Gesellschaft für Informatik (GI) e. V.

# Bildungsstandards Informatik für die Sekundarstufe II

Empfehlungen der Gesellschaft für Informatik e. V.
erarbeitet vom Arbeitskreis »Bildungsstandards SII«

Die Empfehlungen wurden am 29. Januar 2016
vom Präsidium der GI verabschiedet.


**Arbeitskreis »Bildungsstandards SII«**

des Fachausschusses »Informatische Bildung in Schulen« (FA IBS)
und der Fachgruppe »Didaktik der Informatik« (FG DDI)
im Fachbereich »Informatik und Ausbildung/Didaktik der Informatik« (FB IAD)
der Gesellschaft für Informatik e. V. (GI)

Gerhard Röhner (Dieburg), Prof. Dr. Torsten Brinda (Essen), Volker Denke (Nürnberg),
Dr. Lutz Hellmig (Rostock), Theo Heußer (Laudenbach), Dr. Arno Pasternak (Hagen),
Prof. Dr. Andreas Schwill (Potsdam), Monika Seiffert (Hamburg)

Der Arbeitskreis wurde von Gerhard Röhner koordiniert.



Beilage zu LOG IN, 36. Jg. (2016), Heft Nr. 183/184

# Inhalt

Inhalt

# Vorwort

Vorwort

Am 24. Januar 2008 hat die Gesellschaft für Informatik die Empfehlung »Grundsätze und Standards für die Informatik in der Schule – Bildungsstandards Informatik für die Sekundarstufe I« mit dem Ziel verabschiedet, eine zeitgemäße und fachlich substanzielle Bildung in den Schulen zu befördern. Da im Unterschied zur Sekundarstufe I in allen Bundesländern Informatikunterricht in der Sekundarstufe II angeboten wird, ergab sich die Notwendigkeit, auch Bildungsstandards Informatik für die Oberstufe zu definieren. Arbeitsgruppen der »Fachdidaktischen Gespräche« in Königstein haben sich daher in den Jahren 2009 bis 2011 mit diesem Thema auseinandergesetzt, grundlegende Überlegungen zur Strukturierung der Bildungsstandards Informatik für die Sekundarstufe II angestellt und eine Operatorenliste erarbeitet.

Nachdem die Kultusministerkonferenz am 18.10.2012 Bildungsstandards für die Allgemeine Hochschulreife in den Fächern Deutsch, fortgeführte Fremdsprache und Mathematik vorgelegt hatte, war der Zeitpunkt für die Entwicklung der Bildungsstandards Informatik für die Sekundarstufe II gekommen. Im April 2013 wurde folglich vom GI-Fachausschuss »Informatische Bildung in Schulen« der Arbeitskreis »Bildungsstandards SII« eingerichtet, unter Beteiligung der GI-Fachgruppe »Didaktik der Informatik« und des GI-Fachbereichs »Informatik und Ausbildung/Didaktik der Informatik«. Die Mitglieder des Arbeitskreises haben die vorliegende Fassung erarbeitet. Entwürfe wurden in den Jahren 2014 und 2015 anlässlich der »Fachdidaktischen Gespräche« in Königstein kritisch geprüft. Sehr hilfreich waren dabei die Rückmeldungen von Dr. Hermann Puhlmann, der die Arbeiten an den Bildungsstandards für die Sekundarstufe I koordiniert hatte. Besonderer Dank geht an Prof. Dr. Johannes Magenheim, der Anfang 2015 eine Entwurfsfassung ausführlich begutachtet und seine Ergebnisse mit dem Arbeitskreis diskutiert hat. Auf einem Workshop, den der Arbeitskreis auf der Tagung »Informatik und Schule – INFOS2015« an der Technischen Universität Darmstadt durchgeführt hat, wurde der fachdidaktischen Community ein Entwurf vorgestellt und diskutiert. Rückmeldungen aus dem Workshop sind in eine Onlinebefragung eingeflossen, die im Herbst 2015 durchgeführt wurde. An dieser Befragung haben sich Mitglieder der GI-Landesgruppen, Hochschullehrende und Fachwissenschaftler beteiligt. Sehr ausführlich war dabei die Kommentierung von Prof. Dr. Marco Thomas. Alle im Entwicklungsprozess gegebenen Rückmeldungen wurden vom Arbeitskreis gesichtet und auf resultierenden Änderungsbedarf geprüft. Im Ergebnis wurden die Vorschläge übernommen, für die es im Arbeitskreis einen Konsens gab.

Die Bildungsstandards Informatik für die Sekundarstufe II übernehmen die Struktur der Prozess- und Inhaltsbereiche aus den Standards für die Sekundarstufe I, da diese bereits in vielen Bundesländern in Lehrplänen umgesetzt wurde und somit deren Anschlussfähigkeit sichergestellt werden kann. Sie werden um die Anforderungsbereiche aus den einheitlichen Prüfungsanforderungen in der Abiturprüfung (EPA) Informatik ergänzt, da kein empirisch abgesichertes Kompetenzmodell für die Schulinformatik existiert. Die Prozess- und Inhaltsbereiche stellen im Verständnis dieses Dokuments jeweils Sichten auf die spezifizierten Anforderungen dar, einmal aus der Perspektive informatischer Prozesse und einmal aus der Perspektive informatischer Inhalte. Einzelne Anforderungen haben somit immer einen Prozess- und einen Inhaltsbezug; die jeweilige Zuordnung im Dokument drückt somit eine Schwerpunktsetzung in dieser Hinsicht aus.

Die für die Sekundarstufe I formulierten Mindeststandards werden vorausgesetzt. Im Unterschied zur Sekundarstufe I werden Regelstandards für den Abschluss der gymnasialen Oberstufe definiert. Die zugehörigen Kompetenzen sollen bezogen auf das belegte Anforderungsniveau von Schülerinnen und Schülern im Fach Informatik erreicht und in Prüfungen nachgewiesen werden. In den vorliegenden Standards wird zwischen grundlegendem und erhöhtem Anforderungsniveau differenziert, um Anforderungsunterschiede zwischen Grund- und Leistungskurs zu beschreiben.

Im Detail gibt es zwischen den Bildungsstandards für die Sekundarstufen I und II Unterschiede beim strukturellen Aufbau. Eine Differenzierung der Standards nach Jahrgangsstufen erschien für die Sekundarstufe II nicht nötig. Im vorliegenden Dokument werden die formulierten Anforderungen aus der Perspektive der Prozessbereiche gemäß den drei EPA-Anforderungsbereichen geordnet. Ein weiterer Unterschied besteht darin, dass in Analogie zu Standards anderer Fächer bei den vorliegenden Bildungsstandards zuerst die Prozess- und dann die Inhaltsbereiche aufgeführt werden.

Weiterhin muss auf einen Unterschied in der Begriffsverwendung gegenüber den SI-Standards aufmerksam gemacht werden. In den SI-Standards wird unter Modellieren der komplette Modellierungszyklus mit Problemanalyse, Modellbildung, Implementierung und Modellkritik verstanden. Da für die Problemanalyse insbesondere Kompetenzen aus den Prozessbereichen »Strukturieren und Vernetzen« sowie »Darstellen und Interpretieren« gefordert und für die Modellkritik Kompetenzen aus dem Prozessbereich »Begründen und Bewerten« erforderlich sind, wird nun das Modellieren auf die eigentliche Modellbildung beschränkt. In der Konsequenz wurde »Modellieren« ebenfalls in die Operatorenliste aufgenommen.

Die vorliegenden Bildungsstandards enthalten ganz bewusst auch werkzeugbezogene Kompetenzen. So sollen z. B. Entwicklungsumgebungen funktionell zur Implementierung eines Modells eingesetzt und digitale Kommunikations- und Kooperationssysteme genutzt werden. Die reflektierte Nutzung von Informatiksystemen beinhaltet ein grundlegendes Verständnis der Funktionalitäten, gibt Einblick in die Wirkungsweisen digitaler Medien und macht deutlich, welchen spezifischen Beitrag Informatiksysteme gegenüber anderen Medien leisten können. Somit trägt die informatische Bildung wesentlich zu einer zeitgemäßen digitalen Bildung bei.

Mit den vorliegenden Bildungsstandards für die Sekundarstufe II wird das Ziel verfolgt, die Qualität des Informatikunterrichts in der Oberstufe zu verbessern. Dies gelingt durch einen kompetenzorientierten Unterricht, der sich in Planung, Durchführung und Reflexion an den Kompetenzen der Standards orientiert. Dabei beschreiben die Prozessbereiche, wie sich die Schülerinnen und Schüler mit Fachinhalten auseinandersetzen. Die Inhaltsbereiche legen fest, in welchen Gebieten der Informatik Kompetenzen erworben werden.

Das vorliegende Dokument ist im Weiteren wie folgt strukturiert: In Kapitel 1 wird die diesem Dokument zugrundeliegende Sicht auf die Fachwissenschaft Informatik einerseits und den Informatikunterricht andererseits konturiert. In Kapitel 2 wird das Kompetenzmodell der vorliegenden Standards beschrieben. Es baut auf dem Modell auf, das den Bildungsstandards für die Sekundarstufe I zugrunde liegt. Kapitel 3 enthält dann die eigentlichen Standards differenziert nach Prozess- und Inhaltssicht.

In Kapitel 4 werden Aufgabenbeispiele dargestellt, die auf den in Kapitel 3 angegebenen Bildungsstandards basieren. Die einführenden Erläuterungen klären das Verhältnis zwischen Aufgaben zum Lernen und Leisten. Dementsprechend folgt eine Liste schriftlicher und mündlicher Leistungsaufgaben sowie von Lernaufgaben. Die Aufgabenbeispiele sollen beispielhaft aufzeigen, wie Kompetenzen nachgewiesen bzw. erworben werden können, und sie konkretisieren das erwartete fachliche Niveau. Sie sind mithilfe von Operatoren formuliert sowie den Prozess-, Inhalts- und Anforderungsbereichen zugeordnet, wodurch der Bezug zu den Bildungsstandards explizit und transparent gemacht wird.

*Gerhard Röhner*
für den Arbeitskreis »Bildungsstandards SII«
Dezember 2015

# Fachpräambel

1

Informatik ist die Wissenschaft von der theoretischen Analyse und Konzeption, der organisatorischen und technischen Gestaltung und der konkreten Realisierung komplexer Informatiksysteme. Unter einem Informatiksystem wird eine spezifische Zusammenstellung von Hardware-, Software- und Netzwerkkomponenten zur Lösung eines Anwendungsproblems verstanden – einschließlich nichttechnischer Aspekte, die sich durch die Einbettung in gesellschaftliche Kontexte ergeben. Informatik hat daher neben mathematischen und ingenieurwissenschaftlichen auch gesellschafts-, geistes- und naturwissenschaftliche Züge.

Obwohl die Hauptprodukte der Informatik im Unterschied zu den traditionellen Ingenieurwissenschaften immateriell sind, haben sie dennoch eine weitreichende Bedeutung in der Realität. Strategien zur Entwicklung eines Informatiksystems werden vor dem Hintergrund von Konzepten gemacht bzw. gewonnen, die nicht direkt aus Erfahrungen ableitbar sind.

Die Informationsgesellschaft verlangt daher nach einer neuen, zusätzlichen Sichtweise innerhalb der Allgemeinbildung, der informatischen Bildung. Bezugswissenschaft ist die Informatik, die allgemeine Gesetzmäßigkeiten informationsverarbeitender Prozesse in Gesellschaft, Natur und Technik untersucht und diese Prozesse in Informatiksystemen transparent macht. Die Informatik ergänzt und überschreitet die Gegenstandsbereiche und Methodenspektren anderer Fachdisziplinen. Informatisches Entwickeln und Problemlösen ist auch ein kreativer Prozess, in dem Theorie, Abstraktion und Design verknüpft sind. Die Denkweisen und Werkzeuge der Informatik haben inzwischen in allen Gebieten von Wissenschaft, Wirtschaft und Technik Eingang gefunden. Auch wer sich nicht aktiv mit Informatiksystemen beschäftigt, gehört zumindest zur Gruppe der Betroffenen.

Im Informatikunterricht erhalten Schülerinnen und Schüler vielfältige Gelegenheiten zur Entwicklung und Ausbildung von Kompetenzen, die sie befähigen, ihr Leben in einer Informationsgesellschaft selbstbestimmt zu führen und zu gestalten. Sie nutzen dabei informatische Konzepte, um Elemente ihrer Erfahrungswelt zu verstehen, d. h. zu ordnen, zu erklären, zu gestalten und gegebenenfalls zu beeinflussen. Das Verständnis für eine informatische Sicht der Welt erschließt sich für Schülerinnen und Schüler dabei nicht nur aus der alltäglichen Erfahrung mit digitalen Medien, zumal sich diese fortwährend rasch ändern oder erweitern, sondern erfordert einen Perspektivenwechsel von der Lebenswelt hin zu fachlich fundierter, wissenschaftspropädeutischer Auseinandersetzung.

Die Ausbildung von Fach-, Methoden-, Sozial- und Selbstkompetenz erfolgt im Informatikunterricht in der Regel ganzheitlich und wechselseitig. In der Auseinandersetzung mit Problemstellungen aus dem Bereich der Informations- und Kommunikationstechnologien werden Fähigkeiten und Fertigkeiten zur informatischen Analyse von Sachverhalten ausgebildet, die sich auf konkrete Lebenssituationen beziehen. Hierzu gehört der Umgang mit Modellierungs- und Strukturierungskonzepten, Softwarewerkzeugen und Programmiersprachen. Der Grad ihrer Verwendung richtet sich nach dem notwendigen Beitrag für das informatische Verständnis eines Zusammenhangs. Die ausdauernde, ziel- und ergebnisorientierte informatische Bearbeitung komplexerer Fragestellungen in Teamarbeit trägt zur Entwicklung von Qualifikationen für Ausbildung und Studium bei.

# Kompetenzmodell

2

## Prozess- und Inhaltsbereiche

Die Bildungsstandards Informatik für die Sekundarstufe II sind nach den etablierten Prozess- und Inhaltsbereichen der Bildungsstandards Informatik für die Sekundarstufe I strukturiert.

| **Prozessbereiche** | | **Inhaltsbereiche** | |
|---|---|---|---|
| MI | Modellieren und Implementieren | ID | Information und Daten |
| BB | Begründen und Bewerten | AL | Algorithmen |
| SV | Strukturieren und Vernetzen | SA | Sprachen und Automaten |
| KK | Kommunizieren und Kooperieren | IS | Informatiksysteme |
| DI | Darstellen und Interpretieren | IMG | Informatik, Mensch und Gesellschaft |

Die aufgeführten Prozess- und Inhaltsbereiche sind Ergebnis eines langjährigen Diskussionsprozesses der fachdidaktischen Gemeinschaft. Durch sie werden in der Schule wesentliche Kompetenzen informatischer Bildung abgedeckt. Mit der Ausweisung von jeweils fünf Prozess- und Inhaltsbereichen wird deutlich, dass in einem guten Informatikunterricht vielfältige Kompetenzen erworben werden.

Informatische Kompetenzen erwachsen in der aktiven Auseinandersetzung mit den Inhalten. Die Formen der Auseinandersetzung werden in den Prozessbereichen beschrieben. Die Prozess- und Inhaltsbereiche sind untrennbar und vielfältig miteinander verzahnt. Das bedeutet, dass verschiedene Inhalte beispielsweise dargestellt und interpretiert werden. Umgekehrt wird beispielsweise der Inhaltsbereich Informatiksysteme anhand von Tätigkeiten aus verschiedenen Prozessbereichen erschlossen.

## Anforderungsbereiche

Das Kompetenzmodell der Bildungsstandards Informatik der Sekundarstufe II umfasst neben den Prozess- und Inhaltsbereichen mit den Anforderungsbereichen eine dritte Dimension. Dabei beschreiben die Anforderungsbereiche unterschiedliche kognitive Ansprüche von informatischen Aktivitäten, die daher in den Prozessbereichen ausdifferenziert werden. Da ein empirisch abgesichertes Kompetenzmodell fehlt, stellen die Anforderungsbereiche keine Ausprägungen oder Niveaustufen einer Kompetenz dar.

Drei Anforderungsbereiche werden unterschieden (siehe auch Abbildung 2.01, nächste Seite):

- I Reproduktion,
- II Reorganisation und Transfer,
- III Reflexion und Problemlösung.

Die nachfolgende Beschreibung der Anforderungsbereiche entspricht den Einheitlichen Prüfungsanforderungen in der Abiturprüfung Informatik (EPA Informatik, vgl. KMK, 2004). Sie beruhen auf der praktischen Erfahrung in der Schule und auf einschlägigen Aufgabenformaten aus vorhandenen Klausur- und Abituraufgaben. Für Aussagen über die Angemessenheit, Qualität und Komplexität der Anforderungen stellen die Anforderungsbereiche einen Orientierungsrahmen dar, in dem sich die Leistungen von Schülerinnen und Schülern erfahrungsgemäß bewegen.

Der *Anforderungsbereich **I Reproduktion*** umfasst

- die Wiedergabe von bekannten Sachverhalten aus einem abgegrenzten Gebiet im gelernten Zusammenhang,
- die Beschreibung und Darstellung bekannter Verfahren, Methoden und Prinzipien der Informatik,

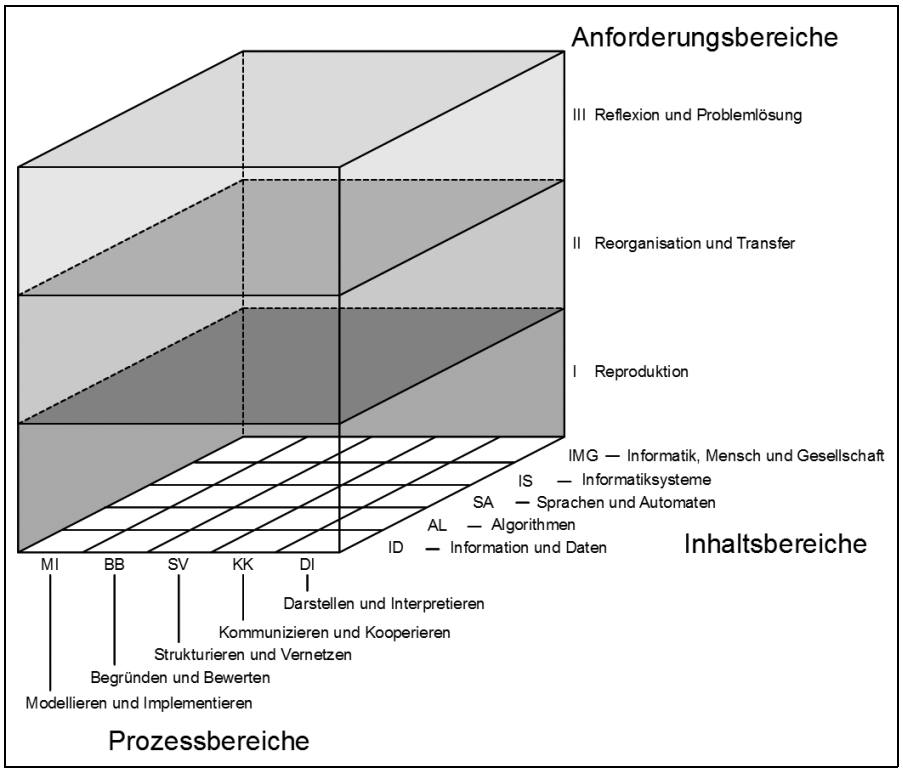



**Abbildung 2.01: Kompetenzmodell der Bildungsstandards Informatik für die Sekundarstufe II.**

▷ die Beschreibung und Verwendung gelernter und geübter Arbeitstechniken und Verfahrensweisen in einem begrenzten Gebiet und in einem wiederholenden Zusammenhang.

Der *Anforderungsbereich **II Reorganisation und Transfer*** umfasst

▷ die selbstständige Verwendung (Auswählen, Anordnen, Verarbeiten und Darstellen) bekannter Sachverhalte zur Bearbeitung neuer Frage- oder Problemstellungen unter vorgegebenen Gesichtspunkten in einem durch Übung bekannten Zusammenhang,

▷ die selbstständige Übertragung des Gelernten auf vergleichbare neue Situationen, wobei es entweder um veränderte Fragestellungen, veränderte Sachzusammenhänge oder um abgewandelte Verfahrensweisen gehen kann,

▷ die Anwendung bekannter Verfahren, Methoden und Prinzipien der Informatik zur Lösung eines neuen Problems aus einem bekannten Problemkreis.

Der *Anforderungsbereich **III Reflexion und Problemlösung*** umfasst

▷ die planmäßige Verarbeitung komplexer Gegebenheiten mit dem Ziel, zu selbstständigen Gestaltungen, Deutungen, Folgerungen, Begründungen bzw. Wertungen zu gelangen,

▷ die bewusste und selbstständige Auswahl und Anpassung geeigneter gelernter Methoden und Verfahren in neuartigen Situationen. Dabei werden aus gelernten Denkmethoden bzw. Lösungsverfahren die zur Bewältigung der Aufgabe geeigneten selbstständig ausgewählt und einer neuen Problemstellung angepasst.

In der Schule können die Anforderungsbereiche als methodisches Grundprinzip zur Gestaltung von Lernprozessen angewendet werden. Sie realisieren das didaktische Prinzip vom Einfachen zum Komplexen. Lernaufgaben, die unter Berücksichtigung der Anforderungsbereiche entworfen sind, ermöglichen einen leichten Einstieg, haben ihren Schwerpunkt in der Analyse, Reorganisation und dem Transfer des Fachinhalts, ermöglichen aber auch Reflexion und selbstständiges Problemlösen. Sie weisen differenzierte und offene Aufgabenstellungen auf. Die Orientierung an Anforderungsbereichen berücksichtigt die Heterogenität im Informatikunterricht und liefert somit auch einen Beitrag zur Binnendifferenzierung und Individualisierung des Unterrichts.

Den Teilaufgaben der in Kapitel 4 dargestellten Aufgabenbeispiele wurden Anforderungsbereichen zugeordnet. Da dies nicht immer eindeutig möglich war, sind manche Teilaufgaben auch mehreren Anforderungsbereichen zugeordnet. Die Zuordnung erfolgt durch die Angabe von Bewertungseinheiten pro Anforderungsbereich. Im Verständnis dieses Dokuments liegt bei gut konzipierten Lern- und Prüfungsaufgaben der Anteil des Anforderungsbereichs II bei 50 %, und Anforderungsbereich I hat einen höheren Anteil als Anforderungsbereich III. Dadurch werden Einseitigkeiten vermieden und die Vergleichbarkeit der Prüfungsaufgaben sowie der Bewertung der Prüfungsleistung verbessert.

Das durch die Prozess-, Inhalts- und Anforderungsbereiche repräsentierte Kompetenzmodell der Bildungsstandards Informatik unterstützt die Übersetzung allgemeiner Bildungsziele in Unterrichtsvorhaben und konkrete Aufgabenstellungen. Es stellt somit ein Bindeglied zwischen den Kompetenzen, den Lernprozessen im Unterricht sowie den Aufgaben in Prüfungssituationen dar.

# Bildungsstandards im Fach Informatik

3

## Prozessbereiche

In den Prozessbereichen wird beschrieben, auf welche Art und Weise die Schülerinnen und Schüler mit Fachinhalten umgehen sollen. Damit sind inhärent kognitive Fähigkeiten und Fertigkeiten verbunden, die fachspezifisch geprägt, aber nicht an spezielle informatische Inhalte gebunden sind. Sie können von Schülerinnen und Schülern nur in aktiver Auseinandersetzung mit Inhalten erworben werden und befähigen sie, ihr erworbenes Wissen und Können anzuwenden, auf neue Situationen zu übertragen und Probleme zu lösen. Die Prozessbereiche werden jeweils durch zwei Verben benannt, deren Bedeutung im informatischen Kontext geklärt wird.

### Modellieren und Implementieren

*Modellieren und Implementieren*

Modellieren und Implementieren sind die zentralen Teile des Modellierungskreislaufes, bei dem ein Problem analysiert, ein informatisches Modell entworfen und auf einem Informatiksystem implementiert, getestet und bewertet wird.

Unter Modellieren wird das Abbilden eines Realitätsausschnitts oder eines geplanten Systems durch Abstraktion zu einem bestimmten Zweck verstanden. Dazu müssen zuvor in einer Problemanalyse Sachverhalte und Abläufe unter informatischer Perspektive mit Blick auf verallgemeinerbare und typische Bestandteile untersucht werden. Das daraus entstehende Modell muss formal darstellbar sein und eine Realisierung mit einem Informatiksystem ermöglichen.

Implementieren ist das Umsetzen eines Modells auf einem Informatiksystem, wodurch das Ergebnis des Modellierens erlebbar wird. Zum Implementieren gehört das Testen der Problemlösung. In einer anschließenden Reflexion werden die Qualität sowie Einsatzmöglichkeiten und Grenzen des entwickelten Produkts bewertet, was zu einer Modifikation des Modells führen kann.

Die drei Anforderungsbereiche zu diesem Prozessbereich lassen sich wie folgt beschreiben:

***I Reproduktion***

Die Schülerinnen und Schüler
- ▷ geben ein bekanntes Modell in vertrauter Darstellung wieder,
- ▷ erkunden ein vorgegebenes Modell oder erproben eine gegebene Implementierung,
- ▷ testen eine Implementierung mit vorgegebenen Testfällen.

***II Reorganisation und Transfer***

Die Schülerinnen und Schüler
- ▷ prüfen die Eignung eines vorhandenen informatischen Modells für die Lösung einer Problemstellung,
- ▷ führen gemäß einer Problemanalyse eine Modellierung und Implementierung mit einem bekannten Modellierungsansatz durch,
- ▷ setzen zur Implementierung eines Modells Entwicklungsumgebungen funktionell ein,
- ▷ testen systematisch eine Implementierung auf korrekte Funktionalität, auch in Bezug auf Sonderfälle.

**Referenzieren der Prozessbereiche**

Um einzelne Standards aus den Prozessbereichen in anderen Dokumenten eindeutig zu referenzieren, werden Identifikationen verwendet, die wie folgt zusammengesetzt sind:

1. Akronym des Prozessbereichs, in dem der zu referenzierende Standard enthalten ist, also MI, BB, SV, KK oder DI (siehe auch Seite 3);
2. Nummer I, II bzw. III des Anforderungsbereichs, in dem der zu referenzierende Standard enthalten ist;
3. laufende Nummer des jeweiligen Standards im entsprechenden Anforderungsbereich, also 1, 2, 3 usw.

Die einzelnen Bestandteile werden durch Bindestriche verbunden.

*Beispiel:* Der dritte Standard im Anforderungsbereich I des Prozessbereichs *Modellieren und Implementieren* (»Die Schülerinnen und Schüler testen eine Implementierung mit vorgegebenen Testfällen«) ist damit referenzierbar über den Identifikator MI-I-3.

***III Reflexion und Problemlösung***

Die Schülerinnen und Schüler
- führen gemäß einer Analyse eines komplexen Problems eine Modellierung und Implementierung mit einem Modellierungsansatz durch,
- überarbeiten die eigene Lösung unter Berücksichtigung von Effizienz, Allgemeinheit und Wiederverwendbarkeit,
- reflektieren den Lösungsprozess und nutzen die Erkenntnisse beim weiteren Modellieren und Implementieren.

## *Begründen und Bewerten*

**Begründen und Bewerten**

Beim Begründen stützen die Schülerinnen und Schüler eine gegebene Aussage oder einen Sachverhalt durch rational nachvollziehbare Argumente. Durch logisches Schließen bilden sie Argumentationsketten. Wird eine Aussage unter Verwendung transparenter, fachlicher Kriterien zusätzlich überprüft und dadurch bestätigt, relativiert, entkräftet oder widerlegt, so spricht man von Beurteilen.

Das Bewerten zielt auf die Formulierung eines Werturteils ab, das unter Einbeziehung des Kontextes und der Verwendung transparenter und sachgerechter Bewertungskriterien und -maßstäbe zustande kommt. Dabei werden gegebenenfalls die Argumente anderer aufgenommen und geprüft, und der persönliche Standpunkt wird unter Zuhilfenahme geeigneter Fachbegriffe dargestellt und begründet.

Die drei Anforderungsbereiche zu diesem Prozessbereich lassen sich wie folgt beschreiben:

***I Reproduktion***

Die Schülerinnen und Schüler
- geben bekannte Argumentationen wieder,
- bestätigen oder widerlegen eine Aussage durch einschrittiges logisches Schließen,
- geben Begründungen in bekannten Zusammenhängen wieder.

***II Reorganisation und Transfer***

Die Schülerinnen und Schüler
- begründen mithilfe eigener Argumente oder Argumentationsketten,
- bewerten fachliche Darstellungen und die Eignung von Darstellungsformen,
- beurteilen informatische Sachverhalte anhand gegebener Kriterien,
- bewerten Informatiksysteme unter z.B. fachlichen, ethischen, ökologischen, ökonomischen und rechtlichen Aspekten.

***III Reflexion und Problemlösung***

Die Schülerinnen und Schüler
- begründen komplexere informatische Sachverhalte und entwickeln dafür Argumentationsketten,
- beurteilen bzw. bewerten informatische Sachverhalte – auch in authentischen Darstellungen – oder die Eignung von Informatiksystemen anhand selbst gewählter fachlicher sowie für die Nutzung relevanter Kriterien,
- bewerten den eigenen oder gemeinsamen Arbeitsprozess und dessen Ergebnisse und ziehen Schlüsse für ihr zukünftiges Handeln.

## *Strukturieren und Vernetzen*

**Strukturieren und Vernetzen**

Beim Strukturieren werden Sachverhalte aus informatischer Sicht analysiert, Gegenstände und Prozesse sowie ihr Zusammenwirken systematisch erfasst. Die Informatik verfügt dazu über einen Vorrat an Strukturierungsmethoden zur Problemlösung und zur Repräsentation von Information.

Beim Vernetzen werden bestehende Zusammenhänge, Wirkungen und Analogien in und außerhalb der Informatik erkannt, neue Inhalte und Prozesse in das eigene Denk- und Wissensschema integriert und kognitiv verknüpft sowie in anderen Kontexten und Anwendungsbereichen eingesetzt.

Die drei Anforderungsbereiche zu diesem Prozessbereich lassen sich wie folgt beschreiben:

***I Reproduktion***

Die Schülerinnen und Schüler
- ▷ analysieren die Struktur eines einfachen Ausschnitts der Lebenswelt,
- ▷ identifizieren Elemente und ihre Beziehungen in einem Sachzusammenhang,
- ▷ geben Beziehungen zwischen Fachbegriffen wieder.

***II Reorganisation und Transfer***

Die Schülerinnen und Schüler
- ▷ analysieren im Rahmen eines planvollen Vorgehens die Struktur von Ausschnitten der Lebenswelt,
- ▷ untersuchen Abläufe und Wirkungszusammenhänge unter informatischen Aspekten,
- ▷ beschreiben Beziehungen zwischen informatischen Inhalten oder Vorgehensweisen, um Neues mit Bekanntem zu vernetzen,
- ▷ verwenden sequenzielle, hierarchische oder netzartige Strukturen zur Darstellung von Fachbegriffen und Inhalten.

***III Reflexion und Problemlösung***

Die Schülerinnen und Schüler
- ▷ strukturieren ihr Wissen und ihren Wissenserwerb selbstständig, auch mithilfe von Informatiksystemen,
- ▷ übertragen Erkenntnisse auf neue Problemstellungen,
- ▷ verknüpfen informatische Inhalte mit solchen in und außerhalb der Informatik.

## Kommunizieren und Kooperieren

*Kommunizieren und Kooperieren*

Kommunikation dient sowohl der angemessenen mündlichen und schriftlichen Verständigung unter Verwendung der Fachsprache als auch der Anwendung von Methoden zur Informationserschließung aus unterschiedlichen Quellen. Zur mündlichen Kommunikation gehören adressatengerechtes Sprechen über Fachinhalte, aktives Zuhören, zielführende Beiträge zu Diskussionen sowie Präsentieren umfangreicher Beiträge unter Verwendung adäquater Medien. Die Dokumentation von Problemlösungen und Projekten mit fachsprachlich genauen, aufgabenadäquaten sowie inhaltlich und formal gut strukturierten Darstellungen, z. B. Texte oder Diagramme, kennzeichnen gute schriftliche Kommunikation. Texterschließung basiert auf Methoden des Sprachunterrichts, wobei Alltags- und Fachwissen genutzt werden, um sich neue Inhalte und Kontexte anhand von Fachtexten selbstständig anzueignen.

Kooperationsfähigkeit ist Voraussetzung für gute Zusammenarbeit im Informatikunterricht und insbesondere für Team- und Projektarbeit. Gemeinsame Ziele, die Arbeitsaufteilung und Zuständigkeiten werden vereinbart, Vereinbarungen getroffen, Schnittstellen definiert und Termine geplant. Kooperationsfähigkeit zeigt sich in der Übernahme von Verantwortung für den eigenen Bereich und das gemeinsame Ziel, in der Einhaltung von Absprachen, der gegenseitigen Hilfe und dem effektiven Zusammenarbeiten in guter Atmosphäre. Auftretende Konflikte werden respektvoll und sachbezogen gelöst.

Zur Kommunikation und Kooperation werden auch netzbasierte Plattformen genutzt und deren Möglichkeiten, Chancen und Risiken reflektiert.

Die drei Anforderungsbereiche zu diesem Prozessbereich lassen sich wie folgt beschreiben:

***I Reproduktion***

Die Schülerinnen und Schüler
- ▷ erschließen aus leicht erfassbaren Texten und Diagrammen Informationen mit informatischem Gehalt,
- ▷ geben einfache informatische Sachverhalte unter Benutzung der Fachsprache schriftlich oder mündlich wieder,
- ▷ kommunizieren fachgerecht über Texte und Diagramme mit informatischem Gehalt,
- ▷ nutzen digitale Kommunikations- und Kooperationssysteme,
- ▷ organisieren und koordinieren ihre Arbeitsschritte mit Unterstützung.

***II Reorganisation und Transfer***

Die Schülerinnen und Schüler
- ▷ erläutern informatische Sachverhalte fachsprachlich genau, kommunizieren adressatengerecht und stellen problembezogene Fragen,
- ▷ wählen selbstständig digitale Kommunikations- und Kooperationssysteme zweckangemessen aus und verwenden sie sachgerecht,
- ▷ beachten vereinbarte Konventionen bezüglich der Werkzeuge, Techniken und Dokumente,
- ▷ nutzen geeignete informatische Methoden zur Dokumentation, Versionierung und Archivierung,
- ▷ kooperieren bei der Lösung informatischer Probleme und wenden ein Vorgehensmodell bei der Durchführung ihrer Projekte an.

***III Reflexion und Problemlösung***

Die Schülerinnen und Schüler
- ▷ erläutern adressatengerecht eine komplexe informatische Lösung in Präsentation und Diskussion,
- ▷ reflektieren den Einsatz digitaler Kommunikations-, Kooperations- und Kollaborationssysteme,
- ▷ diskutieren Strategien und Methoden der Problemlösung und reflektieren diese.

## *Darstellen und Interpretieren*

**Darstellen und Interpretieren**

Konzepte und Sachverhalte der Informatik werden in vielfältigen Formen und verschieden stark formalisiert dargestellt. Die typische Vorgehensweise der Informatik beginnt mit der begründeten Auswahl einer Darstellungsform zu einem Sachverhalt, die häufig durch den Modellierungsansatz bedingt ist. Daran schließt sich die Darstellung mit informatischen Werkzeugen und die Übertragung in andere Darstellungsformen an. Formale Darstellungen ermöglichen die automatische Informationsverarbeitung und dienen der fachlichen Kommunikation.

Darstellungen werden im Hinblick auf den modellierten Realitätsausschnitt interpretiert. Dies beinhaltet ihre detaillierte Analyse sowie die Untersuchung und Deutung der enthaltenen Elemente und ihrer Beziehungen. Das Interpretieren ist eine Grundlage für die Beurteilung von Sachverhalten. Dazu gehört auch die Berücksichtigung des Kontextes. Aus Daten wird durch Interpretation Information gewonnen.

Die drei Anforderungsbereiche zu diesem Prozessbereich lassen sich wie folgt beschreiben:

***I Reproduktion***

Die Schülerinnen und Schüler
- ▷ fertigen Darstellungen einfacher informatischer Sachverhalte an,
- ▷ interpretieren Darstellungen von einfachen Modellen und Algorithmen,
- ▷ übertragen mithilfe eines Routineverfahrens eine Darstellung in eine andere Darstellungsform.

***II Reorganisation und Transfer***

Die Schülerinnen und Schüler
- ▷ analysieren und interpretieren gegebene Darstellungen im Detail und im Zusammenhang,
- ▷ stellen Modelle, Algorithmen und andere informatische Inhalte grafisch oder sprachlich strukturiert dar,
- ▷ passen Darstellungen zielgerichtet an neue Anforderungen an,
- ▷ wählen problemadäquat eine Darstellungsform aus,
- ▷ dokumentieren eine Problemlösung mit angemessenen Darstellungsmitteln.

***III Reflexion und Problemlösung***

Die Schülerinnen und Schüler
- ▷ interpretieren Darstellungen höherer Komplexität, in neuen Kontexten oder unvertrauten Formen,
- ▷ reflektieren Darstellungen und Darstellungsformen sowie ihre Auswahl kritisch.

# Inhaltsbereiche

Die Inhaltsbereiche legen fest, in welchen Gebieten der Informatik die Schülerinnen und Schüler Kompetenzen erwerben, über welches fachliche Wissen und Können sie in diesen Gebieten verfügen sollen. Wegen der begrenzten Unterrichtszeit muss eine Beschränkung auf Inhaltsbereiche erfolgen, die einerseits schulisch umsetzbar sind, andererseits aber vielfältige Kompetenzen in relevanten Gebieten der Informatik ermöglichen. Die Auswahl der Inhaltsbereiche ist in den *Bildungsstandards Informatik für die Sekundarstufe I* begründet (vgl. AKBSI, 2008, S. 11 f.). Die Begriffe zur Bezeichnung der Inhaltsbereiche werden soweit nötig erklärt.

Bei den Inhaltsbereichen wird eine Unterscheidung zwischen grundlegendem und erhöhtem Anforderungsniveau gemacht, wobei Anforderungen des grundlegenden Niveaus auch immer zum erhöhten Niveau gehören. Diese Unterscheidung berücksichtigt, dass bei erhöhtem Niveau mehr Lernzeit zur Verfügung steht und somit zusätzliche oder vertiefte Kompetenzen erworben werden. Deshalb ist in Prüfungen des erhöhten Niveaus der Anteil der Aufgaben im Anforderungsbereich III höher und dafür der Anteil aus Anforderungsbereich I niedriger.

**Referenzieren der Inhaltsbereiche**

Identifikatoren für Standards aus den Inhaltsbereichen werden analog zu den Prozessbereichen (siehe Seite 5) aus folgenden Komponenten gebildet:

1. Akronym des Inhaltsbereichs, in dem der zu referenzierende Standard enthalten ist, also ID, AL, SA, IS oder IMG (siehe auch Seite 3);
2. Akronym GEA, falls der zu referenzierende Standard im grundlegenden und erhöhten Anforderungsniveau zu verorten ist, und sonst EA (erhöhtes Anforderungsniveau);
3. laufende Nummer des jeweiligen Standards im entsprechenden Anforderungsniveau, also 1, 2, 3 usw.

Die einzelnen Bestandteile werden durch Bindestriche verbunden.

*Beispiel:* Der dritte Standard im grundlegenden und erhöhten Anforderungsniveau des Inhaltsbereichs *Algorithmen* (»Die Schülerinnen und Schüler entwerfen Algorithmen und stellen sie in geeigneter Form dar«, siehe Seite 10) ist damit referenzierbar über den Identifikator AL-GEA-3.

## *Information und Daten*

**Information und Daten**

Information ist der kontextbezogene Bedeutungsgehalt einer Aussage, Beschreibung, Anweisung, Mitteilung oder Nachricht. In der Informatik dominiert die systematische Darstellung und automatische Verarbeitung von Daten als Träger von Information.

Daten sind eine Darstellung von Information in formalisierter Art, geeignet zur Kommunikation, Interpretation und Verarbeitung. Sie werden durch Zeichenfolgen repräsentiert, deren Aufbau einer vereinbarten Syntax folgt. Daten werden wieder zu Information, wenn sie in einem Bedeutungskontext interpretiert werden.

***Grundlegendes und erhöhtes Anforderungsniveau***

Die Schülerinnen und Schüler
- ▷ unterscheiden zwischen Zeichen, Daten und Information sowie zwischen Syntax und Semantik,
- ▷ analysieren Daten hinsichtlich ihrer Struktur,

▷ bilden Information als Daten mit Datentypen und in Datenstrukturen ab,
▷ verwenden, modellieren und implementieren Operationen auf statischen und dynamischen Datenstrukturen,
▷ erstellen zu einem Realitätsausschnitt ein Datenmodell und implementieren es als Datenbank,
▷ untersuchen und organisieren Daten unter Beachtung von Redundanz, Konsistenz und Persistenz,
▷ verwenden eine Abfragesprache zur Anzeige und Manipulation von Daten und interpretieren die Daten.

***Erhöhtes Anforderungsniveau***

Die Schülerinnen und Schüler
▷ verwenden, modellieren und implementieren Operationen auf komplexen Datenstrukturen,
▷ entwickeln zu einem Ausschnitt der Lebenswelt mit komplexen Beziehungen eine Datenbank.

## *Algorithmen*

**Algorithmen**

Algorithmen sind endliche Beschreibungen von Abläufen zur Lösung von Problemen und ergeben bei einer Ausführung eine eindeutig definierte Abfolge von Handlungen. Eine automatische Ausführung auf einem Computer bedarf der Formulierung in einer Programmiersprache.

Komplexe Probleme lassen sich lösen, wenn neben den algorithmischen Grundbausteinen geeignete Entwurfsmethoden genutzt und Datenstrukturen entwickelt werden. Die Implementierung eines Algorithmus bedarf ausreichender Tests und gegebenenfalls Überarbeitungen.

***Grundlegendes und erhöhtes Anforderungsniveau***

Die Schülerinnen und Schüler
▷ verwenden algorithmische Grundbausteine (Sequenz, Alternative, Wiederholung) und implementieren diese mithilfe einer Programmiersprache,
▷ analysieren gegebene Programme hinsichtlich der Grundkonzepte, einschließlich Variable, Referenz, Schachtelung und funktionaler Zerlegung,
▷ entwerfen Algorithmen und stellen sie in geeigneter Form dar,
▷ wenden Modularisierung zur Strukturierung von Algorithmen und bei deren Implementierung an,
▷ verwenden Softwarebibliotheken oder bereitgestellte Module bei der Implementierung von Algorithmen,
▷ testen und überarbeiten Programme systematisch.

***Erhöhtes Anforderungsniveau***

Die Schülerinnen und Schüler
▷ modellieren und implementieren iterative und rekursive Algorithmen und Datenstrukturen,
▷ vergleichen und beurteilen Algorithmen zur Lösung eines Problems, unter anderem hinsichtlich der Effizienz,
▷ analysieren an Beispielen die Komplexität von Algorithmen und beurteilen die praktischen und theoretischen Grenzen der Algorithmisierung.

## *Sprachen und Automaten*

**Sprachen und Automaten**

Formale Sprachen sind Grundlage der Kommunikation mit Automaten und kommen in vielfältigen Anwendungsszenarien in Informatiksystemen zum Einsatz. Im Unterschied zu natürlichen Sprachen haben formale Sprachen eine eindeutig definierte Syntax, die durch Grammatiken, Syntaxdiagramme oder Sprachbeschreibungen dargestellt werden kann. Nach der Form der Produktionen einer Grammatik lassen sich verschiedene Sprachtypen unterscheiden.

Automaten sind zustandsbasierte Systeme, die eine Eingabe zeichenweise lesen und verarbeiten. Automatentypen lassen sich nach der Konzeption ihres Speichers und damit nach ihren prinzipiellen Möglichkeiten und Grenzen unterscheiden. Den Automatentypen sind entsprechende Sprachtypen zugeordnet.

***Grundlegendes und erhöhtes Anforderungsniveau***

Die Schülerinnen und Schüler
- ▷ vergleichen formale mit natürlichen Sprachen,
- ▷ untersuchen den Zusammenhang zwischen einer Grammatik und ihrer Sprache, leiten Wörter einer Sprache ab und stellen Ableitungsbäume dar,
- ▷ verwenden Sprachdefinitionen (z.B. Grammatiken, Syntaxdiagramme) zur Analyse, Beschreibung und Entwicklung formaler Sprachen,
- ▷ überführen Grammatiken in endliche Automaten und umgekehrt.

***Erhöhtes Anforderungsniveau***

Die Schülerinnen und Schüler
- ▷ erläutern den Zusammenhang zwischen Grammatiken, Sprachen und Automaten,
- ▷ analysieren und implementieren Programme zu Problemstellungen auf Kellerautomaten, Turingmaschinen oder Registermaschinen,
- ▷ erläutern prinzipielle und praktische Grenzen der Berechenbarkeit.

## Informatiksysteme



Ein Informatiksystem ist eine spezifische Zusammenstellung von Hardware-, Software- und Netzwerkkomponenten zur Lösung eines Anwendungsproblems. Eingeschlossen sind auch nichttechnische Aspekte, die durch die Einbettung in ein soziokulturelles System relevant werden, z.B. Einbeziehung der potenziellen Nutzer in den Entwicklungsprozess, die ökonomischen und ökologischen Folgen. Zur kompetenten Nutzung, Gestaltung und Bewertung von Informatiksystemen ist ein grundlegendes Verständnis ihres Aufbaus und ihrer Funktionsweise notwendig. Sie bestehen aus mehreren logisch getrennten Schichten, in denen verschiedene Komponenten interagieren. Zur Entwicklung von Informatiksystemen werden maschinell verarbeitbare Sachverhalte der realen Welt identifiziert und modelliert. Typische Einsatzbereiche von Informatiksystemen sind Datenmanagement, Kommunikation, Grafik, Simulation, Robotik, Prozesssteuerung und -regelung oder Sprachverarbeitung.

***Grundlegendes und erhöhtes Anforderungsniveau***

Die Schülerinnen und Schüler
- ▷ erläutern prinzipielle Funktionsweisen und das Zusammenwirken wesentlicher Hardware-, Software- und Netzwerkkomponenten,
- ▷ beschreiben und erklären wesentliche Schichten und Komponenten der Architektur gegebener Informatiksysteme und die damit verbundenen Prozesse,
- ▷ entwickeln ein Netzwerk mithilfe geeigneter Strukturierungs- und Darstellungsmethoden,
- ▷ verwenden den objektorientierten Ansatz, indem sie Klassen mit ihren Attributen, Methoden und Beziehungen modellieren und implementieren,
- ▷ analysieren die Kommunikation und die Datenhaltung in vernetzten Systemen und beurteilen diese auch unter den Gesichtspunkten des Datenschutzes und der Datensicherheit.

***Erhöhtes Anforderungsniveau***

Die Schülerinnen und Schüler
- ▷ wenden Konzepte und Methoden der Softwareentwicklung zur Gestaltung und Entwicklung von Informatiksystemen an, auch unter Berücksichtigung von Aspekten der Softwareergonomie,

▷ gestalten Informatiksysteme auf Basis von Qualitätskriterien (z. B. Robustheit, Wiederverwendbarkeit, Korrektheit, Effizienz, Komplexität).

## Informatik, Mensch und Gesellschaft

**Informatik, Mensch und Gesellschaft**

Informatiksysteme prägen unsere Informationsgesellschaft und stehen in Wechselwirkungen mit den Menschen und der Gesellschaft. Ausgehend von gesellschaftlich relevanten Fragestellungen oder eigenen Erfahrungen im Umgang mit Informatiksystemen werden Wechselwirkungen zwischen Informatiksystemen und ihrer gesellschaftlichen Einbettung analysiert. In Auseinandersetzung mit normativen, rechtlichen, ethischen und sozialen Aspekten entwickeln sich ein Orientierungsrahmen sowie Verantwortungsbewusstsein im Umgang mit moderner Informationstechnik. Entscheidungsfreiheiten im Umgang mit Informatiksystemen, Handeln in Übereinstimmung mit gesellschaftlichen Normen und angemessenes Reagieren auf Risiken bei der Nutzung von Informatiksystemen werden reflektiert.

***Grundlegendes und erhöhtes Anforderungsniveau***

Die Schülerinnen und Schüler
▷ analysieren und beschreiben Wechselwirkungen zwischen Informatiksystemen, Individuen und Gesellschaft,
▷ beschreiben Chancen, Risiken und Missbrauchsmöglichkeiten von Informatiksystemen,
▷ diskutieren und bewerten wesentliche Aspekte des Datenschutz- und Urheberrechts anhand von Anwendungsfällen,
▷ beurteilen und bewerten die gesellschaftlichen Folgen der Einführung und Nutzung von Informatiksystemen,
▷ verwenden und beschreiben Verfahren zur Sicherung von Vertraulichkeit, Authentizität und Integrität von Daten,
▷ ziehen Rückschlüsse für das eigene Verhalten beim Einsatz von Informatiksystemen.

***Erhöhtes Anforderungsniveau***

Die Schülerinnen und Schüler
▷ analysieren und beurteilen Verfahren zur Sicherung von Vertraulichkeit, Authentizität oder Integrität von Daten in konkreten aktuellen Anwendungskontexten,
▷ konzipieren Maßnahmen zur Realisierung von Datensicherheit für konkrete Anwendungsfälle, insbesondere Zugriffskontrolle.

# Aufgaben

4

## Aufgaben zum Lernen und Leisten

**Lern- und Leistungssituationen**

Lernen kann in engerem Sinne als ein Prozess verstanden werden, in dem es um den Erwerb neuen Wissens, Könnens und neuer Überzeugungen geht. Neben dem *Kompetenzerwerb*, der durch das Erkunden, Entdecken und Erfinden geprägt ist, muss der expliziten weiteren *Ausprägung der Kompetenzen* (Sammeln, Sichern, Systematisieren) sowie dem *Kompetenzerhalt* (Üben und Wiederholen) ausreichende Aufmerksamkeit zuteilwerden. Fehler dienen als Anlass zu einer vertieften Auseinandersetzung mit dem Thema – mit dem Ziel, die Zahl der Fehler mit fortschreitendem Lernprozess zu reduzieren.

Leistungssituationen sind Bestandteil des Lernprozesses. In ihnen sollen bereits erworbene Kompetenzen unter Vermeidung von Fehlern unter Beweis gestellt werden. Der Anlass hierfür kann sich sowohl unmittelbar aus dem Unterricht als auch durch schulinterne oder zentrale Prüfungsformate ergeben. Die Ziele einer Leistungserbringung können vielfältig sein. Neben dem Prüfen, der klassischen Leistungsmessung durch die Lehrkraft zum Zweck der Bewertung, können Leistungssituationen auch durch die Lernenden selbst gestaltet werden. Hierzu gehören das Anwenden mit dem Ziel des motivierenden Kompetenzerlebens und die Selbstüberprüfung. Des Weiteren kann die Diagnose des Leistungsvermögens wertvolle Hinweise für die weitere Gestaltung des Lehr- und Lernprozesses geben.

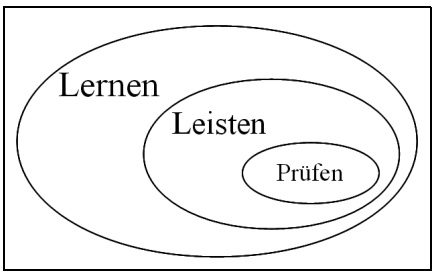



**Abbildung 4.01: Verhältnis von Lernen, Leisten und Prüfen.**

**Funktionen von Aufgaben im Unterricht**

Eine Aufgabe kann je nach Einbettung in den Unterricht unterschiedlichen Zwecken gerecht werden. Prüfungsaufgaben sollen sich auf wesentliche Prozess- und Inhaltsbereiche konzentrieren. Ziel ist es, auf die Vielfalt und Ausgewogenheit abgeforderter Leistungsbereiche zu achten. Der Grad der Offenheit von Prüfungsaufgaben ist begrenzt, Teilaufgaben sollten weitgehend voneinander unabhängig sein. Bewertungskriterien müssen transparent und die Bewertung muss nachvollziehbar sein.

Für Leistungssituationen geeignete Aufgaben können zumeist auch zum Üben und Wiederholen eingesetzt werden. Die Vorbereitung auf Testsituationen darf den Unterricht aber nicht dominieren. Um die Aneignung von Kompetenzen zu fördern, sind deshalb weitere Aufgabenformate jenseits der Eignung für Leistungssituationen vonnöten.

Die Konstruktion und Festigung von Wissen und Können in Lernsituationen ist ein stetiger Vorgang. Anders als bei den Leistungsaufgaben steht hier der *Prozess* des konstruierenden Wissenserwerbs durch aktive Lerner im Fokus. Die Subjektivität des Wissens und seiner Aneignung impliziert ein größeres Maß an Offenheit für entsprechende Lernaufgaben. Eine formale Bewertbarkeit der Aufgaben ist nicht erforderlich. Die Aneignung des Wissens erfolgt individuell auf Basis der Lernvoraussetzungen und des bereits vorhandenen semantischen Netzes. Aufgaben, die in reinen Lernsituationen eingesetzt werden, regen die selbstständige Arbeit der Schülerinnen und Schüler und die Verknüpfung mit bereits Bekanntem durch die bewusste Anwendung des Wissens an.

Die Output-Orientierung von Standards erfordert primär Aufgaben zur Leistungsüberprüfung. Um die Entwicklung der Kompetenzen anschaulich zu beschreiben, wird darüber hinaus eine Zusammenstellung von Lernaufgaben gegeben.

# Operatoren

Lern- und Prüfungsaufgaben müssen so formuliert sein, dass sie von Schülerinnen und Schülern verstanden werden, d. h. dass sie eindeutig hinsichtlich des Arbeitsauftrags und der erwarteten Leistung formuliert sind. Nur bei Klarheit über das erwartete Ergebnis einer Lernaufgabe bzw. die erwartete Leistung in einer Prüfungsaufgabe kann eine Beurteilung bzw. Bewertung objektiv und gerecht erfolgen.

Operatoren sind ausgewählte Verben wie z. B. *erläutern*, *darstellen* und *begründen*, die im Rahmen einer Aufgabe zu einer bestimmten Tätigkeit auffordern, deren Bedeutung möglichst genau spezifiziert ist, und die erwartete Leistung klar beschreiben. Durch die Verwendung einer standardisierten Liste von Operatoren in Lern- und Prüfungsaufgaben wird der Gefahr von Fehldeutungen von Aufgabentexten erfolgreich entgegengewirkt. Die Anzahl der Operatoren ist beschränkt, damit die Liste praktikabel und eine ausreichende Trennschärfe zwischen den Operatoren gegeben ist. Die Verwendung einer Operatorenliste führt insbesondere in Prüfungssituationen zu einer besseren Verständlichkeit der Arbeitsaufträge.

Eine Zuordnung von Operatoren zu Anforderungsbereichen erfolgt nicht, weil eine eindeutige Zuordnung nicht möglich ist. Die Zuordnung kann erst erfolgen, wenn ein Operator zur Formulierung einer konkreten Aufgabe verwendet wird.

**Operatorenliste**

| Operator | Bedeutung |
|---|---|
| Analysieren | eine Materialgrundlage untersuchen, Elemente identifizieren, deren Merkmale und Beziehungen erfassen und das Ergebnis darstellen |
| Angeben | ohne nähere Erläuterungen und Begründungen nennen |
| Begründen | durch rational nachvollziehbare Argumente stützen |
| Berechnen | Ergebnisse durch Rechenoperationen gewinnen |
| Beschreiben | unter Verwendung der Fachsprache in eigenen Worten verständlich wiedergeben |
| Bestimmen | anhand von Kriterien einen Sachverhalt feststellen und das Ergebnis formulieren |
| Beurteilen | unter Verwendung von Kriterien ein Sachurteil formulieren |
| Bewerten | unter Verwendung von Kriterien ein Werturteil formulieren |
| Darstellen | in formalisierter Form grafisch oder fachsprachlich wiedergeben |
| Entwerfen | in seinen wesentlichen Zügen erstellen |
| Entwickeln | umfassend und nachvollziehbar herstellen oder konstruieren |
| Erklären | einen Sachverhalt erfassen, in einen kausalen Zusammenhang einordnen und deuten |
| Erläutern | mithilfe zusätzlicher Angaben (Beispiele, Vergleiche, …) verständlich machen |
| Implementieren | ein informatisches Modell auf einem oder für ein Informatiksystem umsetzen |
| Modellieren | gemäß einer Problemanalyse ein informatisches Modell anfertigen |
| Stellung nehmen | die eigene Meinung zu einem Problem argumentativ entwickeln und darlegen |
| Übertragen | in eine andere Darstellungsform bringen |
| Vergleichen | nach vorgegebenen oder selbst gewählten Gesichtspunkten Gemeinsamkeiten und Unterschiede angeben |
| Zeichnen | eine hinreichend exakte grafische Darstellung anfertigen |

# Schriftliche Leistungsaufgaben

Die folgenden schriftlichen Leistungsaufgaben sind Aufgaben aus zentralen Abiturprüfungen einiger Bundesländer, die teilweise leicht bearbeitet wurden.

## DVD-Verleih

*DVD-Verleih*

***Anmerkungen***

▷ grundlegendes Anforderungsniveau
▷ vorgesehene Bearbeitungszeit: 120 min

***Aufgabe***

Beim DVD-Verleih »Wunschkino« sollen registrierte Kunden aus den DVDs dieses Verleihs online einen Wunschzettel mit bis zu 10 DVDs zusammenstellen und diese Auswahl speichern können. Fordert ein Kunde dann eine DVD an, wird die erste verfügbare DVD seines Wunschzettels ausgewählt und von »Wunschkino« an ihn verschickt. Erst wenn er diese DVD zurückgesendet hat, kann er wieder eine DVD anfordern.

Die Softwarefirma IT12 ist mit der Erstellung des entsprechenden Systems beauftragt und hat dafür eine Klasse DVD und eine Klasse KUNDE gemäß des folgenden Klassendiagramms entworfen (siehe folgende Abbildung 4.02).

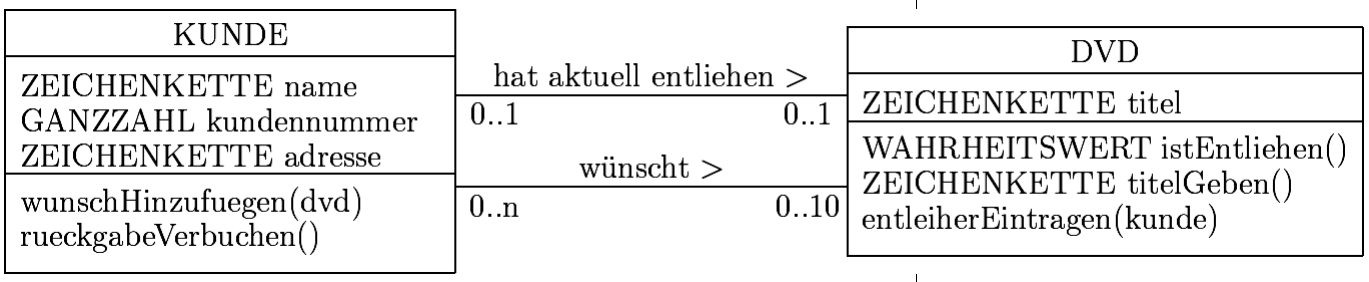



**Abbildung 4.02.**

Im Attribut titel ist der Titel der DVD gespeichert. Vereinfachend wird davon ausgegangen, dass es zu jedem Titel genau eine DVD gibt. Die Methode istEntliehen() gibt genau dann den Wahrheitswert wahr zurück, wenn die DVD aktuell entliehen ist. Die Methode entleiherEintragen(kunde) trägt das übergebene Kundenobjekt als aktuellen Entleiher ein und veranlasst den Versand der DVD.

Mithilfe der Klasse KUNDE werden von jedem Kunden eine ganzzahlige Kundennummer, sein Name, seine Adresse sowie gegebenenfalls die von ihm entliehene DVD gespeichert. Mit der Methode wunschHinzufuegen(dvd) kann der Kunde ein DVD-Objekt seinem Wunschzettel hinten hinzufügen. Ist der Wunschzettel jedoch bereits voll, wird eine entsprechende Meldung ausgegeben. Die Methode rueckgabeVerbuchen() aktualisiert den Datenbestand, wenn der Kunde seine entliehene DVD an »Wunschkino« zurückgesendet hat.

1. In der Klasse KUNDE wird zur Implementierung des Wunschzettels ein Feld verwendet. Sie hat zusätzlich die folgenden Methoden:
   - Mit der Methode wunschLoeschen(i) wird der Wunsch entfernt, der im Wunschzettel an der i-ten Stelle steht. Die dahinter stehenden Wünsche werden zudem nach vorne gerückt.
   - Bei Aufruf der Methode wunschdvdAnfordern() wird der Wunschzettel von vorne startend durchsucht, bis eine entleihbare DVD gefunden ist. Dann werden der Kunde als Entleiher der DVD und die DVD als von diesem Kunden entliehen eingetragen sowie dieser Wunsch aus dem Wunschzettel entfernt. Ist kein Entleihvorgang möglich, da der Kunde bereits eine DVD entliehen hat oder auf seinem Wunschzettel keine entleihbare DVD gefunden wurde, wird eine entsprechende Meldung ausgegeben.

   a) Geben Sie in einer objektorientierten Programmiersprache eine mögliche Implementierung der Methode wunschdvdAnfordern() an. Alle anderen

Methoden der Klasse KUNDE dürfen als implementiert vorausgesetzt werden. Geben Sie auch die Deklarationen der Attribute der Klasse KUNDE an.

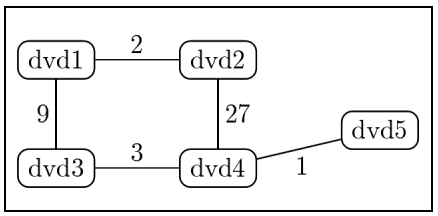
b) Stellen Sie in einem Sequenzdiagramm die einzelnen Schritte des Methodenablaufs von wunschdvdAnfordern() dar. Gehen Sie beispielhaft davon aus, dass die zweite DVD auf dem Wunschzettel entliehen wird. Geben Sie eine Problemsituation an, die sich ergeben kann, wenn bei einem weiteren Kunden ebenfalls wunschdvdAnfordern() aufgerufen wird, und nennen Sie eine Möglichkeit, dieses Problem zu vermeiden.

2. Von jedem Kunden soll die Historie der von ihm entliehenen DVDs in einer einfach verketteten Liste gespeichert werden. Entleiht ein Kunde eine DVD mehrmals, soll sie auch mehrmals in dieser Liste enthalten sein. Beachten Sie bei folgender Aufgabe, dass die Klasse DVD wie eingangs beschrieben als vollständig implementiert vorausgesetzt werden darf.
   a) Zeichnen Sie ein Klassendiagramm, das als Grundlage zur Implementierung der Historie dienen kann. Verwenden Sie dabei das Entwurfsmuster Kompositum und berücksichtigen Sie auch das Prinzip der Trennung von Struktur und Inhalt. Auf die Angabe von Attributen und Methoden kann verzichtet werden. Begründen Sie, warum eine DVD mehrfach in der Historie stehen kann.
   b) Implementieren Sie in einer objektorientierten Programmiersprache alle betroffenen Klassen der Listenstruktur so, dass in die Liste eine DVD vorne eingefügt werden kann und dass die Titel aller gespeicherten DVDs in der Reihenfolge des Einfügens ausgegeben werden können. Wenden Sie soweit wie möglich das Prinzip der Rekursion an. Auf Attribute und Methoden, die zur Lösung nicht benötigt werden, kann verzichtet werden.

3. »Wunschkino« will seinen Bestand an DVDs in einem alphabetisch geordneten Binärbaum speichern. Als Schlüssel dient dabei der Titel der DVD. Es darf vorausgesetzt werden, dass die Titel eindeutig sind.
   a) Zeichnen Sie den Baum, der sich ergibt, wenn in den zunächst leeren Baum nacheinander die DVDs »Batman«, »Monaco Franze«, »High Noon«, »Fluch der Karibik«, »Cars 3«, »König der Löwen« und »Titanic« eingefügt werden. Geben Sie außerdem eine Einfügereihenfolge an, die zu einem Baum mit möglichst wenigen Ebenen führt.
   b) Nennen und beschreiben Sie kurz einen Algorithmus, der alle im Baum gespeicherten DVD-Titel in alphabetischer Reihenfolge ausgibt.
   c) »Wunschkino« hat in einem solchen Baum 20 000 DVDs gespeichert. Beschreiben Sie, wie der Baum aufgebaut sein sollte, damit die Suche nach einer beliebigen DVD möglichst effizient ist. Schätzen Sie ab, wie lange eine Suche in diesem Fall maximal dauert, wenn pro untersuchtem Baumelement 1 ns nötig ist.

4. Für Marketingzwecke fasst »Wunschkino« die Vorlieben aller seiner Kunden in einem gemeinsamen Graphen mit sämtlichen angebotenen DVDs als Knoten zusammen. Gibt ein Kunde eine DVD zurück, wird dazu die Historie der bislang von diesem Kunden entliehenen DVDs ausgewertet, indem das Gewicht der Kante zwischen der zurückgegebenen DVD und jeder anderen DVD aus der Historie dieses Kunden um 1 erhöht wird. Falls diese Kante noch nicht existiert, wird eine Kante mit Kantengewicht 1 erstellt. Ein möglicher Graph ist nebenstehend abgebildet (siehe Abbildung 4.03).
   a) Stellen Sie diesen Graphen als Adjazenzmatrix dar.
   b) Zeichnen Sie den Graphen, der aus dem oben stehenden Graphen entsteht, wenn ein Kunde die DVD *dvd2* zurückgibt und seine Historie ansonsten aus den DVDs *dvd3* und *dvd4* besteht.
   c) Nennen Sie einen Algorithmus zum Durchlaufen eines Graphen. Beschreiben Sie eine Ausleihsituation, in der bei Abarbeitung dieses Algorithmus nicht alle Knoten des Graphen besucht werden.
   d) Eine Klasse GRAPH enthält zur Speicherung der Kanten und Knoten des oben beschriebenen Graphen eine $n \times n$-Matrix $a$ und ein Feld $d$ der Länge $n$. Der Wert von $n$ ist dabei die (als konstant angenommene) Anzahl aller DVDs von »Wunschkino«. Betrachten Sie die folgenden Methoden *m1*,

**Abbildung 4.03.**



*m2* und *m3* und beschreiben Sie, was der Aufruf der Methode *m3* bewirkt. Geben Sie auch den Zweck der Methode an.

```
Methode m1(GANZZAHL k)
  GANZZAHL m=0
  wiederhole für i=0,1,2,…,n-1
    wenn (a[i][k]>m)
      m=a[i][k]
    endeWenn
  endeWiederhole
  gib m zurück
endeMethode

Methode m2(GANZAHL k, GANZZAHL w)
  wiederhole für i=0,1,2,…, n-1
    wenn (a[i][k] ist gleich w)
      gib Titel von d[i] aus
    endeWenn
  endeWiederhole
endeMethode

Methode m3(GANZZAHL k)
  GANZZAHL u=m1(k)
  wenn (u ungleich 0)
    m2(k,u)
  endeWenn
endeMethode
```

## Lösungshinweise

***Lösungshinweise zu den Aufgaben***

1a – mögliche Lösung

Attributdeklarationen:

```
private String name;
private int kundennr;
private String adresse;
private DVD[] wuensche;
private int anzahlWuensche;
private DVD entlieheneDVD;
```

Geforderte Methode:

```
public void wunschdvdAnfordern(){
  int i=0;
  if (entlieheneDVD != null){
    System.out.println("bereits eine DVD entliehen");
  } else {
    while ((i<anzahlWuensche) && (wuensche[i].istEntliehen())){
      i++;
    }
    if (i==anzahlWuensche){
      System.out.println("keine entleihbare DVD gefunden");
    } else {
       wuensche[i].entleiherEintragen(this);
       entlieheneDVD = wuensche[i];
       wunschLoeschen(i);
    }
  }
}
```

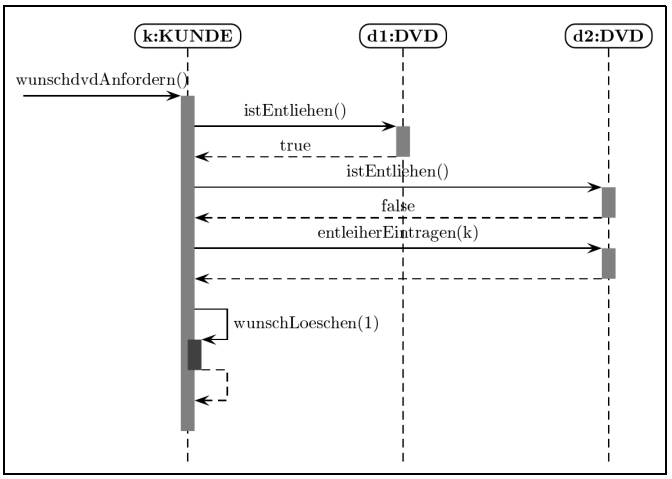
1b – mögliche Lösung

Siehe Abbildung 4.04.

*Hinweis:* Je nach Implementierung der Zählung in Teilaufgabe 1a muss statt wunschLoeschen(1) ggf. wunschLoeschen(2) aufgerufen werden.

Ein Problem ergibt sich zum Beispiel, wenn ein weiterer Kunde nach dem obigen zweiten Aufruf von istEntliehen bei dieser DVD entleiherEintragen aufruft. Vermeiden ließe sich dieses Problem, wenn z.B. durch ein Semaphor sichergestellt würde, dass der Zugriff eines Kundenobjekts auf eine DVD nicht unterbrochen werden darf.

**Abbildung 4.04.**



2a – mögliche Lösung

Siehe folgende Abbildung 4.05.

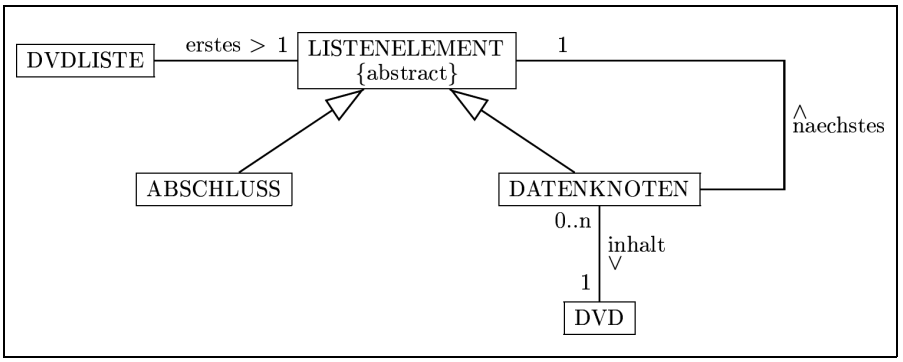



**Abbildung 4.05.**

2b – mögliche Lösung

```
class DVDLISTE{
  private LISTENELEMENT erstes;

  DVDLISTE() {
    erstes = new ABSCHLUSS();
  }
  void dvdHinzufuegen(DVD d){
    erstes = new KNOTEN(d, erstes);
  }
  void alleTitelAnzeigen(){
    erstes.alleTitelAnzeigen();
  }
}
abstract class LISTENELEMENT{
  abstract void alleTitelAnzeigen();
}
class KNOTEN extends LISTENELEMENT{
  private LISTENELEMENT naechstes;
  private DVD inhalt;

  KNOTEN(DVD eineDVD, LISTENELEMENT n){
    inhalt = eineDVD;
    naechstes = n;
  }
  void alleTitelAnzeigen(){
    naechstes.alleTitelAnzeigen();
    System.out.println(inhalt.titelGeben());
  }
}

class ABSCHLUSS extends LISTENELEMENT{
  void alleTitelAnzeigen(){
  }
}
```

3a – mögliche Lösung

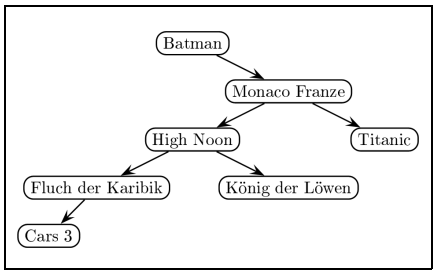



**Abbildung 4.06.**

Siehe Abbildung 4.06.

Ausbalanciert wäre der Baum z. B. bei der Reihenfolge »High Noon«, »Cars 3«, »Monaco Franze«, »Fluch der Karibik«, »König der Löwen«, »Batman«, »Titanic«.

3b – mögliche Lösung

Inorder-Traversierung: Alle im Baum gespeicherten DVD-Titel in alphabetischer Reihenfolge können durch eine rekursive Methode, die in jedem Knoten jeweils erst den linken Teilbaum, dann den Inhalt und schließlich den rechten Teilbaum bearbeitet, ausgegeben werden.

3c – mögliche Lösung

Der Baum sollte eine möglichst geringe Höhe haben, hier sind das 15 Ebenen. Begründung:

In einem Baum mit 1 Ebene kann $1 = 2^1-1$ Element gespeichert werden,
in einem Baum mit 2 Ebenen
können $1+2 = 3 = 2^2-1$ Elemente gespeichert werden,
in einem Baum mit 3 Ebenen
können $1+2+4 = 7 = 2^3-1$ Elemente gespeichert werden,
...,
in einem Baum mit k Ebenen
können $2^k-1$ Elemente gespeichert werden.

Nun gilt: $2^{14}-1 = 16\,383$, $2^{15}-1 = 32\,767$, also sind für 20 000 Elemente 15 Ebenen nötig.
Eine Suche dauert also maximal 1 ns×15 = 15 ns.

4a – mögliche Lösung

| | dvd1 | dvd2 | dvd3 | dvd4 | dvd5 |
|---|---|---|---|---|---|
| dvd1 | | 2 | 9 | | |
| dvd2 | 2 | | | 27 | |
| dvd3 | 9 | | | 3 | |
| dvd4 | | 27 | 3 | | 1 |
| dvd5 | | | | 1 | |

4b – mögliche Lösung

Siehe folgende Abbildung 4.07.

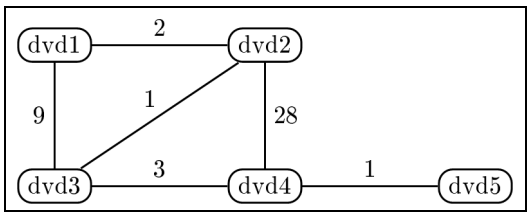



**Abbildung 4.07.**

4c – mögliche Lösung

Ein Algorithmus zum Graphendurchlauf ist die Tiefensuche.

Nicht alle Knoten werden besucht, wenn z. B. eine DVD nie entliehen worden ist. Es könnten sich auch zwei verschiedene Teilgraphen gebildet haben, z. B. wenn eine Gruppe von Kunden immer nur Kinder-, eine andere nur Action-DVDs entliehen hat.

4d – mögliche Lösung

Es werden alle DVDs angezeigt, die am häufigsten mit der angegebenen DVD in einer Historie standen.

Denn *m3* ermittelt zunächst – durch Aufruf von *m1* – das Maximum *u* aller Kantengewichte der Kanten, die vom Knoten mit dem Index *k* ausgehen. Ist dieses nicht 0, dann werden – durch Aufruf von *m2* – die Titel aller DVDs angezeigt, die mit der DVD mit dem Index *k* durch eine Kante mit Kantengewicht *u* verbunden sind.

Quelle:
ISB – Staatsinstitut für Schulqualität und Bildungsforschung München (Hrsg.): Abiturprüfung 2013 – Informatik. München: ISB, 2013, S. 2–6.
https://www.isb.bayern.de/download/15005/abituraufgaben_2013_informatik.pdf

***Zuordnung zu den Prozess-, Inhalts- und Anforderungsbereichen***

| Aufgabe | Prozessbereiche | | | | | Inhaltsbereiche | | | | | Bewertungseinheiten in Anforderungsbereichen | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MI | BB | SV | KK | DI | ID | AL | SA | IS | IMG | I | II | III |
| 1a | X | | X | | X | X | X | | | | | 12 | |
| 1b | | X | | X | X | | X | | X | | | 10 | |
| 2a | X | | X | | X | X | | | X | | 4 | 2 | |
| 2b | X | | X | | X | X | X | | | | 7 | 7 | |
| 3a | | X | X | | X | X | | | | | 3 | 2 | |
| 3b | | | | X | | | X | | | | 4 | | |
| 3c | | X | | X | | X | | | | | | 2 | 5 |
| 4a | | | X | | X | X | | | | | 4 | | |
| 4b | | | X | | X | X | | | | | | 3 | |
| 4c | | X | | X | | X | X | | | | | 2 | 2 |
| 4d | | X | | X | X | | X | | | | | | 11 |
| | | | | | | | | | | Summe 80 | 22 | 40 | 18 |

## *Rangierbahnhof*

**Rangierbahnhof**

***Anmerkungen***

▷ grundlegendes Anforderungsniveau
▷ vorgesehene Bearbeitungszeit: 90 min

***Aufgabe***

Auf einem Rangierbahnhof werden Güterzüge in Waggons zerlegt und diese zu neuen Zügen zusammengestellt. Zum Rangieren verwendet man einen Ablaufberg. Die Rangierlokomotive kann einen auf dem Rangiergleis stehenden Zug auf den Ablaufberg ziehen. Auf dem Ablaufberg werden die Waggons der Reihe nach entkuppelt. Sie rollen dann eigenständig das Gefälle hinab und gelangen gemäß der jeweiligen Weichenstellung auf die vor Beginn des Rangiervorgangs leeren Rangiergleise.

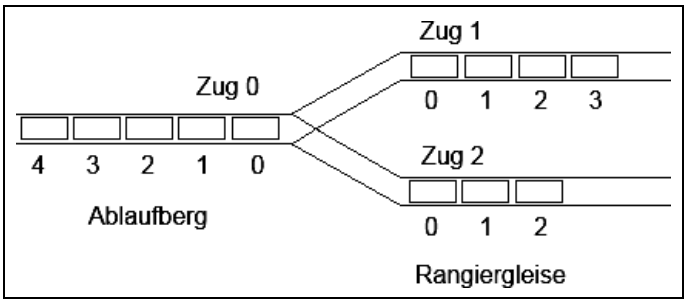



**Abbildung 4.08.**

1. Für die Modellierung des Rangiervorgangs besteht ein Güterzug aus keinem, einem oder mehreren Waggons und hat keine Lokomotive. Wir beschränken uns auf Züge mit maximal zwölf Waggons. Von jedem Waggon sind seine Start- und Zielbahnhofsnummer zu speichern.

1.1 Entwerfen Sie für einen Güterzug ein UML-Klassendiagramm mit den beiden Klassen ZUG und WAGGON. Sehen Sie in der Klasse WAGGON Attribute und Methoden so vor, dass die im Text genannten Informationen modelliert und später abgefragt werden können. Bei der Modellierung der Klasse ZUG sind auch *Material 1* und aus *Material 2* die in der Methode Wag

gonsAbstossen() verwendeten Methoden sowie die sich daraus ergebenden Attribute zu berücksichtigen.
Begründen Sie die von Ihnen ausgewählte Klassenbeziehung.

1.2 Beschreiben Sie die beiden Konstruktoren der Klasse ZUG im *Material 1*.

2. Das An- und Abkuppeln von Waggons kann beim Rangieren nur am Zugende stattfinden. Es wird davon ausgegangen, dass der letzte Waggon an der Position 0 im Feld gespeichert ist.

2.1 Implementieren Sie die Methode ankuppeln(Waggon einWaggon) der Klasse ZUG, um einen Waggon an einen Zug anzukuppeln.

2.2 Die Methode sucheMaxZiel() bestimmt die größte Zielbahnhofsnummer der Waggons eines Zuges. Implementieren Sie sucheMaxZiel().

3. Die Klasse RANGIERBAHNHOF im *Material 2* besitzt als Attribute die drei Züge, die in obiger Grafik dargestellt sind. *Zug0* ist dabei der zu rangierende Güterzug, dessen Waggons mithilfe der zwei anfangs leeren Züge *Zug1* und *Zug2* umrangiert werden sollen.

3.1 Analysieren Sie die beiden im *Material 2* dargestellten Methoden WaggonsAbstossen() und ZugAufDenAblaufberg(Zug einZug).

3.2 Analysieren und erklären Sie unter Einbezug vom *Material 2* den Algorithmus *Rangieren*, der im *Material 3* als Struktogramm dargestellt ist.

4. Erläutern Sie unter Berücksichtigung aller Aufgabenteile, warum die Verwendung zweier Konstruktoren innerhalb der Klasse Zug sinnvoll ist.

*Material 1*

```
public class Zug {
  private int Anzahl;
  private int MaxZahl = 12;
  private Waggon[] Waggons = new Waggon[MaxZahl];

  public Zug() {
    for (int i = 0; i < MaxZahl; i++)
      Waggons[i] = null;
    Anzahl = 0;
  }

  public Zug(int Start)  { // ein Testzug
    Waggons[ 0] = new Waggon(Start, 4);
    Waggons[ 1] = new Waggon(Start, 6);
    Waggons[ 2] = new Waggon(Start, 7);
    Waggons[ 3] = new Waggon(Start, 5);
    Waggons[ 4] = new Waggon(Start, 1);
    Waggons[ 5] = new Waggon(Start, 8);
    Waggons[ 6] = new Waggon(Start, 6);
    Waggons[ 7] = new Waggon(Start, 2);
    Waggons[ 8] = new Waggon(Start, 9);
    Waggons[ 9] = new Waggon(Start, 1);
    Waggons[10] = new Waggon(Start, 7);
    Waggons[11] = new Waggon(Start, 6);
    Anzahl = MaxZahl;
  }

  public int sucheMaxZiel() {
    // zu implementieren in Aufgabe 2.2
  }
  ...
}
```

*Material 2*

```
public class Rangierbahnhof {
  private Zug Zug0;
  private Zug Zug1;
  private Zug Zug2;

  public void WaggonsAbstossen() {
    Waggon einWaggon;
    int MaxZiel = Zug0.sucheMaxZiel();
    int AnzahlWaggons = Zug0.getAnzahl();
```

```
    for (int i = 0; i < AnzahlWaggons; i++) {
      einWaggon = Zug0.abkuppeln();
      if (einWaggon.getZiel() < MaxZiel)
        Zug2.ankuppeln(einWaggon);
      else
        Zug1.ankuppeln(einWaggon);
    }
  }

  public void ZugAufDenAblaufberg(Zug einZug) {
    int AnzahlWaggons = einZug.getAnzahl();
    for (int i = 0; i < AnzahlWaggons; i++)
      Zug0.ankuppeln(einZug.abkuppeln());
  }

  ...
}
```

*Material 3*

Siehe folgende Abbildung 4.09.

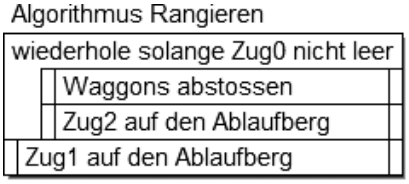



**Abbildung 4.09.**

## Lösungshinweise

***Lösungshinweise zu den Aufgaben***

1.1 – mögliche Lösung

Siehe folgende Abbildung 4.10.

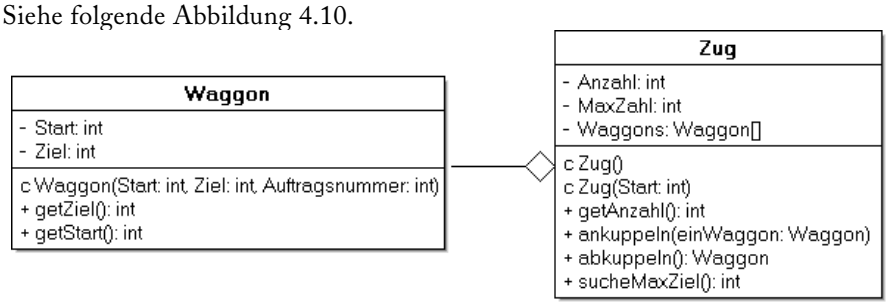



**Abbildung 4.10.**

Da ein Zug aus keinem, einem oder mehreren Waggons besteht, haben die beiden Klassen WAGGON und ZUG eine Aggregationsbeziehung.

1.2 – mögliche Lösung

Der erste Konstruktor hat keinen Parameter. Mittels einer for-Schleife wird jedes Element des Feldes Waggons mit null initialisiert. Die Anzahl der Waggons wird auf 0 gesetzt.

Der zweite Konstruktor hat den Parameter Start vom Datentyp int. Durch Aufrufe des Konstruktors der Klasse WAGGON mit dem Parameter Start sowie den angegebenen Werten für Ziel werden zwölf Waggons erzeugt und Verweise darauf in den Elementen des Feldes gespeichert. Die Anzahl der Waggons wird auf MaxZahl gesetzt.

2.1 – mögliche Lösung

```
public void ankuppeln(Waggon einWaggon) {
  if (Anzahl < MaxZahl) {
    for (int i = Anzahl; i > 0; i--)
      Waggons[i] = Waggons[i-1];
    Waggons[0] = einWaggon;
    Anzahl++;
  }
}
```

2.2 – mögliche Lösung

```
public int sucheMaxZiel() {
  int MaxZiel = 0;
  for (int i = 0; i < Anzahl; i++)
    if (MaxZiel < Waggons[i].getZiel())
      MaxZiel = Waggons[i].getZiel();
  return MaxZiel;
}
```

3.1 – mögliche Lösung

Die Methode WaggonsAbstossen() hat keinen Parameter. Eine lokale Variable einWaggon vom Datentyp Waggon wird deklariert. Die lokale Variable MaxZiel erhält mit Zug0.sucheMaxZiel() die aktuell größte Zielbahnhofsnummer von Zug0 als Wert und die lokale Variable AnzahlWaggons mit Zug0.getAnzahl() die Anzahl der Waggons von Zug0. In der for-Schleife wird bei jedem Schleifendurchlauf ein Waggon von Zug0 abgekuppelt und in einWaggon zwischengespeichert. In einer bedingten Anweisung wird geprüft, ob die Zielbahnhofsnummer des Waggons kleiner als MaxZiel ist. Trifft dies zu, so wird der Waggon an Zug2 angekuppelt, ansonsten an Zug1. Die Methode WaggonsAbstossen() kuppelt die Waggons von Zug0 an die Züge Zug1 und Zug2 an. Waggons mit der aktuell maximalen Zielbahnhofsnummer von Zug0 werden an Zug1 angekuppelt, die anderen an Zug2.

Die Methode ZugAufDenBerg() wird mit dem Parameter einZug vom Datentyp Zug aufgerufen. Die lokale Variable AnzahlWaggons erhält mit einZug.getAnzahl() die Anzahl der Waggons von einZug. In der for-Schleife werden alle Waggons von einZug abgekuppelt und an Zug0 angekuppelt. Wenn die Schleife beendet ist, hat einZug keine Waggons mehr und alle Waggons befinden sich in Zug0 auf dem Ablaufberg.

3.2 – mögliche Lösung

Der Algorithmus beginnt mit einer Schleife, die terminiert, wenn Zug0 leer ist. Solange Zug0 noch Waggons hat, werden die Waggons mit der aktuell größten Zielbahnhofsnummer mittels WaggonsAbstossen() an Zug1 angekuppelt, alle anderen Waggons an Zug2. Letztere werden anschließend wieder an Zug0 angekuppelt. Im nächsten Durchlauf wird die größte Zielbahnhofsnummer der verbliebenen Waggons bestimmt und ebenso verfahren. Der Vorgang ist abgeschlossen, wenn alle Waggons an Zug1 angekuppelt worden sind und somit Zug0 leer ist. Zum Schluss werden alle Waggons wieder von Zug1 an Zug0 angekuppelt. Die Waggons werden durch den Rangier-Algorithmus nach der Zielbahnhofsnummer sortiert.

4 – mögliche Lösung

Mit dem ersten Konstruktor kann man einen leeren Zug erzeugen. Dieser Konstruktor wird für die Erzeugung der anfangs leeren Züge Zug1 und Zug2 benötigt. Der zweite Konstruktor erzeugt einen vollständigen Güterzug mit zwölf Waggons. Mit ihm wird Zug0 erzeugt, damit man einen zu rangierenden Zug simulieren kann.

Quelle:
Hessisches Kultusministerium (Hrsg.): Landesabitur 2013 – Informatik Grundkurs. Wiesbaden: 2013.

***Zuordnung zu den Prozess-, Inhalts- und Anforderungsbereichen***

| Aufgabe | Prozessbereiche | | | | | Inhaltsbereiche | | | | | Bewertungseinheiten in Anforderungsbereichen | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MI | BB | SV | KK | DI | ID | AL | SA | IS | IMG | I | II | III |
| 1.1 | X | X | | | X | | | | X | | 5 | 7 | |
| 1.2 | | | | X | | | X | | | | 3 | 3 | |

| Aufgabe | Prozessbereiche | | | | | Inhaltsbereiche | | | | | Bewertungseinheiten in Anforderungsbereichen | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MI | BB | SV | KK | DI | ID | AL | SA | IS | IMG | I | II | III |
| 2.1 | X | | | | | | X | | | | 2 | 4 | 2 |
| 2.2 | X | | | | | | X | | | | | 4 | 4 |
| 3.1 | | | | X | X | | X | | | | 4 | 7 | 1 |
| 3.2 | | | | X | X | | X | | | | 4 | 4 | 2 |
| 4 | | X | | | | | X | | | | | 1 | 3 |
| Summe 60 | | | | | | | | | | | 18 | 30 | 12 |

## Au-pair-Vermittlung

**Au-pair-Vermittlung**

***Anmerkungen***

▷ grundlegendes Anforderungsniveau
▷ vorgesehene Bearbeitungszeit: 70 min

***Aufgabe***

Als **Au-pair** (frz. *auf Gegenleistung*) bezeichnet man junge Menschen, die für Verpflegung, Unterkunft und Taschengeld in einer Familie im In- oder Ausland tätig sind, um im Gegenzug Sprache und Kultur des Gastlandes bzw. der Gastregion kennenzulernen. Das Au-pair lebt dabei im Haushalt der Gastfamilie, hilft bei der Kinderbetreuung und übernimmt leichte Hausarbeiten. Eine internationale Au-pair-Agentur will die Vermittlung optimieren und eine Datenbank einsetzen. Dazu liegt folgende Beschreibung vor:

Das Au-pair hat eine Nummer, einen Nachnamen, einen Vornamen, ein Geburtsdatum und spricht mehrere Sprachen auf jeweils einem bestimmten Niveau (fließend, gut, gering). Im Falle der Vermittlung lebt das Au-pair bei einer Familie, wobei Beginn und Ende der Aufenthaltsdauer bekannt sind. Familien werden durch eine Nummer, den Namen, die Anzahl der Kinder, die in der Familie vorwiegend gesprochene Sprache und das monatliche Einkommen beschrieben und wohnen in Städten, von denen der eindeutige Name, die Einwohnerzahl und das zugehörige Land bekannt sind.

1. Modellieren Sie eine Datenbank für die Au-pair-Agentur als ER-Diagramm. Geben Sie die Kardinalitäten und die Optionalitäten der Beziehungen im ER-Diagramm an.

2. Entwerfen Sie zu dem ER-Diagramm ein optimiertes relationales Datenbankschema. Markieren Sie Primär- und Fremdschlüssel.

3. Beschreiben Sie den Aufbau und erläutern Sie die inhaltliche Bedeutung der folgenden SQL-Befehle:

3.1
```
SELECT FNr, Name, Kinderzahl
 FROM  Familie, Stadt, lebt_bei
 WHERE Familie.FNr = lebt_bei.FNr AND
       Familie.SName = Stadt.SName AND
       Land = 'Deutschland' AND Kinderzahl > 2 AND
       von >= '2011.02.20' AND bis <= '2011.02.20'
```

3.2
```
SELECT Land, COUNT(FNr) AS Anzahl
 FROM  Stadt, Familie, lebt_bei
 WHERE Stadt.SName = Familie.SName AND
       Familie.FNr = lebt_bei.FNr
 GROUP BY Land
 ORDER BY COUNT(FNr)
```

4. Implementieren Sie eine SQL-Anweisung für folgende Abfrage:
   Die Familie mit der Nummer 214 sucht ein Au-pair, das ihre Sprache

fließend spricht. Gesucht wird eine Liste aller infrage kommenden Au-pairs unabhängig davon, ob sie bereits bei einer Familie leben oder nicht.

5. Die Agentur stellt fest, dass seit einiger Zeit die Zahl der an einem Au-pair interessierten Familien deutlich höher als die zur Verfügung stehenden Au-pairs ist, sodass viele Familien nicht berücksichtigt werden können. Damit mehr Aupairs an Familien vermittelt werden können, entschließt sich die Agentur, Daten der Familien aus ihrer Datenbank im Internet zu veröffentlichen.
   Nehmen Sie Stellung dazu, welche Daten in welcher Form veröffentlicht werden sollen. Berücksichtigen Sie dabei die Interessen aller beteiligten Gruppen.

## Lösungshinweise

***Lösungshinweise zu den Aufgaben***

1 – mögliche Lösung

Siehe folgende Abbildung 4.11.

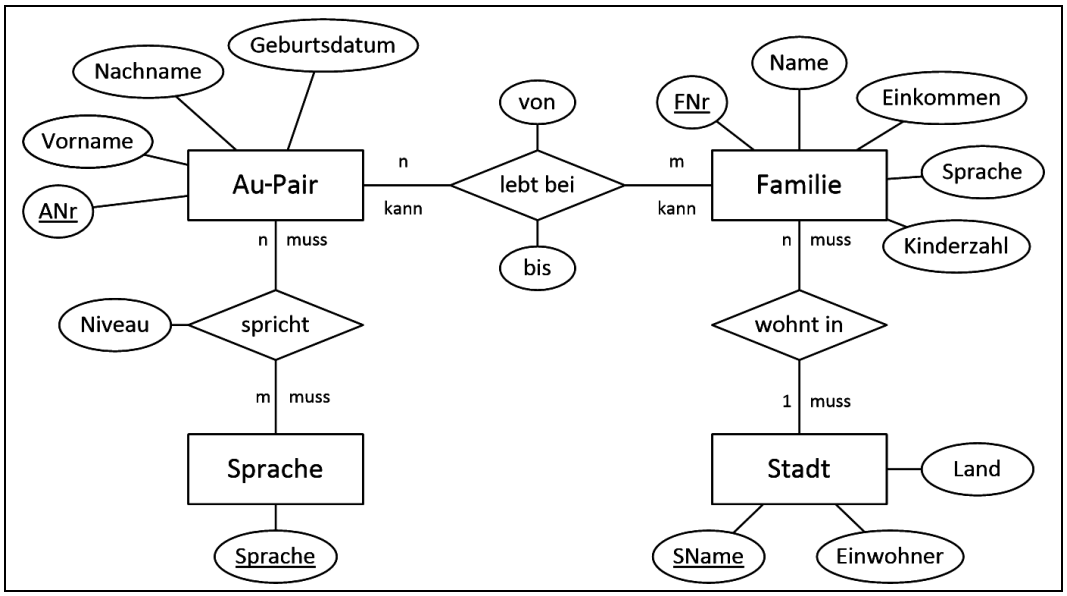



**Abbildung 4.11.**

2 – mögliche Lösung

AuPair(ANr, Nachname, Vorname, Geburtsdatum)
Familie(FNr, Name, Einkommen, Sprache, Kinderzahl, ↑SName)
Stadt(SName, Land, Einwohner)
spricht(↑ANr, ↑Sprache, Niveau)
lebt_bei(↑ANr, ↑FNr, von, bis)

Der Entitätstyp Sprache enthält nur das Attribut Sprache, das direkt in die Beziehungsrelation *spricht* übernommen werden kann.

3.1 – mögliche Lösung

Es wird ein Join der Relationen *Familie*, *Stadt* und *lebt_bei* gebildet, und die gewünschten Datensätze werden durch die Bedingungen bezüglich Land, Kinderzahl und Einsatzdaten selektiert. Abschließend erfolgt eine Projektion auf FNr, Name und Kinderzahl.

Die Select-Anweisung gibt Nummer, Name und Kinderzahl der Familien aus Deutschland mit mehr als zwei Kindern aus, die am 20. Februar 2011 ein Aupair hatten.

3.2 – mögliche Lösung

Es werden ein Join der Relationen *Stadt*, *Familie* und *lebt_bei* gebildet und dann die Datensätze nach dem Land gruppiert. Die Aggregatfunktion COUNT zählt

für jede Gruppe die Anzahl der Familien. Abschließend erfolgt eine Projektion auf Land und Anzahl, die nach der Anzahl aufsteigend sortiert ausgegeben wird.

Ergebnis des SQL-Befehls ist eine aufsteigend geordnete Liste, aus der hervorgeht, aus welchen Ländern wie viele Familien schon einmal ein Au-pair gehabt haben.

4 – mögliche Lösung

```
SELECT ANr, Nachname, Vorname
FROM  Familie, spricht, AuPair
WHERE Familie.Sprache = spricht.Sprache AND
      spricht.ANr = AuPair.ANr AND
      spricht.Niveau = 'fließend' AND
      FNr = 214
```

5 – mögliche Lösung

Ein Au-pair interessiert sich für den Wohnort, die Einwohnerzahl, das Land, die Anzahl der Kinder, die in der Familie gesprochene Sprache und schließlich auch für das monatliche Einkommen von Familien. Eine Internetpräsenz könnte nach der Registrierung bei der Agentur und nach der Eingabe entsprechender Suchkriterien die passenden Angebote auflisten. Bei der Übermittlung der Daten müssen die schutzwürdigen Interessen der Familien und die geschäftlichen Interessen der Agentur gewahrt werden. Zum Beispiel wird die Kinderzahl der Familien erst nach Registrierung und Autorisierung als Au-pair angezeigt, und die Adressen der Familien können nur direkt bei der Agentur erfragt werden. Das Einkommen darf nur mit Einwilligung der Familie veröffentlicht werden.

Quelle:
Hessisches Kultusministerium (Hrsg.): Landesabitur 2011 – Informatik Grundkurs. Wiesbaden: 2011.

***Zuordnung zu den Prozess-, Inhalts- und Anforderungsbereichen***

| Aufgabe | Prozessbereiche | | | | | Inhaltsbereiche | | | | | Bewertungseinheiten in Anforderungsbereichen | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MI | BB | SV | KK | DI | ID | AL | SA | IS | IMG | I | II | III |
| 1 | X | | X | X | X | X | | | | | 5 | 5 | 2 |
| 2 | | | | | X | X | | | | | | 6 | |
| 3.1 | | | | X | X | X | X | | | | 4 | 1 | |
| 3.2 | | | | X | X | X | X | | | | 3 | 2 | |
| 4 | X | | | | | X | X | | | | | 6 | |
| 5 | | X | | | | | | | | X | | 2 | 4 |
| Summe 40 | | | | | | | | | | | 12 | 22 | 6 |

## *Call a Bike*

Foto: Wikipedia / Missy Wagner

**Abbildung 4.12: CallBikes in Hamburg.**

**Call a Bike**

***Anmerkungen***

▷ grundlegendes Anforderungsniveau
▷ vorgesehene Bearbeitungszeit: 90 min

***Aufgabe***

Als *CallBikes* werden Fahrräder bezeichnet, die telefonisch oder online gebucht werden können. Um ein CallBike ausleihen zu können, ist vorab eine Registrierung zwingend notwendig. Bezahlen kann man wahlweise per Kreditkarte unter Angabe der Kreditkartennummer und der dreistelligen Prüfziffer oder per Mobiltelefon durch Abbuchung.

Die CallBikes sind mit elektronischen Nummernschlössern verriegelt. Ruft man die auf dem Schloss angegebene Telefonnummer an, wird einem der Code

zum Öffnen des Nummernschlosses angesagt. Mit der Eingabe des Codes beginnt der Ausleihvorgang.

Die Abgabe des CallBikes erfolgt durch Anschließen des Fahrrads an gekennzeichneten Standorten (siehe Abbildung 4.12, vorige Seite) und anschließendem Drücken des Sperrknopfes. Der Quittungscode erscheint. Der Ausleihvorgang wird mit der Übermittlung des Codes und der Standortnummer abgeschlossen.

Für den Ausleihvorgang eines Fahrrads beschreibt das in der folgenden Abbildung 4.13 dargestellte ER-Modell Teile der verwendeten Datenbank.

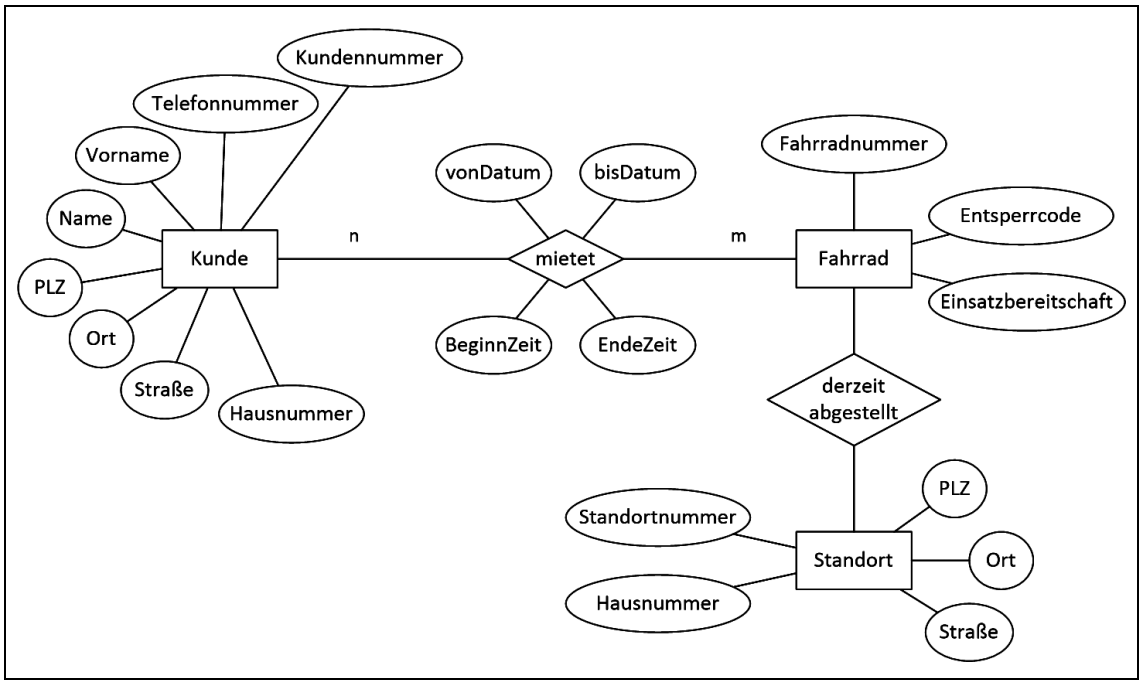



**Abbildung 4.13.**

*Modellierung*

1. Beschreiben Sie den Aufbau eines Datenbanksystems.

2. Bestimmen Sie die Kardinalität der Beziehung *derzeit_abgestellt*. Begründen Sie.

3. Geben Sie für die Attribute *Einsatzbereitschaft* und *Entsperrcode* den zu wählenden Datentyp an. Begründen Sie.

4. Übertragen Sie das ER-Modell in ein Relationenschema. Kennzeichnen Sie Primär- und Fremdschlüssel unterschiedlich.

5. Nennen Sie zwei Maßnahmen zum Schutz der im Ausleihprozess anfallenden personenbezogenen Daten.

*SQL-Abfragen*

Geben Sie folgende Abfragen in SQL an:

6. Erstellen Sie eine Liste defekter Fahrräder.

7. Geben Sie die Namen aller Personen an, die am 14.05.2012 ein Fahrrad zurückgegeben haben.

*Mobilfunk und Co.*

Ein Bahnkunde möchte sich kurz entschlossen auf dem Bahnhof ein Fahrrad mieten. Für die Registrierung bei »Call a Bike« kann er wahlweise mit dem Anbieter telefonieren oder den Hotspot auf dem Bahnhof nutzen.

8. Vergleichen Sie beide Möglichkeiten der Anmeldung hinsichtlich Praxistauglichkeit und Sicherheitsrisiken.

9. Nennen Sie zwei Geräte, die die Nutzung des Hotspots ermöglichen.

10. Beschreiben Sie mithilfe eines Schichtenmodells den Datenfluss der vom Hotspot ausgesendeten Signale bis zur Darstellung im Web-Browser.

11. In diesem Netzwerk werden Übertragungsgeschwindigkeiten von 14,4 MBit pro Sekunde erreicht.
11.1 Berechnen Sie das Datenvolumen, das maximal in einer Minute übertragen werden kann.
11.2 Begründen Sie, dass die benötigte Zeit für die Übertragung des berechneten Datenvolumens in der Realität meist größer ist.

## *Lösungshinweise*

***Lösungshinweise zu den Aufgaben***

1 – mögliche Lösung

Ein Datenbanksystem (DBS) besteht aus einer Datenbasis und einem Datenbank-Managementsystem (DBMS).
Das DBMS stellt die Schnittstelle zwischen Benutzer und Datenbasis dar und dient der effizienten Speicherung und Abfrage der strukturierten Daten.
Die Datenbasis beinhaltet alle notwendigen Daten zur Realisierung der gewünschten Anforderungen.

2 – mögliche Lösung

n:1 (Ein Fahrrad kann zu einem bestimmten Zeitpunkt nur an einem Ort abgestellt werden. An einem Ort können mehrere Fahrräder zur selben Zeit stehen.)

3 – mögliche Lösung

Einsatzbereitschaft: Datentyp boolean (Es gibt genau zwei Möglichkeiten. Entweder ist das Fahrrad einsatzbereit oder nicht.)
Code: Datentyp integer oder text (Der Typ integer reicht aus, um eine genügend große Anzahl von Kombinationen bereitzustellen und eine ausreichende Sicherheit zu gewährleisten. Führende Nullen sind dabei ausgeschlossen und können nur über den Datentyp text realisiert werden.)

4 – mögliche Lösung

KUNDE(**Kundennummer**, Name, Vorname, Telefonnummer, PLZ, Ort, Straße, Hausnummer)
MIETET(***Kundennummer***, ***Fahrradnummer***, **vonDatum**, bisDatum, BeginnZeit, EndeZeit)
FAHRRAD(**Fahrradnummer**, Entsperrcode, Einsatzbereitschaft, *Standortnummer*)
STANDORT(**Standortnummer**, PLZ, Ort, Straße, Hausnummer)

5 – mögliche Lösung

Kennwortauthorisierung, verteilte Datenspeicherung, Trennung von Ausleih- und Bezahlvorgang.

6 – mögliche Lösung

```
SELECT * FROM FAHRRAD
WHERE FAHRRAD.Einsatzbereitschaft = false;
```

7 – mögliche Lösung

```
SELECT KUNDE.Name
FROM MIETET InnerJOIN KUNDE
   ON KUNDE.InnerJOIN = MIETET.Kundennummer
WHERE MIETET.bisDatum = "14.05.2012";
```

8 – mögliche Lösung

Sicherheitsrisiken: Übertragung von Daten.
Praxistauglichkeit: persönlicher Kontakt beim Telefonieren, Zugangsdaten zum Bahnhofshotspot erforderlich.

9 – mögliche Lösung

Laptop mit WLAN-Adapter, WLAN-fähiges Mobiltelefon.

10 – mögliche Lösung

Beschreibung des Datenflusses mithilfe des OSI- bzw. des TCP/IP-Schichtenmodells.
Netzzugangsschicht: Empfang der Funkdaten
Internetschicht: Auflösung der IP-Adresse
Transportschicht: Vollständigkeit der Pakete, Reihenfolge,
Korrektheit der Übertragung, Zuordnung des Anwendungsports
Anwendungsschicht: Anzeige der Webseite im Browser

11.1 – mögliche Lösung

14,4 MBit × 60 / 8 = 108 MByte.

11.2 – mögliche Lösung

Verkleinerung der Bandbreite bei mehreren aktiven Nutzern.

Quelle:
Ministerium für Bildung, Wissenschaft und Kultur Mecklenburg-Vorpommern (Hrsg.): Zentralabitur 2012 – Informatik. Schwerin: 2012, S. 7–8.

***Zuordnung zu den Prozess-, Inhalts- und Anforderungsbereichen***

| Aufgabe | Prozessbereiche | | | | | Inhaltsbereiche | | | | | Bewertungseinheiten in Anforderungsbereichen | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MI | BB | SV | KK | DI | ID | AL | SA | IS | IMG | I | II | III |
| 1 | X | | | | | | | | X | | 3 | | |
| 2 | | X | | | | X | | | | | | 2 | |
| 3 | | X | | | | X | | | | | 1 | 2 | |
| 4 | | | | | X | X | | | | | 1 | 1 | 2 |
| 5 | | X | | | | X | | | | | | 2 | 2 |
| 6 | X | | | | | X | | | | | | 1 | |
| 7 | X | | | | | X | | | | | | 1 | 1 |
| 8 | | X | | | | | | | X | | | 1 | 2 |
| 9 | | | X | | | | | | X | | 2 | | |
| 10 | | | X | | | | | | X | | 1 | 2 | |
| 11.1 | | X | | | | | | | X | | | 2 | |
| 11.2 | | X | | | | | | | X | | | 1 | |
| Summe 30 | | | | | | | | | | | 8 | 15 | 7 |

## *Gartenarchitektur*


**Abbildung 4.14.**

**Gartenarchitektur**

***Anmerkungen***

▷ erhöhtes Anforderungsniveau
▷ vorgesehene Bearbeitungszeit: 150 min

***Aufgabe***

Eine kleine Softwarefirma hat den Auftrag, eine Grafikanwendung zu entwickeln, die die Erstellung kleiner Karten, beispielsweise zur Anlage eines Parks oder eines Gartens, ermöglicht. Der Kunde liefert die in Abbildung 4.14 wiedergegebene Skizze eines Parks, die er mit dem Programm erstellen möchte.

Die Gartenanlagen sollen aus Beeten bzw. Wegen bestehen, deren Begrenzungen klaren Linien folgen. Außerdem soll es möglich sein, Büsche, Bäume und Hecken darzustellen. Im Park soll es außerdem Brunnen geben. Dabei wird vom Auftraggeber ausdrücklich darauf hingewiesen, dass nur ein geringes Budget zur Verfügung steht. Deshalb werden einfache Darstellungen akzeptiert. Das Programm soll nur streng geometrische Formen wie Dreiecke oder Ovale erlauben.

1. Die folgende Abbildung 4.15 zeigt den ersten Klassenentwurf als vereinfachtes UML-Diagramm. Im *Material 1* werden die Methoden und Attribute der einzelnen Klassen im Detail dargestellt.
   Erläutern Sie die Beziehungen zwischen den Klassen.

**Abbildung 4.15.**

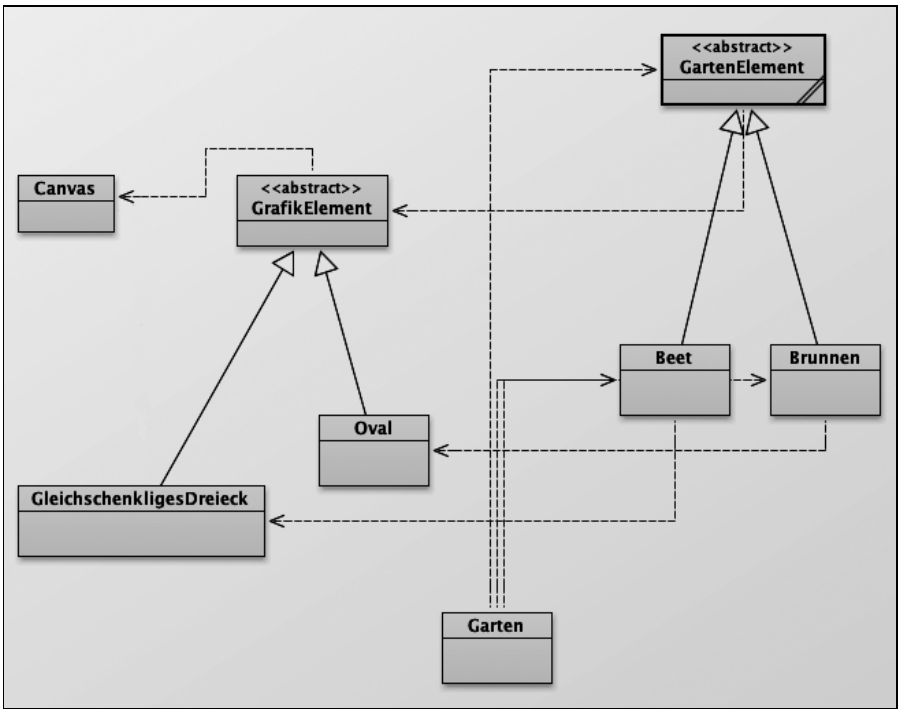



**Abbildung 4.16.**

2. Ein Objekt der Klasse GARTEN wurde dazu genutzt, die Abbildung 4.16 zu zeichnen.
   Nennen Sie die vier Objekte, die von der Klasse GARTEN erzeugt werden.
   Die Klasse GARTEN enthält die Methode zeichne().
   Beschreiben Sie ausführlich, welche Objekte betroffen sind und wie sie agieren, damit dieses Bild auf der Zeichenfläche erscheint.

3. Es soll ein Konstruktor der Klasse BEET ergänzt werden, bei dem die Größe und die Farbe des Gartenelements als Parameter übergeben werden können.
   Erläutern Sie, welche Aufgabe Konstruktoren in der objektorientierten Programmierung haben. Begründen Sie, weshalb häufig mehrere Konstruktoren verwendet werden.

Erläutern und implementieren Sie den neuen Konstruktor. Der aktuelle Konstruktor ist im *Material 2* enthalten.

4. Der Kunde wünscht eine Erweiterung des Programms, sodass auch Beete in der Form eines rechtwinkligen Dreiecks dargestellt werden können.
   Entwickeln Sie eine Klasse RechtwinkligesDreieck. Es ist dabei ausreichend, wenn Sie für die Position des rechten Winkels nur einen einfachen Spezialfall berücksichtigen.
   Erläutern Sie Ihren Entwurf und implementieren Sie diese Klasse.

5. Damit Beete und Brunnen vergrößert und verkleinert werden können, soll eine Methode skaliere implementiert werden.
   Begründen Sie, dass es am einfachsten ist, die Methode skaliere in der Klasse GartenElement zu implementieren.
   Implementieren Sie diese Methode.

6. Mit der Anwendung sollen auch Baumreihen dargestellt werden können. Dazu werden der Anfangspunkt, der Endpunkt und die Anzahl der Bäume in der Baumreihe angegeben.
   Erläutern Sie den Entwurf und implementieren Sie die Klassen.

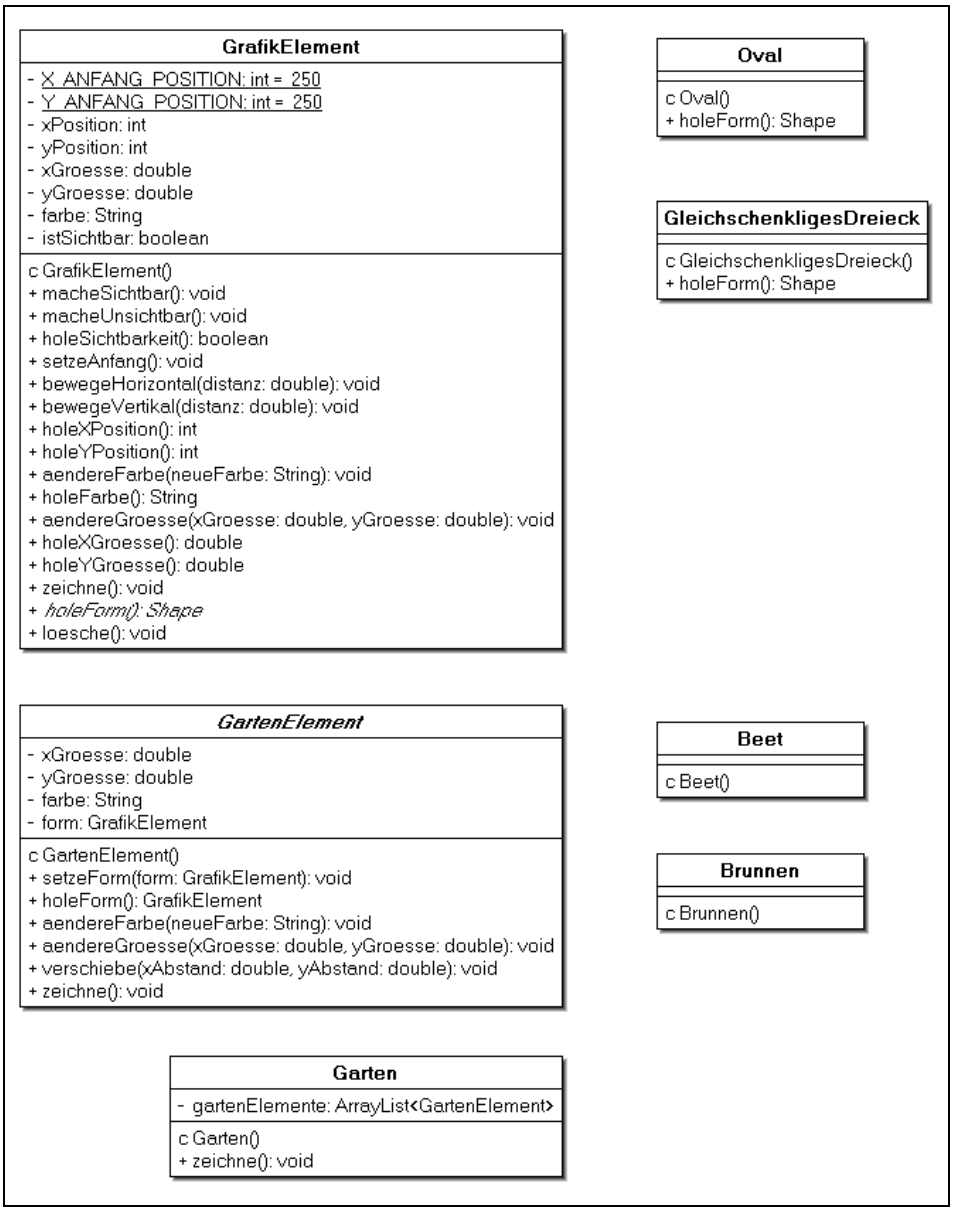



**Abbildung 4.17.**

7. Nachdem der Entwurf des Gartens erstellt ist, möchte der Kunde wissen, welche Gesamtfläche alle Beete und Brunnen seines Gartens zusammen belegen.
   Erläutern Sie, wie diese zusätzliche Funktionalität implementiert werden sollte.
   Erläutern Sie, wie die Berechnung der Gesamtfläche zur Laufzeit konkret erfolgt.

*Material 1*

Siehe Abbildung 4.17.

*Material 2*

```
1 /**
2  * Konstruktor der Klasse Beet
3  */
4 public Beet()
5 {
6     setzeForm(new GleichschenkligesDreieck());
7     aendereFarbe("green");
```

```
8     aendereGroesse(30.0, 80.0);
9 }
```

```
1  // Die Klasse gleichschenkliges Dreieck
2  import java.awt.*;
3  import java.awt.geom.*;
4
5  public class GleichschenkligesDreieck extends GrafikElement
6  {
7     /**
8     * Erzeugt ein Gleichschenkliges Dreieck.
9     */
10    public GleichschenkligesDreieck()
11    {
12           aendereFarbe("yellow");
13           aendereGroesse(100, 30);
14     }
15
16     /**
17      * Holt den Umriss des gleichschenkligen Dreiecks.
18      *
19      * @return Shape    Umriss des gleichschenkligen Dreiecks.
20      */
21     public Shape holeForm()
22     {
23           GeneralPath form = new GeneralPath();
24           form.moveTo(holeXPosition() - holeXGroesse() / 2.0,
               holeYPosition());
25           form.lineTo(holeXPosition(), holeYPosition() +
               holeYGroesse());
26           form.lineTo(holeXPosition() + holeXGroesse() / 2.0,
               holeYPosition());
27           form.closePath();
28           return form;
29     }
30 }
```

## Lösungshinweise

***Lösungshinweise zu den Aufgaben***

1 – mögliche Lösung

Der GARTEN setzt sich aus BRUNNEN und BEETEN zusammen, es handelt sich um eine Teile-Ganzes-Beziehung.

BRUNNEN und BEETE sind Spezialfälle der Klasse GARTENELEMENT; sie erben von dieser Klasse. GLEICHSCHENKLIGESDREIECK und OVAL sind Spezialfälle der Klasse GRAFIKELEMENT; sie erben von dieser Klasse.

Das GARTENELEMENT benutzt GRAFIKELEMENT. Das Beet benutzt GLEICHSCHENKLIGESDREIECK, der BRUNNEN benutzt OVAL.

*Anmerkung:* Die meisten Methoden von GRAFIKELEMENT werden von der allgemeinen Klasse vorgegeben, die abstrakte Methode holeForm wird in den speziellen Klassen überschrieben (implementiert). Die von GARTENELEMENT abgeleiteten Klassen unterscheiden sich nur in den Konstruktoren.

2 – mögliche Lösung

Es gibt einen Brunnen und drei Beete. Es werden folglich diese vier Objekte erzeugt.

Wird die Methode zeichne() aufgerufen, so muss über die Elemente der ArrayList GartenElemente (siehe UML-Diagramm) iteriert werden. Dadurch erhält jedes einzelne GartenElement die Nachricht zeichne(). Je nachdem, ob ein GartenElement ein Beet oder ein Brunnen ist, wird damit deren spezifische Methode zeichne() aufgerufen. Diese Objekte reichen die Nachricht zeichne an ihre jeweiligen GrafikElemente, nämlich GleichschenkligesDreieck und Oval, weiter. Dabei werden Farbe, Abmessungen und Position über das Attribut form kommuniziert.

3 – mögliche Lösung

Der Konstruktor hat die Aufgabe, einen gültigen Anfangszustand für ein Objekt sicherzustellen.

Mehrere Konstruktoren sind sinnvoll, damit Objekte unterschiedlicher Anfangszustände einfach erzeugt werden können.

Für den neuen Konstruktor Beet werden drei Parameter benötigt. Der Parameter für die Farbe muss vom Typ string sein und die Abmessungen des Beetes (xGroesse und yGroesse) vom Typ double. *Anmerkung:* Dieser Satz kann in der Erläuterung fehlen, wenn dies im Programmtext des Konstruktors richtig implementiert ist.

Da die Zustandsgrößen in der übergeordneten Klasse private angelegt sind, kann der Konstruktor die Werte nicht direkt setzen, sondern muss dies über die »Setter« der abstrakten Klasse tun.

```
1 public Beet(String farbe, double xGroesse, double yGroesse)
2 {
3     setzeForm(new GleichschenkligesDreieck());
4     aendereFarbe(farbe);
5     aendereGroesse(xGroesse, yGroesse);
6 }
```

4 – mögliche Lösung

Dazu muss eine Klasse geschrieben werden, die von der abstrakten Klasse GRAFIKELEMENT abgeleitet wird. In der Klasse wird dann die Methode holeForm implementiert.

*Anmerkung:* Für die Lösung ist es ausreichend, wenn der Prüfling ein Dreieck zeichnet, dessen Katheten parallel zur x- bzw. y-Achse liegen.

Bei der folgenden Lösung wird ein rechtwinkliges Dreieck so gezeichnet, dass der rechte Winkel oben links positioniert wird.

```
1 public class RechtwinkligesDreieck extends GrafikElement
2 {
3     public RechtwinkligesDreieck()
4     {
5           aendereGroesse(100, 30);
6           aendereFarbe("green");
7     }
8
9     public Shape holeForm()
10    {
11          GeneralPath form = new GeneralPath();
12          form.moveTo(holeXPosition(), holeYPosition());
13          form.lineTo(holeXPosition(),
14                      holeYPosition() + holeYGroesse());
15          form.lineTo(holeXPosition() + holeXGroesse(),
16                      holeYPosition());
17          form.closePath();
18          return form;
19    }
20 }
```

5 – mögliche Lösung

Die Methode skaliere muss nur einmal in der Klasse GARTENELEMENT implementiert werden, da die Aufgabe der Skalierung für alle GARTENELEMENT-Objekte gleich ist. Es müssen die Attribute für die Größe des Gartenelements mit dem Skalierungsfaktor multipliziert werden.

Sollte die Methode in der Klasse BEET bzw. BRUNNEN implementiert werden, müssen »Getter« für die bestehenden Größen implementiert werden, da die Sichtbarkeit auf den Wert private gestellt wurde.

Alternativ ließe sich die Methode auch mithilfe von aendereGroesse realisieren.

Beispiel einer Methode:

```
1 /**
2  * skaliere aendert die Groesse um den angegebenen Faktor
3  * @param  faktor   Veränderung
4  */
```

```
5  public void skaliere(double faktor)
6  {
7      xGroesse *= faktor;
8      yGroesse *= faktor;
9  }
```

*Anmerkung:* Die Prüflinge könnten auch eine zusätzliche Anweisung form.aendereGroesse(xGroesse, yGroesse); nach Zeile 8 hinzufügen, da für sie nicht erkennbar ist, an welcher Stelle die Attribute xGroesse und yGroesse des Grafikelements aktualisiert werden.

6 – mögliche Lösung

Zuerst sollte eine Klasse BAUM erstellt werden, die entsprechend BRUNNEN und BEET allein den Konstruktor enthält. Ohne diese Klasse ist die Modellierung unbefriedigend.

```
1 public class Baum extends GartenElement
2 {
3     public Baum()
4     {
5         setzeForm(new Oval());
6         aendereFarbe("green");
7         aendereGroesse(50.0, 50.0);
8     }
9 }
```

In jedem Fall ist die Klasse BAUMREIHE erforderlich. Bei der hier vorgestellten Lösung werden insgesamt fünf Attribute benötigt, und zwar *Anfangsposition (x, y)*, *Endposition (x, y)*, *Anzahl*, deren Werte im Konstruktor definiert werden. Notwendig ist der Einsatz einer Sammlungsklasse für die zugehörigen Baum-Objekte und bei den Zugriffen eine Iteration.

```
1 public class Baumreihe extends GartenElement
2 {
3     private double xAnfang, yAnfang;
4     private double xEnde, yEnde;
5     private int anzahl;
6     private ArrayList<Baum> baumReihe;
7
8     /**
9      * Konstruktor für Objekte der Klasse Baumreihe
10     */
11    public Baumreihe(double xAnfang, double yAnfang,
12                     double xEnde, double yEnde, int anzahl)
13    {
14        GartenElement baum;
15        baumReihe = new ArrayList<Baum>();
16        this.xAnfang = xAnfang;
17        this.yAnfang = yAnfang;
18        this.anzahl = anzahl;
19        double dx = (xEnde - xAnfang) / (anzahl - 1);
20        double dy = (yEnde - yAnfang) / (anzahl - 1);
21        for (int i = 0; i < this.anzahl; i++)
22        {
23            baumReihe.add(new Baum());
24            baumReihe.get(i).verschiebe(xAnfang + i*dx,
25                                        yAnfang + i*dy);
26        };
27    }
28
29    /**
30     * Zeichnet die Baumreihe
31     */
32    public void zeichne()
33    {
34        for (Baum baum: baumReihe) baum.zeichne();
35    }
36 }
```

Da eine Baumreihe ein spezielles Gartenelement ist, sollte die Klasse BAUM REIHE von der Klasse GARTENELEMENT erben. Dies ermöglicht zudem den einfachen Zugriff in der Klasse GARTEN.

Für eine vollständige Anwendung müssen noch die »Setter« von GARTENELEMENT überschrieben werden.

*Anmerkung:* Die letzte Aussage wird von den Prüflingen nicht erwartet. Eine Lösung, bei der die Bäume nur horizontal oder vertikal ausgerichtet gezeichnet werden können, ist zulässig.

7 – mögliche Lösung

Dieses kann so realisiert werden, dass eine Methode berechneGesamtFlaeche in GARTEN implementiert wird, die über alle Gartenelemente iteriert. In der Klasse GARTENELEMENT wird eine abstrakte Methode berechneFlaeche benötigt, die in den abgeleiteten Klassen BEET und BRUNNEN überschrieben wird.

Sowohl die Beet- als auch die Brunnenobjekte können über die Methode holeForm() auf das Objekt vom Typ GrafikElement zugreifen. Mithilfe der Methoden holeXGroesse() und holeYGroesse() des Grafikelements können die aktuellen Abmessungen ermittelt und damit die Fläche berechnet werden.

Wird die Methode berechneGesamtFlaeche der Klasse GARTEN aufgerufen, so wird die Methode berechneFlaeche aufgerufen, und zwar in der von dem jeweiligen Objekttyp (Beet- bzw. Brunnenobjekt) überschriebenen Variante und anschließend alle Werte addiert.

Quelle:
Freie und Hansestadt Hamburg – Behörde für Schule und Berufsbildung (Hrsg.): Abituraufgaben Informatik 2014. Hamburg: 2014.

***Zuordnung zu den Prozess-, Inhalts- und Anforderungsbereichen***

| Aufgabe | Prozessbereiche | | | | | Inhaltsbereiche | | | | | Bewertungseinheiten in Anforderungsbereichen | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MI | BB | SV | KK | DI | ID | AL | SA | IS | IMG | I | II | III |
| 1 | X | | | | X | | | | X | | 2 | 3 | |
| 2 | X | | | | X | | | | X | | | 3 | 2 |
| 3 | X | X | | X | | X | X | | X | | 5 | | |
| 4 | X | | | X | | X | X | | X | | 6 | 3 | |
| 5 | X | X | | X | X | X | | | X | | | 6 | 2 |
| 6 | X | | | X | | X | X | | X | | | 5 | 3 |
| 7 | X | X | | X | | X | | | X | | | 5 | 5 |
| | | | | | | | | | | Summe 50 | 13 | 25 | 12 |

## Der Bote

***Anmerkungen***

▷ erhöhtes Anforderungsniveau
▷ vorgesehene Bearbeitungszeit: 150 min

***Aufgabe***

Im Zeitalter des Absolutismus (17. Jahrhundert) herrscht Ludwig XIV. in Frankreich. Die Kommunikation des Königs mit den Befehlshabern des Militärs und innerhalb des Militärs soll stärker abgesichert werden. Bislang werden wichtige Mitteilungen von einem als vertrauenswürdig geltenden Boten übermittelt, entweder *mündlich* oder mit dem Caesar-Verfahren *verschlüsselt in schriftlicher* Form.

Eine Übermittlung von Ludwig XIV. zum militärischen Oberbefehlshaber läuft wie folgt ab:

Der König lässt von seinem Schreiber eine Mitteilung notieren. Der Schreiber verschlüsselt die Mitteilung nach dem Caesar-Verfahren. Der Berater des Königs wählt unter den Boten einen aus, der dem Oberbefehlshaber bekannt ist.



**Abbildung 4.18:**
**Ein französischer Bote zu Pferd.**


Quelle: LOG-IN-Archiv

Der Bote nimmt die verschlüsselte Mitteilung und eilt zum Oberbefehlshaber. Nachdem sich der Oberbefehlshaber davon überzeugt hat, dass ihm der Bote bekannt ist, nimmt er die Botschaft entgegen und entschlüsselt sie.

1. Geben Sie eine Definition der grundlegenden Begriffe *Vertraulichkeit*, *Integrität* und *Authentizität* an und erläutern Sie diese Begriffe anhand des Beispiels.

2. Protokolle sollen den geordneten Ablauf einer Kommunikation ermöglichen. Erläutern Sie, inwieweit beim bisherigen Ablauf der Kommunikation der geordnete Ablauf sichergestellt wird.

3. Beschreiben Sie, wie man einen mit dem Caesar-Verfahren verschlüsselten Text ohne Kenntnis des Schlüssels entschlüsseln kann.

4. Bereits im 16. Jahrhundert entwickelte Blaise de Vigenère (1523–1596) das nach ihm benannte Verfahren.
   Stellen Sie das Vigenère-Verfahren an einem selbst gewählten Beispiel dar.
   Erklären Sie, wovon die Sicherheit des Verfahrens abhängt.

5. Im *Material* ist der Quelltext für eine Vigenère-Verschlüsselung abgedruckt.
   Erläutern Sie die Funktionsweise der Funktion vigenere sowie der zugehörigen Hilfsfunktion anhand der ersten beiden Buchstaben für den Beispielaufruf (vigenere '(I N F O R M A T I K) '(H O L Z)) .
   Erklären Sie darüber hinausgehend, wie die wiederholte Anwendung des Schlüsselwortes realisiert worden ist.

6. Es gibt sogenannte schwache Schlüssel, die einen Angriff auf einen damit erstellten Geheimtext vereinfachen. Beim Vigenère-Verfahren sollte der Schlüssel deshalb möglichst keine Buchstabenwiederholungen enthalten.
   Entwickeln und implementieren Sie eine Funktion (ggf. mit weiteren Unterfunktionen), die den Schlüssel entsprechend überprüft.

7. Es gibt mehrere Gründe, das bisherige Verfahren zu verändern. Dazu gibt es eine Reihe von Vorschlägen:
   1. Die Boten erhalten einen Dienstausweis.
   2. Es werden in Zukunft Mitteilungen nur noch schriftlich übergeben.
   3. Der verwendete Schlüssel wird jeden Monat geändert.
   4. Als Verschlüsselungsverfahren wird das Vigenère-Verfahren eingeführt.
   Bewerten Sie jeden dieser Vorschläge.
   Begründen Sie, welcher oder welche Vorschläge umgesetzt werden sollten.

*Material*

*Hinweis:* statt der Funktion first kann car benutzt werden.
statt der Funktion rest kann cdr benutzt werden.

```
1  (define alphabet '(A B C D E F G H I J K L M N O P Q R S T U V W
2                     X Y Z))
3
4  ;pos-symbol liefert die Position eines Symbols (Buchstabe)
5  ;           in einer Liste.
6  ;Die Nummerierung beginnt mit 0.
7  ;Ist das Symbol nicht Element der Liste,
8  ;liefert die Funktion als Wert die Länge der Liste + 1.
9  ;Beispielaufruf: (pos-symbol 'C alphabet) --> 2,
10 ;                aber (pos-symbol 'c alphabet) --> 27
11 ;
12 (define (pos-symbol symbol liste)
13   (cond ((null? liste) 1)
14         ((equal? (first liste) symbol) 0)
15         (else (+ 1 (pos-symbol symbol (rest liste))))))
16
17 ;symbol-pos liefert das Symbol, das sich an der n-ten Stelle
18 ;                einer Liste befindet.
19 ;Beispielaufruf: (symbol-pos 4 alphabet) --> E
20 ;Liegt ein Index außerhalb der Liste,
21 ;                wird eine leere Liste zurückgeliefert.
```

```
22 (define (symbol-pos index liste)
23   (cond ((null? liste) '())
24         ((= index 0) (first liste))
25         (else (symbol-pos (- index 1) (rest liste)))))
26
27 ;Beispielaufruf (caesar 'A 5) --> F
28 (define (caesar klarzeichen schluessel)
29   (symbol-pos
30    (modulo (+ (pos-symbol klarzeichen alphabet) schluessel)
31               (length alphabet))
32    alphabet))
33
34 ;vigenere-hilf (Hilfsfunktion),
35 ;              da für den Aufruf nicht benutzerfreundlich.
36 ;Beispielaufruf:
37 ;(vigenere-hilf '(I N F O R M A T I K) '(H O L Z) '(H O L Z)) -->
38 ;(P B Q N Y A L S P Y)
39 (define (vigenere-hilf klartextliste schluessel schluesselwort)
40   (cond ((null? klartextliste) '())
41         ((null? schluessel)
42          (vigenere-hilf klartextliste schluesselwort
43               schluesselwort))
44         (else (cons (caesar   (first klartextliste)
45                               (pos-symbol (first schluessel)
46                                alphabet))
47                     (vigenere-hilf
48               (rest klartextliste) (rest schluessel)
49               schluesselwort)))))
50
51 ;vigenere erhält einen Klartext als Liste von Symbolen sowie das
52 ;              Schlüsselwort,
53 ;ebenfalls als Liste.
54 ;Beispielaufruf (vigenere '(I N F O R M A T I K) '(H O L Z)) -->
55               (P B Q N Y A L S P Y)
56 (define (vigenere klartextliste schluesselwortliste)
57   (vigenere-hilf klartextliste schluesselwortliste
58               schluesselwortliste))
```

## Lösungshinweise

***Lösungshinweise zu den Aufgaben***

1 – mögliche Lösung

*Vertraulichkeit:* Nur ausgewählte befugte Personen können die Nachricht lesen.
*Integrität:* Die Nachricht kann während der Übertragung nicht verändert werden.
*Authentizität:* Die Identität des Absenders bzw. die genaue Zuordnung der Absenderadresse ist sichergestellt.

Vertraulichkeit wird mithilfe des Caesar-Verfahrens hergestellt (unsicher, siehe Aufgabe 3).

Integrität wird dadurch hergestellt, dass der Bote die Nachricht transportiert und damit schützt.

Authentizität wird dadurch hergestellt, dass der Absender und der Empfänger den Boten kennen.

2 – mögliche Lösung

Das »Protokoll« legt zwar den Ablauf der Ver- und Entschlüsselung sowie die Übertragung fest, sieht aber keinerlei Regelungen für den Fall etwaiger Fehler oder Probleme vor.

Die Liste der möglichen Fehler ist lang: unbekannter Bote, Bote kommt nicht an, Bote kommt ohne Nachricht an, …

Von den oben aufgeführten Aspekten ist nur der Fall geregelt, dass ein unbekannter Bote beim Empfänger ankommt; aber auch in diesem Fall bleibt offen, ob der Sender von dem Ereignis erfährt.

3 – mögliche Lösung

Mit der Häufigkeitsanalyse wird der häufigste Buchstabe im Geheimtext bestimmt. Dieser entspricht vermutlich dem häufigsten Buchstaben im Klartext (im Deutschen das »E«), daraus kann die Verschiebung (= Schlüsselbuchstabe)

bestimmt werden. Mithilfe der Verschiebung kann der Geheimtext entschlüsselt werden.

Der Prüfling darf auch die Brute-Force-Methode beschreiben. Er könnte gemeinsam mit 25 weiteren Mitschülerinnen und Mitschüler auf die Idee kommen, dass jeder eine Verschiebung ausprobiert. Des Weiteren könnte er auch allein mithilfe der ersten n Buchstaben 26 Verschiebungen ausprobieren, bis etwas Sinnvolles entsteht.

4 – mögliche Lösung

Nach der Festlegung des Schlüsselwortes wird der Klartext zyklisch verschlüsselt. Der Klartext wird in Blöcke von der Länge des Schlüsselwortes unterteilt, der erste (zweite, dritte, …) Buchstabe eines Klartextblocks wird mit dem ersten (zweiten, dritten, …) Buchstaben des Schlüsselwortes nach Caesar verschlüsselt.

*Beispiel:* INFORMATIK - HOLZ → PBQNYALSPY

Die Sicherheit hängt in erster Linie von dem Verhältnis Klartextlänge zu Schlüsselwortlänge ab. Erstens wird die Bestimmung der Schlüsselwortlänge mit zunehmender Schlüsselwortlänge tendenziell aufwendiger, und zweitens liefert die Häufigkeitsanalyse der gleich verschlüsselten Buchstaben mit zunehmender Schlüsselwortlänge immer schlechtere Ergebnisse.

5 – mögliche Lösung

In der Funktion vigenere-hilf werden Klartext und Schlüssel zeichenweise abgearbeitet. Dazu wird die Funktion caesar aufgerufen, die ein Symbol (ein Klartextzeichen) nach dem Caesar-Verfahren verschlüsselt. Klartext- und Schlüsselbuchstabe (I und H bzw. N und O) werden mit der Funktion pos-symbol in Zahlen (8 und 7 bzw. 13 und 14) umgewandelt, addiert und durch die Länge des Alphabets modular geteilt. Die dabei ermittelte Zahl (15 bzw. 1) wird mit der Funktion symbol-pos in einen Buchstaben (P bzw. B) umgewandelt.

Wenn das Schlüsselwort einmal abgearbeitet worden ist (null? schluessel), erfolgt der nächste Aufruf der Funktion vigenere-hilf mit dem vollständigen Schlüsselwort als zweitem Parameter.

Dies geschieht solange, bis die Bedingung (null? klartextliste) erfüllt ist, also der Klartext vollständig abgearbeitet wurde.

Der Geheimtext wird zeichenweise aufgebaut. Er entsteht beim Aufstieg aus der Rekursionstiefe, bei dem von unten nach oben zunächst das letzte, dann das vorletzte usw. und schließlich das erste Geheimzeichen in die anfangs leere Liste mit cons vorn eingefügt wird.

6 – mögliche Lösung

Der Schlüssel wird zeichenweise abgearbeitet. Jedes Zeichen wird in einer Hilfsfunktion daraufhin überprüft, ob es in dem noch folgenden Teil des Schlüssels enthalten ist. Die Funktion wird beendet, wenn ein Buchstabe doppelt im Schlüssel vorkommt oder das Ende bzw. der letzte Buchstabe des Schlüssels erreicht ist.

Andere Ansätze sind in Abhängigkeit von den jeweiligen Vorkenntnissen der Prüflinge ebenfalls möglich.

*Mustercode:*

```
1  (define (vorhanden? buchst liste)
2    (cond ((null? liste) #f)
3          ((equal? buchst (first liste)) #t)
4          (else (vorhanden? buchst (rest liste)))))
5
6  (define (test schluessel)
7    (cond ((null? schluessel) #t)
8          ((vorhanden? (first schluessel) (rest schluessel)) #f)
9          (else (test (rest schluessel)))))
10
11 (test '(H O L Z)) ergibt true
12 (test '(H O H L)) ergibt false
```

7 – mögliche Lösung

1: sinnvoll, dadurch müssen die einzelnen Personen sich nicht vorher schon einmal kennengelernt haben, auch sinnvoll bei Krankheit, aber Fälschungssicherheit ist schwer herzustellen.
2: sinnvoll, da ein Bote nicht unter Druck die Information preisgeben kann.
3: sinnvoll, um die Folgen etwaiger erfolgreicher Angriffe zeitlich zu begrenzen. Die regelmäßige Änderung eines Schlüssels könnte mithilfe eines Schlüsselwortbuchs erfolgen. Das ist relativ aufwendig und anfällig, weil das Schlüsselwortbuch gestohlen werden könnte. Alternativ könnte der Schlüssel auch monatlich durch einen Boten übermittelt werden. Dabei besteht jedoch die Gefahr, dass der Schlüssel abgefangen wird.
4: sinnvoll, da polyalphabetische Verfahren nicht so einfach anzugreifen sind wie monoalphabetische.

Alle Aspekte müssen gegeneinander abgewogen werden (insbesondere 1 und 3 erscheinen als umständlich oder schwierig unter den damaligen Verhältnissen). Eine mögliche Entscheidung würde also in erster Linie die Vorschläge 2 und 4 berücksichtigen. Vorschlag 3 könnte in Form eines Schlüsselwortbuchs für einen bestimmten Zeitraum berücksichtigt werden.

*Anmerkung:* Darüber hinaus können Einzelaspekte konkretisiert und/oder ergänzt werden wie Länge und andere Eigenschaften des Schlüsselwortes, Versiegelung der Nachricht, …

Quelle:
Freie und Hansestadt Hamburg – Behörde für Schule und Berufsbildung (Hrsg.): Abituraufgaben Informatik 2014. Hamburg: 2014

***Zuordnung zu den Prozess-, Inhalts- und Anforderungsbereichen***

| Aufgabe | Prozessbereiche | | | | | Inhaltsbereiche | | | | | Bewertungseinheiten in Anforderungsbereichen | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MI | BB | SV | KK | DI | ID | AL | SA | IS | IMG | I | II | III |
| 1 | | | | X | | | | | | X | 3 | 3 | |
| 2 | | X | | X | | | | | | X | 4 | 2 | |
| 3 | | | | X | | | X | | | X | 4 | 3 | |
| 4 | | | | X | | | X | | | X | | 4 | 3 |
| 5 | | | X | X | X | | X | | | | 2 | 5 | |
| 6 | X | | | | | X | X | | | | | 5 | 6 |
| 7 | | X | | | | | | | | X | | 3 | 3 |
| Summe 50 | | | | | | | | | | | 13 | 25 | 12 |

## CSS-Grammatik



***Anmerkungen***

▷ erhöhtes Anforderungsniveau
▷ vorgesehene Bearbeitungszeit: 80 min

***Aufgabe***

Mithilfe von Cascading Style Sheets (CSS) können Layout und Inhalt von HTML-Dokumenten getrennt dargestellt werden. Die Syntax von CSS wird vom WWW-Konsortium als Grammatik dargestellt. Dabei werden folgende Metasymbole benutzt:

| Metasymbol | Bedeutung |
|---|---|
| * | muss nicht auftreten, kann aber mehrfach wiederholt werden |
| + | muss ein- oder mehrfach wiederholt werden |
| [ ] | tritt in dieser Kombination auf |
| " " | zwischen den Anführungszeichen steht ein Terminalsymbol |

Ein Metasymbol gehört nicht zur eigentlichen CSS-Sprache, sondern hilft, die Sprache zu definieren (Metaebene).

Die für diese Aufgabe vereinfachten Produktionen der CSS-Grammatik sehen wie folgt aus:

```
Stylesheet → Ruleset+
Ruleset → Selector ["," Selector]*
                    "{" Declaration [";" Declaration]* "}"
Selector → Identifier
Declaration → Property ":" Expression
Property → Identifier
Expression → Number | Identifier | Number "%"
```

Dabei ist Identifier ein Bezeichner aus Buchstaben, Ziffern oder dem Bindestrich, der mit einem Buchstaben beginnt, und Number eine Zahl aus den Ziffern 0 bis 9, die mit einer Ziffer beginnt, optional gefolgt von einem Dezimalpunkt und mindestens einer weiteren Ziffer.

1. Geben Sie die Terminale, die Nichtterminale und das Startsymbol der CSS-Grammatik an, beschreiben Sie die angegebenen Produktionen und ordnen Sie die Grammatik begründet in die Chomsky-Hierarchie ein.

2. Stellen Sie die angegebenen Produktionen als Syntaxdiagramme dar. Syntaxdiagramme für Identifier und Number brauchen nicht gezeichnet zu werden.

3. Entwerfen Sie äquivalente Produktionen, die statt mit Schleifen (Metasymbole + und *) mittels Rekursion definiert sind.

4. Geben Sie zum Nachweis, dass h1{color:blue;line-height:120.5%} ein Stylesheet ist, eine Ableitung an oder zeichnen Sie einen Ableitungsbaum.

5.1 Zeichnen Sie je einen endlichen Automaten für Identifier und Number.
5.2 Modellieren Sie einen deterministischen endlichen Automaten als Akzeptor für die durch die CSS-Grammatik definierte Sprache.

## *Lösungshinweise*

***Lösungshinweise zu den Aufgaben***

1 – mögliche Lösung

Terminale: {a, b, …, z, A, B,…, Z, –, 0, 1,…9, ".", ",", ";", ":", "{", "}", "%"}
Nichtterminale: {Stylesheet, Ruleset, Selector, Declaration, Property, Expression, Number, Identifier}
Startsymbol S: Stylesheet

Ein Stylesheet besteht aus einer nichtleeren Reihe von Rulesets. Ein Ruleset beginnt mit einem Selector, auf den weitere mit Komma getrennte Selectoren folgen können. In einem Paar geschweifter Klammern folgen eine oder mehrere Declarations, die mit Semikolons zu trennen sind. Ein Selector ist ein Identifier. Eine Declaration besteht aus einer Property, dem Doppelpunkt als Terminal und einer Expression. Eine Property wird zu einem Identifier abgeleitet. Eine Expression ist eine Number, ein Identifier oder eine Number gefolgt von dem Terminal %.

Es handelt sich um eine kontextfreie Grammatik (Typ 2), weil die Produktionsköpfe jeweils nur aus einem einzigen Nichtterminal bestehen und die Anzahl der Terminale und Nichtterminale in den Produktionsrümpfen mindestens 1 ist. Sie ist nicht regulär, weil z. B. Property → Identifier keine zulässige Produktion für reguläre Grammatiken ist.

2 – mögliche Lösung

Siehe folgende Abbildung 4.19.

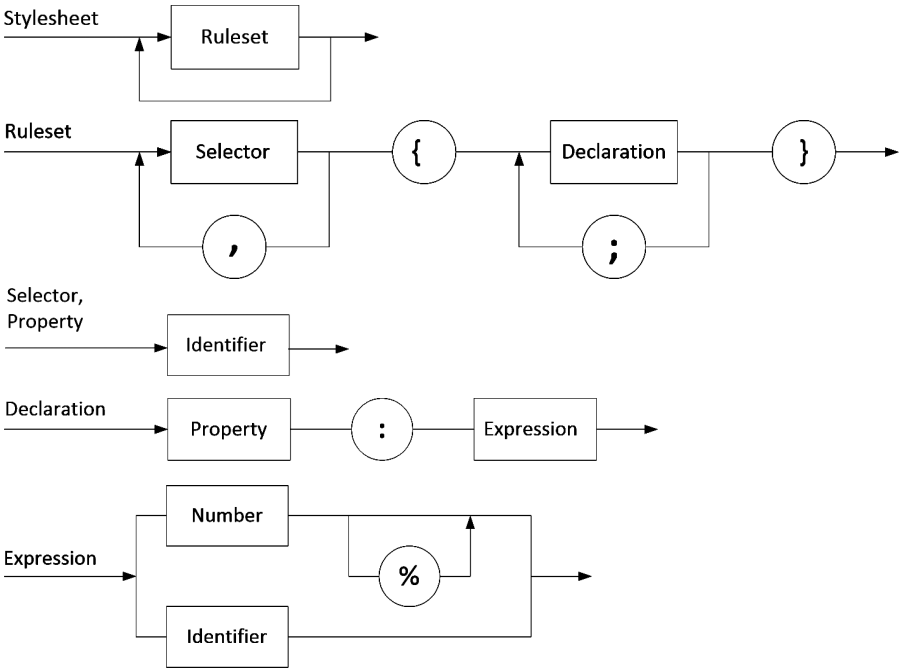



**Abbildung 4.19.**

3 – mögliche Lösung

Stylesheet → Ruleset | Ruleset Stylesheet
Ruleset → Selectors { Declarations }
Selectors → Selector | Selector , Selectors
Declarations → Declaration | Declaration ; Declarations
alle anderen Produktionen wie bisher

4 – mögliche Lösung

Siehe folgende Abbildung 4.20.

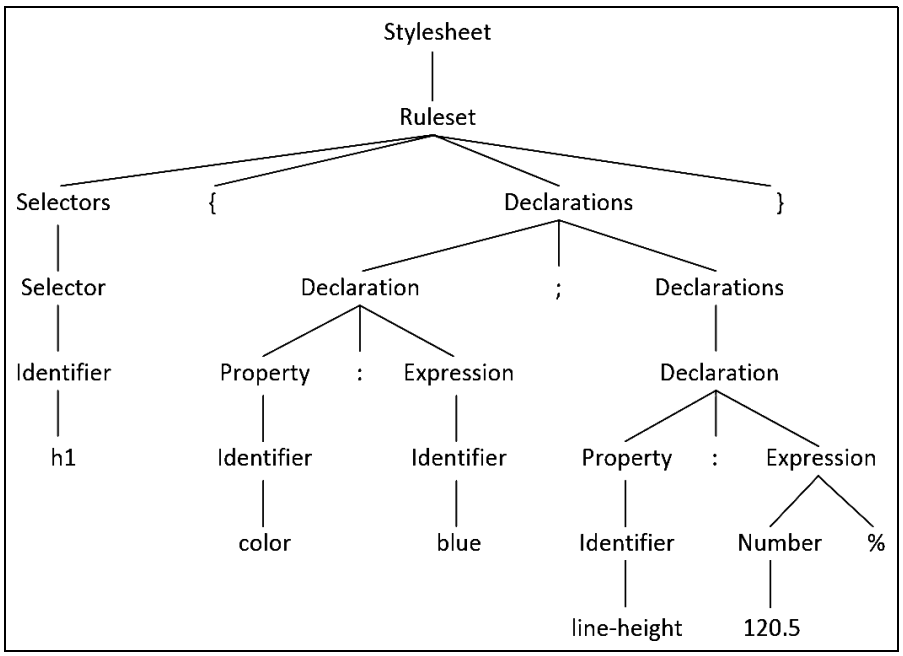



**Abbildung 4.20.**

5.1 – mögliche Lösung

Siehe folgende Abbildung 4.21.

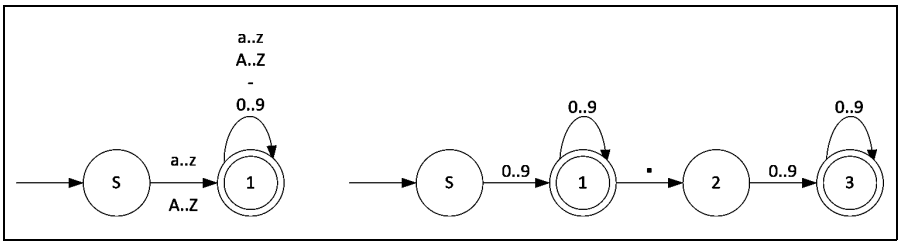



**Abbildung 4.21.**

5.2 – mögliche Lösung

Siehe folgende Abbildung 4.22.

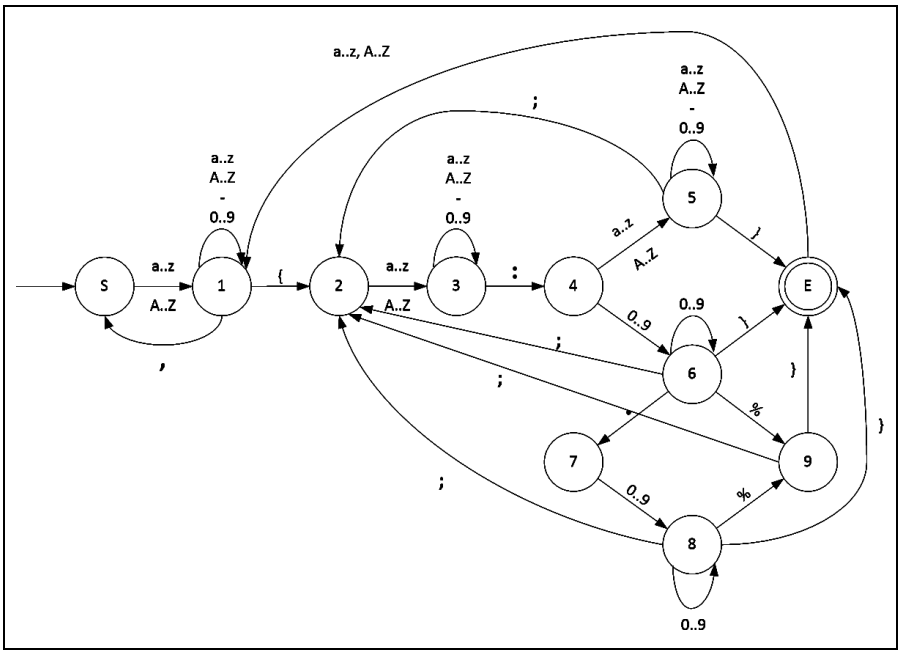



**Abbildung 4.22.**

Quelle:
Hessisches Kultusministerium (Hrsg.): Landesabitur Informatik 2010 – Leistungskurs. Wiesbaden: 2010.

***Zuordnung zu den Prozess-, Inhalts- und Anforderungsbereichen***

| Aufgabe | Prozessbereiche | | | | | Inhaltsbereiche | | | | | Bewertungseinheiten in Anforderungsbereichen | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MI | BB | SV | KK | DI | ID | AL | SA | IS | IMG | I | II | III |
| 1 | X | X | | X | | | | X | | | 5 | 1 | |
| 2 | | | | | X | | | X | | | | 6 | |
| 3 | | | | | X | | | X | | | | 4 | 2 |
| 4 | | | | | X | | | X | | | 3 | 3 | |
| 5.1 | | | | | X | | | X | | | 2 | | |
| 5.2 | | | X | | X | | | X | | | | | 4 |
| Summe 30 | | | | | | | | | | | 10 | 14 | 6 |

**Netzwerkfähiger Plotter**

***Anmerkungen***

▷ erhöhtes Anforderungsniveau
▷ vorgesehene Bearbeitungszeit: 120 min

***Aufgabe***

Für die Auswertung von Messwerten zweier Forschungsgruppen soll ein netzwerkfähiger Plotter entwickelt werden, der die funktionalen Zusammenhänge mithilfe eines Stifts direkt auf Papier zeichnen kann. Der Plotter besteht u. a. aus dem Steuerrechner, einem Minicomputer mit einer LAN- und einer USB-Schnittstelle und der Stiftmechanik.

Für die Programmierung der Stiftbewegung werden die Befehle der Sprache HPGL (Hewlett Packard Graphic Language) genutzt.

*Von-Neumann-Architektur*

1. Erstellen Sie eine beschriftete Skizze der Komponenten eines Von-Neumann-Rechners.

2. Vergleichen Sie Aufbau und Funktionsweise eines Kellerspeichers mit dem Speicher eines Von-Neumann-Rechners.

3. Nennen Sie die Befehlsphasen im Von-Neumann-Zyklus und geben Sie deren Aufgabe an.

*Netzanbindung*

Für die beiden Forschungsgruppen wurden eigenständige Rechnernetze entwickelt. Der Plotter soll so in das Netz integriert werden, dass auf ihn von allen Rechnern aus zugegriffen werden kann. Aus Sicherheitsgründen darf jedoch kein Zugriff zwischen den Teilnetzen der beiden Gruppen existieren.

Einen ersten Lösungsvorschlag zeigt die folgende Abbildung 4.23.

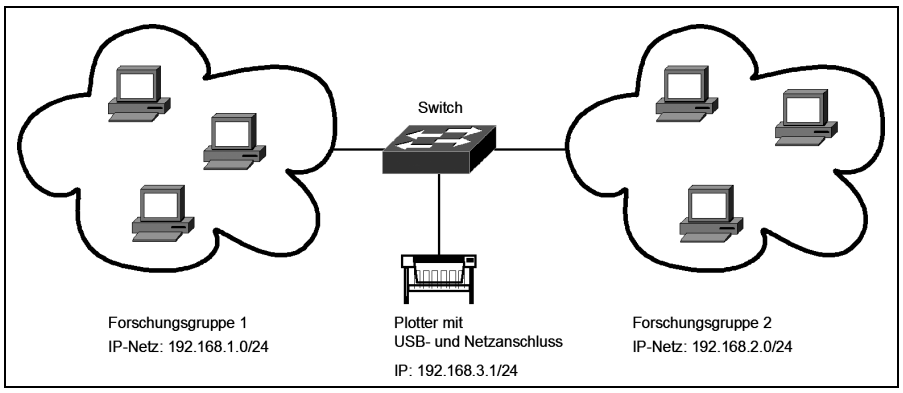



**Abbildung 4.23.**

4. Begründen Sie, dass sich der Plotter und die Geräte der Forschungsgruppe 1 in verschiedenen Teilnetzen befinden.

5. Begründen Sie, dass der Switch in diesem Beispiel nicht als Netzkoppelelement geeignet ist.

6. Beschreiben Sie eine Möglichkeit, wie beide Gruppen unter Berücksichtigung der Sicherheitsrichtlinien auf den Plotter zugreifen können.

*Plottersprache HPGL*

Die Plottersprache HPGL wird in verschiedene Befehlsgruppen unterteilt. Die Befehle für die Gruppe »Zeichnen« können durch Syntaxdiagramme wie im zur Verfügung stehenden *Material* beschrieben werden.

Dabei gilt folgende Befehlszuordnung:

| Schlüsselwort | Beschreibung |
|---|---|
| PA | Stiftbewegung zum angegebenen Punkt |
| PR | Stiftbewegung ausgehend vom aktuellen Punkt entsprechend der Richtungsangabe |
| AA | Stiftkreisbewegung beginnend am aktuellen Punkt um die angegebenen Koordinaten und den angegebenen Winkel |
| AR | Stiftkreisbewegung beginnend vom aktuellen Punkt entsprechend der Richtungsangabe und dem gegebenen Winkel |
| CI | Kreisbewegung um den aktuellen Punkt mit dem angegebenen ganzzahligen Radius |

7. Skizzieren Sie das Ergebnis der folgenden Befehlsfolge, wenn der Stift im Koordinatenurspung O aufgesetzt ist.
   PA100,0;PR0,100;CI50;

8. Entscheiden Sie mithilfe der Syntaxdiagramme, ob folgende Befehlsfolge korrekt ist. Begründen Sie.
   PA2,2.3;AA3,2,90.3;PR3,4;

9. Ein Kreis mit dem Radius r wird durch den Befehl CI<r> an der aktuellen Position gezeichnet. Erstellen Sie ein Syntaxdiagramm für diese Regel.

10. Bestimmen Sie aus den Syntaxdiagrammen den höchsten Typ der Grammatik in der Chomsky-Hierarchie. Begründen Sie.

11. Nennen Sie eine Automatenklasse, die diese Grammatik auf Korrektheit prüfen kann.

*Material*

Siehe Abbildung 4.24 (diese und nächste Seite).

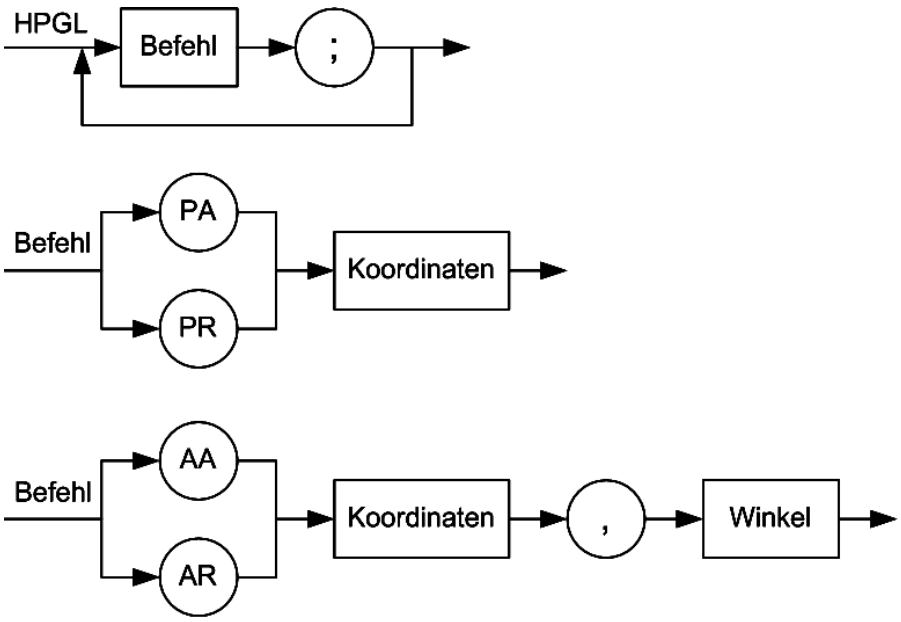



**Abbildung 4.24.**

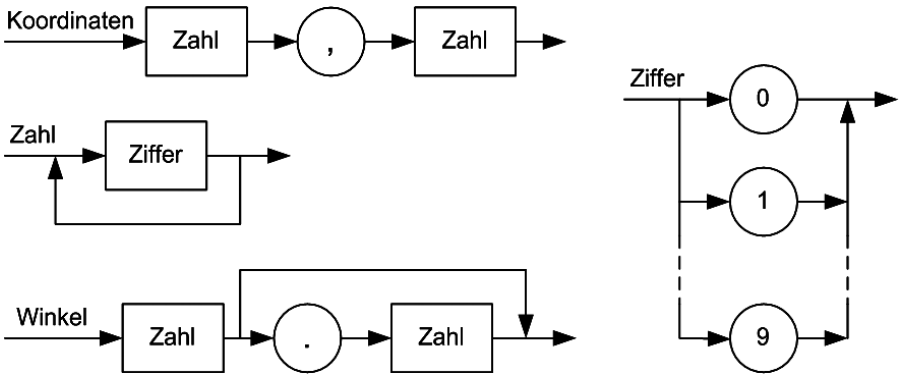



**Abbildung 4.24 (Fortsetzung).**

## Lösungshinweise

***Lösungshinweise zu den Aufgaben***

1 – mögliche Lösung

Siehe folgende Abbildung 4.25.

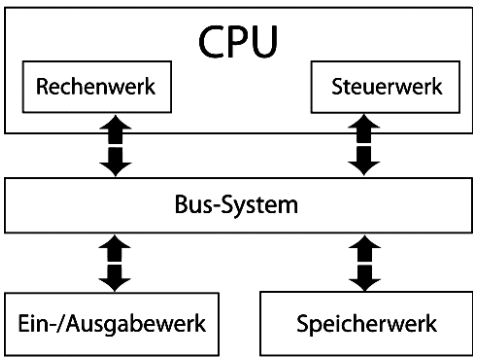



**Abbildung 4.25.**

2 – mögliche Lösung

Aufbau identisch: Speicherzellen gleicher Größe fortlaufend angeordnet; bei VON-NEUMANN: durch Adresse nummeriert.

Funktionsweise Kellerspeicher: Zugriff auf das oberste Element mit POP, Auflegen eines Elements mit PUSH(Element).

Funktionsweise VON-NEUMANN: Zugriff über Adresse auf jedes Element jederzeit möglich (wahlfreier Zugriff).

3 – mögliche Lösung

FETCH-Phase/Befehlshole-Phase:

Aus dem Speicher wird der nächste abzuarbeitende Befehl in das Befehlsregister geladen. Die zu benutzende Speicheradresse gibt der Befehlszähler an.

DECODE-Phase/Dekodier-Phase:

Der Befehlszähler wird um Eins erhöht und der Befehl im Steuerwerk übersetzt/dekodiert.

EXECUTE-Phase/Ausführungs-Phase:

Gegebenenfalls werden aus dem Speicher Operanden geholt (FETCH OPERANDS). Der Befehl wird im Rechenwerk ausgeführt. Im Falle eines Sprungbefehls wird auch der Befehlszähler verändert.

4 – mögliche Lösung

Subnetzmaske in beiden Netzen 255.255.255.0, Gruppe 1 ist demnach im Netz 192.168.1.0 und der Plotter im Netz 192.168.3.0 .

5 – mögliche Lösung

Der Switch arbeitet auf OSI Layer 2 (Netzverbund im gleichen Segment); damit alle Zugriff auf den Plotter haben sollen, müssen die Geräte dann im gleichen Netz sein → Widerspruch zur Sicherheitsforderung.

6 – mögliche Lösung

Einsatz eines Routers mit festgelegten Routen zum Plotter und Routingsperre zwischen den Netzen der beiden Gruppen oder Nutzung beider Plotteranschlüsse.

Netzwerkanschluss – für Forschergruppe 1 mit einer Adresse aus dem 192.168.1.0/24-Netz; USB-Anschluss an Printserver anschließen und diesen als Netzgerät im Netz 192.168.2.0/24 konfigurieren.

7 – mögliche Lösung

Siehe folgende Abbildung 4.26.

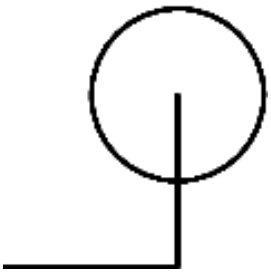


**Abbildung 4.26.**

8 – mögliche Lösung

Der erste Befehl ist fehlerhaft, da dieser eine Dezimalzahl als zweiten Parameter verwendet. Dies ist nach Syntaxdiagramm Koordinaten nicht zulässig. Die anderen Befehle sind korrekt.

9 – mögliche Lösung

Siehe folgende Abbildung 4.27.

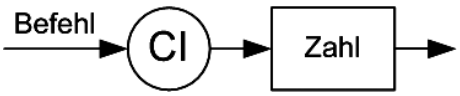



**Abbildung 4.27.**

10 – mögliche Lösung

Typ 2 – kontextfreie Grammatik, da:
Typ 0 – Grammatik.
Typ 1 – Die rechte Seite ist stets länger als die linke Seite.
Typ 2 – Auf der linken Seite steht immer genau ein Nichtterminalsymbol (Bezeichner am jeweiligen Syntaxdiagramm).
Typ 3 – Liegt nicht vor. Auf der rechten Seite steht z. B. <HPGL> → <Befehl> ";" | <Befehl> ";" <HPGL>.

11 – mögliche Lösung

Kellerautomat oder Turingmaschine.

Quelle:
Ministerium für Bildung, Wissenschaft und Kultur Mecklenburg-Vorpommern (Hrsg.): Zentralabitur 2012 – Informatik. Schwerin: 2012, S. 11–12.

***Zuordnung zu den Prozess-, Inhalts- und Anforderungsbereichen***

| Aufgabe | Prozessbereiche | | | | | Inhaltsbereiche | | | | | Bewertungseinheiten in Anforderungsbereichen | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MI | BB | SV | KK | DI | ID | AL | SA | IS | IMG | I | II | III |
| 1 | | | | | X | | | | X | | 2 | | |
| 2 | | | X | | | | | | X | | | 3 | |
| 3 | | | | | X | | | | X | | | 4 | |
| 4 | | X | | | | | | | X | | 1 | | |
| 5 | | X | | | | | | | X | | | | 1 |
| 6 | | | | | X | | | | X | | | 1 | 1 |
| 7 | | | X | | X | | | X | | | 1 | | |
| 8 | | X | | | X | | | X | | | | 2 | |
| 9 | | | | | X | | | X | | | | 1 | |
| 10 | | X | | | X | | | X | | | | 1 | 1 |
| 11 | | | X | | | | | X | | | 1 | | |
| Summe 20 | | | | | | | | | | | 5 | 12 | 3 |

# Mündliche Leistungsaufgaben

Mündliche Prüfungsaufgaben müssen der besonderen Situation gerecht werden, dass die Leistung zunächst im Vortrag, dann im Prüfungsgespräch erbracht wird. Im Unterschied zu schriftlichen Leistungsaufgaben liegt der Schwerpunkt nicht bei der Anwendung von Standardverfahren. Der Schwerpunkt muss vielmehr bei der Darstellung und Reflexion liegen. In der mündlichen Prüfung soll der Prüfling zeigen, dass er Zusammenhänge verstanden hat, sie wiedergeben, erläutern, analysieren, interpretieren, begründen und beurteilen kann. Mündliche Leistungsaufgaben, die nach dem Muster schriftlicher Leistungsaufgaben konzipiert werden, sind dafür ungeeignet. Die Strukturierung mündlicher Leistungsaufgaben in Teilaufgaben und die Beachtung der Anforderungsbereiche erfolgt analog zu schriftlichen Leistungsaufgaben.

Im ersten Teil der mündlichen Prüfung trägt der Prüfling die Lösung der schriftlich gestellten Aufgabe vor. Das folgende Prüfungsgespräch geht von diesem Vortrag aus, spricht aber dann vor allem größere fachliche Zusammenhänge an. Bei den nachfolgenden mündlichen Leistungsaufgaben ist jeweils nur die schriftlich gestellte Aufgabe samt Erwartungshorizont angegeben.

*Au-pair-Vermittlung*

**Au-pair-Vermittlung**

***Anmerkungen***

▷ grundlegendes Anforderungsniveau
▷ Vorbereitungszeit: 30 min

***Aufgabe***

Als **Au-pair** (frz. *auf Gegenleistung*) bezeichnet man junge Menschen, die für Verpflegung, Unterkunft und Taschengeld in einer Familie im In- oder Ausland tätig sind, um im Gegenzug Sprache und Kultur des Gastlandes bzw. der Gastregion kennen zu lernen. Das Au-pair lebt dabei im Haushalt der Gastfamilie, hilft bei der Kinderbetreuung und übernimmt leichte Hausarbeiten. Eine internationale Au-pair-Agentur will die Vermittlung optimieren und eine Datenbank einsetzen.

1. Beschreiben Sie das Relationenmodell für die Datenbank der Au-pair-Vermittlung:
   AuPair(ANr, Nachname, Vorname, Geburtsdatum)
   Familie(FNr, Name, Einkommen, Sprache, Kinderzahl, ↑SName)
   Stadt(SName, Land, Einwohner)
   lebt_bei(↑ANr, ↑FNr, von, bis)
2. Ermitteln Sie aus dem Relationenmodell das ER-Diagramm und erläutern Sie es.
3. Mit einem JAVA-Programm soll die Datenbank verwaltet werden.
   Stellen Sie ohne Angabe von get/set-Methoden die Relation *Familie* als UML-Klassendiagramm dar.
4. Die Relationen *AuPair* und *Familie* können mit diesen beiden CREATE-Befehlen in einer SQL-Datenbank angelegt werden.

```
CREATE TABLE aupair (
  ANr INT,
  Nachname STRING,
  Vorname STRING,
  Geburtsdatum DATE,
  PRIMARY KEY (ANr)
)
```

```
CREATE TABLE familie (
  FNr INT,
  Name STRING,
  Einkommen DOUBLE,
  Sprache STRING,
  Kinderzahl INT,
  SName STRING,
  PRIMARY KEY (FNr)
)
```

   Entwickeln Sie für die Sprache der SQL-CREATE-Befehle eine Grammatik.

5. Bestimmen Sie den Typ der Grammatik gemäß der Chomsky-Hierarchie.

6. Weitere Fragestellungen für das Prüfungsgespräch:
   - Interpretieren Sie den Fremdschlüssel *SName* bei objektorientierter Modellierung der Datenbank.
   - Begründen Sie, ob die Sprache der SQL-CREATE-Befehle von einem endlichen Automaten akzeptiert werden kann.
   - Vergleichen Sie SQL-Anweisungen mit Methoden einer Klasse.

*Hinweise*

Die Aufgabe *Au-pair-Vermittlung* liegt auch als schriftliche Abituraufgabe vor (siehe Seite 24ff.). In der mündlichen Variante findet eine Vernetzung mehrerer Inhaltsbereiche statt.

***Erwartungshorizont***

## *Erwartungshorizont*

1 – mögliche Lösung

Beschreibung des Relationenmodells mit Fachbegriffen:
Relation, Attribut, Primärschlüssel, Fremdschlüssel.

2 – mögliche Lösung

Siehe folgende Abbildung 4.28.

Rekonstruktion und Erläuterung des ER-Modells:

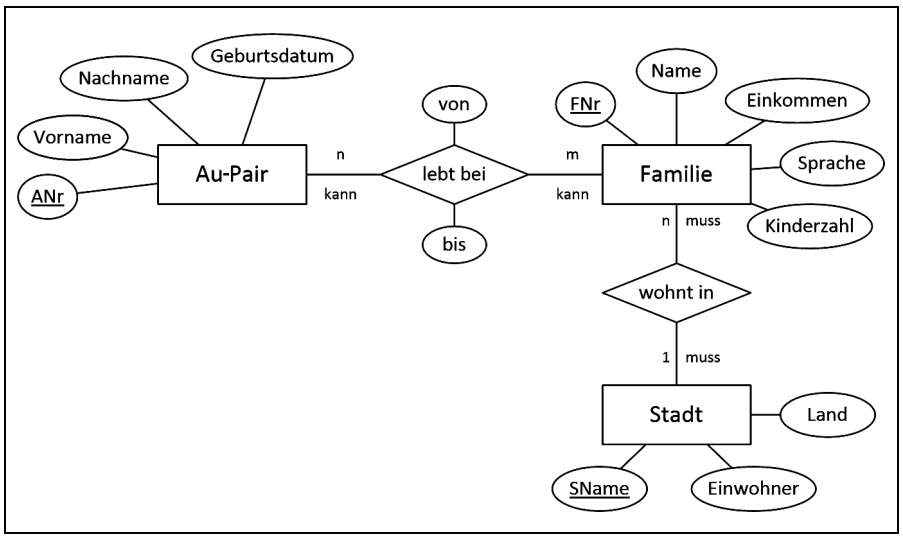



**Abbildung 4.28.**

3 – mögliche Lösung

Siehe folgende Abbildung 4.29.

Modellierung als Klassendiagramm:

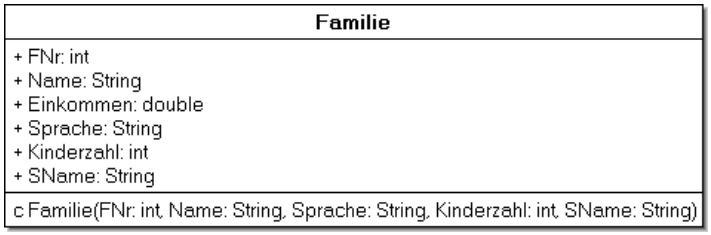



**Abbildung 4.29.**

SName kann als Assoziation mit der Klasse STADT modelliert werden.

4 – mögliche Lösung

Entwicklung einer Grammatik für eine Teilsprache von SQL.

S → CREATE TABLE Tabelle(Attribute, PRIMARY KEY(AttributOhneTyp))
Attribute → Attribut | Attribut, Attribute
Attribut → Bezeichner Datentyp
AttributOhneTyp → Bezeichner
Datentyp → INT | STRING | DATE | DOUBLE
Tabelle → Bezeichner
Bezeichner → ein oder mehrere Buchstaben

Angabe des Startsymbols, der Terminale und Nichtterminale.

5 – mögliche Lösung

Einordnung in Chomsky-Hierarchie.
Kontextfreie Grammatik, also kontextfreie Sprache.
Erläuterung der Chomsky-Hierarchie.

***Zuordnung zu den Prozess-, Inhalts- und Anforderungsbereichen***

| Aufgabe | Prozessbereiche | | | | | Inhaltsbereiche | | | | | Bewertungseinheiten in Anforderungsbereichen | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MI | BB | SV | KK | DI | ID | AL | SA | IS | IMG | I | II | III |
| 1 | | | X | X | | X | | | | | 4 | | |
| 2 | | | X | X | X | X | | | | | 3 | 3 | |
| 3 | | | X | X | X | | | | X | | | 2 | |
| 4 | X | | | X | | | | X | | | | 2 | 2 |
| 5 | | X | | X | | | | X | | | | 2 | |
| 6 | | X | X | X | X | X | | X | X | | | 3 | 3 |
| Summe 24 | | | | | | | | | | | 7 | 12 | 5 |

## *Tischtennisturnier*

**Tischtennisturnier**

***Anmerkungen***

▷ grundlegendes Anforderungsniveau
▷ Vorbereitungszeit: 30 min

***Aufgabe***

Zur Vereinfachung der Organisation der Tischtennisbezirksmeisterschaften der Herren soll eine relationale Datenbank erstellt werden. Die Modellierung ergibt das ER-Diagramm der Abbildung 4.30, nächste Seite.

1. Beschreiben Sie das ER-Diagramm.
2. Übertragen Sie das ER-Diagramm ins Relationenmodell und begründen Sie Ihre Vorgehensweise.
3. Stellen Sie ohne Angabe von get/set-Methoden die Beziehung *Teilnehmer spielt_für Verein* als UML-Klassendiagramm dar.
4. Erläutern Sie die SQL-Anweisung

```
SELECT Schiedsrichter, Name, Vorname
FROM  Teilnehmer, spielt, Spiel
WHERE Teilnehmer.Spielerpassnummer = spielt.Spielerpassnummer
    AND Spiel.Spielnummer = spielt.Spielnummer
    AND Teilnehmer.Name = 'Rosskopf'
```

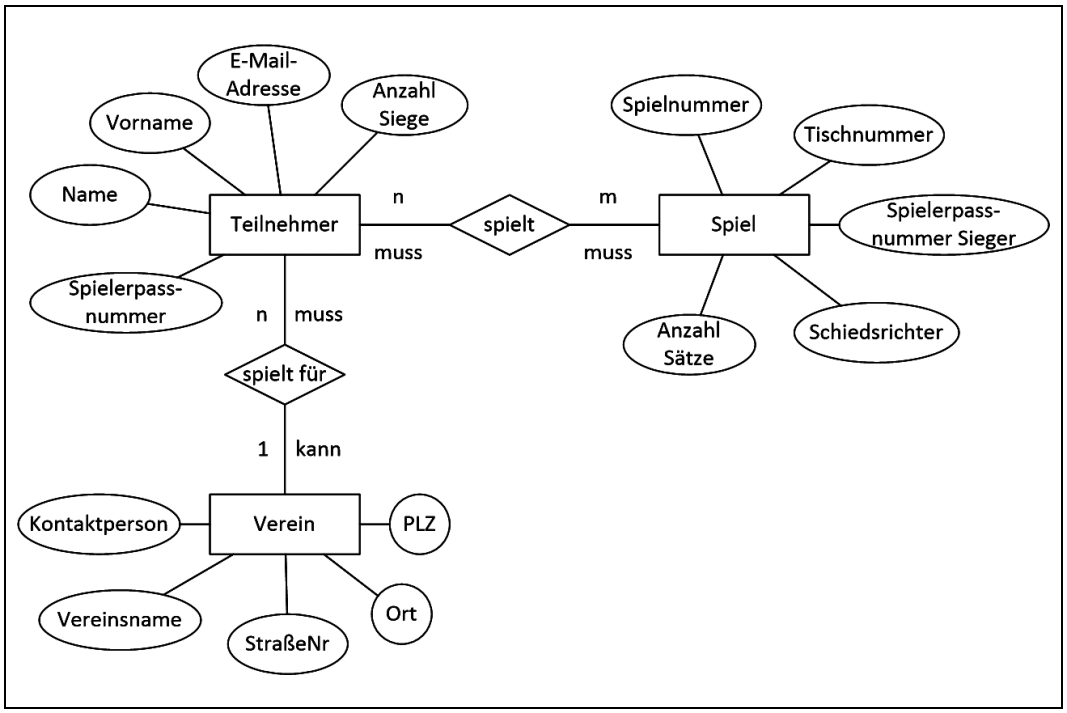



**Abbildung 4.30.**

5. Wenden Sie Methoden der theoretischen Informatik auf diese Sprache an:

SELECT Attribute
FROM Relationen
WHERE Relation.Attribut = Relation.Attribut AND Relation.Attribut = Relation.Attribut AND…

***Erwartungshorizont***

## *Erwartungshorizont*

1 – mögliche Lösung

Beschreibung des ER-Diagramms mit Fachbegriffen:
Entitätstyp, Attribute, Beziehungen, Kardinalität und Optionalität.

2 – mögliche Lösung

Erläuterung der Überführung ins Relationenmodell:
Verein(Vereinsname, PLZ, Ort, StraßeNr, Kontaktperson).
Teilnehmer(Spielerpassnummer, Vorname, Name, EMailAdresse, AnzahlSiege, ↑Vereinsname).
spielt(↑Spielerpassnummer, ↑Spielnummer).
Spiel(Spielnummer, Tischnummer, SpielerpassnummerSieger, Schiedsrichter, AnzahlSätze).

3 – mögliche Lösung

Klassenmodellierung einer ER-Beziehung (siehe folgende Abbildung 4.31):

Aggregation durch Feld Spieler

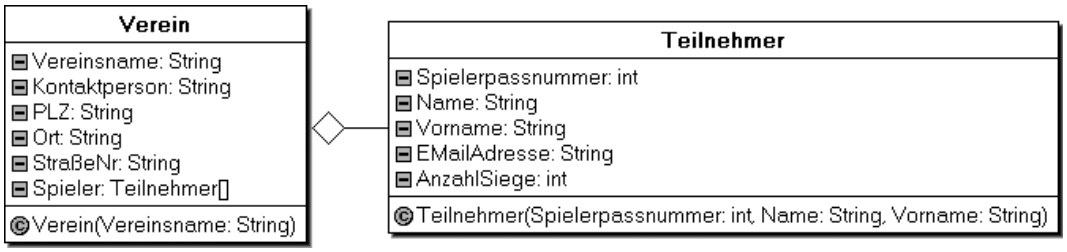



**Abbildung 4.31.**

4 – mögliche Lösung

Doppelter Join über Teilnehmer, spielt, Spiel, Selektion, Projektion.
Ausgabe Schiedsrichter, Name, Vorname für den Spieler *Rosskopf*.

5 – mögliche Lösung

Individuelle Lösungen möglich, z. B. Grammatik oder Syntaxdiagramme.

S → SELECT Attribute FROM Relationen WHERE Bedingungen
Attribute → Bezeichner | Bezeichner, Attribute
Relationen → Bezeichner | Bezeichner, Relationen
Bedingungen → Bedingung | Bedingung AND Bedingungen
Bedingung → Bezeichner.Bezeichner = Bezeichner.Bezeichner
Bezeichner → Buchstabe | Buchstabe Bezeichner

Einordnung in Chomsky-Hierarchie als kontextfreie Sprache, z. B. endlichen Automat angeben.

***Zuordnung zu den Prozess-, Inhalts- und Anforderungsbereichen***

| Aufgabe | Prozessbereiche | | | | | Inhaltsbereiche | | | | | Bewertungseinheiten in Anforderungsbereichen | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MI | BB | SV | KK | DI | ID | AL | SA | IS | IMG | I | II | III |
| 1 | | | X | X | | X | | | | | 4 | | |
| 2 | | X | | X | X | X | | | | | 2 | 2 | |
| 3 | X | | | X | | | | | X | | | 4 | |
| 4 | | | | X | X | X | | | | | | 2 | |
| 5 | X | | X | X | | | | X | | | | 3 | 3 |
| 6 | X | X | X | X | | X | | X | X | | | 3 | 3 |
| Summe 26 | | | | | | | | | | | 6 | 14 | 6 |

## *Sprachen und Automaten*

**Sprachen und Automaten**

***Anmerkungen***

▷ grundlegendes Anforderungsniveau
▷ Vorbereitungszeit: 30 min

***Aufgabe***

Computer müssen für die Abarbeitung von Befehlen in der Lage sein zu erkennen, ob diese Befehle korrekt geschrieben wurden oder nicht. Die mathematische Beschreibung einer solchen erkennenden Einrichtung ist das Modell des endlichen Automaten A = (X, Z, δ, $z_0$, $Z_E$), zu dem auch die Akzeptoren gehören.

1a Nennen Sie drei weitere Beispiele für die Verwendung von Akzeptoren.
1b Geben Sie die Bedeutung der Elemente X, Z, δ, $z_0$, und $Z_E$ in der Definition des Akzeptors an.
1c Beschreiben Sie den Aufbau eines Geräts, mit dem die Arbeitsweise eines Akzeptors veranschaulicht werden kann.

| δ | a | b | c |
|---|---|---|---|
| **$z_0$** | $z_1$ | $z_0$ | $z_0$ |
| **$z_1$** | $z_3$ | $z_2$ | $z_3$ |
| **$z_2$** | $z_1$ | $z_3$ | $z_0$ |
| **$z_3$** | $z_3$ | $z_3$ | $z_3$ |

2. Gegeben sei folgender Akzeptor A = (X, Z, δ, $z_0$, $Z_E$) mit $Z_E$ = {$z_0$, $z_2$} und der tabellarischen Darstellung der Funktion δ (siehe Tabelle links).
   a) Geben Sie die Elemente der Mengen X und Z an.
   b) Zeichnen Sie einen Zustandsgraphen für A.
   c) Geben Sie an, ob die folgenden Wörter vom Akzeptor A erkannt werden, und begründen Sie Ihre Entscheidung: $w_1$ = abc, $w_2$ = bca, $w_3$ = cab.

d) Geben Sie alle vom Akzeptor A erkannten Wörter mit genau zwei Zeichen an.
e) Bestimmen Sie die vom Akzeptor A erkannte Sprache L(A).

3. Beurteilen Sie die Aussage:
»Zu jeder denkbaren formalen Sprache lässt sich ein Akzeptor konstruieren.«

***Erwartungshorizont***

## *Erwartungshorizont*

1a – mögliche Lösung

PIN-Prüfer, Prüfbitgenerator, Parser, …

1b – mögliche Lösung

X … Eingabealphabet (nichtleere, endliche Menge)
Z … Zustandsmenge (nichtleere, endliche Menge)
δ … Überführungsfunktion
$z_0$ … Startzustand
$Z_E$ … Menge der Endzustände (nichtleere, endliche Menge, Teilmenge von Z).

1c – mögliche Lösung

Es sind verschiedene Erwartungsbilder – je nach Abstraktionsebene – denkbar. Abstrakte Gerätebeschreibung mit Eingabeband, Lesekopf, Zentraleinheit und Signallampe. Alternative Beschreibungen auf konkreterer Ebene (realer Automat) sind ebenfalls möglich.

2a – mögliche Lösung

X = {a, b, c}, Z = {$z_0$, $z_1$, $z_2$, $z_3$}

2b – mögliche Lösung

Siehe folgende Abbildung 4.32.

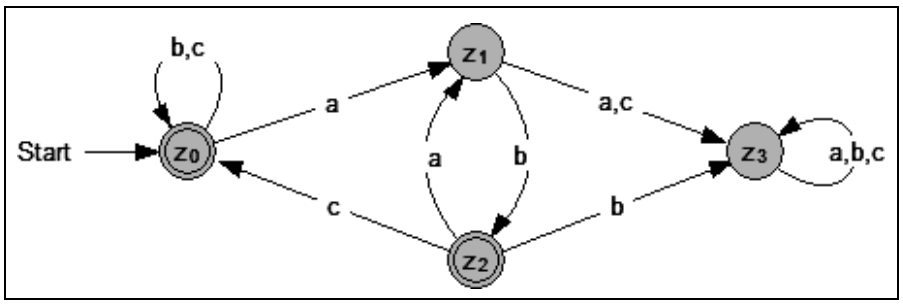



**Abbildung 4.32.**

2c – mögliche Lösung

$w_1 \in L(A)$, da Verarbeitung des Wortes im Endzustand $z_0$ endet.
$w_2 \notin L(A)$, da Verarbeitung des Wortes im Zustand $z_1$ endet und dieser kein Endzustand ist.
$w_3 \in L(A)$, da Verarbeitung des Wortes im Endzustand $z_2$ endet.

2d – mögliche Lösung

{ab, bb, bc, cb, cc}

2e – mögliche Lösung

L(A) ist die Sprache aller Wörter über X, die entweder aus keinem a bestehen (einschließlich dem leeren Wort) oder bei denen nach jedem a genau ein b kommt.

3 – mögliche Lösung

Diese Aussage ist falsch. Es gibt Sprachen, z. B. $L = \{a^n b^n \mid n \in N\}$, die nicht von einem Akzeptor erkannt werden können, da dieser nur endlich viele Zustände haben kann. Zur Erkennung der angegebenen Sprache wären aber unendlich viele Zustände erforderlich, um sich die Anzahl des Zeichen a zu merken.

***Zuordnung zu den Prozess-, Inhalts- und Anforderungsbereichen***

| Aufgabe | Prozessbereiche | | | | | Inhaltsbereiche | | | | | Bewertungseinheiten in Anforderungsbereichen | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MI | BB | SV | KK | DI | ID | AL | SA | IS | IMG | I | II | III |
| 1a | | | | X | | | | X | X | | 2 | | |
| 1b | | | | X | | | | X | | | 2 | | |
| 1c | | | | X | X | | | X | | | 3 | 1 | |
| 2a | | | | X | | | | X | | | | 2 | |
| 2b | | | | | X | | | X | | | | 3 | |
| 2c | | X | | X | | | | X | | | | 3 | |
| 2d | | | X | X | X | | | X | | | | 1 | 1 |
| 2e | | | X | X | | | | X | | | | | 2 |
| 3 | | X | | X | | | | X | | | | | 2 |
| Summe 22 | | | | | | | | | | | 7 | 10 | 5 |

## *UML und Qualität von Software*

**UML und Qualität von Software**

***Anmerkungen***

▷ erhöhtes Anforderungsniveau
▷ Vorbereitungszeit: 30 min

***Aufgabe***

*1. UML-Klassenmodellierung*

Das dargestellte UML-Klassendiagramm stellt einen Teil eines Computer-Online-Shops dar. Zentraler Aspekt des modellierten Ausschnitts ist der Warenkorb.

Siehe Abbildung 4.33, nächste Seite.

a) Betrachten Sie die Klasse WARENKORB.
Geben Sie drei Beispiele für Werte des Attributs Benutzername an. Erläutern Sie, warum die Methode Rausnehmen(einArtikel: Artikel; Anzahl: int) zwei Parameter benötigt. Geben Sie an, welche Attribute in WARENKORB von ihr beeinflusst werden.
b) Beschreiben Sie die Beziehung zwischen den Klassen WARENKORB und ARTIKEL.
c) Erläutern Sie den Begriff *Vererbung* anhand des dargestellten UML-Klassendiagramms.
d) Analysieren Sie das gegebene Modell im Hinblick auf Schwachstellen. Entwickeln Sie Alternativvorschläge.

*2. Qualität von Software*

a) Erläutern Sie die Qualitätskriterien *Benutzerfreundlichkeit* und *Robustheit* anhand des vorgegebenen Beispiels.
b) Begründen Sie, wie das gegebene Modell oder Ihr Alternativvorschlag die Erweiterbarkeit bzw. Wiederverwendbarkeit ermöglicht.

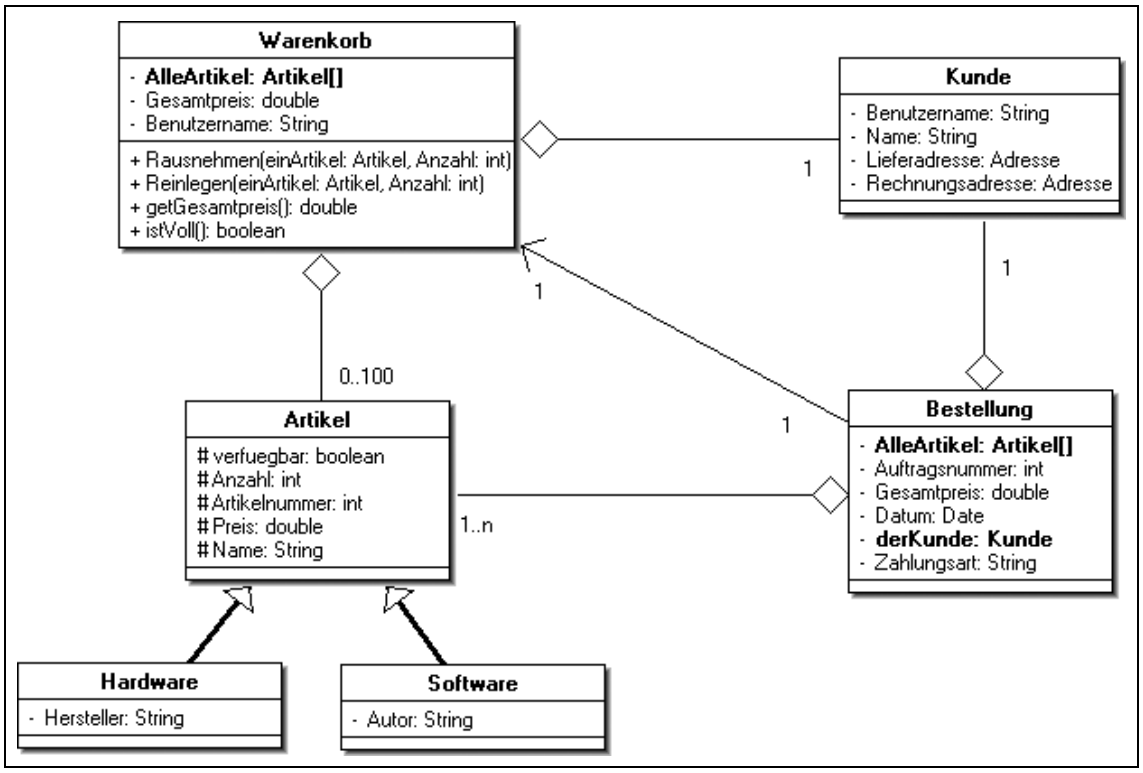



**Abbildung 4.33.**

## Erwartungshorizont

***Erwartungshorizont***

1a – mögliche Lösung

Drei mögliche Werte sind: gregor15, maxkauf, gertigros; weil der Datentyp String ist.

Rausnehmen ist eine Methode mit Parameterübergabe. Übergebene Parameter sind der zu entnehmende Artikel und die zu entnehmende Anzahl dieses Artikels. Es wird der Inhalt des virtuellen Warenkorbs geändert. Dabei muss kontrolliert werden, ob der Artikel mindestens in der übergebenen Anzahl im Warenkorb ist. AlleArtikel und Gesamtpreis werden verändert.

1b – mögliche Lösung

Ein Warenkorb kann 0 bis 100 Artikel enthalten, also auch leer sein. Gegenrichtung: Ein Artikel kann in einem Warenkorb enthalten sein, muss aber nicht.

1c – mögliche Lösung

Das UML-Klassendiagramm zeigt, dass die Klassen SOFTWARE und HARDWARE von der Klasse ARTIKEL abgeleitet sind und damit alle nicht privaten Attribute dieser Klasse erben, ebenso deren Methoden. Da die Attribute die Sichtbarkeit protected haben, werden sie vererbt. HARDWARE z. B. hat darüber hinaus das Attribut Hersteller.

1d – mögliche Lösung

Das Modell hat große Schwächen:

- ▷ durch Doppelung von Einträgen in verschiedenen Klassen, so wird z. B. Benutzername aus KUNDE auch in WARENKORB mitgeführt und Gesamtpreis aus WARENKORB auch in BESTELLUNG,
- ▷ sowie durch die Beschränkung auf 100 Artikel (Reihung).

Man könnte jeder Bestellung ein Objekt der Klasse WARENKORB als Attribut geben; in WARENKORB entfällt damit das Attribut Benutzername; in Bestel

lung entfallen die Artikel-Reihung und der Gesamtpreis, da schon in WARENKORB.

Man könnte wegen der 1:1-Beziehung auch die Klasse BESTELLUNG so erweitern, dass sie alle Attribute von WARENKORB vollständig enthält. Damit wäre ein WARENKORB kein eigenständiges Objekt mehr. Vor-/Nachteile beider Modellierungen erläutern und begründen.

Inkonsistenzen: Teilweise ist eine Implementierung von Beziehungen durch entsprechende Attribute bei den Klassen bereits angedeutet, teilweise nicht; das Diagramm vermischt somit Analyse- und Entwurfsebene.

2a – mögliche Lösung

▷ Beschränkung auf 100 Artikel ist nicht benutzerfreundlich.
▷ Das meiste kann wegen fehlender Benutzungsschnittstelle nicht entschieden werden.
▷ Durch die Doppelungen ist der Entwurf in keiner Weise robust.

2b – mögliche Lösung

▷ Vererbung erlaubt eine leichte Erweiterbarkeit auf weitere spezielle Artikel.
▷ Bestehende Klassen können in anderen Problemlösungskontexten wieder verwendet werden.

***Zuordnung zu den Prozess-, Inhalts- und Anforderungsbereichen***

| Aufgabe | Prozessbereiche | | | | | Inhaltsbereiche | | | | | Bewertungseinheiten in Anforderungsbereichen | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MI | BB | SV | KK | DI | ID | AL | SA | IS | IMG | I | II | III |
| 1a | X | X | X | X | | | | | X | | 1 | 3 | 1 |
| 1b | | | X | X | X | | | | X | | 1 | 1 | |
| 1c | | | X | X | X | | | | X | | 1 | 1 | 1 |
| 1d | X | X | X | X | X | X | X | | X | | | 4 | 3 |
| 2a | | X | | X | | | X | | X | X | 1 | 2 | |
| 2b | X | X | X | | | | | | X | | 1 | 1 | |
| Summe 22 | | | | | | | | | | | 5 | 12 | 5 |

# Lernaufgaben

Lernaufgaben stellen eine Lernumgebung bereit, in der durch passende Materialien und gestufte Aufgabenstellungen die Schülerinnen und Schüler ihren Lernprozess und Kompetenzaufbau selbstständig steuern. In den nachfolgenden Beispielen werden jeweils die Zielstellung sowie gegebenenfalls Voraussetzungen und Materialien angegeben.

## Bootsdatenbank

### *Zielstellung*

Anhand der Lernaufgabe erkennen die Schülerinnen und Schüler zunächst die Probleme inkonsistenter Daten, die durch die redundante Datenhaltung in Tabellenkalkulationen ermöglicht werden. Eine vergleichende Betrachtung mit einer Abfragetabelle einer Datenbank ermöglicht ihnen, die Unterschiede in der Funktionalität zwischen Tabellenkalkulation und Datenbank zu begreifen und die Beziehungsstruktur zwischen den normalisierten Tabellen einer relationalen Datenbank zu entdecken.

Nach der selbstständigen Bearbeitung der Lernaufgabe durch die Schülerinnen und Schüler sollten eine durch die Lehrkraft begleitete Reflexion über das erworbene Wissen und Können erfolgen und offene Fragen verbindlich geklärt werden.

### *Voraussetzungen und Material*

Die Schülerinnen und Schüler, die bereits über Erfahrungen in der tabellarischen Verwaltung von Daten verfügen sollten, erhalten zwei Dateien.

In der Tabellenkalkulation bootseigner.ods liegt eine Tabelle zu Bootseignern und ihren Booten in der ersten Normalform vor:

*Bootsdatenbank*

Foto: LOG-IN-Archiv

**Abbildung 4.34: Eine Bootsdatenbank ist zur Verwaltung nötig.**

| Vorname | Name | Geburtsdatum | Bootsname | Verein | Sitz des Vereins | Kennzeichen | Person_ID |
|---|---|---|---|---|---|---|---|
| Martha | Müller | 19.08.78 | Flipper | Nordwind | Stralsund | HST-AK 588 | 2 |
| Martha | Müller | 19.08.78 | Moby Dick | Nordwind | Stralsund | HST-P 89 | 2 |
| Erika | Meier | 23.05.58 | Alte Liebe | Seenplatte | Waren | MÜR-N 54 | 3 |
| Martha | Müller | 19.08.78 | Titanic II | Blaues Band | Parchim | RZ-XY 4 | 4 |
| Martha | Meier | 23.05.58 | Alte Liebe | Nordwind | Stralsund | HST-AB 13 | 1 |
| Martha | Meier | 23.05.58 | Boddenschwalbe | Achterwasser | Ribnitz | HST-G 4 | 1 |
| Erika | Möller | 14.04.67 | Kon Tiki | Sundflitzer | Stralsund | HST-T 234 | 5 |
| Erika | Möller | 14.04.67 | Moby Dick | Nordwind | Stralsund | HST-P 89 | 5 |

In der Datenbank bootseigner.odb existiert eine Abfrage, deren Ergebnis äußerlich fast mit der Tabelle in der ods-Datei übereinstimmt. Die eigentliche Datenquelle sind drei Tabellen, die in folgender Weise miteinander verknüpft sind (siehe folgende Abbildung 4.35):

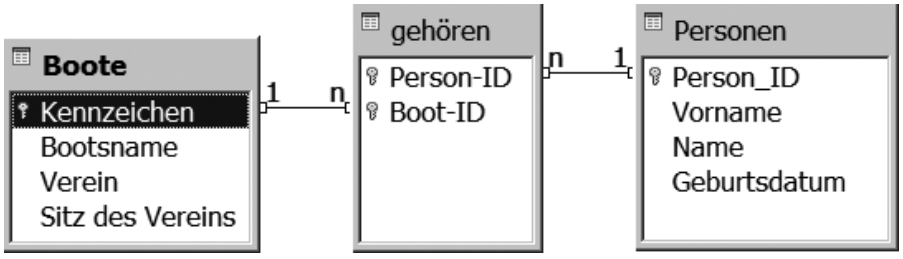



**Abbildung 4.35.**

Beide Beziehungen sind mit dem Attribut der *Referentiellen Integrität mit kaskadierter Änderungsweitergabe* konfiguriert worden. Das bedeutet, dass sich die

Änderung einzelner Daten in Tabellen oder Abfragen auf alle weiteren betroffenen Tabellen auswirkt. Die Aktualisierung muss manuell ausgelöst werden.

Herunterladen der Dateien von: http://informatikstandards.de/download

***Aufgabe***

Ein Dachverband verschiedener Segelvereine möchte Informationen zu Bootsbesitzern und ihren Booten verwalten. Zwei Vereinsmitglieder machen sich über Möglichkeiten der technischen Umsetzung Gedanken. Sie einigen sich darauf, ihren Lösungsvorschlag mit einer Handvoll Datensätzen umzusetzen und daran die Eignung der Lösung zu testen.

Im ersten Entwurf wurden die Datensätze in einer Tabellenkalkulationsdatei gespeichert.

1. Öffnen Sie die Tabelle bootseigner.ods und ändern Sie die Daten anhand folgender Vorgaben:

1.1 Die Eignerin des Bootes »Moby Dick« heiratet und trägt jetzt den Namen »Meier«.

1.2 Das Boot »Alte Liebe« von Erika Meier wird umbenannt und heißt jetzt »Flipper«.

1.3 Der Verein »Nordwind« wechselt seinen Sitz von Stralsund nach Greifswald.

1.4 Die Eignerin des Bootes »Moby Dick« lässt sich wieder scheiden und nimmt wieder Ihren Geburtsnamen »Müller« an.

Begründen Sie anhand dieses Beispiels, dass mit Tabellenkalkulationen zwar Daten gespeichert werden können, aber nicht in jedem Fall effizient und fehlerfrei zu verwalten sind.

Nachdem sich die Tabellenkalkulation für diese Aufgabe als nicht geeignet erwiesen hat, wird nun die zweite Variante unter die Lupe genommen. Das Vereinsmitglied hat sich für die Verwendung eines Datenbanksystems entschieden.

Siehe folgende Abbildung 4.36.

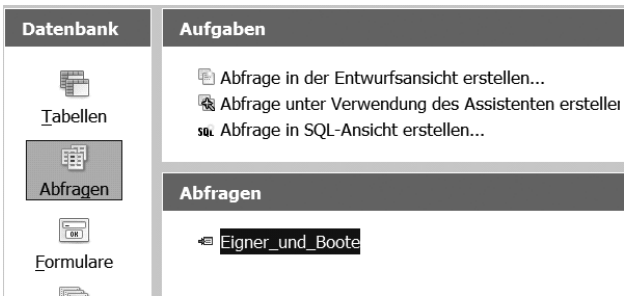



**Abbildung 4.36.**

2. Öffnen Sie die Datei bootseigner.odb und lassen Sie sich die Abfrage Eigner_und_Boote anzeigen. Versuchen Sie, in der Abfrage die gleichen Aufträge wie in Aufgabe 1 zu lösen. Bitte aktualisieren Sie die Datenbank *nach jeder Eingabe* per Klick auf die Schaltfläche:

Entscheiden Sie, welche der Probleme aus Aufgabe 1. durch die Datenbank schon zufriedenstellend gelöst werden und bei welchen die gegebene Datenbank noch verbesserungsbedürftig ist.

Offenbar steckt hinter der angezeigten Abfrage-Tabelle der Datenbank ein Konzept, das die Daten zueinander in Beziehung setzt.

Erkunden Sie nun das Prinzip, nach dem relationale Datenbanken funktionieren.


**Abbildung 4.37.**

3. Sie benutzen bitte weiterhin die Datenbank bootseigner.odb (siehe Abbildung 4.37):

3.1 Wechseln Sie in die Tabellenansicht und lassen Sie sich den Inhalt der Tabellen anzeigen.

3.2 Ändern Sie einzelne Eintragungen in den Tabellen und beobachten Sie die Auswirkungen auf das Abfrageergebnis.

3.3 Max Mayer, geboren am 28.11.1995, ist Mitglied im Warnemünder Verein »Steife Brise« und stolzer Besitzer des Bootes »Kap Hoorn« mit dem amtlichen Kennzeichen HST-K 558. Tragen Sie die Daten in die Tabellen der Datenbank ein. Beobachten Sie die Auswirkungen auf das Abfrageergebnis.
3.4 Die Beziehungen zwischen den Tabellen lassen sich über Extras → Beziehungen auch grafisch verdeutlichen. Beschreiben Sie anhand der Grafik das Zustandekommen des Ergebnisses der Abfragetabelle. Äußern Sie eine Vermutung, warum einige Datenfelder mit Schlüsselsymbolen markiert sind.

Vielleicht haben Sie es schon in Aufgabe 2 bemerkt: Die vorgelegte Lösung ist noch nicht perfekt. Unter anderem können Sie derzeit einem Verein zwei Sitze zuordnen. Das soll nicht erlaubt sein.

4. Erweitern Sie nun den Entwurf der Datenbank bootseigner.odb (siehe Abbildung 4.38):
4.1 Die Struktur der Tabellen können Sie per rechtem Mausklick und Bearbeiten verändern.
Lösen Sie das Problem der widersprüchlichen Einträge in Vereinsname und seinem Sitz durch das Hinzufügen einer weiteren Tabelle und passen Sie die Beziehungen zwischen den Tabellen an.
*Hinweis:* Mit einem Rechtsklick können Sie die Eigenschaften von Beziehungen bearbeiten. Wählen Sie bitte »Kask. Update«, um alle voneinander abhängigen Daten in der Abfrage aktualisieren zu können.
4.2 Entwickeln Sie eigene Ideen für eine sinnvolle Erweiterung der Datenbank.

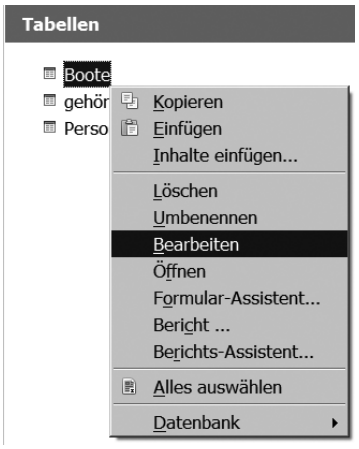



**Abbildung 4.38.**

## Lösungshinweise

***Lösungshinweise zu den Aufgaben***

1. Durch die Gleichheit von Attributwerten bei unterschiedlichen Personen erfordert die konsistente Änderung der Daten äußerste Konzentration und stellt eine potenzielle Fehlerquelle dar – beispielsweise bei der Änderung von Bootsnamen, vor allem, wenn das Boot mehrere gemeinschaftliche Eigner hat. Automatisierte Lösungen wie *suchen und ersetzen* lassen sich offensichtlich nicht zielführend anwenden.

2. Die Änderung von Familien- und Bootsnamen erfolgt nun problemlos. Ein ungelöstes Problem ist die konsistente Änderung des Vereinssitzes aufgrund der Redundanzen in der Tabelle (siehe Teilaufgabe 1.3).

3. Durch die Tabellenansicht wird offensichtlich, dass sich die Daten der Abfragetabelle aus drei einzelnen Tabellen speisen. Durch die Änderung einzelner Daten sind die Abhängigkeiten zwischen den Tabellen in der Abfrageansicht beobachtbar. Der zusätzliche Datensatz kann nur eingetragen werden, wenn das Prinzip von Primär- und Fremdschlüsseleinträgen verstanden wurde. Die grafische Veranschaulichung der Beziehungen zwischen den Tabellen erleichtert die Beschreibung der Verknüpfung zwischen den Tabellen über die Schlüssel.

4. Zwischen den Attributen Verein und Sitz besteht eine Abhängigkeit. Zur Vermeidung von Redundanzen kann das Relationenmodell in der Art – wie in der folgenden Abbildung 4.39 dargestellt – erweitert werden.

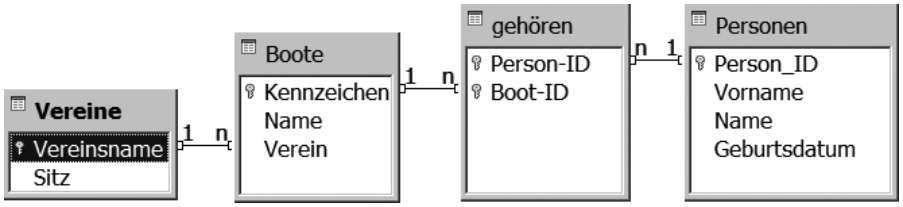



**Abbildung 4.39.**

Damit wird intuitiv eine Normalisierung vorgenommen. Beim Anlegen der Tabelle in der Entwurfsansicht ist die Bedeutung der Datentypen erkennbar.

***Zuordnung zu den Prozess-, Inhalts- und Anforderungsbereichen***

| Aufgabe | Prozessbereiche | | | | | Inhaltsbereiche | | | | | Anforderungsbereiche | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MI | BB | SV | KK | DI | ID | AL | SA | IS | IMG | I | II | III |
| 1 | | X | | | X | X | | | X | | X | X | |
| 2 | | X | X | | X | X | | | X | | | X | |
| 3 | X | | X | | | X | | | X | | | X | X |
| 4 | X | | X | | | X | | | X | | | | X |

## *HTML-Validator*

**HTML-Validator**

***Zielstellung***

In dieser Lernaufgabe erarbeiten sich die Schülerinnen und Schüler die Datenstruktur Keller im Anwendungskontext eines einfachen HTML-Validators. Vorkenntnisse zu HTML sind sinnvoll, aber nicht unbedingt erforderlich. Das Kellerprinzip wird erläutert und am Beispiel der Prüfung der gegebenen Tag-Reihenfolge demonstriert und angewendet. Aufbauend darauf soll ein Prüfalgorithmus entwickelt werden. Die einfach verkettete lineare Liste muss bekannt sein, denn der Keller wird als lineare Liste modelliert und implementiert, wobei die gegebenen Informationen den Lernprozess initiieren und begleiten. Im Sinne der Individualisierung des Lernens werden verschiedene Angebote mit zunehmender Komplexität zur Realisierung eines HTML-Validators gemacht. Abschließend findet eine Reflexion des Kontextes und des Lernprozesses statt. Diese Lernaufgabe ist für das erhöhte Anforderungsniveau konzipiert.

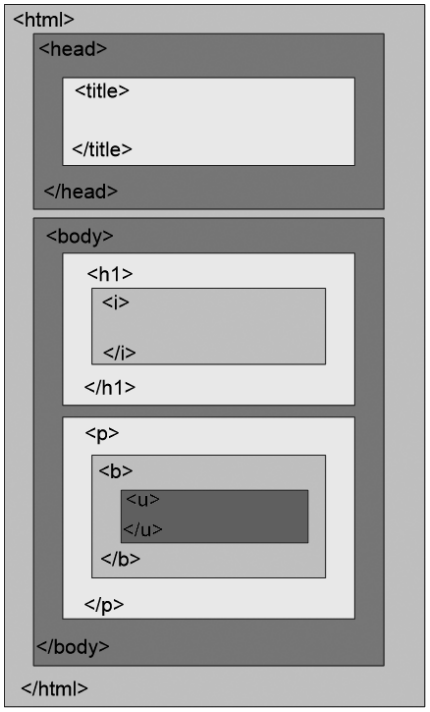



**Abbildung 4.40.**

***Aufgabe***

Mit einem HTML-Validator kann man prüfen, ob ein HTML-Dokument korrekt geschrieben ist. Ein ganz wesentlicher Teil dieser Prüfung befasst sich mit der richtigen Reihenfolge der öffnenden und schließenden HTML-Tags. HTML-Elemente können zwar ineinander geschachtelt werden, aber man kann ein äußeres HTML-Element nicht vor dem inneren HTML-Element schließen. Schachteln ist also wörtlich zu nehmen. Wie bei Schachteln, die ineinander gesetzt werden, müssen die HTML-Elemente durch die zugehörigen Tags geöffnet und geschlossen werden. In der Abbildung 4.40 haben wir eine korrekte Tag-Reihenfolge:

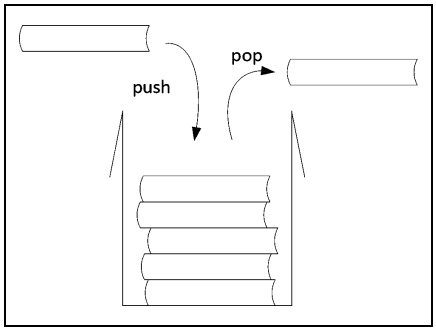
```
<html> <head> <title> </title> </head> <body> <h1> <i> </i> </h1>
<p> <b> <u> </u> </b> </p> </body> </html>
```

1. Geben Sie mit denselben Tags zwei falsche Tag-Reihenfolgen an.
2. Geben Sie mit denselben Tags eine andere richtige Reihenfolge an.

*Datenstruktur Keller*

Ein *Keller* – auch *Stapel* oder *Stack* genannt – ist eine Datenstruktur, bei der die Daten eine Folge bilden, die nur durch die Ein- und Ausfügeoperationen bestimmt wird. Das Prinzip eines Kellers besteht darin, dass stets das zuletzt eingefügte Element eines Kellers als erstes wieder entfernt werden muss. Die Kellerverwaltung arbeitet nach dem *LIFO-Prinzip*: Last In First Out. Was zuletzt in den Keller kam, kommt zuerst aus dem Keller auch wieder heraus. Man kann sich einen Keller als eine Bücherkiste vorstellen, in die man einzeln Bücher legen und wieder herausnehmen kann.

**Abbildung 4.41.**




Üblicherweise stehen folgende Kelleroperationen zur Verfügung (siehe Abbildung 4.41):

*push:* Legt ein Element auf dem Keller ab.
*pop:* Holt ein Element aus dem Keller.
*top:* Schaut nach, welches Element im Keller oben liegt.
*empty:* Prüft, ob ein Keller leer ist.

Mit einem Keller kann die Korrektheit einer Tag-Reihenfolge geprüft werden. Jeder öffnende Tag wird auf den Keller gelegt. Bei einem schließenden Tag wird ein Element aus dem Keller geholt und mit dem eingelesenen Tag verglichen. Passen beide nicht zueinander, ist die Tag-Reihenfolge inkorrekt. Nach dem letzten verarbeiteten Tag muss der Keller leer sein.

Nachfolgend ist für die obige Tag-Reihenfolge der Zustand des Kellers nach jedem verarbeiteten Tag dargestellt. Am Anfang ist der Keller leer.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| | | | | | | | | <i> | |
| | | | <title> | | | | <h1> | <h1> | <h1> |
| | | <head> | <head> | <head> | | <body> | <body> | <body> | <body> |
| | <html> | <html> | <html> | <html> | <html> | <html> | <html> | <html> | <html> |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | <u> | | | | | |
| | | <b> | <b> | <b> | | | | |
| | <p> | <p> | <p> | <p> | <p> | | | |
| <body> | <body> | <body> | <body> | <body> | <body> | <body> | | |
| <html> | <html> | <html> | <html> | <html> | <html> | <html> | <html> | |

3. Stellen Sie die einzelnen Kellerzustände für eine Ihrer beiden fehlerhaften Tag-Reihenfolgen auf Papier dar und erläutern Sie Ihrem Partner, wie der jeweilige Fehler erkannt wird.

4. Entwickeln Sie einen Algorithmus, der mithilfe eines Kellers eine gegebene Tag-Reihenfolge auf Korrektheit prüfen kann.

*Modellierung und Implementierung einer Kellerklasse*

Wir modellieren die Datenstruktur Keller als lineare Liste gemäß dem Klassendiagramm in folgender Abbildung 4.42:

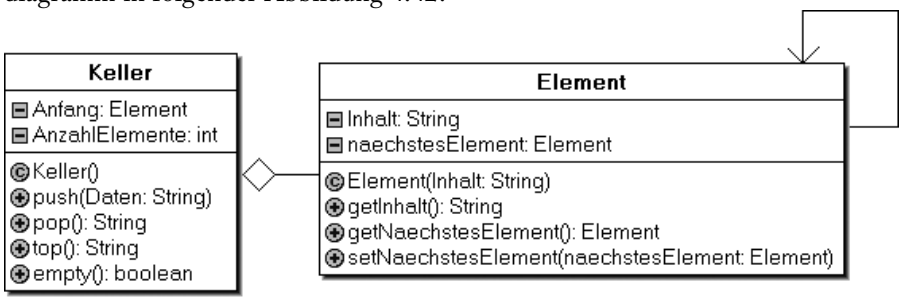


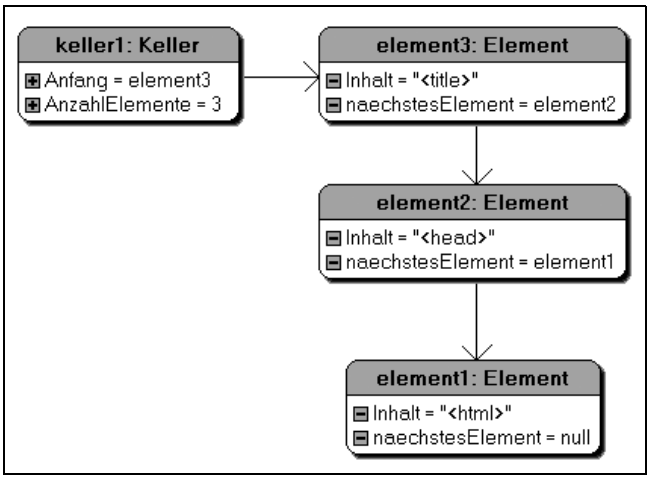

**Abbildung 4.42.**

Die push-Methode kann wie folgt implementiert werden:

```
public void push(String Daten) {
  Element einElement = new Element(Daten);
  einElement.setNaechstesElement(Anfang);
  Anfang = einElement;
  AnzahlElemente++;
}
```

Damit lässt sich beispielsweise das folgende Objektdiagramm eines Kellers erzeugen (siehe nebenstehende Abbildung 4.43).

**Abbildung 4.43.**




5. Implementieren Sie das Klassendiagramm in der Entwicklungsumgebung.

6. Analysieren Sie die Funktionsweise der push-Methode.

7. Erzeugen Sie den in der Abbildung dargestellten Keller.

8. Implementieren Sie die weiteren Methoden der Klasse KELLER.

*HTML-Validator*

Die Implementierung des HTML-Validators kann je nach persönlichen Interessen und Kompetenzen in unterschiedlichen Ausbaustufen erfolgen:

a) Man gibt die HTML-Tags der Reihe nach manuell über die Konsole oder ein TextField schrittweise ein und lässt sie vom Validator prüfen. Dazu braucht man String-Funktionen wie z. B. startsWith, um beispielsweise mit einString.startsWith("</") prüfen zu können, ob der Tag öffnend oder schließend ist, oder auch substring(int beginIndex, int endIndex), um den Namen des Tags zu extrahieren.
b) Man gibt ein HTML-Dokument in eine TextArea ein, holt sich mit getText() den Text in eine String-Variable, sucht mit einString.indexOf('<') und indexOf('>') das erste HTML-Tag im Text, entfernt dieses Element aus dem Text und lässt es vom Validator verarbeiten. In einer TextArea kann man leichter Änderungen vornehmen als in einer List-Komponente.
c) Man liest mit

```
import java.net.URL;
import java.io.InputStreamReader;
import java.io.BufferedReader;
private void getHTMLDokumentFromURL(String HTTPAdresse) {
  try {
    URL url = new URL(HTTPAdresse);  // z. B. "http://www.mathe.de/"
    BufferedReader in = new BufferedReader(new
                        InputStreamReader(url.openStream()));
    String inputLine = in.readLine();
    while (inputLine != null) {
      taHTML.append(inputLine + "\n");
      inputLine = in.readLine();
    }
    in.close();
  } catch (Exception e) {
      e.printStackTrace();
  }
}
```

eine Webseite in die TextArea taHTML ein und verarbeitet diese wie in b). Das ist dann schon deutlich schwieriger, weil die HTML-Elemente auch Attribute haben, die man entfernen muss, und es HTML-Elemente gibt, die kein schließendes Tag haben, wie z. B. <br> oder <img>.

9. Implementieren Sie einen HTML-Validator.

10. Bei validator.w3.org kann man seine Webseiten validieren lassen. Recherchieren Sie Gründe zum Validieren von Webseiten.

11. Reflektieren Sie Ihren Lernfortschritt.

## *Lösungshinweise*

***Lösungshinweise zu den Aufgaben***

1. <html> <head> <title> </title> </head> <body> <h1>  </h1>    <u> </u>   </body> </html>
<html> <head> <title> </title> </head> <body> <h1>  <u>  </h1>   </u>   </body> </html>

2. <html> <head> <title> </title> </head> <body> <h1>  <u>   </u>  </h1>   </body> </html>

3. Beim schließenden Tag </h1> wird ein Tag vom Keller geholt, also der Tag . Da </h1> nicht zu  passt, ist die Tag-Reihenfolge nicht korrekt.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | |  |
| | | | <title> | | | | <h1> | <h1> |
| | | <head> | <head> | <head> | | <body> | <body> | <body> |
| | <html> | <html> | <html> | <html> | <html> | <html> | <html> | <html> |

4. Siehe folgende Abbildung 4.44.

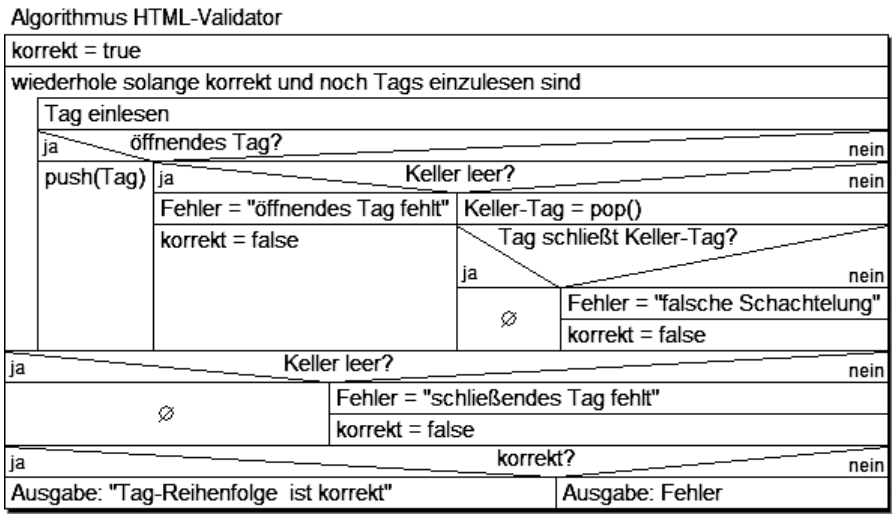



**Abbildung 4.44.**

5. bis 8.

```
public class Element {

  private String Inhalt;
  private Element naechstesElement;

  public Element(String Inhalt) {
    this.Inhalt = Inhalt;
    this.naechstesElement = null;
  }

  public String getInhalt() {
  return Inhalt;
  }

  public Element getNaechstesElement() {
    return naechstesElement;
}

  public void setNaechstesElement(Element naechstesElement) {
    this.naechstesElement = naechstesElement;
  }
}

public class Keller {

  public Element Anfang;
  public int AnzahlElemente;

  public Keller() {
    this.Anfang = null;
    this.AnzahlElemente = 0;
  }

  public void push(String Daten) {
    Element einElement= new Element(Daten);
    einElement.setNaechstesElement(Anfang);
    Anfang = einElement;
    AnzahlElemente++;
  }

  public String pop() {
    if (empty()) {
      return "";
    } else {
      Element einElement = Anfang;
      Anfang = Anfang.getNaechstesElement();
      AnzahlElemente--;
      return einElement.getInhalt();
    }
  }
```

```
  public String top() {
    if (empty())
      return "";
    else
      return Anfang.getInhalt();
  }

  public boolean empty() {
    return (AnzahlElemente==0);
  }
}
```

6. Die Methode wird mit dem Parameter Daten vom Datentyp String aufgerufen. Für diese Daten wird ein neues Element erzeugt. Dessen Verweis auf das nächste Element wird auf den Anfang der bisherigen Liste gesetzt. Der Anfang wird auf das neue Element gesetzt und die Anzahl der Elemente um 1 erhöht. Der Keller wird also als einfach verkettete Liste implementiert, dessen Anfangselement das oberste Element des Kellers darstellt.

9. Lösung nach b) – siehe folgende Abbildung 4.45.

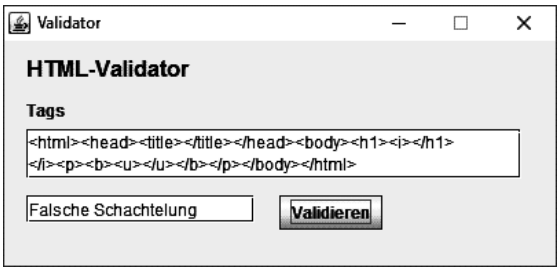



**Abbildung 4.45.**

```
public void bValidieren_ActionPerformed(ActionEvent evt) {
  Keller einKeller = new Keller();
  String HTML = taEingabe.getText();
  String Fehler = "";
  boolean korrekt = true;
  while (korrekt && (HTML.indexOf('<') >= 0)) {
    int BeginIndex = HTML.indexOf('<');
    int EndIndex = HTML.indexOf('>');
    String Tag = HTML.substring(BeginIndex, EndIndex+1);
    HTML = HTML.substring(EndIndex+1);
    if (!Tag.startsWith("</")) {
     einKeller.push(Tag);
    } else {
      if (einKeller.empty()) {
        Fehler = "öffnendes Tag fehlt";
        korrekt = false;
      } else {
        String KellerTag = einKeller.pop();
        KellerTag = "</" + KellerTag.substring(1);
        if (! Tag.equals(KellerTag)) {
          Fehler = "Falsche Schachtelung";
          korrekt = false;
        }
      }
    }
  }
  if (korrekt) {
    if (einKeller.empty()) {
      tfAusgabe.setText("Alles richtig");
    } else {
      tfAusgabe.setText("schließendes Tag fehlt.");
    }
  } else {
    tfAusgabe.setText(Fehler);
  }
}
```

10. Verbesserung der Qualität von Webseiten, leichte Wartung, Kennzeichen für Professionalität, korrekte Darstellung in allen Browsern, Voraussetzung für barrierefreie Webseiten.

***Zuordnung zu den Prozess-, Inhalts- und Anforderungsbereichen***

| Aufgabe | Prozessbereiche | | | | | Inhaltsbereiche | | | | | Anforderungsbereiche | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MI | BB | SV | KK | DI | ID | AL | SA | IS | IMG | I | II | III |
| 1 | X | | | X | X | X | | | | | X | | |
| 2 | X | | | X | X | X | | | | | X | | |
| 3 | X | | | X | X | | X | | | | | X | |
| 4 | X | | X | | | | X | | | | | | X |
| 5 | X | | | | | | | | X | | X | | |
| 6 | X | | | | X | X | X | | | | | X | |
| 7 | X | | | | | X | X | | X | | | X | |
| 8 | X | | | | | X | | | | | | | X |
| 9 | X | | X | | X | X | X | | X | | | X | X |
| 10 | | X | | X | | | | X | | X | | X | |
| 11 | | X | | | | X | X | | X | | | X | |

**Soziale Netzwerke**

## *Soziale Netzwerke*

***Zielstellung***

Diese Aufgabe soll weniger dem Erlernen neuer Inhalte dienen, sondern ist als Anwendungs- bzw. Übungsaufgabe gedacht. Sie kann am Ende der Unterrichtsphase zur objektorientierten Datenbankentwicklung eingesetzt werden.

***Voraussetzungen***

Nötiges Vorwissen sind die objektorientierte Datenbankmodellierung und einfache SQL-Abfragen. Komplexeres Wissen über Datenbankabfragen ist prinzipiell nicht notwendig, jedoch erleichtert es die Lösung der Teilaufgaben b und c, da es für Schülerinnen und Schüler in diesem Fall nicht nachvollziehbar wäre, wie mit einer Datenbank praktisch gearbeitet wird.

***Aufgabe***

Soziale Netzwerke erheben von dem einzelnen Benutzer eine Vielzahl persönlicher Daten. Neben dem Namen und dem Alter werden auch die Schule, Hobbys, Interessen und weitere Kontaktdaten gespeichert. Die einzelnen Benutzer können miteinander befreundet sein, sie können sich Nachrichten schicken, sie können sich zu Gruppen zusammenschließen und sie können den »LIKE-Button« setzen. Auch ganze Fotoalben können erstellt und für einzelne Benutzer oder für Gruppen zugänglich gemacht werden.

a) Erstellen Sie ein objektorientiertes Datenbankmodell, um die Daten, die in einem derartigen sozialen Netzwerk anfallen, sinnvoll zu ordnen.
b) Das Unternehmen, das das soziale Netzwerk administriert, verlangt von den Benutzern keine Gebühren und finanziert sich stattdessen über Werbeeinnahmen. Erläutern Sie, weshalb die Benutzer nun häufig Werbeeinblendungen sehen, die genau zu ihren Hobbys passen.
c) Die Daten gelangen über illegale Kanäle in die Öffentlichkeit. Erläutern Sie welche Gefahren damit verbunden sind.

Quelle: LOG-IN-Archiv

**Abbildung 4.46: Zu den Daten in sozialen Netzwerken gehören u. a. die Ergebnisse zum »Liken«.**

## *Lösungshinweise*

***Lösungshinweise zu den Aufgaben***

a) Das gesuchte Datenbankmodell könnte wie in folgender Abbildung 4.47 aussehen.

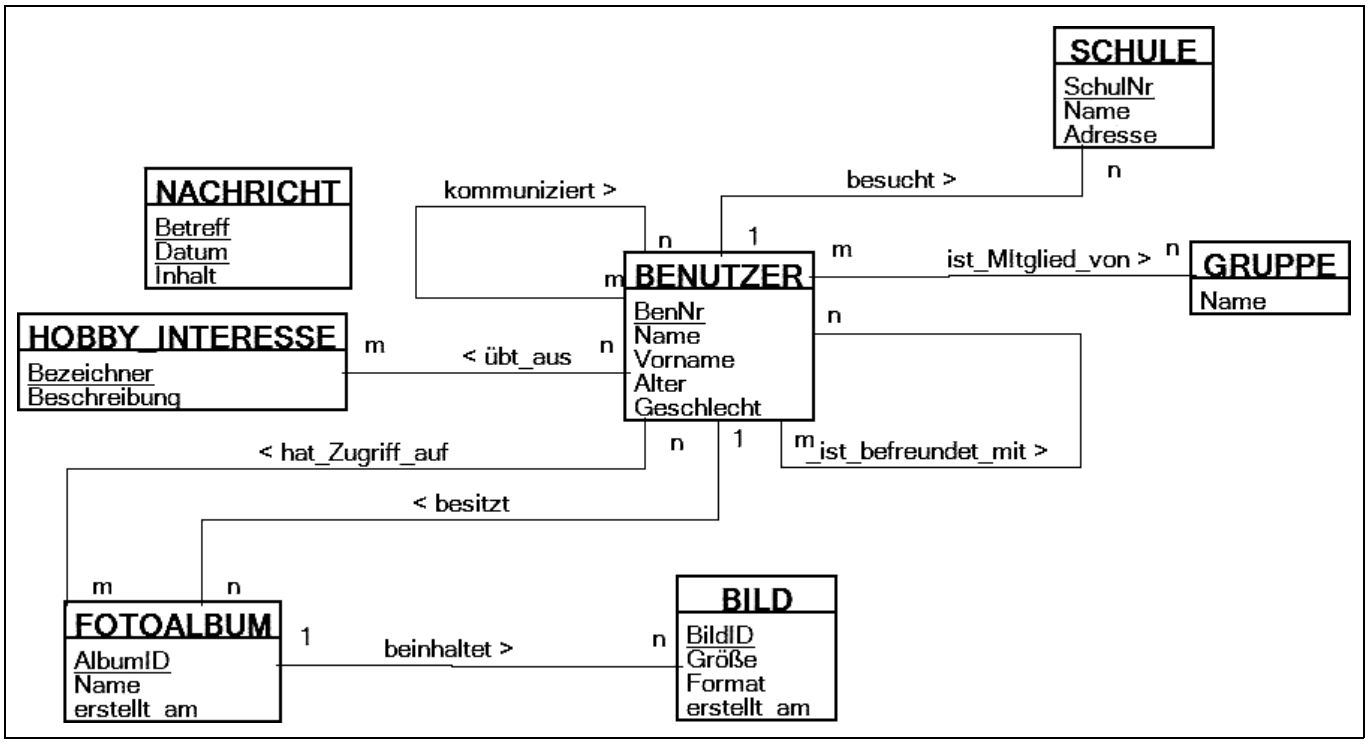



**Abbildung 4.47.**

*Bemerkung:* Die Hobbys und Interessen wurden im Datenbankmodell ausgelagert, um auf die in den nächsten Teilaufgaben angesprochenen Problematiken vorzubereiten. ER-Diagramme sind in Bayern nicht verbindlicher Lehrplaninhalt.

b) Die Hobbys und Interessen der Schülerinnen und Schüler können z. B. durch eine einfache SQL-Abfrage ermittelt werden. So kann das Unternehmen, das das Netzwerk betreibt, gezielt Werbung platzieren. Sind die Interessen bzw. Hobbys zudem mit entsprechenden Werbeblöcken verknüpft – dazu wäre eine zusätzliche Klasse WERBUNG notwendig – könnte der Benutzer in einfacher Weise individuell beworben werden. Im obigen Klassenmodell sind nur die wichtigsten Klassen und Beziehungen angegeben. Man könnte sich zusätzlich noch eine Klasse KONTAKTE vorstellen, die alle Kontakte eines Benutzers archiviert und so dem Unternehmen die Möglichkeit gibt, die Werbeblöcke auch auf häufig kontaktierte andere Benutzer zu schalten.

c) Neben der in der vorhergehenden Teilaufgabe bereits skizzierten Möglichkeit, Werbung gezielt zu platzieren, ergeben sich aus der illegalen Weitergabe der Daten Probleme, die über störende Werbung weit hinausgehen:
- Missbrauch der Bilder: Die persönlichen Bilder könnten heruntergeladen werden, mittels geeigneter Werkzeuge verunglimpfend modifiziert und ins Netz gestellt werden.
- Die Nachrichten und Bilder können über Jahre hinweg gespeichert werden. Man stelle sich vor, die Daten des Benutzers eines sozialen Netzwerks gelangen auf derart illegale Weise in die Öffentlichkeit. Zehn Jahre später bewirbt sich der Benutzer bei einem Arbeitgeber, der über diese Informationen verfügt. Wenn die zehn Jahre alten Kommunikationsvorgänge kritische Aspekte enthalten, wird der Arbeitgeber den Benutzer gegebenenfalls erst gar nicht zum Gespräch einladen, obwohl die Inhalte mit der derzeitigen Haltung des Benutzers nichts mehr zu tun haben müssen!

Quelle:
ISB – Staatsinstitut für Schulqualität und Bildungsforschung München (Hrsg.): Kompetenzorientierte Aufgaben für das Fach Informatik am Gymnasium. Augsburg: Brigg Pädagogik Verlag, 2013, S. 103–106.
http://www.isb.bayern.de/download/12311/hr_kompetenzorientierte_aufgaben_informatik.pdf

***Zuordnung zu den Prozess-, Inhalts- und Anforderungsbereichen***

| Aufgabe | Prozessbereiche | | | | | Inhaltsbereiche | | | | | Anforderungsbereiche | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MI | BB | SV | KK | DI | ID | AL | SA | IS | IMG | I | II | III |
| a | X | | X | | X | X | | | | | | X | X |
| b | | X | | X | | | | | X | X | | | X |
| c | | X | | | | | | | X | X | | | X |

## Chuck a Luck

**Chuck a Luck**

***Zielstellung***

Diese Aufgabe soll weniger dem Erlernen neuer Inhalte dienen, sondern ist als Anwendungs- bzw. Projektaufgabe gedacht.

***Material***

Aufgabenstellung, Lösungshinweise und BLUEJ-Projekte können folgender Quelle entnommen und aus dem Internet heruntergeladen werden:

ISB – Staatsinstitut für Schulqualität und Bildungsforschung München (Hrsg.): Kompetenzorientierte Aufgaben für das Fach Informatik am Gymnasium. Augsburg: Brigg Pädagogik Verlag, 2013, S.143–149.
http://www.isb.bayern.de/download/12311/hr_kompetenzorientierte_aufgaben_informatik.pdf

***Aufgabe***

Chuck a Luck, was auf Deutsch etwa »Glückswurf« bedeutet, ist ein einfaches Würfelglücksspiel mit drei Würfeln, das in vielen Kasinos der Welt gespielt wird. Dabei setzt jeder der Spieler zunächst einen selbst gewählten Einsatz auf eine der Zahlen eins bis sechs, anschließend wirft der Spielleiter die drei Würfel. Zeigt nun mindestens ein Würfel die gesetzte Zahl, bekommt der Spieler seinen Einsatz zurück und zusätzlich einen Einsatz für jeden Würfel, der seine Zahl zeigt. Das heißt, er gewinnt seinen Einsatz, wenn ein Würfel die gesetzte Zahl zeigt, er gewinnt den doppelten Einsatz, wenn zwei Würfel die gesetzte Zahl zeigen, und er gewinnt den dreifachen Einsatz, wenn alle drei Würfel die gesetzte Zahl zeigen. Zeigt kein Würfel die gesetzte Zahl, so ist der Einsatz verloren.

a) Entwickeln Sie in Kleingruppen eine Softwaresimulation von Chuck a Luck für einen oder mehrere (beliebig viele) Spieler. Entwerfen Sie zunächst ein Klassendiagramm mit allen Attributen und Methoden und implementieren Sie dann die einzelnen Klassen arbeitsteilig. Vereinbaren Sie vorab verbindliche Termine für die Fertigstellung und verfassen Sie am Ende eine Dokumentation Ihrer Software.

b) Nun soll simuliert werden, wie hoch der durchschnittliche Gewinn des Kasinos bei diesem Spiel ist. Erweitern Sie dazu Ihre Simulation so, dass ein Spieler automatisch 100 000-mal spielt und dabei jedes Mal einen Euro setzt.
Begründen Sie, wie sich das Ergebnis ändert, wenn der Spieler bei jedem Spiel auf dieselbe Zahl setzen würde.
Das Simulationsergebnis soll nun rechnerisch überprüft werden. Versuchen Sie, eine Formel herzuleiten, indem Sie zunächst berechnen, wie wahrscheinlich es ist, drei Euro, zwei Euro und einen Euro zu gewinnen bzw. einen Euro zu verlieren. Überlegen Sie dann, wie hoch Ihr Gewinn pro Spiel im Schnitt sein wird.

**Abbildung 4.48: Professioneller Chuck-a-Luck-Tisch eines Spielkasinos.**

Foto: Blackdog Casino Equipment

c) Durch Hinzufügen einer weiteren Wettmöglichkeit, dem sogenannten »Field«, entsteht aus Chuck a Luck das Kasinospiel Mini Dice. Bei »Field« wettet der Spieler darauf, dass die Augensumme der drei Würfel einen der Werte 5, 6, 7, 8, 13, 14, 15 oder 16 annimmt. Im Gewinnfall erhält er seinen Einsatz zurück und einen weiteren Einsatz zusätzlich, d. h., er gewinnt einen Einsatz. Fügen Sie Ihrer Simulation von Teilaufgabe a die Wettmöglichkeit »Field« hinzu.

d) Auch bei »Field« gewinnt im Schnitt immer das Kasino. Simulieren Sie wieder wie in Teilaufgabe b den durchschnittlichen Gewinn eines Spielers, der 100 000-mal mit einem Einsatz von einem Euro spielt.
Versuchen Sie wieder, Ihr Simulationsergebnis rechnerisch zu überprüfen. Überlegen Sie sich dazu, wie viele Möglichkeiten es gibt, mit drei Würfeln die Augensummen 5, 6, 7 etc. zu erzielen, und wie viele Möglichkeiten es insgesamt gibt. Falls Ihnen das Zählen der Möglichkeiten zu mühsam ist, können Sie es auch dem Computer überlassen, indem Sie ein kleines Programm dafür entwickeln.

*Hinweise zur Aufgabe*

Die Lösung der Teilaufgabe a kann auch umfangreicher ausfallen. Denkbar wäre etwa, dass mehrere Spiele von denselben Spielern gespielt werden und die Attribute es (Einsatz), z (gesetzte Zahl) und g (Gewinn) des Spielers und das Ergebnis eines Spieles jeweils abhängig von einer Spielnummer in einer anderen Klasse implementiert werden.

Die rechnerische Überprüfung der Simulationsergebnisse in den Teilaufgaben b und d ist jeweils eine fächerübergreifende Aufgabenstellung mit mathematischen Inhalten und kann auch weggelassen werden. Alternativ wäre es möglich, die Erwartungswerte anzugeben und den Schülerinnen und Schülern die Aufgabe zu geben, sie herzuleiten. Ebenso denkbar wäre, dass die Schülerinnen und Schüler die Erwartungswerte mithilfe einer Internetrecherche selbst finden und anschließend verifizieren.

## *Lösungshinweise*

***Lösungshinweise zu den Aufgaben***

a) Siehe folgende Abbildung 4.49.

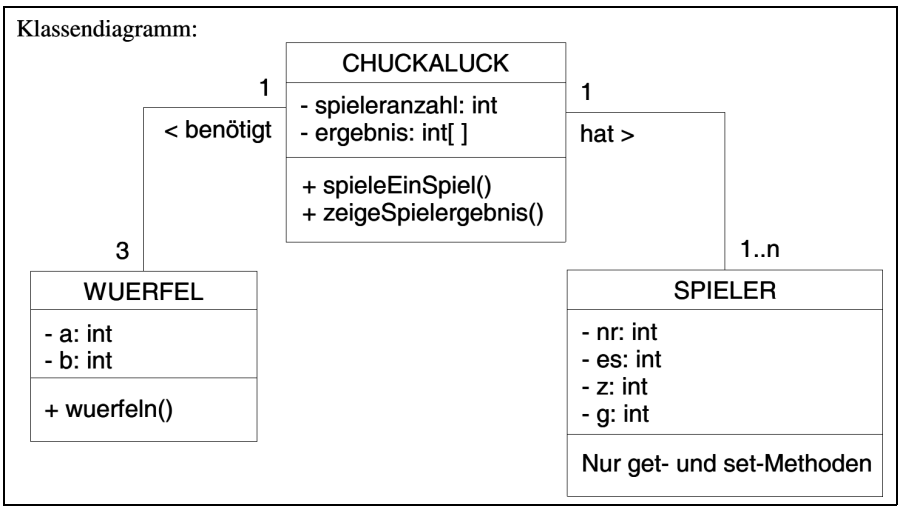



**Abbildung 4.49.**

***Implementierung der Klasse*** CHUCKALUCK

```
public class Chuckaluck {
    private Wuerfel[] wuerfel;
    private int spielerzahl;
    private Spieler[] spieler;
    private int[] ergebnis;
```

```
    public Chuckaluck(int anzahlSpieler) {
        wuerfel = new Wuerfel[3];
        ergebnis = new int[3];
        spielerzahl = anzahlSpieler;
        spieler = new Spieler[spielerzahl];
        for (int i=0; i<3; i++) {
            wuerfel[i] = new Wuerfel();
        }
        for (int i=0; i<spielerzahl; i++) {
            spieler[i] = new Spieler(i);
        }
    }

    public void spieleEinSpiel() {
        int[] treffer = new int[spielerzahl];
        for (int i=0; i<3; i++) {

            ergebnis[i] = wuerfel[i].wuerfeln();
        }
        for (int i=0; i<spielerzahl; i++) { treffer[i] = 0;
            for (int k=0; k<3; k++) {
                if (ergebnis[k]==spieler[i].gibZahl()) {
                    treffer[i] = treffer[i] + 1;
                }
            }
            if (treffer[i]>0) {
               int Gewinn = treffer[i]*spieler[i].gibEinsatz();
               spieler[i].setzeGewinn(Gewinn);
            } else {
                spieler[i].setzeGewinn(-spieler[i].gibEinsatz());
            }
        }
    }

    public void zeigeSpielergebnis() {
        System.out.println("Spielergebnis:");
        for (int i=0; i<3; i++) {
            System.out.println("Würfel " + i + ": " + ergebnis[i]);
        }
        for (int i=0; i<spielerzahl; i++) {
            System.out.println("Spieler " + i + ": gesetze Zahl " +
              spieler[i].gibZahl() + " / Einsatz " +
              spieler[i].gibEinsatz() +
              " / Gewinn " + spieler[i].gibGewinn());
        }
    }
}
```

b) Es kommt eine weitere Klasse SIMULATION hinzu:

```
public class Simulation {
    private Chuckaluck chuckaluck;

    public Simulation() {
        chuckaluck = new Chuckaluck();
    }

    public void simuliere(int anzahlSpiele) {
        int gesamtgewinn = 0;
        for (int i=0; i<anzahlSpiele; i++) {
            chuckaluck.spieleEinSpiel();
            gesamtgewinn = gesamtgewinn +
            chuckaluck.gibErstenSpieler().gibGewinn();
        }
        System.out.println("Der Spieler hat " + gesamtgewinn +
              " gewonnen.");
    }
}
```

Das Ergebnis würde sich nicht ändern, wenn man bei jedem Spiel auf dieselbe Zahl setzen würde, denn die Würfel haben kein »Gedächtnis«.

Rechnerische Überprüfung:

Erwarteter Gewinn pro Spiel bei einem Einsatz von einem Euro (siehe folgende Seite):

$$\left(\frac{1}{6}\right)^3 \cdot 3€ + 3 \cdot \left(\frac{1}{6}\right)^2 \cdot \frac{5}{6} \cdot 2€ + 3 \cdot \frac{1}{6} \cdot \left(\frac{5}{6}\right)^2 \cdot 1€ + \left(\frac{5}{6}\right)^3 \cdot (-1€) =$$

$$\frac{1}{216} \cdot 3€ + \frac{15}{216} \cdot 2€ + \frac{75}{216} \cdot 1€ + \frac{125}{216} \cdot (-1€) = -\frac{17}{216}\,€ \approx -0{,}07870€$$

Bei 100 000 Spielen würde man also im Schnitt 7870 Euro verlieren.

c) Angepasste Methoden der Klasse CHUCKALUCK:

```
public void spieleEinSpiel() {
    augensumme = 0;
    int[] treffer = new int[spielerzahl];
    for (int i=0; i<3; i++) {
        ergebnis[i] = wuerfel[i].wuerfeln();
        augensumme = augensumme + ergebnis[i];
    }
    for (int i=0; i<spielerzahl; i++) {
        if (!spieler[i].gibField()) {
            treffer[i] = 0;
            for (int k=0; k<3; k++) {
                if (ergebnis[k]==spieler[i].gibZahl()) {
                    treffer[i] = treffer[i] + 1;
                }
            }
            if (treffer[i] > 0) {
                spieler[i].setzeGewinn
                        (treffer[i]*spieler[i].gibEinsatz());
            } else {
                spieler[i].setzeGewinn(-spieler[i].gibEinsatz());
            }

        } else {
            if ((augensumme > 4 && augensumme < 9) ||
                (augensumme > 12 && augensumme < 17)) {
                spieler[i].setzeGewinn(spieler[i].gibEinsatz());
            } else {
                spieler[i].setzeGewinn(-spieler[i].gibEinsatz());
            }
        }
    }
}

public void zeigeSpielergebnis() {
    System.out.println("Spielergebnis:");
    for (int i=0; i<3; i++) {
        System.out.println("Würfel " + i + ": " + ergebnis[i]);
    }
    System.out.println("Augensumme: " + augensumme);
    for (int i=0; i<spielerzahl; i++) {
        if (!spieler[i].gibField()) {
            System.out.println("Spieler " + i + ": gesetze Zahl " +
            spieler[i].gibZahl() + " / Einsatz " +
            spieler[i].gibEinsatz()+ " / Gewinn " +
            spieler[i].gibGewinn());
        } else {
            System.out.println("Spieler " + i +
              ": auf Field gesetzt" +
              " / Einsatz " + spieler[i].gibEinsatz()+
              " / Gewinn " + spieler[i].gibGewinn());
        }
    }
}
```

d) Die Klasse SIMULATION ändert sich gegenüber Teilaufgabe b nicht, die Klassen CHUCKALUCK und SPIELER erhalten gegenüber Teilaufgabe c zusätzliche Konstruktoren, ähnlich wie in Teilaufgabe b.
Rechnerische Überprüfung:
Anzahl der Möglichkeiten mit drei (unterscheidbaren) Würfeln eine bestimmte Augensumme zu erzielen:

| Augensumme | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Möglichkeiten | 1 | 3 | 6 | 10 | 15 | 21 | 25 | 27 | 27 | 25 | 21 | 15 | 10 | 6 | 3 | 1 |

Insgesamt gibt es $6^3 = 216$ Möglichkeiten, d. h. die Wahrscheinlichkeit zu gewinnen beträgt:

$$\frac{6}{216}+\frac{10}{216}+\frac{15}{216}+\frac{21}{216}+\frac{21}{216}+\frac{15}{216}+\frac{10}{216}+\frac{6}{216}=\frac{104}{216}\approx 0{,}4815$$

Damit beträgt die Wahrscheinlichkeit zu verlieren:

$$1-\frac{104}{216}=\frac{112}{216}\approx 0{,}5185$$

Man erwartet also folgenden Gewinn pro Spiel bei einem Einsatz von einem Euro:

$$\frac{104}{216}\cdot 1€+\frac{112}{216}\cdot(-1€)=-\frac{8}{216}€\approx -0{,}03704€$$

Bei 100 000 Spielen würde man also im Schnitt 3704 Euro verlieren.

Die oben aufgeführte Wahrscheinlichkeitsverteilung lässt sich schnell mithilfe eines kleinen JAVA-Programms berechnen:

```
public class Wvert {
    private int[] augensumme;

    public Wvert() {
        augensumme = new int[19];
        for (int i=0; i<19; i++) {
            augensumme[i] = 0;
        }
    }

    public void verteilungBerechnen() {
        int summe;
        for (int i=1; i<7; i++) {
            for (int j=1; j<7; j++) {
                for (int k=1; k<7; k++) {
                    summe = i + j + k;
                    augensumme[summe] = augensumme[summe] + 1;
                }
            }
        }
    }

    public void verteilungAusgeben() {
        for (int i=3; i<19; i++) {
            System.out.println("Anzahl der Möglichkeiten
            für Augensumme " + i + ": " + augensumme[i]);
        }
    }
}
```

Quelle:
ISB – Staatsinstitut für Schulqualität und Bildungsforschung München (Hrsg.): Kompetenzorientierte Aufgaben für das Fach Informatik am Gymnasium. Augsburg: Brigg Pädagogik Verlag, 2013, S. 143–149.
http://www.isb.bayern.de/download/12311/hr_kompetenzorientierte_aufgaben_informatik.pdf

***Zuordnung zu den Prozess-, Inhalts- und Anforderungsbereichen***

| Aufgabe | Prozessbereiche | | | | | Inhaltsbereiche | | | | | Anforderungsbereiche | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MI | BB | SV | KK | DI | ID | AL | SA | IS | IMG | I | II | III |
| a | X | | | X | X | X | X | | X | | | X | X |
| b | X | X | | | | | X | | | | | X | X |
| c | X | | | X | | | X | | X | | | X | |
| d | X | | | | | | X | | | | | X | X |

## E-Mail-Protokolle

### E-Mail-Protokolle

***Zielstellung***

Die Schülerinnen und Schüler erkunden mithilfe dieser Aufgabe die Wirkungsweise zweier Protokolle der Anwendungsschicht. Sie analysieren und beschreiben den Datenaustausch der E-Mail-Protokolle SMTP und POP3. Im Weiteren betrachten die Schülerinnen und Schüler qualitative Aspekte der Protokolle und beurteilen die Protokolle bezüglich der Verfügbarkeit, Konsistenz, Sicherheit und Authentizität der Daten.

***Voraussetzungen und Material***

Für die Schülerinnen und Schüler steht ein reales oder simuliertes Netz, bestehend aus einem E-Mail-Server und mindestens zwei weiteren Rechnern, zur Verfügung. Ein grundlegendes Verständnis der IP-Adressierung und Kenntnisse zu einem Schichtenmodell werden vorausgesetzt. Das System sei bereits so konfiguriert, dass zwei Nutzer miteinander per E-Mail kommunizieren können (siehe folgende Abbildung 4.50).

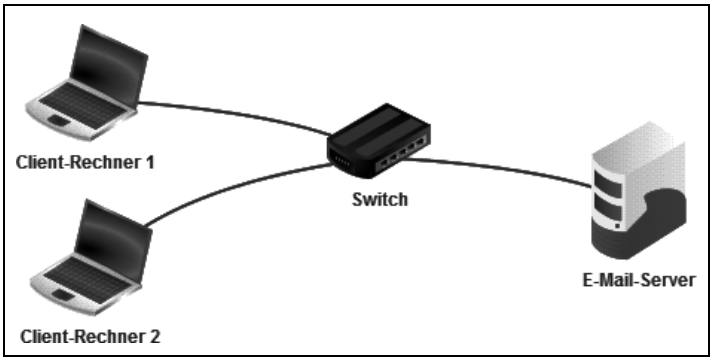



**Abbildung 4.50.**

Die Sicherheitseinstellungen des SMTP-Protokolls seien deaktiviert, wie in der folgenden Abbildung 4.51 verdeutlicht wird.


**Abbildung 4.51.**

Auf allen Geräten werden die Kommunikationsschritte entweder durch die Simulationssoftware selbst oder durch entsprechende Software dokumentiert und sind jeweils als Log-File durch die Schülerinnen und Schüler einsehbar.

***Aufgabe***

E-Mails gehören zu den ältesten Möglichkeiten der digitalen Kommunikation. Nach wie vor steigt die Zahl der versendeten E-Mails – die unerwünschten Spam-Mails nicht mitgerechnet – jährlich an. Im Folgenden sollen Sie herausfinden, was beim Senden und Empfangen von E-Mails genau passiert. In Protokollen werden Regeln für Kommunikationsprozesse festgelegt. Mit dem gegebenen System aus E-Mail-Server und Clients erkunden und vergleichen Sie die Protokolle SMTP und POP3, die den Versand und den Empfang von E-Mails regeln.

1. Die Clients sind schon für die Nutzer Horst bzw. Hugo eingerichtet. Die Kennwörter lauten jeweils »test«.

Versenden und empfangen Sie mindestens je eine E-Mail im Namen von Horst und Hugo.

2. Die einzelnen Schritte des Datenaustauschs sind für jeden beteiligten Rechner in Log-Files festgehalten worden. Lassen Sie sich die Log-Files anzeigen und betrachten Sie den Datenaustausch auf der Anwendungsschicht.

2.1 Vervollständigen Sie anhand des entsprechenden Log-Files das Sequenzdiagramm (siehe folgende Abbildung 4.52) für den Versand einer E-Mail vom Client zum Server mithilfe des SMTP-Protokolls.

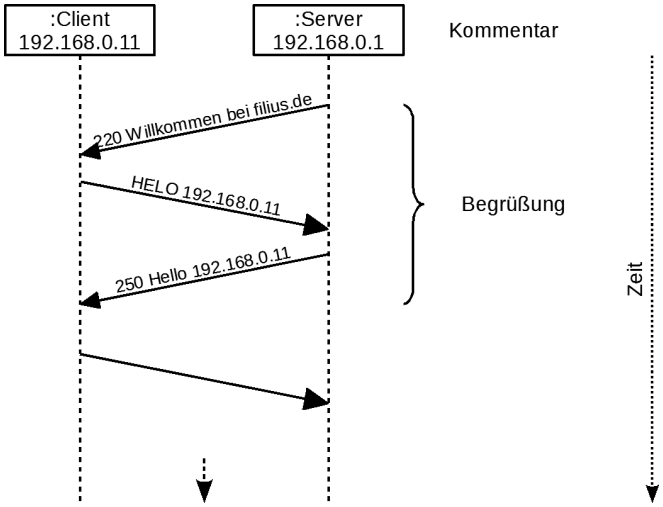



**Abbildung 4.52.**

2.2 Zeichnen Sie anhand der Log-Files ein Sequenzdiagramm für den Abruf einer E-Mail durch den Client vom Server mithilfe des POP3-Protokolls.

3. Das SMTP-Protokoll gilt in seiner ursprünglichen Form als unsicher, da es leicht ist, Absenderadressen zum Beispiel für das Versenden von Spam-E-Mails zu fälschen.

3.1 Begründen Sie diese Aussage mithilfe des SMTP-Sequenzdiagramms.

3.2 Eine Möglichkeit, das Fälschen von E-Mail-Absendern etwas zu erschweren, stellt das Prinzip »POP-before-SMTP« dar, bei dem ein Nutzer erst mithilfe des POP3-Protokolls versuchen muss, E-Mails vom Provider abzurufen, bevor er per SMTP eigene E-Mails senden darf.
Vergleichen Sie die Sequenzdiagramme von POP3 und SMTP und begründen Sie die Erhöhung der Sicherheit durch dieses Verfahren.

4. Analysieren Sie die Übertragung der Nutzerdaten (Anmeldename und Passwort) durch das POP3-Protokoll und bewerten Sie die Sicherheit der übertragenen Daten.

5. Fügen Sie in Ihrem E-Mail-System einen weiteren Rechner hinzu und installieren Sie einen E-Mail-Client für den Nutzer Horst.
Schreiben Sie mehrere E-Mails an Horst und rufen Sie zwischendurch die E-Mails für Horst von verschiedenen Rechnern aus ab.

5.1 Beschreiben Sie die auftretenden Probleme.

5.2 Informieren Sie sich über das IMAP-Protokoll und formulieren Sie eine Argumentation für die Verwendung von IMAP anstelle von POP3.

*Hinweise zur Bearbeitung der Aufgabe*

Um den Schülerinnen und Schülern das Problemfeld der Authentizität der Daten im Zusammenhang mit dem Spam-Problem zu verdeutlichen, sollten die erweiterten Sicherheitseinstellungen des SMTP-Protokolls deaktiviert sein. Sollte die Übertragung von Benutzernamen und ggf. Passwort nicht im Klartext möglich sein, ist ein Hinweis zur Base64-Codierung erforderlich.

Zur Zeitersparnis bei der Auswertung können exemplarische Log-Files als Kalkulationstabelle mit Filterfunktion bereitgehalten werden.

## Lösungshinweise

***Lösungshinweise zu den Aufgaben***

2. Übertragen der Log-File-Einträge der Anwendungsschicht in das Sequenzdiagramm (siehe Abbildungen 4.53 bis 4.55).

SMTP-Log (exemplarisch) – siehe folgende Abbildung 4.53.

| Nr | Zeit | Quelle | Ziel | Schicht | |
|---|---|---|---|---|---|
| 65 | 10:38:28.339 | 192.168.0.1:25 | 192.168.0.11:53676 | Anwendung | 220 Willkommen bei filius.de |
| 67 | 10:38:28.392 | 192.168.0.11:53676 | 192.168.0.1:25 | Anwendung | HELO 192.168.0.11 |
| 69 | 10:38:28.651 | 192.168.0.1:25 | 192.168.0.11:53676 | Anwendung | 250 Hello 192.168.0.11 |
| 71 | 10:38:28.704 | 192.168.0.11:53676 | 192.168.0.1:25 | Anwendung | MAIL FROM: <horst@filius.de> |
| 73 | 10:38:28.967 | 192.168.0.1:25 | 192.168.0.11:53676 | Anwendung | 250 Sender OK |
| 75 | 10:38:29.021 | 192.168.0.11:53676 | 192.168.0.1:25 | Anwendung | RCPT TO:<hugo@filius.de> |
| 77 | 10:38:29.289 | 192.168.0.1:25 | 192.168.0.11:53676 | Anwendung | 250 Recipient OK |
| 79 | 10:38:29.344 | 192.168.0.11:53676 | 192.168.0.1:25 | Anwendung | DATA |
| 81 | 10:38:29.603 | 192.168.0.1:25 | 192.168.0.11:53676 | Anwendung | 354 End data with <CR><LF>.<CR><LF> |
| 83 | 10:38:29.656 | 192.168.0.11:53676 | 192.168.0.1:25 | Anwendung | From: Horst <horst@filius.de> To: <hugo@filius.de> Subject: Testmail hurz . |
| 85 | 10:38:29.917 | 192.168.0.1:25 | 192.168.0.11:53676 | Anwendung | 250 Mail queued for delivery |
| 87 | 10:38:29.971 | 192.168.0.11:53676 | 192.168.0.1:25 | Anwendung | QUIT |
| 89 | 10:38:30.238 | 192.168.0.1:25 | 192.168.0.11:53676 | Anwendung | 221 Server beendet Verbindung. |

POP3-Log (exemplarisch) – siehe folgende Abbildung 4.54.

| Nr | Zeit | Quelle | Ziel | Schicht | |
|---|---|---|---|---|---|
| 44 | 10:48:23.343 | 192.168.0.1:110 | 192.168.0.12:40437 | Anwendung | +OK POP3 server ready |
| 46 | 10:48:23.395 | 192.168.0.12:40437 | 192.168.0.1:110 | Anwendung | USER hugo |
| 48 | 10:48:23.664 | 192.168.0.1:110 | 192.168.0.12:40437 | Anwendung | +OK enter password |
| 50 | 10:48:23.717 | 192.168.0.12:40437 | 192.168.0.1:110 | Anwendung | PASS test |
| 52 | 10:48:23.979 | 192.168.0.1:110 | 192.168.0.12:40437 | Anwendung | +OK Mailbox locked and ready |
| 54 | 10:48:24.034 | 192.168.0.12:40437 | 192.168.0.1:110 | Anwendung | STAT |
| 56 | 10:48:24.309 | 192.168.0.1:110 | 192.168.0.12:40437 | Anwendung | +OK 1 54 |
| 58 | 10:48:24.362 | 192.168.0.12:40437 | 192.168.0.1:110 | Anwendung | LIST |
| 60 | 10:48:24.624 | 192.168.0.1:110 | 192.168.0.12:40437 | Anwendung | +OK 1 54 0 54 |
| 62 | 10:48:24.626 | 192.168.0.12:40437 | 192.168.0.1:110 | Anwendung | RETR 0 |
| 64 | 10:48:24.944 | 192.168.0.1:110 | 192.168.0.12:40437 | Anwendung | +OK message follows From: Horst <horst@filius.de> To: <hugo@filius.de> Subject: Testmail hurz |
| 66 | 10:48:24.946 | 192.168.0.12:40437 | 192.168.0.1:110 | Anwendung | DELE 0 |
| 68 | 10:48:25.261 | 192.168.0.1:110 | 192.168.0.12:40437 | Anwendung | +OK message marked for delete |
| 70 | 10:48:25.319 | 192.168.0.12:40437 | 192.168.0.1:110 | Anwendung | QUIT |
| 72 | 10:48:25.583 | 192.168.0.1:110 | 192.168.0.12:40437 | Anwendung | +OK |

**Abbildungen 4.53 (oben) und 4.54.**

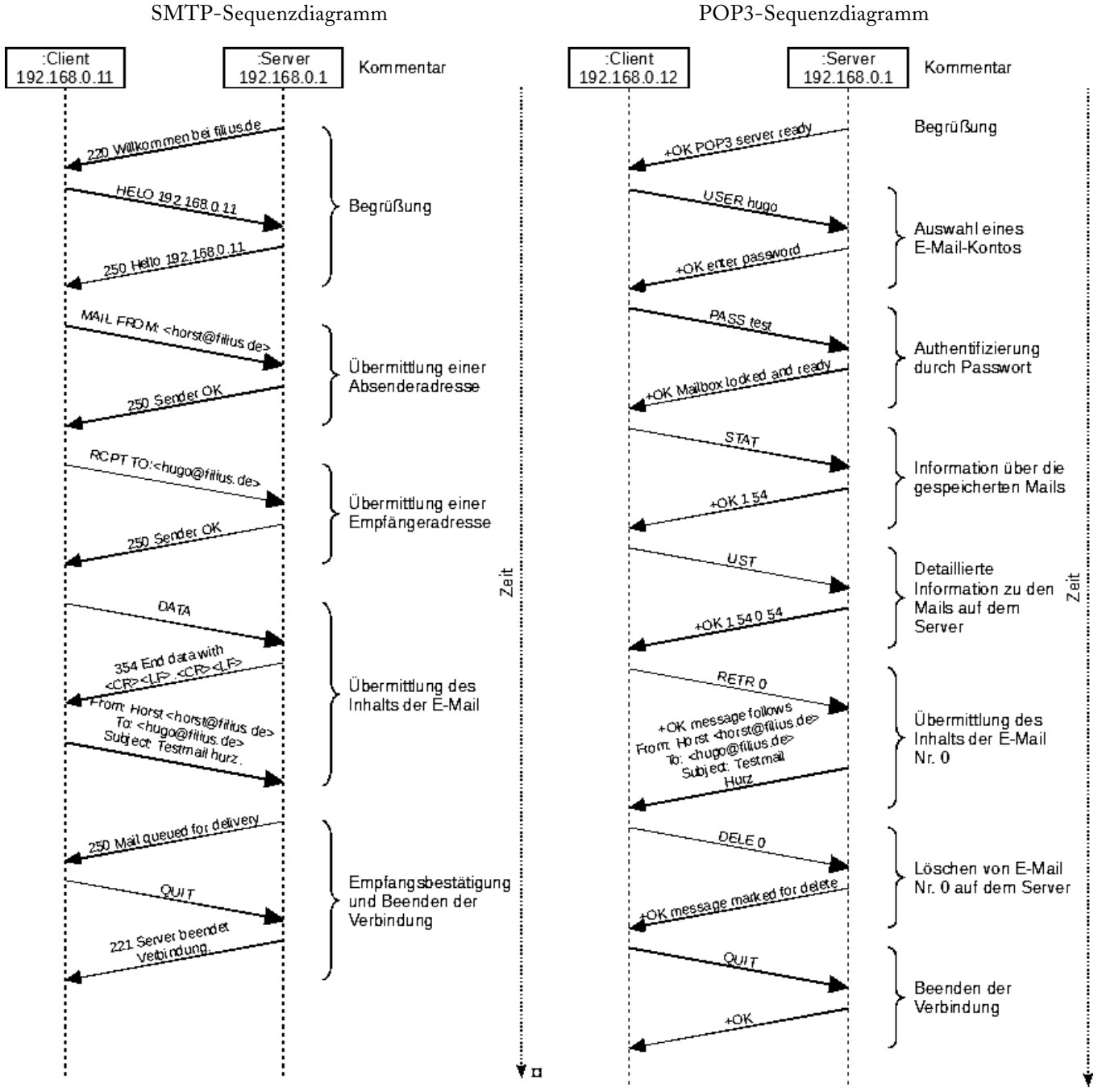



**Abbildung 4.55.**

3.1 Das SMTP-Protokoll erfordert keine Authentifizierung. Jeder kann im Namen eines anderen Mails versenden.

3.2 Das POP3-Protokoll erfordert eine Authentifizierung. Somit ist das Senden per SMTP erst nach einer vorherigen gültigen Authentifizierung mit POP3 möglich.

4. Das POP3-Protokoll verlangt zwar ein Passwort, die Übertragung erfolgt aber unverschlüsselt. Somit können die Anmeldedaten ausgelesen werden und sind nicht sicher. Dies ist auch der Grund, warum das in Aufgabe 3 untersuchte »POP-before-SMTP«-Verfahren insgesamt als unsicher eingestuft werden muss.

5.1 Es gibt zwei Möglichkeiten. Wird eine E-Mail nach Abruf auf dem Server wie in obigem Beispiel gelöscht, kann sie nicht mehr auf dem zweiten Client gelesen werden. Wird sie gelöscht, existieren mehrere Kopien der Mail

auf verschiedenen Rechnern, die nicht konsistent verwaltet (z. B. gelöscht) werden können; jede Kopie muss einzeln behandelt werden.

5.2 Durch die Verwendung des IMAP-Protokolls existiert immer nur eine Instanz einer E-Mail auf dem Server, auf die jeweils von den Clients referenziert wird. Das gewährleistet sowohl die ortsunabhängige, vollständige Verfügbarkeit der E-Mails als auch die Konsistenz der Daten durch Vermeidung von Redundanzen.

Quelle:
Gramm, A.; Hornung, M.; Witten, H.: E-Mail (nur?) für Dich – Eine Unterrichtsreihe des Projekts »Informatik im Kontext«. In: LOG IN, 31. Jg. (2011), Nr. 169/170, Beilage, S. 6–7.
und
Gramm, A.; Hornung, M.; Witten, H.: E-Mail (nur?) für Dich – Lernabschnitt 1: Wie kommt eine E-Mail von meinem Computer auf den Computer des Empfängers?
http://medienwissenschaft.uni-bayreuth.de/informatik-im-kontext/index.php/entwuerfe/email-nur-fuer-dich/lernabschnitt-1-weg-einer-e-mail/

***Zuordnung zu den Prozess-, Inhalts- und Anforderungsbereichen***

| Aufgabe | Prozessbereiche | | | | | Inhaltsbereiche | | | | | Anforderungsbereiche | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | MI | BB | SV | KK | DI | ID | AL | SA | IS | IMG | I | II | III |
| 1 | | | | X | | | | | X | | X | | |
| 2.1 | X | | X | | X | | | X | X | | | X | |
| 2.2 | X | | X | | X | | | X | X | | | | X |
| 3.1 | | X | | | X | | | X | X | | | X | |
| 3.2 | | X | X | | | | | X | X | | | | X |
| 4 | | X | X | | | | | X | X | | | | X |
| 5.1 | | | | X | X | | | | X | | | X | |
| 5.2 | | X | | | | | | X | X | | | | X |

# Anhang

5

## Verwendete Literatur und Internetquellen


Abraham, U.; Müller, A.: Aus Leistungsaufgaben lernen. In: Praxis Deutsch, 36. Jg. (2009), Nr. 214, S. 4–12.

AKBSI – Arbeitskreis »Bildungsstandards« der Gesellschaft für Informatik (Hrsg.): Grundsätze und Standards für die Informatik in der Schule – Bildungsstandards Informatik für die Sekundarstufe I. Empfehlungen der Gesellschaft für Informatik e. V. vom 24.01.2008. In: LOG IN, 28. Jg. (2008), Nr. 150/151, Beilage.

Baumann, R.: Auf dem Weg zu Bildungsstandards Informatik für die Sekundarstufe II – Probleme und Lösungsvorschläge. In: LOG IN, 31. Jg. (2011), Nr. 169/170, S. 60–71.

BLK – Bund-Länder-Kommission für Bildungsplanung und Forschungsförderung: Gutachten zur Vorbereitung des Programms »Steigerung der Effizienz des mathematisch-naturwissenschaftlichen Unterrichts«. Reihe »Materialien zur Bildungsplanung und zur Forschungsförderung«, Heft 60. Bonn: Geschäftsstelle der BLK, 1997.
http://www.blk-bonn.de/papers/heft60.pdf

Büchter, A.; Leuders, T.: Mathematikaufgaben selbst entwickeln. Lernen fördern – Leistung überprüfen. Berlin: Cornelsen Scriptor, 2005.

Büchter, A.: Verstehensorientierte Aufgaben als Kern einer neuen Kultur der Leistungsüberprüfung. Reihe »SINUS-Transfer«, Erläuterung zu Modul 10. Bayreuth: Universität Bayreuth Z-MNU, 2006.
http://www.sinus-transfer.de/fileadmin/MaterialienBT/Buechter_Modul_10.pdf

Claus, V.; Schwill A. (Bearb.): Duden – Informatik A–Z – Fachlexikon für Studium, Ausbildung und Beruf. Mannheim: Bibliographisches Institut, 42006.

Fothe, M.: Bildungsstandards Informatik für die Sekundarstufe II – Vorüberlegungen zur Entwicklung. In: T. Brinda, M. Fothe, P. Hubwieser, K.Schlüter (Hrsg.): Didaktik der Informatik – Aktuelle Forschungsergebnisse. 5. Workshop der GI-Fachgruppe »Didaktik der Informatik« 24.–25.09.2008 in Erlangen. Reihe »LNI – Lecture Notes in Informatics – Proceedings«, Band P-135. Bonn: Köllen Verlag, 2008, S. 107-117.
http://subs.emis.de/LNI/Proceedings/Proceedings135/gi-proc-135-010.pdf

Gieseke, W. (Hrsg.): Abituraufgaben Informatik Niedersachsen. Reihe »Duden Informatik«. Berlin: Cornelsen Schulbuchverlage, 2013.

ISB – Staatsinstitut für Schulqualität und Bildungsforschung München (Hrsg.): Abiturprüfung 2013 – Informatik. München: ISB, 2013.
https://www.isb.bayern.de/download/15005/abituraufgaben_2013_informatik.pdf

KMK – Ständige Konferenz der Kultusminister der Länder in der Bundesrepublik Deutschland: Bildungsstandards im Fach Mathematik für die Allgemeine Hochschulreife. Beschluss der Kultusministerkonferenz vom 18.10.2012. Bonn; Berlin; Köln: KMK; Wolters Kluwer, 2015.
http://www.kmk.org/fileadmin/Dateien/veroeffentlichungen_beschluesse/2012/2012_10_18-Bildungsstandards-Mathe-Abi.pdf

KMK – Ständige Konferenz der Kultusminister der Länder in der Bundesrepublik Deutschland: Einheitliche Prüfungsanforderungen in der Abiturprüfung Informatik. Beschluss vom 1.12.1989 i. d. F. vom 5.2.2004. München; Neuwied: Wolters Kluwer; Luchterhand, 2004.
http://www.kmk.org/fileadmin/Dateien/veroeffentlichungen_beschluesse/1989/1989_12_01-EPA-Informatik.pdf

Leisen, J.: Kompetenzorientiert unterrichten – Fragen und Antworten zu kompetenzorientiertem Unterricht und einem entsprechenden Lehr-Lern-Modell. In: Unterricht Physik, 22. Jg. (2011), Nr. 123/124, S. 4–10.

Ministerium für Schule und Weiterbildung des Landes Nordrhein-Westfalen: Kernlehrplan für die Sekundarstufe II Gymnasium/Gesamtschule in Nordrhein-Westfalen Informatik. Heft 4725. Düsseldorf: Schulministerium, 2014.
http://www.schulentwicklung.nrw.de/lehrplaene/upload/klp_SII/if/KLP_GOSt_Informatik.pdf

MoKoM – Entwicklung von qualitativen und quantitativen Messverfahren zu Lehr-Lern-Prozessen für Modellierung und Systemverständnis in der Informatik. Letzte Änderungen am: 16.11.2011. http://ddi.uni-paderborn.de/forschung/mokom.html

Röhner, G.: Abituraufgaben. In: LOG IN, 33. Jg. (2013/2014), Nr. 176/177, S. 123–128.

Walpuski, M.; Kauertz, A.; Fischer, H.E.; Kampa, N.; Mayer, J.; Sumfleth, E.; Wellnitz, N.: ESNaS – Evaluation der Standards für die Naturwissenschaften in der Sekundarstufe I. In: A. Gehrmann, U. Hericks, M. Lüders (Hrsg.): Bildungsstandards und Kompetenzmodelle – Beiträge zu einer aktuellen Diskussion über Schule, Lehrerbildung und Unterricht. Bad Heilbrunn: Julius Klinkhardt, 2010, S. 171–184.

Alle Internetquellen wurden zuletzt am 15. Februar 2016 geprüft.

# Mitwirkende

An der Entwicklung dieser Bildungsstandards Informatik für die Sekundarstufe II sind sehr viele Informatiklehrkräfte und Hochschullehrende beteiligt gewesen. Sie haben Entwürfe kritisch kontrolliert, Beiträge zu den Aufgaben geliefert, im Workshop der INFOS-Tagung in Darmstadt Feedback gegeben oder an der Onlinebefragung teilgenommen. Im Arbeitskreis wurden alle Rückmeldungen sorgfältig geprüft und verarbeitet. Besonders hilfreich waren das Gutachten von Prof. Dr. Johannes Magenheim und die schriftlichen Stellungnahmen von Dr. Hermann Puhlmann und Prof. Dr. Marco Thomas.

Durch die namentliche Nennung soll allen für ihre aktive Beteiligung gedankt werden:

Sven Alisch (Hamburg), Rüdeger Baumann (Garbsen), Peter Brichzin (Ottobrunn), Roland Ebner (Königs Wusterhausen), Michael Fothe (Jena), Steffen Friedrich (Dresden), Marc Hannappel (Lüneburg), Tino Hempel (Ribnitz-Damgarten), Silke Herbst (Zimkendorf), Walter Hower (Albstadt-Sigmaringen), Christian Kemmer (Rheine), Monika Klaaßen (Sanitz), Bernhard Koerber (Berlin), Martin Lönertz (Esch), Peter Micheuz (Klagenfurt), Reinhard Oldenburg (Augsburg), Torsten Otto (Hamburg), Hermann Puhlmann (Altdorf), Eva Segeritz (Offenbach), Wolf Spalteholz (Dresden), Marco Thomas (Münster), Albert Wiedemann (München).

Im Arbeitskreis »Bildungsstandards SII« haben mitgearbeitet:

Gerhard Röhner (Dieburg), Prof. Dr. Torsten Brinda (Essen), Volker Denke (Nürnberg), Dr. Lutz Hellmig (Rostock), Theo Heußer (Laudenbach), Dr. Arno Pasternak (Hagen), Prof. Dr. Andreas Schwill (Potsdam), Monika Seiffert (Hamburg).

Der Arbeitskreis wurde von Gerhard Röhner koordiniert.

**Beilage zu LOG IN, 36. Jg. (2016), Heft Nr. 183/184**

Gesellschaft für Informatik e. V. (GI)
Wissenschaftszentrum
Ahrstraße 45
53175 Bonn
E-Mail: gs@gi-ev.de
URL: http://www.gi-ev.de/

LOG IN Verlag GmbH
Redaktion LOG IN
Friedrichshaller Straße 41
14199 Berlin
E-Mail: redaktionspost@log-in-verlag.de
URL: http://www.log-in-verlag.de/

LOG IN
VERLAG