Anzahl an Workshop-Anmeldungen sollen in einen zunächst leeren Baum eingefügt werden:

christine@abc.de (1)

felix@abc.de (4)

lydia@lnet.de (1)

nina@lnet.de (4)

paul@abc.de (3)

peter@wind.de (2)

robert@wind.de (1)

tim@lnet.de (3)

tobi@wind.de (1)

Zeichnen Sie einen Baum mit möglichst wenigen Ebenen, der genau die gegebenen Regioteilnehmerdaten enthält. Bezeichnen Sie die Knoten des Baums vereinfachend mit den in den E-Mail-Adressen genannten Vornamen und der Anzahl der Anmeldungen. Geben Sie für diesen Baum eine mögliche Einfügereihenfolge an.

(Fortsetzung nächste Seite)

- b) Für die Workshops eines Regio-Hubs haben sich 987 Personen angemeldet. Bestimmen Sie die Mindestanzahl der Ebenen des zugehörigen Binärbaums. Ermitteln Sie zudem, wie viele weitere Elemente bestenfalls eingefügt werden können, bevor die Anzahl der Ebenen vergrößert werden muss.

Bei der Implementierung des geordneten Binärbaums in der Klasse REGIOTEILNEHMERBAUM kommen wieder das Entwurfsmuster Kompositum sowie das Konzept der Trennung von Struktur und Daten zum Einsatz.

- c) Die Klasse REGIOTEILNEHMERBAUM soll die Methode *anzahlAnmeldungenErhöhen(teilnehmerEmail)* besitzen, die für das Regioteilnehmer-Objekt mit der übergebenen E-Mail-Adresse den Wert des Attributs *anzahlAnmeldungen* um 1 erhöht. Falls ein entsprechendes Regioteilnehmer-Objekt im Baum noch nicht enthalten ist, weil die zugehörige Person noch für keinen Workshop im Regio-Hub angemeldet war, soll ein entsprechendes Objekt erzeugt und sortiert eingefügt werden.

Notieren Sie in einer auf dem Deckblatt angegebenen Programmiersprache eine mögliche Implementierung dieser Methode und aller dazu nötigen Methoden in der Baumstruktur. Wenden Sie dabei so weit wie möglich das Prinzip der Rekursion an und nutzen Sie die Vorteile des geordneten Binärbaums.

Eine Klasse REGIOTEILNEHMER mit den Attributen *email* und *anzahlAnmeldungen* kann einschließlich der Methoden zum Geben und Setzen von Attributwerten als vollständig implementiert vorausgesetzt werden. Ebenso gibt es eine Methode *vergleichenMit(zk)* der Klasse ZEICHENKETTE, die

- eine negative Zahl zurückgibt, falls das ausführende Objekt lexikografisch kleiner als die übergebene Zeichenkette *zk* ist,
- die Zahl 0 zurückgibt, falls beide Zeichenketten übereinstimmen,
- und ansonsten eine positive Zahl zurückgibt.

(Fortsetzung nächste Seite)

d) In den Klassen des Binärbaums sind die Methoden *m1* und *m2* implementiert.

Klasse REGIOTEILNEHMERBAUM:

```
Methode m1(zahl)
    wurzel.m2(zahl)
endeMethode
```

Klasse KNOTEN:

```
Methode m2(zahl)
    linkerNachfolger.m2(zahl)
    wenn inhalt.anzahlAnmeldungen ≥ zahl dann
        ausgeben(inhalt.email)
    endeWenn
    rechterNachfolger.m2(zahl)
endeMethode
```

Klasse ABSCHLUSS:

```
Methode m2(zahl)
endeMethode
```

Notieren Sie die Ausgabe, nachdem *m1(3)* vom Baum aus Teilaufgabe 4a ausgeführt wurde.

Beschreiben Sie allgemein, was beim Aufruf *m1(zahl)* ausgegeben wird.

(Fortsetzung nächste Seite)

5. Eine Besonderheit beim Vergabeverfahren von Workshopplätzen ist, dass der letzte freie Platz eines jeden Workshops prinzipiell erst freigehalten und später im Rahmen einer Restplatzvergabe zugeteilt wird. Interessierte Teilnehmende können sich für diese Restplätze bewerben.

Folgende Adjazenzmatrix repräsentiert den Graphen, der die Bewerbungen der Teilnehmenden Anna (A), Bruno (B), Carl (C) und Daniela (D) auf die Restplätze der Workshops Löten (L), Mikrocontrollerprogrammierung (M), Netzwerke (N) und Objektorientierung (O) vollständig modelliert:

	A	B	C	D	L	M	N	O
A						w	w	
B						w		
C					w			w
D					w			
L								
M								
N								
O								

Wenn eine Kante vorhanden ist, wird *wahr* (w) eingetragen, ansonsten *falsch*. Zur besseren Übersicht wird in obiger Adjazenzmatrix nur w dargestellt.

Zum Beispiel hat sich Anna auf die Restplätze der Workshops Mikrocontrollerprogrammierung und Netzwerke beworben, Bruno dagegen nur auf den Restplatz des Workshops Mikrocontrollerprogrammierung.

- 4 a) Zeichnen Sie den zur gegebenen Adjazenzmatrix gehörenden Graphen. Geben Sie zwei Eigenschaften des Graphen an.

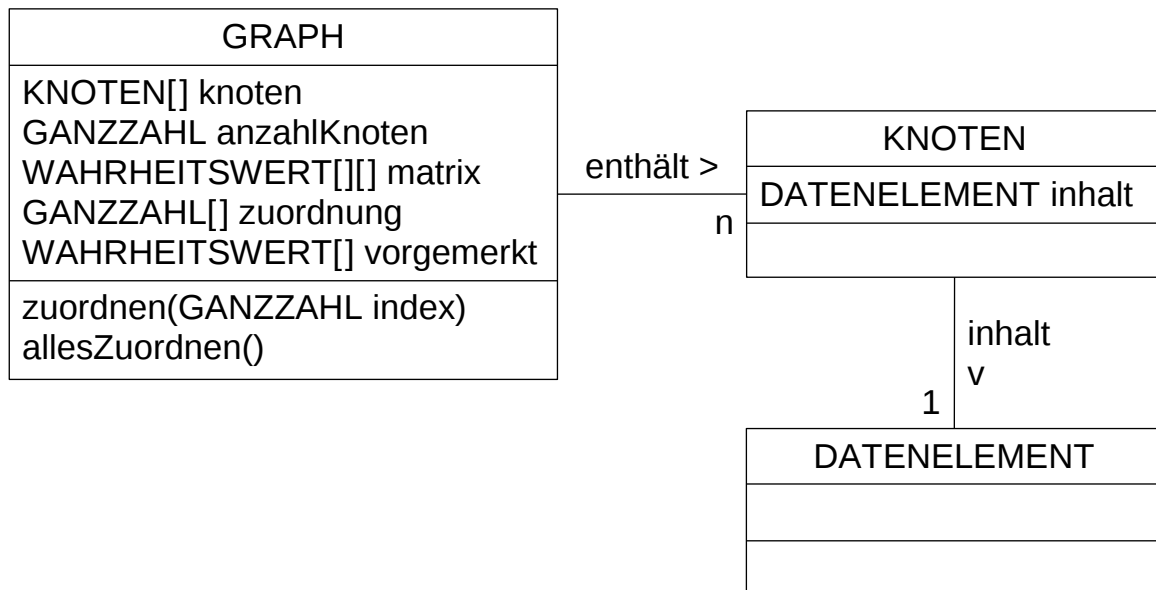
Begründen Sie, dass die Adjazenzmatrix nur im oberen rechten Teil Einträge mit dem Wert *wahr* enthalten kann.

Nun wird eine Zuordnung der Bewerberinnen und Bewerber zu den Workshops ermittelt, bei der möglichst viele von ihnen einen Restplatz erhalten.

- 2 b) Geben Sie für die gegebene Situation eine solche Zuordnung an.

(Fortsetzung nächste Seite)

Der Implementierung des Graphen liegt folgender Ausschnitt des Klassendiagramms zugrunde:



2

- c) Bewerber und Workshops als Datenelemente haben unterschiedliche Attribute. Erweitern Sie das Klassendiagramm so, dass dies berücksichtigt werden kann. Auf die Angabe von Attributen und Methoden kann verzichtet werden.

Der Zuordnungsvorgang soll automatisiert werden. Im Feld *zuordnung* wird die Zuordnung der Bewerberinnen und Bewerber zu den Workshops gespeichert. Wird beispielsweise dem Workshop M (Index 5) der Bewerber A (Index 0) zugeordnet, wird im Feldelement *zuordnung[5]* der Wert 0 gespeichert. Solange den Workshops keine Bewerberinnen bzw. Bewerber zugeordnet wurden, sind alle Feldelemente mit dem Wert -1 vorbelegt.

Das Feld *vorgemerkt* dient der folgenden Methode *zuordnen* zur Speicherung von Zwischenergebnissen. Die Feldelemente des Feldes *vorgemerkt* sind mit dem Wert *falsch* vorbelegt. Die Indizes des Feldes *vorgemerkt* entsprechen denen des Feldes *knoten*.

(Fortsetzung nächste Seite)

Die Methode *zuordnen* ist durch folgenden Pseudocode gegeben:

```

Methode zuordnen(index)
  zähle i von 0 bis anzahlKnoten - 1
    wenn matrix[index][i] gleich wahr und vorgemerkt[i] gleich falsch dann
      vorgemerkt[i] = wahr
      wenn zuordnung[i] gleich -1 dann
        zuordnung[i] = index
        gib wahr zurück
      sonst
        wenn zuordnen(zuordnung[i]) gleich wahr dann
          zuordnung[i] = index
          gib wahr zurück
        endeWenn
      endeWenn
    endeZähle
  gib falsch zurück
endeMethode

```

8

d) Nun wird *zuordnen(0)* aufgerufen.

Geben Sie an, welche Feldelemente von *vorgemerkt* nach Ausführung von *zuordnen(0)* den Wert *wahr* haben, welche Feldelemente von *zuordnung* welchen Wert ungleich -1 haben und welcher Wert zurückgegeben wird.

Anschließend werden alle Feldelemente des Feldes *vorgemerkt* wieder auf *falsch* gesetzt und *zuordnen(1)* wird aufgerufen.

Geben Sie wieder an, welche Feldelemente von *vorgemerkt* nun den Wert *wahr* haben, welche Feldelemente von *zuordnung* welchen Wert ungleich -1 haben und welcher Wert zurückgegeben wird.

5

e) Die Methode *allesZuordnen()* der Klasse GRAPH führt eine Zuordnung durch, indem sie dem Attribut *zuordnung* unter Verwendung der Methode *zuordnen* geeignete Werte zuweist.

Notieren Sie in einer auf dem Deckblatt angegebenen Programmiersprache eine Implementierung der Methode *allesZuordnen*.

III.

BE

1. Die Bayerische Staatsbahn führte zu Beginn des 20. Jahrhunderts folgendes Schema zur Bezeichnung von Lokomotivtypen ein:

- Die Bezeichnung beginnt mit einem Großbuchstaben, wobei S für eine Schnellzuglok, P für eine Personenzuglok, G für eine Güterzuglok und R für eine Rangierlok steht. Falls eine Lok elektrisch betrieben ist, wird zusätzlich ein E vorangestellt.
- Gegebenenfalls folgen zur Angabe weiterer Informationen bis zu drei Kleinbuchstaben: t steht für eine Lok mit Tender, z für eine Zahnradlok und s für eine Schmalspurlok. Falls mehrere Informationen zutreffen, müssen die Kleinbuchstaben in der angegebenen Reihenfolge erscheinen.
- Nach einem Leerzeichen folgt zunächst die Anzahl der angetriebenen Achsen und dann nach einem Schrägstrich die Anzahl der insgesamt vorhandenen Achsen. Jeder Lokomotivtyp hat mindestens zwei und höchstens neun Achsen, von denen mindestens eine und höchstens drei angetrieben werden.

Gültige Bezeichnungen sind demnach z. B. ES 3/6 und Pts 2/2.

Die Menge der nach obiger Beschreibung gültigen Bezeichnungen für Lokomotivtypen bildet die formale Sprache L über dem Alphabet $\{1, 2, \dots, 9, E, G, P, R, S, s, t, z, /, _ \}$, wobei das Zeichen $_$ für das Leerzeichen steht.

4 a) Notieren Sie in einer formalen Textnotation (z. B. in EBNF) die Produktionsregeln einer Grammatik, die L erzeugt.

7 b) Geben Sie ein Zustandsübergangsdiagramm eines endlichen Automaten an, der genau die Wörter der Sprache L akzeptiert.

(Fortsetzung nächste Seite)

2. Zur Überwachung eines potenziell aktiven Vulkans wird ein Netz aus 100 Sensoren installiert. Dabei soll jeder Sensor mit jedem anderen Sensor durch ein Kabel direkt verbunden sein. Zwei Sensoren haben im Mittel einen Abstand von 150 Metern.

2 a) Berechnen Sie die Gesamtlänge der Kabel, die in etwa für dieses Netz aus Sensoren nötig ist.

2 b) Begründen Sie, dass eine derartige Topologie im Sachzusammenhang sinnvoll sein kann.

4 3. Im vollautomatisierten Restaurant „Coffman’s Inn“ arbeiten im Gastraum zwei Servierroboter und in der Küche zwei Küchenroboter. Zwischen Küche und Gastraum gibt es zwei Durchreiche, in denen jeweils maximal sechs Tablett Platz haben. In Durchreiche 1 werden Tablett mit zubereiteten Speisen und Getränken von den Küchenrobotern abgelegt und anschließend von den Servierrobotern zu den Gästen gebracht. In Durchreiche 2 werden Tablett mit abgeräumtem Geschirr von den Servierrobotern abgelegt und anschließend von den Küchenrobotern weggeräumt. Sollte in einer der beiden Durchreichen kein Platz mehr für ein Tablett sein, warten die Servier- bzw. die Küchenroboter so lange, bis wieder ein Platz frei geworden ist.

Erklären Sie, wie es in diesem Szenario zu einer Verklemmung kommen kann, und beschreiben Sie eine Möglichkeit, wie die von Ihnen beschriebene Verklemmung verhindert werden kann.

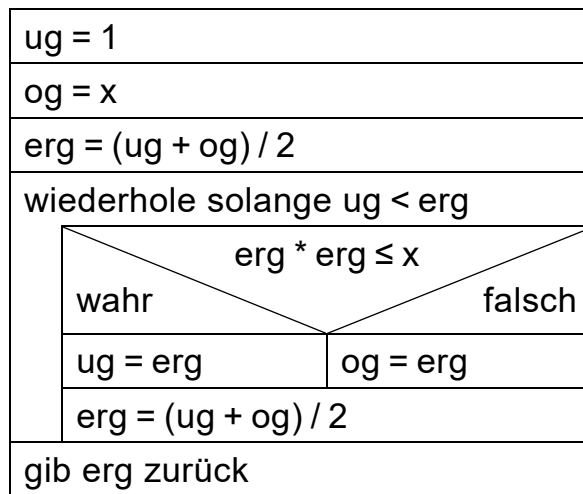
(Fortsetzung nächste Seite)

4. Gegeben ist eine Registermaschine mit folgendem Befehlssatz:

load x	kopiert den Wert aus der Speicherzelle x in den Akkumulator
loadi n	lädt die ganze Zahl n in den Akkumulator
store x	kopiert den Wert aus dem Akkumulator in die Speicherzelle x
add x	addiert den Wert aus der Speicherzelle x zum Wert im Akkumulator
addi n	addiert die ganze Zahl n zum Wert im Akkumulator
sub x	subtrahiert den Wert aus der Speicherzelle x vom Wert im Akkumulator
subi n	subtrahiert die ganze Zahl n vom Wert im Akkumulator
mul x	multipliziert den Wert im Akkumulator mit dem Wert in Speicherzelle x
muli n	multipliziert den Wert im Akkumulator mit der ganzen Zahl n
div x	dividiert den Wert im Akkumulator durch den Wert in Speicherzelle x (ganzzahlige Division)
divi n	dividiert den Wert im Akkumulator durch die ganze Zahl n (ganzzahlige Division)
jmp x	springt zum Befehl in Speicherzelle x
jeq x	springt zum Befehl in Speicherzelle x, falls der Wert im Akkumulator gleich 0 ist
jne x	springt zum Befehl in Speicherzelle x, falls der Wert im Akkumulator ungleich 0 ist
jge x	springt zum Befehl in Speicherzelle x, falls der Wert im Akkumulator positiv oder gleich 0 ist
jle x	springt zum Befehl in Speicherzelle x, falls der Wert im Akkumulator negativ oder gleich 0 ist
jgt x	springt zum Befehl in Speicherzelle x, falls der Wert im Akkumulator positiv ist
jlt x	springt zum Befehl in Speicherzelle x, falls der Wert im Akkumulator negativ ist
hold	beendet die Abarbeitung des Programms

(Fortsetzung nächste Seite)

Gegeben ist der folgende Algorithmus *a1* mit einer positiven ganzen Zahl x als Eingabe:



Hinweis: Der Operator „/“ liefert den ganzzahligen Wert des Quotienten (ohne Rest).

- 4 a) Führen Sie *a1* exemplarisch für die Eingabe $x = 11$ aus und geben Sie dabei tabellarisch an, welche Werte die Variablen *ug*, *og* und *erg* vor jedem Durchlauf der Wiederholung und nach Ablauf des Algorithmus annehmen.
- 7 b) Schreiben Sie ein Programm für die gegebene Registermaschine, das den Algorithmus *a1* umsetzt. Machen Sie deutlich (z. B. durch symbolische Adressierung), in welchen Speicherzellen die Werte von x , *ug*, *og* und *erg* abgelegt werden.

(Fortsetzung nächste Seite)

Gegeben ist das folgende Programm für die gegebene Registermaschine, das einen Algorithmus *a2* umsetzt. Darin sind die Adressen der Speicherzellen, in denen Werte von Variablen abgelegt sind, mit Symbolen bezeichnet, die genauso heißen wie die entsprechenden Variablen. Die mit dem Symbol *x* bezeichnete Speicherzelle steht für eine positive ganze Zahl als Eingabe, die Adresse der Speicherzelle für den Rückgabewert wird mit dem Symbol *erg* bezeichnet.

```
        loadi 0
        store erg
wh:     load  erg
        addi  1
        store tmp
        mul   tmp
        sub   x
        jgt   ende
        load  erg
        addi  1
        store erg
        jmp   wh
ende:    hold
```

- 4 c) Formulieren Sie z. B. in Pseudocode zum gegebenen Algorithmus *a2* eine Methode.
- 2 d) Die beiden Algorithmen *a1* und *a2* liefern bei gleicher Eingabe das gleiche Ergebnis, das jeweils in *erg* gespeichert ist. Geben Sie die mathematische Bedeutung an, die der Wert in *erg* hat.
- 4 e) Um das Laufzeitverhalten der beiden Algorithmen *a1* und *a2* zu vergleichen, wird gezählt, wie oft die Sequenz in der jeweiligen Wiederholung durchlaufen wird.

Geben Sie für beide Algorithmen *a1* und *a2* jeweils an, wie sich die Anzahl der Durchläufe verändert, wenn der Wert von *x* vervierfacht wird. Beurteilen Sie, welchen der beiden Algorithmen Sie für große Werte von *x* vorziehen würden.

IV.

BE

Die Agenten James Blond (007), Jack Black (008) und Charles Brown (009) sind in geheimer Mission unterwegs und tauschen untereinander sowie mit der Zentrale Informationen aus. Ihre Kontaktpersonen in der Zentrale sind der Geheimdienstleiter N, die Entwicklungsleiterin O sowie der Sekretär Mister Moneycent.

Uhrzeiten für geheime Treffen müssen sicher kommuniziert werden können. Dazu wird zunächst ein besonderes Format festgelegt. Man vereinbart, die Uhrzeit in der Form h:mm-d anzugeben. Dabei steht d für das jeweilige Tagesdrittel (1, 2 oder 3), h für die Stundenangabe innerhalb des Tagesdrittels (von 0 bis 8) und mm für die Minutenangabe (von 00 bis 59). Das erste Tagesdrittel beginnt um 00:01 Uhr (dargestellt als 0:01-1) und endet um 08:00 Uhr (dargestellt als 8:00-1). Entsprechendes gilt für das zweite bzw. dritte Tagesdrittel.

Beispielsweise wird 21:45 Uhr als 5:45-3 oder 16:00 Uhr als 8:00-2 dargestellt.

1. Die Menge aller Uhrzeiten im vereinbarten Format bildet die formale Sprache L über dem Alphabet $\{0, 1, \dots, 9, :, -\}$.

6 a) Geben Sie in formaler Textnotation (z. B. in EBNF) die Produktionsregeln einer Grammatik an, die L erzeugt.

7 b) Erstellen Sie das Zustandsübergangsdiagramm eines endlichen Automaten, der genau die Wörter der Sprache L erkennt.

(Fortsetzung nächste Seite)

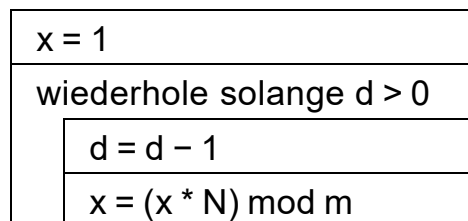
2. Agent Blond übermittelt eine geheime Uhrzeit an Agent Black, indem er Lichtsignale sendet. Zunächst wird die Uhrzeit verschlüsselt, indem jede Ziffer durch ihr „Neunerkomplement“ ausgetauscht wird, d. h. eine Ziffer z wird durch $9 - z$ ersetzt. Doppelpunkt und Bindestrich werden entfernt. Beispielsweise wird die Uhrzeit 0:38-2 als Ziffernfolge 9617 verschlüsselt. Zur Übermittlung per Lichtsignal verwenden die Agenten das Morsealphabet. 9617 wird somit als Folge von kurzen und langen Lichtsignalen dem Partner zugesandt.
- 4 a) Stellen Sie die Kommunikation in einem Schichtenmodell mit vier Schichten dar. Geben Sie dabei den Schichten aussagekräftige Bezeichner.
- 2 b) Beschreiben Sie, welche Schichten in welcher Weise geändert werden müssen, wenn anstelle der Lichtsignale Pfeiftöne mit kurzen bzw. langen Tönen verwendet und anstelle des Morsecodes die Uhrzeiten binär codiert werden.

(Fortsetzung nächste Seite)

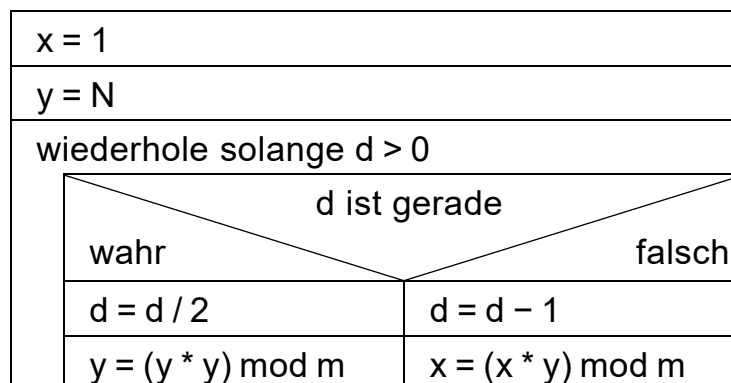
3. Die Verschlüsselung einer Uhrzeit mit dem Neunerkomplement aus Aufgabe 2 ist unsicher. Deswegen verwenden die Agenten für die Verschlüsselung ein sichereres Verschlüsselungsverfahren. Dazu werden zunächst Doppelpunkt und Bindestrich aus der Uhrzeit entfernt und die verbleibende Ziffernfolge als natürliche Zahl N ohne führende Nullen interpretiert. Anschließend wählt man zwei natürliche Zahlen d und m . Für die Verschlüsselung muss der Wert des Terms $N^d \bmod m$ berechnet werden, wobei der Operator $\bmod$ den ganzzahligen Rest bei der Division von N^d durch m liefert. Da d und m aus Sicherheitsgründen meist sehr große Zahlen sind, ist es nicht praktikabel, zuerst den Wert der Potenz N^d zu bestimmen und danach den Rest bei Division durch m zu ermitteln.

Die beiden nachfolgenden Struktogramme beschreiben Algorithmen, mit denen der Wert des Terms $N^d \bmod m$ berechnet werden kann, ohne den Wert der Potenz N^d zu berechnen. Die Variable x enthält nach Ausführung des jeweiligen Algorithmus das Ergebnis.

Algorithmus 1:



Algorithmus 2:



- 7 a) Im Folgenden soll exemplarisch die Uhrzeit 0:01-3, d. h. $N = 13$, durch die beiden Algorithmen mit den kleinen Werten $d = 5$ und $m = 11$ verschlüsselt werden. Geben Sie dazu jeweils für jeden Durchlauf der Wiederholung die Werte für d und x bzw. d , x und y tabellarisch an.
- 3 b) Beurteilen Sie, welcher Algorithmus ein besseres Laufzeitverhalten in Abhängigkeit von d aufweist.

(Fortsetzung nächste Seite)

- c) Schreiben Sie ein Programm für eine Registermaschine mit folgendem Befehlssatz, das den gegebenen Algorithmus 2 umsetzt. Machen Sie deutlich (z. B. durch symbolische Adressierung), in welchen Speicherzellen die Werte der Variablen stehen. Sie können davon ausgehen, dass die entsprechenden Speicherzellen mit den Werten von N , d und m vor der Ausführung des Programms bereits vorbelegt sind.

load x	kopiert den Wert aus der Speicherzelle x in den Akkumulator
loadi n	lädt die ganze Zahl n in den Akkumulator
store x	kopiert den Wert aus dem Akkumulator in die Speicherzelle x
add x	addiert den Wert aus der Speicherzelle x zum Wert im Akkumulator
addi n	addiert die ganze Zahl n zum Wert im Akkumulator
sub x	subtrahiert den Wert aus der Speicherzelle x vom Wert im Akkumulator
subi n	subtrahiert die ganze Zahl n vom Wert im Akkumulator
mul x	multipliziert den Wert im Akkumulator mit dem Wert in Speicherzelle x
muli n	multipliziert den Wert im Akkumulator mit der ganzen Zahl n
div x	dividiert den Wert im Akkumulator durch den Wert in Speicherzelle x (ganzzahlige Division)
divi n	dividiert den Wert im Akkumulator durch die ganze Zahl n (ganzzahlige Division)
mod x	dividiert den Wert im Akkumulator durch den Wert in Speicherzelle x und speichert den ganzzahligen Rest im Akkumulator
modi n	dividiert den Wert im Akkumulator durch die ganze Zahl n und speichert den ganzzahligen Rest im Akkumulator
jmp x	springt zum Befehl in der Speicherzelle x
jeq x	springt zum Befehl in Speicherzelle x, falls der Wert im Akkumulator gleich 0 ist
jge x	springt zum Befehl in Speicherzelle x, falls der Wert im Akkumulator positiv oder gleich 0 ist
jle x	springt zum Befehl in Speicherzelle x, falls der Wert im Akkumulator negativ oder gleich 0 ist
jgt x	springt zum Befehl in Speicherzelle x, falls der Wert im Akkumulator positiv ist
jlt x	springt zum Befehl in Speicherzelle x, falls der Wert im Akkumulator negativ ist
hold	beendet die Abarbeitung des Programms

(Fortsetzung nächste Seite)

4. Jeder Agent verfügt über genau eine sichere Leitung zur Zentrale. Kommuniziert ein Agent über eine solche Leitung mit einer Person der Zentrale, ist er für keine weitere Person aus der Zentrale erreichbar.

Nun möchte der Geheimdienstleiter N eine Dreierkonferenz mit den Agenten 007 und 008 aufbauen. Dazu kontaktiert er zunächst einen der beiden Agenten über dessen sichere Leitung. Ist die Leitung bereits belegt, so wartet N, bis sie frei ist, und beginnt dann das Gespräch mit dem Agenten. Anschließend kontaktiert N den anderen Agenten über dessen sichere Leitung. Ist diese belegt, so wartet N, bis auch sie frei ist. Danach kann er mit beiden Agenten die aktuelle Lage besprechen und schließlich die Dreierkonferenz beenden. Auf die gleiche Weise will O eine Dreierkonferenz mit 007 und 009 sowie Mister Moneycent eine Dreierkonferenz mit 008 und 009 aufbauen.

Erläutern Sie, wie es dabei zu einer Verklemmung kommen kann.

Beschreiben Sie eine Strategie, nach der die Mitglieder der Zentrale vorgehen könnten, damit beim Aufbau von Dreierkonferenzen durch je ein Mitglied der Zentrale mit je zwei Agenten Verklemmungen grundsätzlich nicht eintreten können.