

Abiturprüfung 2012

Hinweise zur Korrektur und Bewertung
der Abiturprüfungsarbeiten in

Informatik

Nicht für den Prüfling bestimmt.

Da es sich bei Informatik um ein vergleichsweise neues Abiturprüfungsfach handelt, sind die Lösungs- und Korrekturhinweise verhältnismäßig ausführlich gehalten. Dennoch können nicht alle Lösungsalternativen dargestellt werden.

Nicht genannte, aber `gleichwertige` Lösungswege und Begründungsansätze sind `gleichberechtigt`.

Die in den Lösungshinweisen gewählte objektorientierte Programmiersprache ist Java.

Die Bewertung der erbrachten Prüfungsleistungen hat sich an der bei jeder Teilaufgabe am linken Rand des Angabenblattes vermerkten, maximal erreichbaren Zahl von Bewertungseinheiten (BE) zu orientieren.

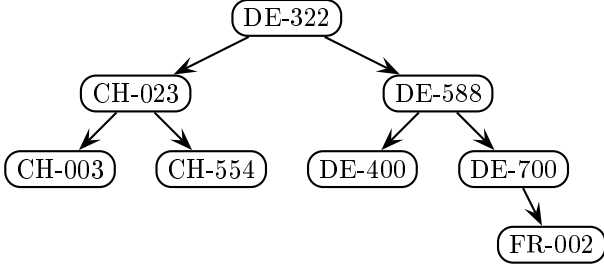
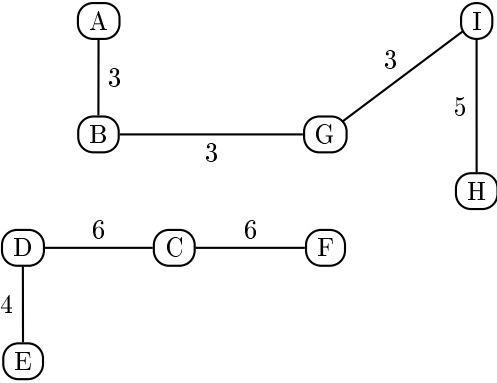
Umrechnung der erreichten Bewertungseinheiten in Notenpunkte:

Die insgesamt erreichten Bewertungseinheiten werden nach der folgenden Tabelle in Notenpunkte umgesetzt:

Notenpunkte	Notenstufen	Bewertungseinheiten	Intervalle in %
15	+1	120 ... 115	15
14	1	114 ... 109	
13	1–	108 ... 103	
12	+2	102 ... 97	15
11	2	96 ... 91	
10	2–	90 ... 85	
9	+3	84 ... 79	15
8	3	78 ... 73	
7	3–	72 ... 67	
6	+4	66 ... 61	15
5	4	60 ... 55	
4	4–	54 ... 49	
3	+5	48 ... 41	20
2	5	40 ... 33	
1	5–	32 ... 25	
0	6	24 ... 0	20

Aufgabe	BE	Hinweise
1. a)	6	<pre> graph LR n2012[nationenliste2012 : SORTIERTELISTE] -- erster --> d1[d1 : DATENKNOTEN] d1 -- nächster --> d2[d2 : DATENKNOTEN] d2 -- nächster --> d3[d3 : DATENKNOTEN] d3 -- nächster --> d4[d4 : DATENKNOTEN] d4 -- nächster --> a0[a0 : ABSCHLUSS] d1 -- inhalt --> china[china : NATION] d2 -- inhalt --> deutschland[deutschland : NATION] d3 -- inhalt --> frankreich[frankreich : NATION] d4 -- inhalt --> polen[polen : NATION] </pre>
b)	20	<pre> class SortierteListe{ private Listenelement erster; int anzahlAllerAthletenGeben(){ return erster.anzahlAllerAthletenGeben(); } void einfuegen(Nation nationNeu){ erster = erster.sortiertEinfuegen(nationNeu); } ... } abstract class Listenelement{ abstract int anzahlAllerAthletenGeben(); abstract Datenknoten sortiertEinfuegen(Nation inhaltNeu); ... } class Datenknoten extends Listenelement{ private Listenelement naechster; private Nation inhalt; Datenknoten(Listenelement next, Nation nat){ naechster = next; inhalt = nat; } int anzahlAllerAthletenGeben(){ return inhalt.anzahlAthletenGeben() + naechster.anzahlAllerAthletenGeben(); } Datenknoten sortiertEinfuegen(Nation inhaltNeu){ if (inhalt.istKleiner(inhaltNeu)){ naechster = naechster.sortiertEinfuegen(inhaltNeu); return this; } else { return new Datenknoten(this, inhaltNeu); } } ... } </pre>

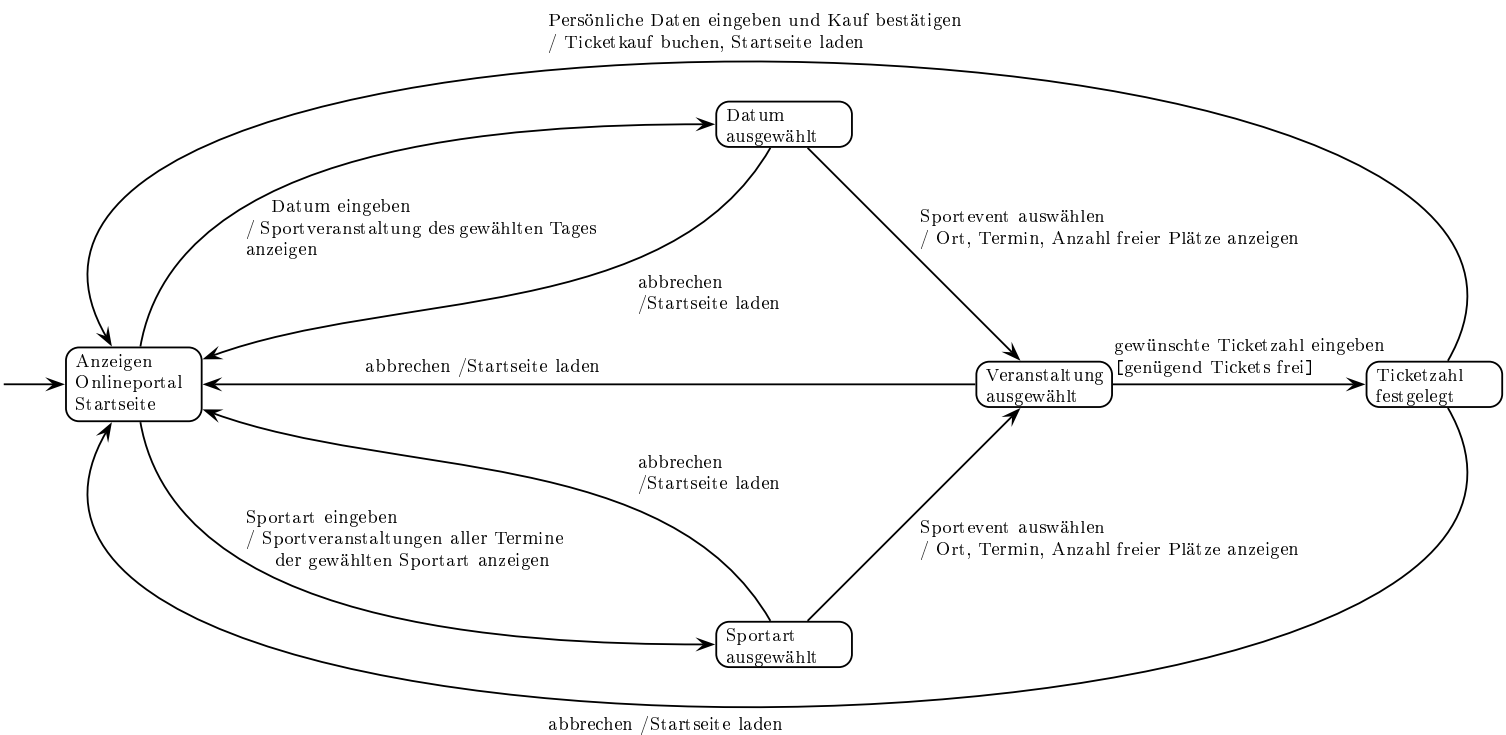
		<pre> class Abschluss extends Listenelement { int anzahlAllerAthletenGeben(){ return 0; } Datenknoten sortiertEinfuegen(Nation inhaltNeu){ return new Datenknoten(this,inhaltNeu); } ... } class Nation { private String landesbezeichner; private int anzahlAthleten; int anzahlAthletenGeben(){ return anzahlAthleten; } ... } </pre>
c)	6	Eine Möglichkeit wäre, Griechenland und das Gastgeberland nicht in der sortierten Liste zu speichern, sondern diese in der Klasse SORTIERTELISTE direkt zu verwalten. Dann sind folgende Änderungen in der Klasse SORTIERTELISTE notwendig: In der Methode zur Bestimmung der Anzahl aller beteiligten Athleten müssen zum Ergebnis der rekursiv definierten Methode für die verkettete Liste die Werte für die Anzahl der Athleten aus Griechenland und dem Gastgeberland addiert werden. Die Methode zum Einfügen neuer Nationen muss zusätzlich um bedingte Anweisungen ergänzt werden. Falls es sich um Griechenland oder das Gastgeberland handelt, werden die Daten als lokale Referenz gespeichert, andernfalls wie bisher sortiert in die Liste eingefügt. Eine zweite Möglichkeit wäre, die Methode <i>istKleiner</i> der Klasse NATION so anzupassen, dass die tatsächliche Reihenfolge richtig berücksichtigt ist: Griechenland ist also im Vergleich immer die „kleinste“, das Gastgeberland immer die „größte“ Nation. Dabei müssen zusätzlich bedingte Anweisungen in der Methode <i>istKleiner</i> ergänzt werden: Falls Griechenland verglichen werden soll, ist das entsprechende Datenelement immer „kleiner“ als alle anderen, falls das Gastgeberland verglichen werden soll, ist das entsprechende Datenelement immer „größer“ als alle anderen Datenelemente.
2.	6	<pre> boolean istBesser(Nation nat){ return ((medaillen[0] > nat.medaillen[0]) (medaillen[0] == nat.medaillen[0] && medaillen[1] > nat.medaillen[1]) (medaillen[0] == nat.medaillen[0] && medaillen[1] == nat.medaillen[1] && medaillen[2] > nat.medaillen[2])) } </pre>

3. a)	7	 Mögliche Reihenfolge beim Einsortieren: DE-322, CH-023, DE-588, CH-003, CH-554, DE-400, DE-700, FR-002
b)	6	Anzahl der Vergleiche im ausgeglichenen Baum: maximal etwa $\log_2(12000) = 13,55 \dots \approx 14$ Zeit für 14 Vergleiche: $14 \cdot 0,2 \cdot 10^{-7} s = 2,8 \cdot 10^{-7} s$ Anzahl der Vergleiche im entarteten Baum: maximal 12000 Zeit für 12000 Vergleiche: $12000 \cdot 0,2 \cdot 10^{-7} s = 2,4 \cdot 10^{-4} s$ Mögliche Bewertungsalternativen: Ein Anwender bemerkt den Unterschied bei einer einzigen Suchanfrage wohl kaum, d. h. die Reihenfolge der Eintragungen spielt bei dieser Anzahl für die Suche keine Rolle. Bei einer Vielzahl von Suchanfragen hingegen fällt der Faktor 1000 durchaus ins Gewicht.
c)	6	Die Methode <i>k1</i> der Klasse ATHLETENBAUM gibt den Namen und die Sportart aller in der Baumstruktur gespeicherten Athleten aus, welche die Nation Deutschland vertreten, und zwar sortiert nach dem Schlüssel.
d)	3	<pre>SELECT Name, Sportart FROM athlet WHERE Nation='Deutschland';</pre>
4. a)	4	

b)	4	<table><tr><td></td><td>A</td><td>B</td><td>C</td><td>D</td><td>E</td><td>F</td><td>G</td><td>H</td><td>I</td></tr><tr><td>A</td><td></td><td>3</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>B</td><td>3</td><td></td><td></td><td></td><td></td><td></td><td>3</td><td></td><td></td></tr><tr><td>C</td><td></td><td></td><td></td><td>6</td><td></td><td>6</td><td></td><td></td><td></td></tr><tr><td>D</td><td></td><td></td><td>6</td><td></td><td>4</td><td></td><td></td><td></td><td></td></tr><tr><td>E</td><td></td><td></td><td></td><td>4</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>F</td><td></td><td></td><td>6</td><td></td><td></td><td></td><td></td><td></td><td></td></tr><tr><td>G</td><td></td><td>3</td><td></td><td></td><td></td><td></td><td></td><td></td><td>3</td></tr><tr><td>H</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>5</td></tr><tr><td>I</td><td></td><td></td><td></td><td></td><td></td><td></td><td>3</td><td>5</td><td></td></tr></table>		A	B	C	D	E	F	G	H	I	A		3								B	3						3			C				6		6				D			6		4					E				4						F			6							G		3							3	H									5	I							3	5	
	A	B	C	D	E	F	G	H	I																																																																																													
A		3																																																																																																				
B	3						3																																																																																															
C				6		6																																																																																																
D			6		4																																																																																																	
E				4																																																																																																		
F			6																																																																																																			
G		3							3																																																																																													
H									5																																																																																													
I							3	5																																																																																														
c)	2	Der Graph kann nicht als Baum dargestellt werden, da er nicht zusammenhängend ist.																																																																																																				

5.

10



80

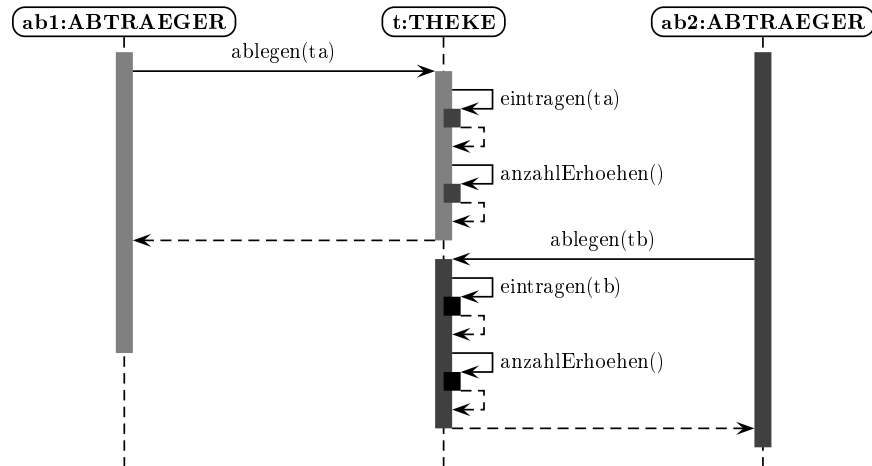
Aufgabe	BE	Hinweise
1.	9	EINKAUF gesamtbetrag einkaufszeit <tätigt 0..1 1 KUNDE kundennr name vorname geburtsdatum guthaben <verwaltet n 1 KASSENSYSTEM tagesumsatz abbuchen(kundennr, betrag) gutschreiben(kundennr, betrag)
2. a)	6	Z. B. eine Auswahl aus folgenden Argumenten: i) Die Liste hat variable Länge, das Feld nicht. ii) In der Liste wird nur der aktuell notwendige Speicherplatz benötigt, beim Feld wird Speicher gegebenenfalls ungenutzt belegt. iii) Einzelne Listenelemente sind leicht zu entfernen, während man im Feld beispielsweise aufwändige Umkopieraktionen benötigt. Mögliche Bewertungen: zu i) Ein Feld würde die Anzahl der Einkäufe begrenzen. zu ii) Ein Feld würde für jeden Kunden Speicher vorhalten, auch wenn dieser nicht einkauft. zu iii) Einzelne Einkäufe werden in der Regel selten storniert, deshalb sind Umkopieraktionen sehr selten.
b)	11	In der abstrakten Klasse LISTENELEMENT: <pre>abstract int gesamtsummeAbhierGeben();</pre> In der Klasse KNOTEN: <pre>int gesamtsummeAbhierGeben(){ return daten.gesamtbetragGeben() + naechster.gesamtsummeAbhierGeben(); }</pre> In der Klasse ABSCHLUSS: <pre>int gesamtsummeAbhierGeben(){ return 0; }</pre> In der Klasse EINKAUFLISTE: <pre>int gesamtsummeGeben(){ return erster.gesamtsummeAbhierGeben(); }</pre>

c)	3	Die Klasse POSTEN muss neu festgelegt werden. EINKAUF erhält eine Beziehung zur Liste der Posten mit den gewünschten Daten. EINKAUF gesamtbetrag einkaufszeit 1 enthält > 1..n POSTEN bezeichner preis
3. a)	4	Die Struktur des geordneten Binärbaumes erlaubt die schnellere Suche eines Schlüssels. Bei 1200 Kunden müssen nur maximal in der Größenordnung $\log_2 1200$ Datenelemente durchsucht werden. Das bedeutet, dass spätestens nach 11 Vergleichen feststeht, ob der Kunde existiert, im Gegensatz zu durchschnittlich 600 Vergleichen bei der sortierten Liste.
b)	14	BINBAUM > 1 BAUMELEMENT {abstract} 2 wurzel ABSCHLUSS KNOTEN nachfolger[^] daten ∨ 1 KUNDE In der Klasse BINBAUM: <pre>Kunde kundeSuchen(int kdnr_gesucht){ return wurzel.kundeSuchen(kdnr_gesucht); }</pre> In der Klasse BAUMELEMENT: <pre>abstract Kunde kundeSuchen(int kdnr_gesucht);</pre> In der Klasse KNOTEN: <pre>Kunde kundeSuchen(int kdnr_gesucht){ if (daten.kundennummerGeben() == kdnr_gesucht){ return daten; }else if (daten.kundennummerGeben() > kdnr_gesucht){ return linkerNachfolger.kundeSuchen(kdnr_gesucht); }else{ return rechterNachfolger.kundeSuchen(kdnr_gesucht); } }</pre> In der Klasse ABSCHLUSS: <pre>Kunde kundeSuchen(int kdnr_gesucht){ return null; }</pre>
c)	3	Der neu eingefügte Teilbaum ist entartet (Liste): Die Sortierung des Baumes bringt hier einen geringeren Geschwindigkeitsvorteil, da der Teilbaum wie eine Liste durchsucht wird.

d)	6	Die Inorder-Ausgabe liefert das Gewünschte. Die Methode <i>inorderAusgeben()</i> von KNOTEN wird rekursiv formuliert: <pre> linkerNachfolger.inorderAusgeben() datenDesEigenenDatenelementsAusgeben() rechterNachfolger.inorderAusgeben() </pre> Die Methode <i>inorderAusgeben()</i> von ABSCHLUSS tut nichts. Die Liste ist aufsteigend sortiert, falls die Kundennummer des linken Nachfolgers eines Baumelements kleiner und die des rechten Nachfolgers größer als die des Baumelements ist.
4. a)	5	Gerichteter Graph, da die Matrix nicht symmetrisch ist:
b)	3	Ein Graph, der als Baum darstellbar sein soll, darf keine Zyklen enthalten. Der Graph aus Teilaufgabe 4a enthält Zyklen.
c)	3	Z. B.: $G - R - K$: Länge 4 $G - M - K$: Länge 5
d)	4	Ein Algorithmus zum Graphendurchlauf ist die Tiefensuche. Dabei wird aber die Gewichtung der Verbindungen zwischen zwei Knoten nicht ausgewertet. Die Tiefensuche ist also für die genannte Aufgabe nicht zielführend. <i>Hinweise:</i> Ebenso gilt das Argument, dass bei der Tiefensuche nicht alle Kanten berücksichtigt werden. Weitere Algorithmen sind z. B. die Breitensuche oder der Dijkstra-Algorithmus, der die genannte Aufgabe erfüllt.
5. a)	6	Z. B. Nennung und Beschreibung eines Phasenmodells. Analyse mit Erstellung des Pflichtenhefts: Ermittlung der Anforderungen an das geplante System und des Leistungsumfangs des fertigen Produkts Entwurf mit Erstellung der Entwurfsspezifikation: Modellierung, Erstellung der notwendigen Diagramme Implementierung: Umsetzung in ausführbaren Programmcode Test mit Integration: Einbau der entwickelten Software in die Hard- und Softwareumgebung und systematisches Testen Bewertung und Abnahme: Abnahmetest beim Auftraggeber zum Nachweis der Erfüllung des Pflichtenhefts, gegebenenfalls Wartung
b)	3	Meilensteine sind festgelegte Zeitpunkte, bis zu denen ein bestimmter Teil des Auftrags erledigt sein muss. Allen Beteiligten sind die einzelnen Teilziele des Projekts klar. Beispiel: Bis zum 1.12.2012 muss das Umfrageergebnis der Eltern zu diesem neuen Bezahlssystem vorliegen.
	80	

Aufgabe	BE	Hinweise
1. a)	4	Kennzeichen = ((Verwaltungsbezirk "–" Kennbuchstaben) ("Y" "–")) Kennziffern . Verwaltungsbezirk = BuchstabeOhneY { Buchstabe } . Kennbuchstaben = Buchstabe { Buchstabe } . Kennziffern = Ziffer { Ziffer } . Buchstabe = BuchstabeOhneY "Y" . BuchstabeOhneY = "A" "B" ... "X" "Z" . Ziffer = "0" "1" "2" "3" "4" "5" "6" "7" "8" "9" .
b)	5	
2. a)	3	Ein Abträger kann in die Situation kommen, dass alle Plätze auf der Theke bereits belegt sind. Der Einräumer kann in die Situation kommen, dass kein Tablett auf der Theke liegt. In beiden Fällen muss der Mitarbeiter warten und es später wieder versuchen, so lange bis die Aktion erfolgreich war.
b)	5	<i>Hinweis:</i> Die nächste freie Position wird hier in einem Attribut vermerkt; sie könnte auch dynamisch festgestellt werden. Der aufeinanderfolgende Aufruf von <i>eintragen(ta)</i> und <i>eintragen(tb)</i> legt beide Tabletts auf dem gleichen Feldelement ab.

- c) 3 Ein Monitor besteht aus einem oder mehreren Programmbereichen, die zu einer Zeit von nur einem Prozess ausgeführt werden können. Wird *ablegen* als Monitor deklariert, können die Tablette nur vollständig nacheinander abgelegt werden.



Hinweis: Das Sequenzdiagramm muss nicht neu gezeichnet werden; die textuelle Beschreibung der Änderung genügt.

- d) 7
- ```

class Theke{
 private int zaehler;
 private Tablett[] platz;
 Theke(){
 platz = new Tablett[10];
 zaehler = 0;
 }
 synchronized boolean ablegen(Tablett t){
 if (zaehler < platz.length){
 platz[zaehler] = t;
 zaehler = zaehler + 1;
 return true;
 }else{
 return false;
 }
 }
 synchronized Tablett wegnehmen(){
 if (zaehler > 0){
 zaehler = zaehler - 1;
 return platz[zaehler];
 }else{
 return null;
 }
 }
}

```

*Hinweise:* Die Art des Wartens bei voller/leerer Theke ist hier nicht angegeben und auch nicht verlangt. Schüler können z. B. das Warten mit *wait/notify* ergänzen. *zaehler* könnte alternativ auch das höchste belegte Feldelement bezeichnen.

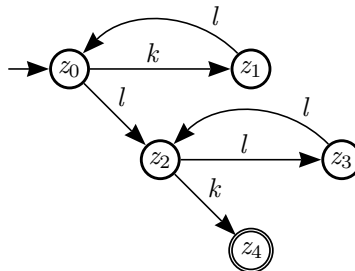
|       |   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|-------|---|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 3. a) | 2 | Geändert wurde nur die Speicherzelle mit der Adresse 102; ihr neuer Wert ist 3.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| b)    | 2 | Die Operation berechnet den Rest bei der ganzzahligen Division.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| c)    | 1 | Die Bedingung prüft, ob Zahl durch Divisor ohne Rest teilbar ist.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| d)    | 8 | <p>Für die Angabe der Registerreferenzen sowie der Datenzellen sind verschiedene Notierungen möglich; der Vorbesetzungswert der Datenzellen kann auch in Prosa beschrieben sein. Als zu testender Wert wurde die Zahl 13 eingesetzt.</p> <p>Die folgende Beispiellösung nutzt symbolische Adressierung mit Marken der Form „name:“ sowie den Ablagebefehl „word“.</p> <pre> startsolange:     load    divisor     cmp     zahl     jmpge   endesolange     load    zahl     div     divisor     mul     divisor     cmp     zahl     jmpne   endewenn1     loadi   1     store   hatteiler  endewenn1:     load    divisor     add     erhoehung     store   divisor     loadi   2     cmp     erhoehung     jmpne   else2     loadi   4     store   erhoehung     jmp     endewenn2  else2:     loadi   2     store   erhoehung  endewenn2:     jmp     startsolange  endesolange:     hold  zahl:     word    13    // zu testender Wert  divisor:     word    5  hatteiler:     word    0  erhoehung:     word    2 </pre> |
| 40    |   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |

| Aufgabe | BE | Hinweise                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------|----|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1. a)   | 3  | <p>i) Der Automat durchläuft die Zustände <math>z_0, z_1, z_2, z_3</math> und ist nach Abarbeiten des Wortes in <math>z_4</math>. Das ist kein Endzustand; also ist llkk nicht zulässig.</p> <p>ii) Der Automat durchläuft die Zustände <math>z_0, z_1, z_2, z_3, z_1, z_2, z_3, z_4</math> und ist nach Abarbeiten des Wortes im Endzustand <math>z_5</math>; also ist llkllkkk zulässig.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| b)      | 9  | <pre> class Automat{     private int zustand;     private void zeichenVerarbeiten(ZEICHEN c){         switch (zustand){             ...             case 3:                 switch (c){                     case 'l': zustand = 1; break;                     case 'k': zustand = 4; break;                     default: zustand = -1; // nicht verlangt                 }                 break;             ...             case 5:                 zustand = -1;                 break;         }     }     boolean codeTesten(ZEICHENKETTE w){         zustand = 0;         for (int i=0; i&lt;w.laenge(); i++){             zeichenVerarbeiten(w.zeichenAn(i));         }         return (zustand == 5);     } } </pre> <p><i>Hinweise:</i></p> <p>Statt ZEICHEN und ZEICHENKETTE dürfen natürlich auch die in der verwendeten Programmiersprache enthaltenen Datentypen und Klassen verwendet werden. Für Java sind das <code>char</code> bzw. <code>String</code> mit den entsprechenden Methoden.</p> <p>Auch eine Auflösung der <code>switch-case</code>-Schachtelung durch eigene Verarbeitungsmethoden für jeden Zustand ist möglich:</p> <pre> ... switch (zustand){     ...     case 3: zeichenVerarbeiten3(c); break;     ... } </pre> |

```
private void zeichenVerarbeiten3(ZEICHEN c){
 switch (c) {
 case 'l': zustand = 1; break;
 case 'k': zustand = 4; break;
 default: zustand = -1; // nicht verlangt
 }
}
```

c)

5



2.

4

Mögliche Argumente:

Die gemeinsame Nutzung hat den Vorteil, dass der Arbeitsspeicher flexibel aufgeteilt werden kann, je nachdem ob mehr Platz für Daten oder für Programmcode benötigt wird.

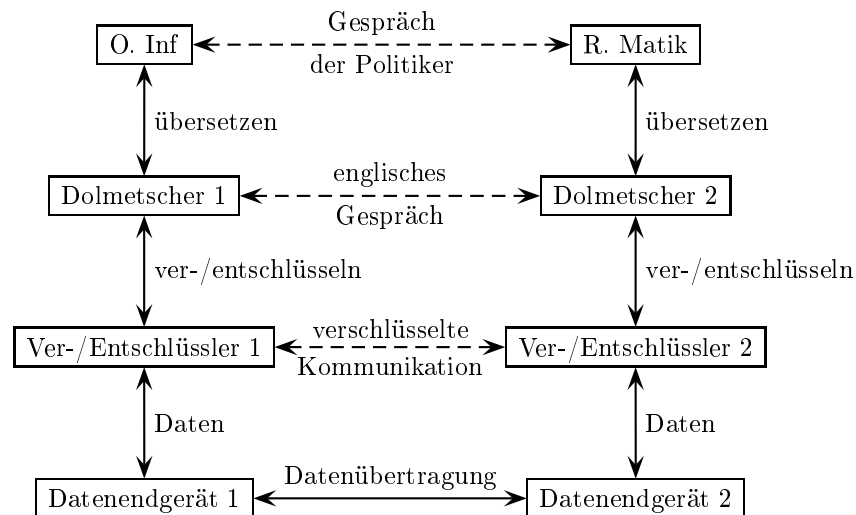
Auch die Möglichkeit, zur Laufzeit Programmcode zu generieren, beruht auf der gemeinsamen Nutzung. Die Daten, welche der Compiler erzeugt, werden als Programmcode verwendet.

Dieses Argument zeigt aber auch die Problematik: Ein Programm könnte zur Laufzeit seinen eigenen Code überschreiben und sich somit leicht zerstören.

Auf den gemeinsam genutzten Speicher muss auch über einen gemeinsamen Bus zugegriffen werden, was zu Engpässen führen kann (Von-Neumann'scher Flaschenhals).

3. a)

6

*Hinweise:*

Auch eine fünfte Schicht, bei der beispielsweise Digitalisierung und Verschlüsselung getrennt werden, ist zulässig.

Für die Verdeutlichung der Aufgaben jeder Schicht genügt, wie in der vorliegenden Lösung angedeutet, ein Stichwort.

|       |    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |   |    |   |   |   |   |   |   |   |   |   |   |
|-------|----|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---|----|---|---|---|---|---|---|---|---|---|---|
| b)    | 4  | <div><div><div><div><div>O. Inf</div><div></div></div><div><div>Dolmetscher 1</div><div></div></div><div><div>Dolmetscher 2</div><div></div></div><div><div>R. Matik</div><div></div></div></div><div><div>Frage<br/>auf Deutsch</div><div>Frage<br/>auf Englisch</div><div>Frage<br/>auf Slowenisch</div><div>Antwort<br/>auf Slowenisch</div><div>Antwort<br/>auf Englisch</div><div>Antwort<br/>auf Deutsch</div></div></div><p><i>Hinweis:</i> Details wie die Länge der Aktivitätsbalken oder die Darstellung der Antwortpfeile sind für die Gültigkeit der Lösung nicht relevant.</p></div> |   |    |   |   |   |   |   |   |   |   |   |   |
| 4. a) | 3  | <table><tr><td>a</td><td>14</td><td>6</td><td>6</td><td>4</td><td>2</td></tr><tr><td>b</td><td>8</td><td>8</td><td>2</td><td>2</td><td>2</td></tr></table> <p>Am Ende haben a und b den Wert 2.</p>                                                                                                                                                                                                                                                                                                                                                                                               | a | 14 | 6 | 6 | 4 | 2 | b | 8 | 8 | 2 | 2 | 2 |
| a     | 14 | 6                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | 6 | 4  | 2 |   |   |   |   |   |   |   |   |   |
| b     | 8  | 8                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | 2 | 2  | 2 |   |   |   |   |   |   |   |   |   |
| b)    | 6  | <p>a steht zu Beginn und am Ende in Speicherzelle 100, b in 101.</p> <pre>1:  load 100 2:  sub 101      // a-b 3:  je 11        // =0, Abbruch 4:  jg 9         // a-b&gt;0, wahr-Fall 5:  load 101     // falsch-Fall 6:  sub 100 7:  store 101    // b=b-a 8:  jmp 1 9:  store 100    // a=a-b 10: jmp 1 11: end</pre> <p>In der gezeigten Lösung wurde der Code durch mehrfaches Ausnutzen der Differenz <math>a - b</math> optimiert. Lösungen, die sich direkter dem Algorithmus anpassen, werden natürlich ebenso akzeptiert.</p>                                                           |   |    |   |   |   |   |   |   |   |   |   |   |
|       | 40 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |   |    |   |   |   |   |   |   |   |   |   |   |