

Schulinterner Lehrplan Informatik Q1

Qualifikationsphase 1	
<u>Unterrichtsvorhaben Q1-I</u>	<u>Unterrichtsvorhaben Q1-II</u>
Thema: <i>Wiederholung der objektorientierten Modellierung und Programmierung anhand einer kontextbezogenen Problemstellung</i>	Thema: <i>Modellierung und Implementierung von Anwendungen mit dynamischen, linearen Datenstrukturen</i>
Zentrale Kompetenzen: <ul style="list-style-type: none">• Argumentieren• Modellieren• Implementieren• Darstellen und Interpretieren• Kommunizieren und Kooperieren	Zentrale Kompetenzen: <ul style="list-style-type: none">• Argumentieren• Modellieren• Implementieren• Darstellen und Interpretieren• Kommunizieren und Kooperieren
Inhaltsfelder: <ul style="list-style-type: none">• Daten und ihre Strukturierung• Algorithmen• Formale Sprachen und Automaten• Informatiksysteme	Inhaltsfelder: <ul style="list-style-type: none">• Daten und ihre Strukturierung• Algorithmen• Formale Sprachen und Automaten
Inhaltliche Schwerpunkte: <ul style="list-style-type: none">• Objekte und Klassen• Analyse, Entwurf und Implementierung von Algorithmen• Syntax und Semantik einer Programmiersprache• Nutzung von Informatiksystemen	Inhaltliche Schwerpunkte: <ul style="list-style-type: none">• Objekte und Klassen• Analyse, Entwurf und Implementierung von Algorithmen• Algorithmen in ausgewählten informatischen Kontexten• Syntax und Semantik einer Programmiersprache
Zeitbedarf: 8 Stunden	Zeitbedarf: 20 Stunden

Qualifikationsphase 1

Unterrichtsvorhaben Q1-III

Thema:

Suchen und Sortieren auf linearen Datenstrukturen

Zentrale Kompetenzen:

- Argumentieren
- Modellieren
- Implementieren
- Darstellen und Interpretieren
- Kommunizieren und Kooperieren

Inhaltsfelder:

- Algorithmen
- Formale Sprachen und Automaten

Inhaltliche Schwerpunkte:

- Analyse, Entwurf und Implementierung von Algorithmen
- Algorithmen in ausgewählten informatischen Kontexten
- Syntax und Semantik einer Programmiersprache

Zeitbedarf: 16 Stunden

Unterrichtsvorhaben Q1-IV

Thema:

Modellierung und Nutzung von relationalen Datenbanken in Anwendungskontexten

Zentrale Kompetenzen:

- Argumentieren
- Modellieren
- Implementieren
- Darstellen und Interpretieren
- Kommunizieren und Kooperieren

Inhaltsfelder:

- Daten und ihre Strukturierung
- Algorithmen
- Formale Sprachen und Automaten
- Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:

- Datenbanken
- Algorithmen in ausgewählten informatischen Kontexten
- Syntax und Semantik einer Programmiersprache
- Sicherheit

Zeitbedarf: 20 Stunden

Qualifikationsphase 1

Unterrichtsvorhaben QI-V

Thema:

Sicherheit und Datenschutz in Netzstrukturen

Zentrale Kompetenzen:

- Argumentieren
- Darstellen und Interpretieren
- Kommunizieren und Kooperieren

Inhaltsfelder:

- Informatiksysteme
- Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:

- Einzelrechner und Rechnernetzwerke
- Sicherheit
- Nutzung von Informatiksystemen, Wirkungen der Automatisierung

Zeitbedarf: 10 Stunden

Summe Qualifikationsphase 1: 74 Stunden

Wiederholung der objektorientierten Modellierung und Programmierung (Unterrichtsvorhaben Q1-I)

Leitfragen: *Wie modelliert und implementiert man zu einer Problemstellung in einem geeigneten Anwendungskontext Java-Klassen inklusive ihrer Attribute, Methoden und Beziehungen? Wie kann man die Modellierung und die Funktionsweise der Anwendung grafisch darstellen?*

Vorhabenbezogenen Konkretisierung:

Zu einer Problemstellung in einem Anwendungskontext soll eine Java-Anwendung entwickelt werden. Die Problemstellung soll so gewählt sein, dass für diese Anwendung die Verwendung einer abstrakten Oberklasse als Generalisierung verschiedener Unterklassen sinnvoll erscheint und eine Klasse durch eine Unterklasse spezialisiert werden kann. Um die Aufgabe einzugrenzen, können (nach der ersten Problemanalyse) einige Teile (Modellierungen oder Teile von Java-Klassen) vorgegeben werden.

Die Schülerinnen und Schülern erläutern und modifizieren den ersten Entwurf und modellieren sowie implementieren weitere Klassen und Methoden für eine entsprechende Anwendung. Klassen und ihre Beziehungen werden in einem Implementationsdiagramm dargestellt. Dabei werden Sichtbarkeitsbereiche zugeordnet. Exemplarisch wird eine Klasse dokumentiert. Der Nachrichtenaustausch zwischen verschiedenen Objekten wird verdeutlicht, indem die Kommunikation zwischen zwei ausgewählten Objekten grafisch dargestellt wird. In diesem Zusammenhang wird das Nachrichtenkonzept der objektorientierten Programmierung wiederholt.

Zeitbedarf: 8 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Wiederholung und Erweiterung der objektorientierten Modellierung und Programmierung durch Analyse und Erweiterung eines kontextbezogenen Beispiels</p> <p>(a) Analyse der Problemstellung</p> <p>(b) Analyse der Modellierung (Implementationsdiagramm)</p> <p>(c) Erweiterung der Modellierung im</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> analysieren und erläutern objektorientierte Modellierungen (A), beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M), ordnen Klassen, Attributen und Methoden ihre 	<p><i>Beispiel:</i> Wetthuepfen</p> <p>Für ein Wetthüpfen zwischen einem Hasen, einem Hund und einem Vogel werden die Tiere gezeichnet. Alle Tiere springen wiederholt nach links. Die Höhe und Weite jedes Hüpfers ist zufällig. Evtl. marschieren sie anschließend hintereinander her.</p> <p>oder</p> <p><i>Beispiel:</i> Tannenbaum</p>

<p>Implementationsdiagramm (Vererbung, abstrakte Klasse)</p> <p>(d) Kommunikation zwischen mindestens zwei Objekten (grafische Darstellung)</p> <p>(e) Dokumentation von Klassen</p> <p>(f) Implementierung der Anwendung oder von Teilen der Anwendung</p>	<p>Sichtbarkeitsbereiche zu (M),</p> <ul style="list-style-type: none"> • modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M), • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • wenden eine didaktisch orientierte Entwicklungsumgebung zur Demonstration, zum Entwurf, zur Implementierung und zum Test von Informatiksystemen an (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D), • dokumentieren Klassen (D), • stellen die Kommunikation zwischen Objekten grafisch dar (D). 	<p>Ein Tannenbaum soll mit verschiedenen Arten von Schmuckstücken versehen werden, die durch unterschiedliche geometrische Objekte dargestellt werden. Es gibt Kugeln, Päckchen in der Form von Würfeln und Zuckerringe in Form von Toren.</p> <p>Ein Prototyp, der bereits mit Kugeln geschmückt werden kann, kann zur Verfügung gestellt werden. Da alle Schmuckstücke über die Funktion des Auf- und Abschmückens verfügen sollen, liegt es nahe, dass entsprechende Methoden in einer gemeinsamen Oberklasse realisiert werden.</p> <p><i>Materialien:</i></p> <p>Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.1-Wiederholung</p>
---	---	---

Modellierung und Implementierung von Anwendungen mit dynamischen, linearen Datenstrukturen (Unterrichtsvorhaben Q1-II)

Leitfrage: *Wie können beliebig viele linear angeordnete Daten im Anwendungskontext verwaltet werden?*

Vorhabenbezogene Konkretisierung:

Nach Analyse einer Problemstellung in einem geeigneten Anwendungskontext, in dem Daten nach dem First-In-First-Out-Prinzip verwaltet werden, werden der Aufbau von Schlangen am Beispiel dargestellt und die Operationen der Klasse Queue erläutert. Anschließend werden für die Anwendung notwendige Klassen modelliert und implementiert. Eine Klasse für eine den Anforderungen der Anwendung entsprechende Oberfläche sowie die Klasse Queue wird dabei von der Lehrkraft vorgegeben. Anschließend wird die Anwendung modifiziert, um den Umgang mit der Datenstruktur zu üben. Anhand einer Anwendung, in der Daten nach dem Last-In-First-Out-Prinzip verwaltet werden, werden Unterschiede zwischen den Datenstrukturen Schlange und Stapel erarbeitet. Um einfacher an Objekte zu gelangen, die zwischen anderen gespeichert sind, wird die Klasse List eingeführt und in einem Anwendungskontext verwendet. In mindestens einem weiteren Anwendungskontext wird die Verwaltung von Daten in Schlangen, Stapeln oder Listen vertieft. Modellierungen werden dabei in Entwurfs- und Implementationsdiagrammen dargestellt.

Zeitbedarf: 20 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Die Datenstruktur Schlange im Anwendungskontext unter Nutzung der Klasse Queue</p> <p>(a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen</p> <p>(b) Erarbeitung der Funktionalität der Klasse Queue</p> <p>(c) Modellierung und Implementierung der Anwendung unter Verwendung eines oder</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (A), • analysieren und erläutern Algorithmen und Programme (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare 	<p><i>Beispiel:</i> Patientenwarteschlange (jeder kennt seinen Nachfolger bzw. alternativ: seinen Vorgänger)</p> <p>Sobald ein Patient in einer Arztpraxis eintrifft, werden sein Name und seine Krankenkasse erfasst. Die Verwaltung der Patientenwarteschlange geschieht über eine Klasse, die hier als Wartezimmer bezeichnet wird. Wesentliche Operationen sind das „Hinzufügen“ eines Patienten und das „Entfernen“ eines Patienten, wenn er zur Behandlung gerufen wird.</p> <p>Die Simulationsanwendung stellt eine GUI zur Verfügung, legt ein</p>

<p>mehrerer Objekte der Klasse Queue</p>	<p>Datensammlungen zu (M),</p> <ul style="list-style-type: none"> • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • modifizieren Algorithmen und Programme (I), • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • testen Programme systematisch anhand von Beispielen (I), • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D). 	<p>Wartezimmer an und steuert die Abläufe. Wesentlicher Aspekt des Projektes ist die Modellierung des Wartezimmers mit Hilfe der Klasse Queue.</p> <p>Anschließend wird der Funktionsumfang der Anwendung erweitert: Patienten können sich zusätzlich in die Warteschlange zum Blutdruckmessen einreihen. Objekte werden von zwei Schlangen verwaltet.</p> <p><i>Materialien:</i></p> <p>Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben – Warteschlange</p>
<p>2. Die Datenstruktur Stapel im Anwendungskontext unter Nutzung der Klasse Stack</p> <p>(a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen</p> <p>(b) Erarbeitung der Funktionalität der Klasse Stack</p> <p>(c) Modellierung und Implementierung der Anwendung unter Verwendung eines oder mehrerer Objekte der Klasse Stack</p>	<p>und zur Analyse von Programmen (I),</p>	<p><i>Beispiel:</i> Heftstapel</p> <p>In einem Heftstapel soll das Heft einer Schülerin gefunden werden.</p> <p>oder</p> <p><i>Beispiel:</i> Kisten stapeln</p> <p>In einem Stapel nummerierter Kisten soll eine bestimmte Kiste gefunden und an einen Kunden geliefert werden. Dazu müssen Kisten auf verschiedene Stapel gestapelt und wieder zurückgestellt werden.</p>
<p>3. Die Datenstruktur lineare Liste im Anwendungskontext unter Nutzung der Klasse List</p> <p>(a) Erarbeitung der Vorteile der Klasse List</p>		<p><i>Beispiel:</i> Abfahrtslauf</p> <p>Bei einem Abfahrtslauf kommen die Skifahrer nacheinander an und werden nach ihrer Zeit in eine Rangliste eingeordnet. Diese Rangliste wird in einer Anzeige ausgegeben. Ankommende</p>

<p>im Gegensatz zu den bereits bekannten linearen Strukturen</p> <p>(b) Modellierung und Implementierung einer kontextbezogenen Anwendung unter Verwendung der Klasse List.</p>		<p>Abfahrer müssen an jeder Stelle der Struktur, nicht nur am Ende oder Anfang eingefügt werden können.</p> <p><i>Materialien:</i></p> <p>Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben - Listen</p>
<p>4. Vertiefung - Anwendungen von Listen, Stapeln oder Schlangen in mindestens einem weiteren Kontext</p>		<p><i>Beispiel:</i> Skispringen</p> <p>Ein Skispringen hat folgenden Ablauf: Nach dem Sprung erhält der Springer eine Punktzahl und wird nach dieser Punktzahl in eine Rangliste eingeordnet. Die besten 30 Springer qualifizieren sich für den zweiten Durchgang. Sie starten in umgekehrter Reihenfolge gegenüber der Platzierung auf der Rangliste. Nach dem Sprung erhält der Springer wiederum eine Punktzahl und wird nach der Gesamtpunktzahl aus beiden Durchgängen in die endgültige Rangliste eingeordnet.</p> <p><i>Beispiel:</i> Terme in Postfix-Notation</p> <p>Die sog. UPN (<i>Umgekehrt-Polnische-Notation</i>) bzw. <i>Postfix-Notation</i> eines Terms setzt den Operator hinter die Operanden. Um einen Term aus der gewohnten Infixschreibweise in einen Term in UPN umzuwandeln oder um den Wert des Terms zu berechnen, kann ein Stack verwendet werden.</p> <p><i>Beispiel:</i> Rangierbahnhof</p> <p>Auf einem Güterbahnhof gibt es drei Gleise, die nur zu einer Seite offen sind. Wagons können also von einer Seite auf das Gleis fahren und nur rückwärts wieder hinausfahren. Die Wagons tragen</p>

Nummern, wobei die Nummer jedoch erst eingesehen werden kann, wenn der Wagon der vorderste an der offenen Gleisseite ist. (Zwischen den Wagons herumzuturnen, um die anderen Wagonnummern zu lesen, wäre zu gefährlich.) Zunächst stehen alle Wagons unsortiert auf einem Gleis. Ziel ist es, alle Wagons in ein anderes Gleis zu fahren, so dass dort die Nummern der Wagons vom Gleisende aus aufsteigend in richtiger Reihenfolge sind. Zusätzlich zu diesen beiden Gleisen gibt es ein Abstellgleis, das zum Rangieren benutzt werden kann.

Beispiel: Autos an einer Ampel zur Zufahrtsregelung

Es soll eine Ampel zur Zufahrtsregelung in Java simuliert werden. An einem geradlinigen, senkrecht von unten nach oben verlaufenden Straßenstück, das von Autos nur einspurig in eine Richtung befahren werden kann, ist ein Haltepunkt markiert, an dem die Ampel steht. Bei einem Klick auf eine Schaltfläche mit der Aufschrift „Heranfahen“ soll ein neues Auto an den Haltepunkt heranfahren bzw. bis an das letzte Auto, das vor dem Haltepunkt wartet. Grünphasen der Ampel werden durch einen Klick auf eine Schaltfläche mit der Aufschrift „Weiterfahren“ simuliert. In jeder Grünphase darf jeweils nur ein Auto weiterfahren. Die anderen Autos rücken nach.

Materialien:

Ergänzungsmaterialien zum Lehrplannavigator

		Unterrichtsvorhaben – Anwendungen für lineare Datenstrukturen
--	--	---

Suchen und Sortieren auf linearen Datenstrukturen (Unterrichtsvorhaben Q1-III)

Leitfrage: *Wie kann man gespeicherte Informationen günstig (wieder-)finden?*

Vorhabenbezogene Konkretisierung:

In einem Anwendungskontext werden zunächst Informationen in einer linearen Liste bzw. einem Feld gesucht. Hierzu werden Verfahren entwickelt und implementiert bzw. analysiert und erläutert, wobei neben einem iterativen auch ein rekursives Verfahren thematisiert wird und mindestens ein Verfahren selbst entwickelt und implementiert wird. Die verschiedenen Verfahren werden hinsichtlich Speicherbedarf und Zahl der Vergleichsoperationen miteinander verglichen.

Anschließend werden Sortierverfahren entwickelt und implementiert (ebenfalls für lineare Listen und Felder). Hierbei soll auch ein rekursives Sortierverfahren entwickelt werden. Die Implementationen von Quicksort sowie dem Sortieren durch Einfügen werden analysiert und erläutert. Falls diese Verfahren vorher schon entdeckt wurden, sollen sie hier wiedererkannt werden. Die rekursive Abarbeitung eines Methodenaufrufs von Quicksort wird grafisch dargestellt.

Abschließend werden verschiedene Sortierverfahren hinsichtlich der Anzahl der benötigten Vergleichsoperationen und des Speicherbedarfs beurteilt.

Zeitbedarf: 16 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Suchen von Daten in Listen und Arrays</p> <p>(a) Lineare Suche in Listen und in Arrays</p> <p>(b) Binäre Suche in Arrays als Beispiel für rekursives Problemlösen</p> <p>(c) Untersuchung der beiden Suchverfahren hinsichtlich ihrer Effizienz (Laufzeitverhalten, Speicherbedarf)</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> analysieren und erläutern Algorithmen und Programme (A), beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), beurteilen die Effizienz von Algorithmen unter Berücksichtigung des Speicherbedarfs und der Zahl der Operationen (A), entwickeln iterative und rekursive Algorithmen unter 	<p><i>Beispiel:</i> Karteiverwaltung</p> <p>Für ein Adressverwaltungsprogramm soll eine Methode zum Suchen einer Adresse geschrieben werden.</p> <p>oder</p> <p><i>Beispiel:</i> Bundesjugendspiele</p> <p>Die Teilnehmer an Bundesjugendspielen nehmen an drei Disziplinen teil und erreichen dort Punktzahlen. Diese werden in einer Wettkampfkarte eingetragen und an das Wettkampfbüro gegeben. Zur Vereinfachung</p>

	<p>Nutzung der Strategien „Modularisierung“ und „Teilen und Herrschen“ (M),</p> <ul style="list-style-type: none"> • modifizieren Algorithmen und Programme (I), • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I), • implementieren und erläutern iterative und rekursive Such- und Sortierverfahren (I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • testen Programme systematisch anhand von Beispielen (I), • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D). 	<p>sollte sich das Modell auf die drei Disziplinen „Lauf“, „Sprung“ und „Wurf“ beschränken.</p> <p>Im Wettkampfbüro wird das Ergebnis erstellt. Das Programm soll dafür zunächst den Besten einer Disziplin herausuchen können und später das gesamte Ergebnis nach gewissen Kriterien sortieren können.</p> <p><i>Materialien:</i></p> <p>Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.3 - Suchen und Sortieren</p>
<p>2. Sortieren in Listen und Arrays - Entwicklung und Implementierung von iterativen und rekursiven Sortierverfahren</p> <p>(a) Entwicklung und Implementierung eines einfachen Sortierverfahrens für eine Liste</p> <p>(b) Implementierung eines einfachen Sortierverfahrens für ein Feld</p> <p>(c) Entwicklung eines rekursiven Sortierverfahrens für ein Feld (z.B. Sortieren durch Mischen)</p>		<p><i>Beispiel:</i> Karteiverwaltung</p> <p>(s.o.)</p> <p>oder</p> <p><i>Beispiel:</i> Bundesjugendspiele</p> <p>(s.o.)</p> <p><i>Materialien:</i></p> <p>(s.o.)</p>
<p>3. Untersuchung der Effizienz der Sortierverfahren „Sortieren durch direktes Einfügen“ und „Quicksort“ auf linearen Listen</p> <p>(a) Grafische Veranschaulichung der Sortierverfahren</p>		<p><i>Beispiel:</i> Karteiverwaltung</p> <p>(s.o.)</p> <p>oder</p> <p><i>Beispiel:</i> Bundesjugendspiele</p> <p>(s.o.)</p> <p><i>Materialien</i></p>

<p>(b) Untersuchung der Anzahl der Vergleichsoperationen und des Speicherbedarf bei beiden Sortierverfahren</p>		<p><i>s.o.</i></p>
<p>(c) Beurteilung der Effizienz der beiden Sortierverfahren</p>		

Modellierung und Nutzung von relationalen Datenbanken in Anwendungskontexten (Unterrichtsvorhaben Q1-IV)

Leitfragen: *Wie können Fragestellungen mit Hilfe einer Datenbank beantwortet werden? Wie entwickelt man selbst eine Datenbank für einen Anwendungskontext?*

Vorhabenbezogene Konkretisierung:

Ausgehend von einer vorhandenen Datenbank entwickeln Schülerinnen und Schüler für sie relevante Fragestellungen, die mit dem vorhandenen Datenbestand beantwortet werden sollen. Zur Beantwortung dieser Fragestellungen wird die vorgegebene Datenbank von den Schülerinnen und Schülern analysiert und die notwendigen Grundbegriffe für Datenbanksysteme sowie die erforderlichen SQL-Abfragen werden erarbeitet.

In anderen Anwendungskontexten müssen Datenbanken erst noch entwickelt werden, um Daten zu speichern und Informationen für die Beantwortung von möglicherweise auftretenden Fragen zur Verfügung zu stellen. Dafür ermitteln Schülerinnen und Schüler in den Anwendungssituationen Entitäten, zugehörige Attribute, Relationen und Kardinalitäten und stellen diese in Entity-Relationship-Modellen dar. Entity-Relationship-Modelle werden interpretiert und erläutert, modifiziert und in Datenbankschemata überführt. Mit Hilfe von SQL-Anweisungen können anschließend im Kontext relevante Informationen aus der Datenbank extrahiert werden.

Ein Entity-Relationship-Diagramm kann auch verwendet werden, um die Entitäten inklusive ihrer Attribute und Relationen in einem vorgegebenen Datenbankschema darzustellen.

An einem Beispiel wird verdeutlicht, dass in Datenbanken Redundanzen unerwünscht sind und Konsistenz gewährleistet sein sollte. Die 1. bis 3. Normalform wird als Gütekriterium für Datenbankentwürfe eingeführt. Datenbankschemata werden hinsichtlich der 1. bis 3. Normalform untersucht und (soweit nötig) normalisiert.

Zeitbedarf: 20 Stunden

Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Nutzung von relationalen Datenbanken (a) Aufbau von Datenbanken und Grundbegriffe <ul style="list-style-type: none"> Entwicklung von Fragestellungen zur vorhandenen Datenbank 	Die Schülerinnen und Schüler <ul style="list-style-type: none"> erläutern die Eigenschaften und den Aufbau von Datenbanksystemen unter dem Aspekt der sicheren Nutzung (A), analysieren und erläutern die Syntax 	<i>Beispiel:</i> VideoCenter VideoCenter ist die Simulation einer Online-Videothek für den Informatik-Unterricht mit Webfrontends zur Verwaltung der Kunden, der Videos und der Ausleihe. Außerdem ist es möglich direkt SQL-Abfragen einzugeben. Es ist auch möglich, die Datenbank herunter zu laden und lokal zu installieren. Unter

<ul style="list-style-type: none"> Analyse der Struktur der vorgegebenen Datenbank und Erarbeitung der Begriffe Tabelle, Attribut, Datensatz, Datentyp, Primärschlüssel, Fremdschlüssel, Datenbankschema 	<ul style="list-style-type: none"> und Semantik einer Datenbankabfrage (A), analysieren und erläutern eine Datenbankmodellierung (A), erläutern die Eigenschaften normalisierter Datenbankschemata (A), bestimmen Primär- und Sekundärschlüssel (M), ermitteln für anwendungsbezogene Problemstellungen Entitäten, zugehörige Attribute, Relationen und Kardinalitäten (M), modifizieren eine Datenbankmodellierung (M), modellieren zu einem Entity-Relationship-Diagramm ein relationales Datenbankschema (M), bestimmen Primär- und Sekundärschlüssel (M), überführen Datenbankschemata in vorgegebene Normalformen (M), verwenden die Syntax und Semantik einer Datenbankabfrage, um Informationen aus 	<p>http://dokumentation.videocenter.schule.de/old/video/index.html</p> <p>(abgerufen: 30. 03. 2014) findet man den Link zu dem VideoCenter-System sowie nähere Informationen. Lesenswert ist auch die dort verlinkte „Dokumentation der Fallstudie“ mit didaktischem Material, welches alternativ bzw. ergänzend zu der im Folgenden beschriebenen Durchführung verwendet werden kann.</p>
<p>(b) SQL-Abfragen</p> <ul style="list-style-type: none"> Analyse vorgegebener SQL-Abfragen und Erarbeitung der Sprachelemente von SQL (SELECT (DISTINCT) ...FROM, WHERE, AND, OR, NOT) auf einer Tabelle Analyse und Erarbeitung von SQL-Abfragen auf einer und mehrerer Tabelle zur Beantwortung der Fragestellungen (JOIN, UNION, AS, GROUP BY, ORDER BY, ASC, DESC, COUNT, MAX, MIN, SUM, Arithmetische Operatoren: +, -, *, /, (...), Vergleichsoperatoren: =, <>, >, <, >=, <=, LIKE, BETWEEN, IN, IS NULL) 	<ul style="list-style-type: none"> Bestimmen Primär- und Sekundärschlüssel (M), ermitteln für anwendungsbezogene Problemstellungen Entitäten, zugehörige Attribute, Relationen und Kardinalitäten (M), modifizieren eine Datenbankmodellierung (M), modellieren zu einem Entity-Relationship-Diagramm ein relationales Datenbankschema (M), bestimmen Primär- und Sekundärschlüssel (M), überführen Datenbankschemata in vorgegebene Normalformen (M), verwenden die Syntax und Semantik einer Datenbankabfrage, um Informationen aus 	<p><i>Beispiel: Schulbuchausleihe</i></p> <p>Unter www.brd.nrw.de/lerntreffs/informatik/structure/material/sek2/datenbanken.php (abgerufen: 30. 03. 2014) wird eine Datenbank zur Verfügung gestellt, die Daten einer Schulbuch-Ausleihe enthält (über 1000 Entleiher, 200 Bücher mit mehreren tausend Exemplaren und viele Ausleihvorgänge). Die Datenbank kann in OpenOffice eingebunden werden.</p>
<p>(c) Vertiefung an einem weiteren Datenbankbeispiel</p>	<p>und Semantik einer Datenbankabfrage (A),</p>	

<p>2. Modellierung von relationalen Datenbanken</p> <p>(a) Entity-Relationship-Diagramm</p> <ul style="list-style-type: none"> • Ermittlung von Entitäten, zugehörigen Attributen, Relationen und Kardinalitäten in Anwendungssituationen und Modellierung eines Datenbankentwurfs in Form eines Entity-Relationship-Diagramms • Erläuterung und Modifizierung einer Datenbankmodellierung <p>(b) Entwicklung einer Datenbank aus einem Datenbankentwurf</p> <ul style="list-style-type: none"> • Modellierung eines relationalen Datenbankschemas zu einem Entity-Relationship-Diagramm inklusive der Bestimmung von Primär- und Sekundärschlüsseln <p>(c) Redundanz, Konsistenz und Normalformen</p> <ul style="list-style-type: none"> • Untersuchung einer Datenbank hinsichtlich Konsistenz und Redundanz in einer Anwendungssituation 	<p>Datenbanksystem zu extrahieren (I),</p> <ul style="list-style-type: none"> • ermitteln Ergebnisse von Datenbankabfragen über mehrere verknüpfte Tabellen (D), • stellen Entitäten mit ihren Attributen und die Beziehungen zwischen Entitäten in einem Entity-Relationship-Diagramm grafisch dar (D), • überprüfen Datenbankschemata auf vorgegebene Normalisierungseigenschaften (D). 	<p><i>Beispiel: Fahrradverleih</i></p> <p>Der Fahrradverleih <i>BTR (BikesToRent)</i> verleiht unterschiedliche Typen von Fahrrädern diverser Firmen an seine Kunden. Die Kunden sind bei <i>BTR</i> registriert (Name, Adresse, Telefon). <i>BTR</i> kennt von den Fahrradfirmen den Namen und die Telefonnummer. Kunden von <i>BTR</i> können CityBikes, Treckingräder und Mountainbikes ausleihen.</p> <p><i>Beispiel: Reederei</i></p> <p>Die Datenverwaltung einer Reederei soll in einem Datenbanksystem umgesetzt werden. Ausgehend von der Modellierung soll mit Hilfe eines ER-Modells und eines Datenbankschemas dieser erste Entwurf normalisiert und in einem Datenbanksystem umgesetzt werden. Es schließen sich diverse SQL-Abfragen an, wobei auf die Relationenalgebra eingegangen wird.</p> <p><i>Beispiel: Buchungssystem</i></p> <p>In dem Online-Buchungssystem einer Schule können die Lehrer Medienräume, Beamer, Laptops, Kameras, usw. für einen bestimmten Zeitpunkt buchen, der durch Datum und die Schulstunde festgelegt ist.</p> <p>Dazu ist die Datenbank zu modellieren, ggf. zu normalisieren und im Datenbanksystem umzusetzen. Weiter sollen sinnvolle Abfragen entwickelt werden.</p> <p>Unter http://mrbs.sourceforge.net (abgerufen: 30.03. 2014) findet man ein freies Online-Buchungssystem inklusive Demo, an Hand derer man erläutern kann, worum es in dem Projekt geht.</p>
--	--	--

<ul style="list-style-type: none">• Überprüfung von Datenbankschemata hinsichtlich der 1. bis 3. Normalform und Normalisierung (um Redundanzen zu vermeiden und Konsistenz zu gewährleisten)		<p><i>Beispiel: Schulverwaltung</i></p> <p>In einer Software werden die Schulhalbjahre, Jahrgangsstufen, Kurse, Klassen, Schüler, Lehrer und Noten einer Schule verwaltet. Man kann dann ablesen, dass z.B. Schüler X von Lehrer Y im 2. Halbjahr des Schuljahrs 2011/2012 in der Jahrgangsstufe 9 im Differenzierungsbereich im Fach Informatik die Note „sehr gut“ erhalten hat. Dazu ist die Datenbank zu modellieren, ggf. zu normalisieren und im Datenbanksystem umzusetzen. Weiter sollen sinnvolle Abfragen entwickelt werden und das Thema Datenschutz besprochen werden.</p>
--	--	--

Sicherheit und Datenschutz in Netzstrukturen (Unterrichtsvorhaben Q1-V)

Leitfragen: *Wie werden Daten in Netzwerken übermittelt? Was sollte man in Bezug auf die Sicherheit beachten?*

Vorhabenbezogene Konkretisierung:

Anschließend an das vorhergehende Unterrichtsvorhaben zum Thema Datenbanken werden der Datenbankzugriff aus dem Netz, Topologien von Netzwerken, eine Client-Server-Struktur, das TCP/IP-Schichtenmodell sowie Sicherheitsaspekte beim Zugriff auf Datenbanken und verschiedene symmetrische und asymmetrische kryptografische Verfahren analysiert und erläutert. Fallbeispiele zur Datenschutzproblematik und zum Urheberrecht runden das Unterrichtsvorhaben ab.

Zeitbedarf: 10 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Daten in Netzwerken und Sicherheitsaspekte in Netzen sowie beim Zugriff auf Datenbanken</p> <p>(a) Beschreibung eines Datenbankzugriffs im Netz anhand eines Anwendungskontextes und einer Client-Server-Struktur zur Klärung der Funktionsweise eines Datenbankzugriffs</p> <p>(b) Netztopologien als Grundlage von Client-Server-Strukturen und TCP/IP-Schichtenmodell als Beispiel für eine Paketübermittlung in einem Netz</p> <p>(c) Vertraulichkeit, Integrität, Authentizität in Netzwerken sowie symmetrische und asymmetrische kryptografische Verfahren (Cäsar-, Vigenère-, RSA-</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • beschreiben und erläutern Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken (A), • analysieren und erläutern Eigenschaften und Einsatzbereiche symmetrischer und asymmetrischer Verschlüsselungsverfahren (A), • untersuchen und bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen, die Sicherheit von Informatiksystemen sowie die Einhaltung der Datenschutzbestimmungen und des Urheberrechts (A), • untersuchen und bewerten Problemlagen, die sich aus dem Einsatz von 	<p><i>Materialien:</i></p> <p>Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben, Verschlüsselung Q1.5 - Zugriff auf Daten in Netzwerken</p>

<p>Verfahren) als Methoden Daten im Netz verschlüsselt zu übertragen</p>	<p>Informatiksystemen ergeben, hinsichtlich rechtlicher Vorgaben, ethischer Aspekte und gesellschaftlicher Werte unter Berücksichtigung unterschiedlicher Interessenlagen (A),</p> <ul style="list-style-type: none"> • nutzen bereitgestellte Informatiksysteme und das Internet reflektiert zum Erschließen, zur Aufbereitung und Präsentation fachlicher Inhalte (D). 	<p><i>Materialien:</i></p> <p><i>Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1 5 - Datenschutz beim Videocenter, Materialblatt- Datenschutzgesetz</i></p>
<p>2. Fallbeispiele zur Datenschutzproblematik und zum Urheberrecht</p>		