Abiturprüfung 2022

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:

INF1. MODELLIERUNG UND PROGRAMMIERUNG

I.

BE

1. Für einen Radiosender soll eine Software entwickelt werden, die Informationen zu den Sendungen mithilfe einer Klasse SENDUNG verwaltet. Sendungen können Wortbeiträge und Songs enthalten. Außerdem sollen Informationen über Mitarbeitende gespeichert werden. Dabei handelt es sich entweder um Moderatorinnen und Moderatoren der Sendungen oder um Redakteurinnen und Redakteure. Letztere erarbeiten die Wortbeiträge zu den Sendungen, wobei ein Wortbeitrag immer von genau einer Person erarbeitet wird. Eine Sendung kann dagegen auch von mehreren Personen moderiert werden.

10 a) Zeichnen Sie nach diesen Vorgaben ein Klassendiagramm. Verwenden Sie das Konzept der Vererbung. Geben Sie für drei Klassen, die keine Personen beschreiben, insgesamt mindestens sechs sinnvolle Attribute an. Auf die Angabe von Methoden kann verzichtet werden.

4 b) In den Sendungen werden Werbespots eingespielt, die durch Objekte einer Klasse WERBESPOT repräsentiert werden. Ein Werbespot kann in verschiedenen Sendungen vorkommen und in einer Sendung können verschiedene Werbespots abgespielt werden. Dies kann z. B. auf folgende Arten umgesetzt werden:

   i. Ein Objekt der Klasse WERBESPOT verwaltet ein Feld mit allen Sendungen, in denen dieser Werbespot enthalten ist.

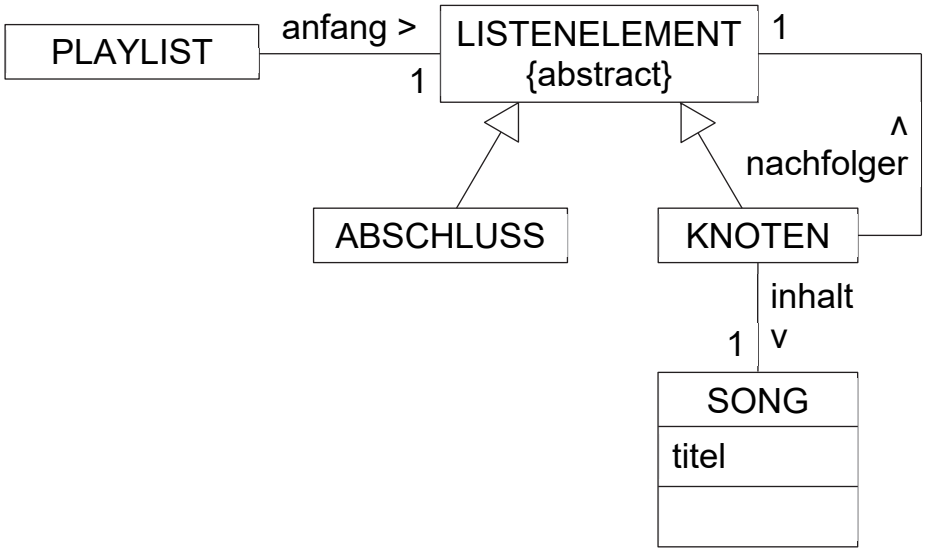
   ii. Ein Objekt der Klasse SENDUNG verwaltet ein Feld mit allen Werbespots, die in dieser Sendung abgespielt werden.

   Beschreiben Sie je eine Anwendungssituation, in der die jeweilige Umsetzung von Vorteil ist.

(Fortsetzung nächste Seite)

BE

Die Songs einer Sendung werden in einer Playlist als einfach verkettete Liste gemäß folgendem Klassendiagramm verwaltet. Zur Vereinfachung wird in der Klasse SONG nur das Attribut *titel* aufgeführt.



Damit Songs an der gewünschten Stelle einer bereits vorhandenen Playlist eingefügt werden können, steht in der Klasse PLAYLIST die rekursive Methode *einfuegenNach(titelVorher, titelNeu)* zur Verfügung. Sie fügt den Song mit dem in der Zeichenkette *titelNeu* übergebenen Titel unmittelbar nach dem Song mit dem in der Zeichenkette *titelVorher* übergebenen Titel in die Liste ein.

5

c) Eine Playlist enthält zunächst nur die zwei Songs „You're Simply the First" gefolgt von „Stairway to Hell". Nun erfolgen nacheinander diese Methodenaufrufe:

i. *einfuegenNach(„You're Simply the First", „Born to Run away")*

ii. *einfuegenNach(„You're Simply the First", „Don't Speak! Write!")*

iii. *einfuegenNach(„Born to Run away", „Wanna have it my Way")*

Stellen Sie die Playlist, die sich nach der Ausführung dieser Einfüge-Operationen ergibt, in einem Objektdiagramm dar. Den Wert des Attributs *titel* können Sie mit dem ersten Wort des Songtitels abkürzen.

(Fortsetzung nächste Seite)

BE	
10	d) Notieren Sie mithilfe einer auf dem Deckblatt angegebenen Programmiersprache eine mögliche Implementierung der Methode *einfuegenNach* der Klasse PLAYLIST und aller dazu nötigen Methoden in der Listenstruktur. Verwenden Sie so weit wie möglich das Prinzip der Rekursion. Sie können davon ausgehen, dass die Klasse SONG einen Konstruktor *SONG(titel)* sowie eine Methode *istGleich(songtitel)* besitzt. Diese gibt genau dann *wahr* zurück, wenn der Songtitel des ausführenden Objekts mit dem übergebenen Songtitel übereinstimmt. Es darf angenommen werden, dass ein Songtitel nicht mehrfach in der Liste vorkommt. Wenn kein Song mit dem Titel *titelVorher* in der Liste enthalten ist, soll der neue Song nicht eingefügt, sondern eine entsprechende Meldung auf dem Bildschirm ausgegeben werden.
3	e) In der Sendung „Wünsch dir was“ können sich Hörerinnen und Hörer Songtitel wünschen. Die Wünsche werden per Formular auf der Webseite des Senders eingereicht. Erläutern Sie, in welcher Hinsicht das Speichern der Wünsche in einem Stapel einzelne Hörerinnen bzw. Hörer benachteiligen könnte.
2	f) Geben Sie jeweils an, welche Methoden eine allgemeine Datenstruktur Liste für die Realisierung eines Stapels bzw. einer Schlange zur Verfügung stellen muss.

2. Die Online-Plattform musicStream, die eine sehr große Zahl an Songs bereitstellt, bietet die Möglichkeit, Informationen über die Songs einzuholen. Die Songs werden mit den zugehörigen Informationen in einem binären Suchbaum unter Verwendung des Entwurfsmusters Kompositum gespeichert. Als Schlüssel dient der International Standard Recording Code (ISRC). Der ISRC ist eine 12-stellige Kennung, die einen Song eindeutig identifiziert.

2	a) Geben Sie an, wie das in Aufgabe 1 gegebene Klassendiagramm der einfach verketteten Liste im Wesentlichen verändert werden muss, damit es einen binären Suchbaum geeignet modelliert.

(Fortsetzung nächste Seite)

BE
6

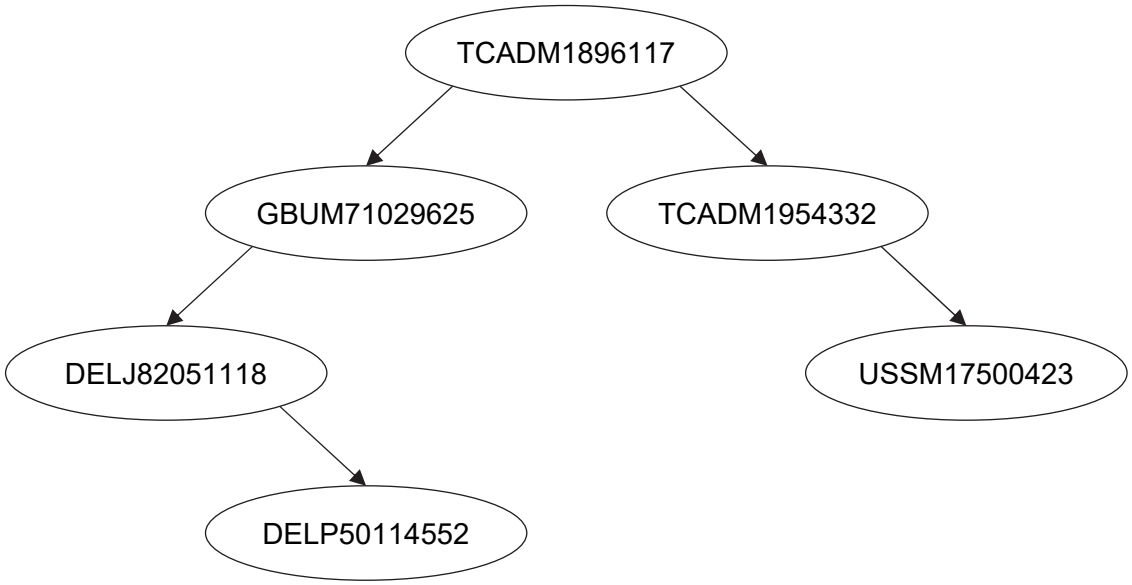
b) Eine Mitarbeiterin von musicStream ist der Meinung, dass in einem Binärbaum durch die vielen Abschluss-Objekte bei Verwendung des Entwurfsmusters Kompositum unnötig Speicherplatz belegt wird.

Begründen Sie, dass ein solcher Binärbaum mit n Knoten stets n+1 Abschlüsse enthält.

Berechnen Sie, wie viel Speicherplatz durch die Abschlüsse bei einem Baum mit 70 Millionen Knoten belegt wird. Gehen Sie davon aus, dass pro Abschluss 32 Byte Speicherplatz belegt wird.

Ein Mitarbeiter ist der Ansicht, dass Speicherplatz gespart wird, wenn die Songs in einem Feld statt in einem Baum gespeichert werden.

Beispielhaft wird dazu zunächst folgender Suchbaum betrachtet:



(Fortsetzung nächste Seite)

BE

5

c) Mithilfe eines Verfahrens zum vollständigen Durchlaufen des geordneten Binärbaums werden alle Songs in einer bestimmten Reihenfolge ausgegeben. Die Songs könnten in dieser Reihenfolge in einem Feld gespeichert werden. Nennen Sie ein Verfahren zum Durchlaufen eines Baumes und wenden Sie dieses auf den gegebenen Baum an.

Entscheiden Sie begründet, ob dieses Verfahren dazu geeignet ist, einen Baum so in einem Feld darzustellen, dass daraus wieder ein identisch aufgebauter Baum erzeugt werden kann.

6

d) Um die Songs eines beliebigen Binärbaums so in einem Feld zu speichern, dass der Baum wieder eindeutig rekonstruiert werden kann, wird nach folgendem Verfahren jeder Knoten zunächst nummeriert:

- Die Wurzel erhält die Nummer 1.
- Wurde einem Knoten die Nummer i zugeordnet, wird dessen linkem Nachfolger die Nummer 2i und dessen rechtem Nachfolger die Nummer 2i+1 zugeordnet.

Ein Song, der in einem Knoten mit der Nummer i referenziert ist, wird im Feldelement mit entsprechendem Index i gespeichert.

Wenden Sie dieses Verfahren auf den oben gegebenen Baum an und notieren Sie das sich ergebende Feld.

5

e) Ein Baum mit 10 Knoten soll mit dem Verfahren aus Teilaufgabe 2d in einem Feld gespeichert werden. Ermitteln Sie, wie groß das Feld dafür im schlechtesten Fall mindestens sein muss.

Berechnen Sie, welcher Anteil des Feldes in diesem Fall nicht mit Knoten belegt wäre.

(Fortsetzung nächste Seite)

BE

3. Die Online-Plattform musicStream wertet das Hörverhalten ihrer Kundschaft aus und generiert daraus u. a. Profile mit Vorschlägen für weitere Songs. Dazu werden Songs in Knoten eines Graphen gespeichert. Die Kanten geben an, welche Songs direkt nacheinander gehört wurden, wobei es keine Rolle spielt, welcher Song zuerst gehört wurde. Außerdem soll anhand des Graphen ersichtlich sein, wie häufig eine solche direkte Abfolge für zwei Songs aufgetreten ist.

4 a) Geben Sie zwei Eigenschaften eines solchen Graphen an und begründen Sie Ihre Angaben im Sachzusammenhang.

6 b) Zu den fünf Songs „Born“, „Don’t“, „Stairway“, „Wanna“, „You’re“ (abgekürzt jeweils durch das erste Wort des Titels) wurden folgende Daten gesammelt:

- „Born“ wurde bisher zweimal vor „You’re“ gehört.
- „Don’t“ wurde schon fünfmal nach „You’re“ gehört.
- „Wanna“ wurde einmal nach „Don’t“ und einmal nach „You’re“ gehört.
- „You’re“ wurde einmal vor „Born“ und einmal nach „Don’t“ gehört.

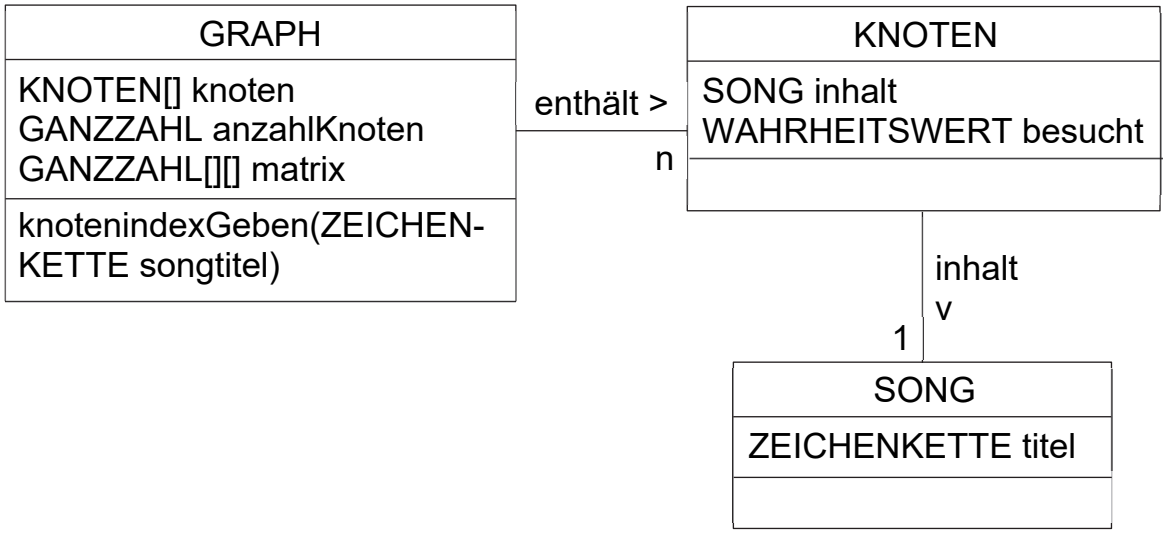
Zeichnen Sie den dazugehörigen Graphen und geben Sie die Adjazenzmatrix an.

(Fortsetzung nächste Seite)

BE

12

c) Der Implementierung des Graphen liegt folgendes Klassendiagramm zugrunde:



Die Methode *knotenindexGeben(songtitel)* gibt zu einem Songtitel den entsprechenden Knotenindex zurück. Wenn kein Knoten existiert, der ein Song-Objekt referenziert, das den eingegebenen Titel als Attributwert besitzt, wird −1 zurückgegeben.

Die Kunden von musicStream sollen sich eine Favoritenliste von Songs auf dem Bildschirm anzeigen lassen können. Dabei soll zunächst ein ausgewählter Startsong ausgegeben werden. Anschließend wird von den Songs, die noch nicht in der Favoritenliste vorkommen, immer derjenige als nächstes angezeigt, der bisher am häufigsten in direkter Abfolge mit dem zuletzt angezeigten Song gespielt wurde. Die Klasse GRAPH soll zu diesem Zweck eine Methode *favListeAusgeben(startsongtitel)* besitzen, die eine entsprechende Ausgabe erzeugt.

Falls es zu einem Song mehrere weitere gibt, die gleich häufig in direkter Abfolge gehört wurden, soll der zuerst gefundene Song ausgegeben werden. Falls zum eingegebenen Titel *startsongtitel* kein entsprechender Knoten existiert, wird eine Fehlermeldung ausgegeben.

Notieren Sie in einer auf dem Deckblatt angegebenen Programmiersprache eine Implementierung der Methode *favListeAusgeben(startsongtitel)*. Dazu können Sie auch weitere Methoden implementieren, falls es Ihnen sinnvoll erscheint. Methoden zum Geben und Setzen sämtlicher Attributwerte sowie die Methode *knotenindexGeben* können Sie als bereits implementiert voraussetzen. Außerdem sind in sämtlichen Knoten die Werte des Attributs *besucht* mit *falsch* vorbelegt.

80

INF1. MODELLIERUNG UND PROGRAMMIERUNG

II.

BE

Für die John-von-Neumann-Realschule soll eine Schulverwaltungssoftware entwickelt werden.

1. Es sollen die Daten von allen Schülerinnen und Schülern, Lehrkräften, Schulklassen, Erziehungsberechtigten und Räumen verwaltet werden. Von jeder Schülerin und jedem Schüler soll der Name, das Geburtsdatum und die Schulklasse gespeichert werden. Diese wird anhand ihrer Bezeichnung (z. B. 7a) identifiziert. Von Lehrkräften soll der Name und das eindeutige Kürzel registriert werden. Ebenso soll für Schulklassen gespeichert werden, welche Lehrkräfte sie unterrichten. Von Räumen soll die eindeutige Nummer sowie die Anzahl der Sitzplätze erfasst werden. Jeder Schulklasse ist genau ein Raum als Klassenraum und jeder Schülerin bzw. jedem Schüler mindestens eine erziehungsberechtigte Person zugeordnet, deren Name erfasst wird.

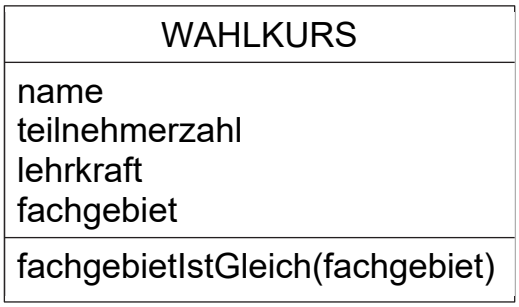
9 a) Erstellen Sie ein zur beschriebenen Situation passendes Klassendiagramm. Auf die Angabe von Methoden und nicht im Text genannten Attributen kann verzichtet werden.

4 b) Erläutern Sie im Sachzusammenhang ein mögliches Szenario, für das man das Prinzip der Spezialisierung für eine der Klassen Ihres Diagramms aus Teilaufgabe 1a sinnvoll anwenden kann. Stellen Sie diese Vererbungsbeziehung in einem weiteren Klassendiagramm dar.

(Fortsetzung nächste Seite)

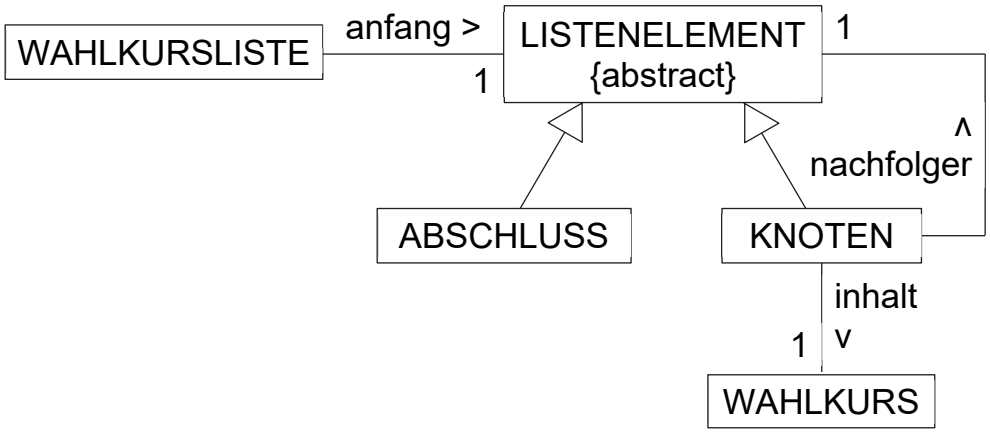
BE

Die Schule bietet verschiedene Wahlkurse an, die als Objekte der folgenden Klasse WAHLKURS verwaltet werden:



Die Methode *fachgebietIstGleich(fachgebiet)* gibt *wahr* zurück, wenn das als Parameter übergebene Fachgebiet mit dem Wert des Attributs *fachgebiet* übereinstimmt, ansonsten *falsch*.

2. Die Wahlkurse werden in einer einfach verketteten Liste gemäß folgendem Klassendiagramm verwaltet.



5 a) Erklären Sie, warum man sich für eine einfach verkettete Liste anstelle eines Feldes entschieden haben könnte. Erläutern Sie außerdem, warum es sinnvoll ist, dabei das Entwurfsmuster Kompositum zu verwenden, und stellen Sie dar, inwiefern bei der gegebenen Umsetzung die Trennung von Struktur und Daten berücksichtigt wurde.

(Fortsetzung nächste Seite)

BE

16

b) In der Klasse WAHLKURSLISTE sollen folgende Methoden zur Verfügung stehen:

- *anzahlKurseGeben(fachgebiet)* gibt die Anzahl der Wahlkurse des übergebenen Fachgebiets zurück.
- *zuKleineKurseLoeschen(minTeilnehmerzahl)* entfernt sämtliche Wahlkurs-Objekte aus der Liste, deren Teilnehmerzahl unter der übergebenen Mindestteilnehmerzahl liegt.

Notieren Sie in einer auf dem Deckblatt angegebenen Programmiersprache eine Implementierung aller Klassen der Listenstruktur sowie eine mögliche Implementierung der beiden oben aufgeführten Methoden und aller dazu nötigen Methoden in der Listenstruktur (ohne Konstruktoren). Verwenden Sie so weit wie möglich das Prinzip der Rekursion.

Dabei können Sie davon ausgehen, dass die oben gegebene Klasse WAHLKURS einschließlich der Methoden zum Geben und Setzen der Attributwerte bereits vollständig implementiert ist.

8

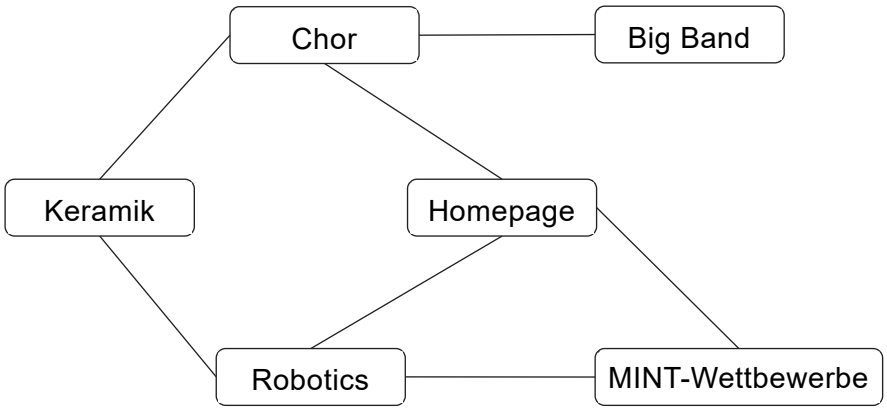
c) Die Schulverwaltungssoftware ermöglicht es, dass verschiedene Personen Wahlkurslisten gemäß der gegebenen Listenstruktur erstellen, die anschließend in eine Gesamtliste zusammengeführt werden. Dazu soll eine rekursive Methode *hintenEinfuegen(wahlkursliste)* zur Verfügung stehen, welche die in der übergebenen Liste *wahlkursliste* enthaltenen Wahlkurse hinten an jene Liste anhängt, von der die Methode ausgeführt wird.

Stellen Sie für eine mögliche Implementierung den Ablauf des Methodenaufrufs *wkliste1.hintenEinfuegen(wkliste2)* in einem Sequenzdiagramm dar, wobei *wkliste1* zwei Knoten *knoten1* und *knoten2* verwaltet. Sie können davon ausgehen, dass Methoden zum Geben und Setzen von Attributwerten zur Verfügung stehen.

(Fortsetzung nächste Seite)

BE

3. Die Wahlkurse werden zusätzlich in einem Graphen verwaltet. Die Knoten des Graphen enthalten dabei Referenzen auf die Wahlkurse. Zwei Knoten, deren Wahlkurse gemeinsame Teilnehmer besitzen, werden durch eine Kante verbunden. Die folgende Abbildung zeigt den Graphen für sechs Wahlkurse.



Um den Graphen zu implementieren, speichert die Klasse GRAPH die Knoten in einem Feld *knoten*, die aktuelle Knotenanzahl im Attribut *anzahlKnoten* und die Adjazenzmatrix in einem zweidimensionalen Feld *matrix* ab.

3 a) Geben Sie die zugehörige Adjazenzmatrix an.

3 b) Zusätzlich zu den oben dargestellten Wahlkursen gibt es noch die Wahlkurse „Schülerzeitung“, „Fotografie“ und „Tennis“.

Es gibt eine Schülerin, die am Wahlkurs „Schülerzeitung“ und am Wahlkurs „Keramik“ teilnimmt. Drei Schüler nehmen sowohl am Wahlkurs „Fotografie“ als auch am Wahlkurs „Tennis“ teil. Außerdem besuchen fünf Schülerinnen bzw. Schüler zusätzlich zum Wahlkurs „Tennis“ auch den Wahlkurs „Robotics“. Zwei Schüler aus dem Wahlkurs „Fotografie“ sind auch in der Big Band aktiv.

Zeichnen Sie den resultierenden Graphen.

(Fortsetzung nächste Seite)

BE

Für eine Projektarbeit, an der alle Wahlkurse beteiligt sind, müssen Materialien, die in einem bestimmten Wahlkurs angefertigt werden, an alle anderen Wahlkurse verteilt werden. Dazu nehmen diejenigen Schülerinnen und Schüler, die mehrere Wahlkurse besuchen, die Materialien einfach zu ihren weiteren Wahlkursen mit. Damit man weiß, über wie viele Wahlkurse die Weitergabe der Materialien erfolgt, ordnet man den Wahlkursen nach folgendem Schema Kategorien zu:

- Ein Wahlkurs, der Materialien anfertigt, die verteilt werden müssen, erhält die Kategorie 1.
- Ein Wahlkurs, der mit einem Wahlkurs der Kategorie 1 gemeinsame Schülerinnen oder Schüler besitzt, wird als Kategorie 2 eingestuft.
- Ein Wahlkurs, der mit einem Wahlkurs der Kategorie 2 gemeinsame Teilnehmerinnen bzw. Teilnehmer besitzt, wird als Kategorie 3 eingestuft, sofern ihm noch keine niedrigere Kategorie zugewiesen wurde.

Diese Kategorisierung lässt sich entsprechend fortsetzen.

Die Methode *wahlkurseKategorisieren(kursname)* in der Klasse GRAPH kategorisiert alle Wahlkurse gemäß der oben angegebenen Regel für den Fall, dass ausgehend vom Wahlkurs mit dem übergebenen Namen *kursname* Materialien verteilt werden müssen.

2 c) Der Wahlkurs „MINT-Wettbewerbe“ hat als erster Wahlkurs Materialien zu verteilen. Geben Sie für alle Wahlkurse im ursprünglichen Graphen die Kategorien nach Ausführen der Methode *wahlkurseKategorisieren("MINT-Wettbewerbe")* an.

9 d) Formulieren Sie einen Algorithmus (z. B. in Pseudocode) für die Methode *wahlkurseKategorisieren(kursname)*. Gehen Sie dabei davon aus, dass ein Wahlkurs mit dem übergebenen Kursnamen existiert und die Knoten ein ganzzahliges Attribut *kategorie* besitzen, das mit dem Wert 0 vorbelegt ist. Methoden zum Geben und Setzen von Werten dieses Attributs sowie eine Methode *indexGeben(kursname)*, die den Index des Knotens zurückgibt, der den entsprechenden Wahlkurs referenziert, können als gegeben vorausgesetzt werden.

(Fortsetzung nächste Seite)

BE

4. Die Schülerinnen und Schüler des Wahlkurses „Homepage“ möchten zum neuen Schuljahr einen Schulnewsletter einrichten, um z. B. über Veranstaltungen der Schule zu informieren. Interessierte können den Newsletter abonnieren. Für die Verwaltung der Abonnentinnen und Abonnenten wird ein lexikographisch geordneter Binärbaum verwendet, wobei eine eindeutige E-Mail-Adresse als Schlüssel dient.

6 a) Vor Einführung des Schulnewsletters testen die Schülerinnen und Schüler ihre Implementierung des Baums mithilfe von Beispieldaten. Ein Testdurchlauf für alle im Baum enthaltenen E-Mail-Adressen ergibt folgende Ausgabereihenfolge:

georg@gumail.de
dana@t-line.de
paul@ebrief.de
maja@mueller.de
werner@email.de
silke@aul.de
karl@pear.de

Begründen Sie, dass die gegebene Ausgabe weder durch Inorder-Durchlauf noch durch Preorder-Durchlauf erzeugt worden sein kann.

Zeichnen Sie unter der Annahme, dass die Ausgabe durch einen Postorder-Durchlauf erzeugt wurde, den dazugehörigen Baum. Dabei können Sie sich bei den Bezeichnern der Knoten auf die gegebenen Vornamen beschränken.

3 b) Geben Sie eine Einfügereihenfolge der in Teilaufgabe 4a genannten E-Mail-Adressen an, sodass der Binärbaum möglichst wenige Ebenen besitzt. Dabei genügt es, wieder nur die Vornamen anzugeben. Der Baum muss nicht erneut gezeichnet werden.

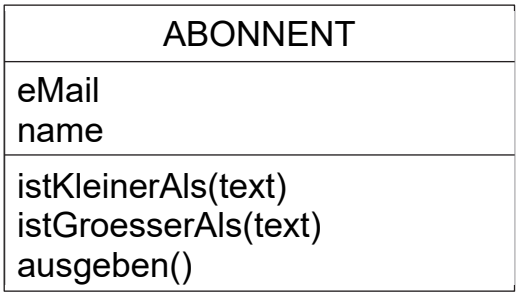
4 c) Der Schulnewsletter wird von 2000 Personen abonniert. Geben Sie begründet die maximal sowie die minimal nötige Anzahl der Ebenen für einen dazugehörigen Binärbaum an.

(Fortsetzung nächste Seite)

BE

8

d) Der Binärbaum wurde unter Verwendung der Klassen BAUM, BAUMELEMENT, KNOTEN, ABSCHLUSS und ABONNENT sowie des Entwurfsmusters Kompositum unter Berücksichtigung der Trennung von Struktur und Daten umgesetzt. Die Klasse ABONNENT wird durch nachstehendes Klassendiagramm beschrieben:



Für die Methoden der Klasse ABONNENT gilt:

- *istKleinerAls(text)* gibt genau dann *wahr* zurück, wenn der Wert des Attributs *eMail* des ausführenden Abonnent-Objekts lexikographisch kleiner als der Wert des Parameters *text* ist, wobei zwischen Groß- und Kleinschreibung nicht unterschieden wird.
- *istGroesserAls(text)* gibt genau dann *wahr* zurück, wenn der Wert des Attributs *eMail* des ausführenden Abonnent-Objekts lexikographisch größer als der Wert des Parameters *text* ist, wobei zwischen Groß- und Kleinschreibung nicht unterschieden wird.
- *ausgeben()* gibt die E-Mail-Adresse auf dem Bildschirm aus.

In der Klasse BAUM soll nun eine Methode *eMailsAusgeben(von, bis)* programmiert werden, die alle E-Mail-Adressen in alphabetischer Reihenfolge ausgibt, die lexikographisch zwischen den beiden übergebenen Zeichenketten *von* und *bis* eingeordnet sind, wobei die Grenzen *von* bzw. *bis* nicht mit ausgegeben werden. Z. B. soll der Methodenaufruf *eMailsAusgeben("dana", "karl")* für den oben beschriebenen Baum die E-Mail-Adressen dana@t-line.de und georg@gumail.de ausgeben, da dana@t-line.de lexikographisch größer als dana ist.

Notieren Sie in einer auf dem Deckblatt angegebenen Programmiersprache eine mögliche Implementierung der Methode *eMailsAusgeben(von, bis)* in der Klasse BAUM und aller dazu nötigen Methoden in den weiteren Klassen der Baumstruktur. Verwenden Sie so weit wie möglich das Prinzip der Rekursion und nutzen Sie die Vorteile des geordneten Binärbaums. Die Klasse ABONNENT kann als bereits implementiert vorausgesetzt werden.

80

INF2. THEORETISCHE UND TECHNISCHE INFORMATIK

III.

BE

1. Auf einer großen Anzeigetafel eines Bahnhofs werden Informationen der abfahrenden Züge aufgelistet. Dabei setzt sich die Information zu einer Zugabfahrt aus folgenden Bestandteilen zusammen, die jeweils durch ein Leerzeichen voneinander getrennt werden. Zur Darstellung des Leerzeichens wird das Zeichen ␣ verwendet.

   (1) Uhrzeit im Format hh:mm

   (2) Zugnummer, bestehend aus zwei oder drei Großbuchstaben gefolgt von genau vier Ziffern

   (3) Zielort, bestehend aus drei oder mehr Großbuchstaben

   (4) Ggf. Ortsnamen der Zwischenhalte, die jeweils ebenfalls aus drei oder mehr Großbuchstaben bestehen; mehrere Zwischenhalte werden durch Kommata (ohne Leerzeichen) getrennt

   (5) Gleisnummer, bestehend aus ein oder zwei Ziffern

   (6) Ggf. Status, in dem durch das Zeichen V bzw. Ø angezeigt wird, falls der Zug verspätet abfährt bzw. gestrichen wurde

   Zwei Beispiele für die Darstellung einer korrekten Zuganzeige sind:

   12:00␣ICE1234␣ULM␣5

   15:15␣RE2468␣MUENCHEN␣NUERNBERG,INGOLSTADT␣3␣Ø

6 a) Die Menge der Zuganzeigen, die wie oben beschrieben erzeugt werden, bildet die formale Sprache L. Geben Sie eine Grammatik für die Sprache L an und stellen Sie dabei die Produktionsregeln in formaler Textnotation (z. B. in EBNF) dar. Vereinfachend dürfen Sie davon ausgehen, dass für Uhrzeit und Gleisnummer alle Ziffern verwendet werden dürfen.

2 b) Erklären Sie die Begriffe Syntax und Semantik anhand eines Beispiels im Kontext der Sprache L.

(Fortsetzung nächste Seite)

BE	
4	c) Zeichnen Sie ein Syntaxdiagramm zur Darstellung des Zielorts einschließlich evtl. vorhandener Zwischenhalte wie in den Bestandteilen (3) und (4) der Zuganzeige beschrieben.
4	d) Zeichnen Sie das Zustandsübergangsdiagramm eines erkennenden endlichen Automaten, der nur gültige Zeitformate einer Uhrzeit von 00:00 bis 23:59 akzeptiert. Als Zeichenvorrat stehen die Ziffern von 0 bis 9 und der Doppelpunkt zur Verfügung.

2. Die ICE-Waggons bestehen aus Metall und auch die Fenster sind metallbeschichtet, weshalb Mobilfunksignale nur abgeschwächt ins Innere der Züge gelangen. Um Fahrgästen stabilen Internetzugang zu ermöglichen, gibt es innerhalb der Waggons WLAN-Accesspoints, die den Internetzugang über Mobilfunk mithilfe einer Antenne am Zug herstellen. Zwei Bahnreisende in zwei unterschiedlichen Zügen kommunizieren über das Internet miteinander.

4	a) Stellen Sie den Kommunikationsvorgang zwischen den beiden Fahrgästen in einem Schichtenmodell mit mindestens drei Schichten dar. Benennen Sie die Schichten geeignet.
2	b) Erläutern Sie in diesem Sachzusammenhang zwei Vorteile bei der Aufteilung der Kommunikation in mehrere Schichten.

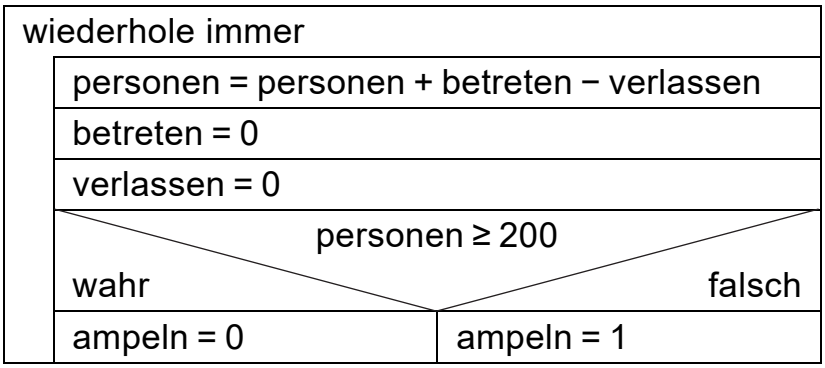
(Fortsetzung nächste Seite)

BE

3. An einem Bahnhof gibt es eine Business-Lounge, in der sich aus Brandschutzgründen zur selben Zeit höchstens 200 Personen aufhalten dürfen. Deshalb wird an allen Ein- und Ausgängen registriert, wenn eine Person den Durchgang passiert. Dazu wird jedes Mal, wenn eine Person die Lounge betritt, automatisch der Wert in Speicherzelle 101 einer Registermaschine um 1 erhöht. Entsprechend wird der Wert in Speicherzelle 102 automatisch um 1 erhöht, wenn eine Person die Lounge verlässt.

   An den Eingängen sind Ampeln montiert, die signalisieren, ob die Lounge betreten werden darf (grün) oder nicht (rot). Die Ampeln werden von der Registermaschine gesteuert. Die Ampelfarben werden durch den Wert 0 für Rot bzw. 1 für Grün in einer dritten Speicherzelle 103 festgelegt.

   Das folgende Struktogramm zeigt den Algorithmus, gemäß dem die Registermaschine die Ampeln steuert, wobei die Variablen *betreten* und *verlassen* für die Werte in den Speicherzellen 101 bzw. 102 stehen.



(Fortsetzung nächste Seite)

BE	
6	a) Der Befehlssatz der Registermaschine umfasst folgende Befehle:

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
sub x	subtrahiert den Wert aus der Speicherzelle x vom Wert im Akkumulator
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

Geben Sie ein Programm für die Registermaschine an, das den dargestellten Algorithmus umsetzt. Nennen Sie dazu die Adresse der Speicherzelle, in welcher der Wert der Variablen *personen* abgelegt ist. Gehen Sie davon aus, dass alle Variablen mit dem Wert 0 vorbesetzt sind.

2 b) Beschreiben Sie ein Szenario, in dem es mit der oben geschilderten Vorgehensweise zu einer unzulässig hohen Personenzahl in der Business-Lounge kommt.

(Fortsetzung nächste Seite)

BE

4. Es ist die folgende Methode zur Sortierung eines Feldes *array* in Pseudocode gegeben:

```
Methode sortieren(array)
    n = array.länge
    wiederhole solange n größer 1
        i = 0
        tauschen = falsch
        wiederhole solange i kleiner n − 1
            wenn array[i] größer array[i + 1] dann
                c = array[i]
                array[i] = array[i + 1]
                array[i + 1] = c
                tauschen = wahr
            endeWenn
            i = i + 1
        endeWiederhole
        wenn tauschen gleich falsch dann
            beende sortieren
        endeWenn
        n = n − 1
    endeWiederhole
endeMethode
```

6 a) Geben Sie schrittweise an, wie sich das folgende Feld beim Ausführen der Methode *sortieren* verändert.

8	7	6	5

Beschreiben Sie allgemein die Vorgehensweise des zugrunde liegenden Algorithmus und erläutern Sie, warum nach dessen Ablauf das übergebene Feld aufsteigend sortiert ist.

4 b) Bestimmen Sie in Abhängigkeit von der Feldlänge *n* die Anzahl der Vergleiche von Feldelementen im günstigsten und im ungünstigsten Fall. Geben Sie jeweils den Laufzeitaufwand der Methode *sortieren* für den günstigsten und den ungünstigsten Fall an.

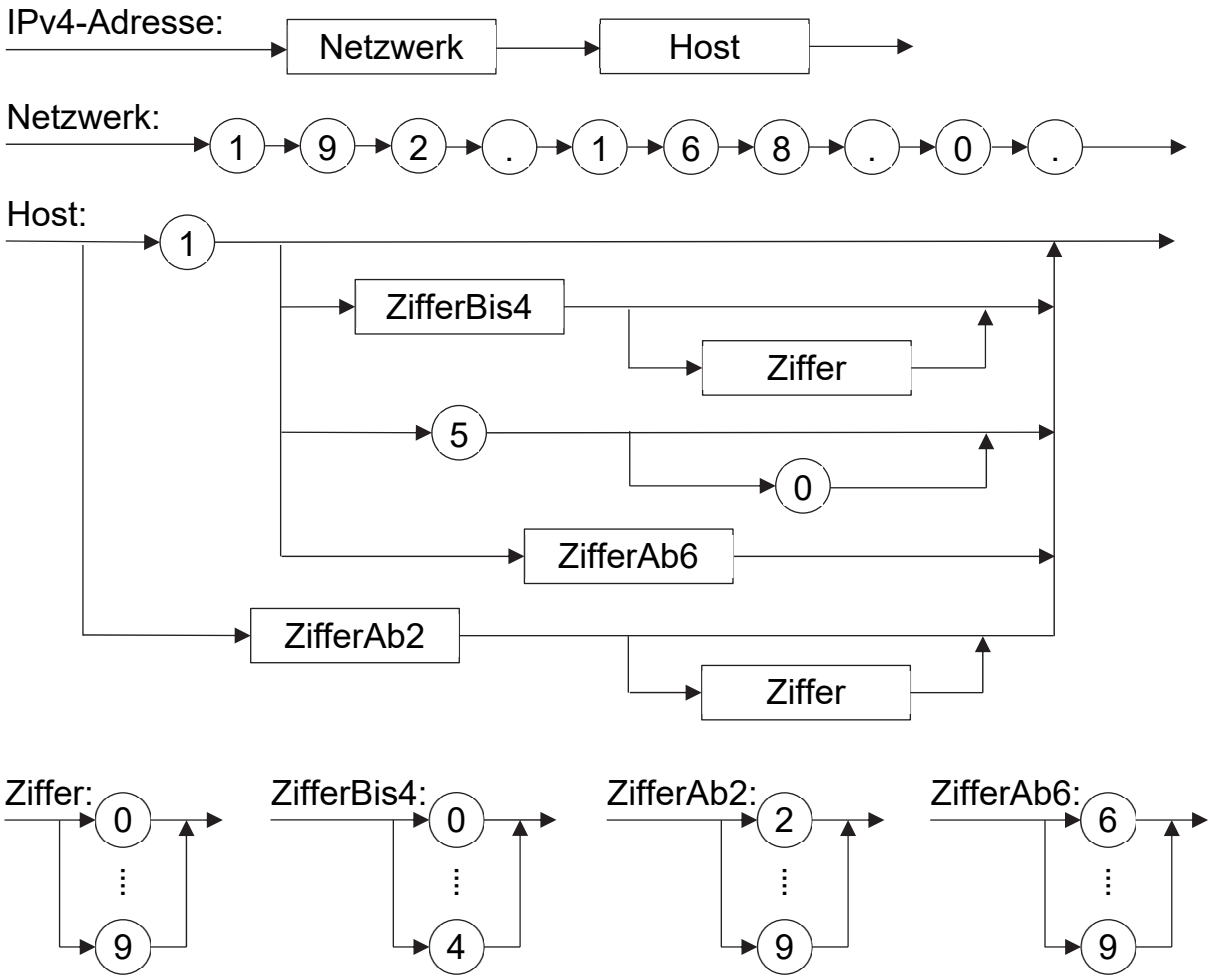
Hinweis: Sie können zur Vereinfachung die Gauß'sche Summenformel verwenden: $1 + 2 + 3 + \ldots + n = \frac{n\,(n+1)}{2}$.

40

IV.

BE

1. Der Router eines Heimnetzwerks weist den angebundenen Geräten (Hosts) IPv4-Adressen zu. Der dafür vorgesehene Adressbereich wird durch das folgende Syntaxdiagramm beschrieben:



3 a) Geben Sie den Bereich der IPv4-Adressen an, die in dem Heimnetzwerk gemäß dem Syntaxdiagramm zur Verfügung stehen.

6 b) Geben Sie das Zustandsübergangsdiagramm eines erkennenden endlichen Automaten mit Eingabealphabet {0; 1; …; 9} an, der genau den nach dem oben angegebenen Syntaxdiagramm gebildeten Host-Anteil der IPv4-Adressen akzeptiert.

(Fortsetzung nächste Seite)

BE

2. In einem Online-Rollenspiel können Spielcharaktere in einer interaktiven Welt bestimmte Missionen (Quests) erfüllen.

3 a) Um eine Quest gemeinsam zu meistern, können sich zwei Spielende zusammenschließen. Hierzu sendet z. B. Spieler1 eine Anfrage an Spieler2 und wartet auf Bestätigung. Während des Wartens können von Spieler1 keine weiteren Aktionen ausgeführt werden. Sobald Spieler2 die Anfrage annimmt, startet die Quest.

Vereinfacht dargestellt funktioniert dies auf mobilen Endgeräten wie folgt:

Erläutern Sie ein Problem, das auftreten kann, wenn sich zwei Spielende gegenseitig zur gleichen Zeit eine Anfrage schicken, und nennen Sie eine Möglichkeit, wie das Problem vermieden werden kann.

(Fortsetzung nächste Seite)

BE
2

b) Eine Quest besteht darin, vier Rätsel zu lösen. Für jedes gelöste Rätsel erhält man einen Großbuchstaben von A bis Z. Diese vier Großbuchstaben werden zu einem Lösungswort zusammensetzt, womit anschließend ein Tresor geöffnet werden kann. Ermitteln Sie, ob der Tresor mittels Brute-Force-Verfahren sicher innerhalb einer Minute geöffnet werden kann, wenn für die automatisierte Eingabe einer Kombination aus vier Buchstaben 0,2 Millisekunden benötigt werden.

3. 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)
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
jmpn x	springt zum Befehl in der Speicherzelle x, falls der Wert im Akkumulator negativ ist
jmpz x	springt zum Befehl in der Speicherzelle x, falls der Wert im Akkumulator gleich 0 ist
jmpp x	springt zum Befehl in der Speicherzelle x, falls der Wert im Akkumulator positiv ist
hold	beendet die Abarbeitung des Programms

(Fortsetzung nächste Seite)

BE

Buchstaben werden gespeichert, indem sie gemäß folgender Tabelle durch ganze Zahlen repräsentiert werden (ASCII-Tabelle):

A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**
65	66	67	68	69	70	71	72	73	74	75	76	77
N	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**
78	79	80	81	82	83	84	85	86	87	88	89	90
a	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**k**	**l**	**m**
97	98	99	100	101	102	103	104	105	106	107	108	109
n	**o**	**p**	**q**	**r**	**s**	**t**	**u**	**v**	**w**	**x**	**y**	**z**
110	111	112	113	114	115	116	117	118	119	120	121	122

Gegeben ist das folgende Programm für die Registermaschine, wobei in Speicherzelle 100 der nach obiger Tabelle kodierte Wert eines Zeichens steht.

```
Start: load 100
       subi 97
       jmpn Kleiner
       load 100
       subi 122
       jmpp XXX
       load 100
       subi 32
       jmp Ende
Kleiner: load 100
       subi 65
       jmpn XXX
       load 100
       subi 90
       jmpp XXX
       load 100
       jmp Ende
XXX:   loadi −1
Ende:  store 101
       hold
```

(Fortsetzung nächste Seite)

BE

6 a) Das Programm wird jeweils mit den Werten 65, 98 und 105 als Inhalt der Speicherzelle 100 ausgeführt. Geben Sie jeweils den berechneten Wert in Speicherzelle 101 und den zugeordneten Buchstaben an.

Beschreiben Sie im Sachzusammenhang kurz, was das Programm in Abhängigkeit des Startwerts in Speicherzelle 100 allgemein leistet. Gehen Sie dabei auch auf die Bedeutung der Zeile mit Sprungmarke XXX ein.

6 b) Stellen Sie den zugrunde liegenden Algorithmus graphisch dar (z. B. durch ein Struktogramm). Bezeichnen Sie dabei den in Speicherzelle 100 abgelegten Wert mit *a* und den berechneten Wert in Speicherzelle 101 mit *b*.

5 c) In dem Online-Rollenspiel aus Aufgabe 2 kann ein Spielcharakter Gegenstände kaufen. Zur Bezahlung stehen Goldmünzen im Wert von 5 Spieldollar ($) und Silbermünzen im Wert von 1 $ zur Verfügung.

Nachfolgender Algorithmus berechnet bei ausreichendem Vermögen des Käufers die kleinstmögliche Anzahl an Münzen, mit denen der Kaufpreis passend gezahlt werden kann. Hierbei entspricht *x* der Anzahl der Goldmünzen und *y* der Anzahl der Silbermünzen.

```
ergebnis = −1
wenn kaufpreis ≤ vermögen dann
    x = kaufpreis / 5
    y = kaufpreis mod 5
    ergebnis = x + y
endeWenn
```

Hinweis: Bei einer Division mit Rest liefert der Operator „/“ den ganzzahligen Quotienten (ohne Rest) und der Operator „mod“ den Rest. Beispielsweise gilt: 17 / 5 = 3 und 17 mod 5 = 2.

Schreiben Sie ein Programm für die gegebene Registermaschine, das den obigen Algorithmus umsetzt. Machen Sie dabei deutlich, in welchen Speicherzellen die einzelnen Variablen abgelegt sind.

(Fortsetzung nächste Seite)

BE

In besagtem Online-Rollenspiel kann man Zaubertränke im Wert von 1, 2, 3, …, *m* Spieldollar ($) erwerben, wobei die ganze Zahl *m* dem Preis des wertvollsten Zaubertranks entspricht, d. h. für *m* = 4 gibt es Zaubertränke im Wert von 1 $, 2 $, 3 $ und 4 $. Mit einem Betrag von 2 $ kann man entweder zwei Zaubertränke im Wert von 1 $ oder einen Zaubertrank im Wert von 2 $ kaufen. Demnach gibt es für 2 $ zwei verschiedene Möglichkeiten, Zaubertränke zu kaufen.

Zur Berechnung der Anzahl der verschiedenen Möglichkeiten, für einen bestimmten Betrag Zaubertränke zu erhalten, wird der unten stehende rekursive Algorithmus verwendet. Dabei entspricht die Zahl *b* dem Betrag in Spieldollar.

```
Methode anzahl(b, m)
    wenn b kleiner 0 oder m gleich 0 dann
        gib 0 zurück
    sonst wenn b gleich 0 dann
        gib 1 zurück
    sonst
        gib anzahl(b, m−1) + anzahl(b−m, m) zurück
    endeWenn
endeMethode
```

3 d) Zeigen Sie, dass dieser Algorithmus für *b* = 2 und *m* = 2 als Ergebnis den Wert 2 liefert, indem Sie die Berechnung von *anzahl(2, 2)* als Folge von Methodenaufrufen angeben. Geben Sie zudem die Anzahl der auf den Aufruf von *anzahl(2, 2)* folgenden rekursiven Aufrufe von *anzahl* an.

4 e) Zeigen Sie, dass *anzahl(2, m)* = *anzahl(2, m−1)* für *m* > 2 gilt. Ermitteln Sie daraus das Laufzeitverhalten von *anzahl(2, m)* in Abhängigkeit von *m*.

2 f) Begründen Sie im Sachzusammenhang, dass *anzahl(2, m)* = *anzahl(2, 2)* für *m* > 2 gilt.

40