

Schulinternes Curriculum Q1/Q2

I. Verbindliche Festlegungen der Fachkonferenz:

Die Fachkonferenz des Hardtberg-Gymnasiums hat Themen, Leitfragen und die Ausführungen dieses Curriculums verbindlich vereinbart, ebenso die ausgewiesenen Kompetenzen. Die Sequenzierung der Unterrichtsvorhaben ist modular aufgebaut. Alle Mitglieder der Fachkonferenz haben sich darauf verständigt, in ihrem Unterricht Lerngelegenheiten anzubieten, so dass Schülerinnen und Schüler diese Kompetenzen im Rahmen der festgelegten Unterrichtssequenzen erwerben oder vertiefen können. Die Unterrichtssequenzen sind als Module zu verstehen, deren Ausprägung und Reihenfolge je nach Interesse, individuellen Bedürfnissen sowie möglichen Bezug auf aktuellen Ereignissen variabel sind.

In der Qualifikationsphase werden die Unterrichtsvorhaben unter Berücksichtigung des Kernlehrplanes Informatik in NRW sowie der Vorgaben für das Zentralabitur konkretisiert. Diese sind zu beziehen unter der Adresse:

http://www.standardsicherung.schulministerium.nrw.de/lehrplaene/upload/klp_SII/if/GOSt_Informatik_Endfassung.pdf

Materialien:

Verbindlich wird von allen Schülerinnen und Schüler benötigt:

- ein USB-Stick / eine portable Festplatte (mind. 256MB)
- „Informatik 3“ Schönigh Verlag ISBN 9778-3-14-037125-4 (durch die Schulbuchleihe bei den Schülerinnen und Schüler vorhanden)

Für den Unterricht der EF am HBG benötigte Software:

Ist auf den Rechnern am HBG installiert und soll als portable-Version auf den USB-Sticks / portablen Festplatten installiert sein.

- BlueJ (Installation im Rahmen von Greenfoot)
- Greenfoot (<http://www.greenfoot.org/download> - empfohlen wird die „Stand-Alone-Version“)
- GLOOP (<http://www.standardsicherung.schulministerium.nrw.de/materialdatenbank/nutzersicht/materialeintrag.php?matId=4302> sowie <http://www.standardsicherung.schulministerium.nrw.de/materialdatenbank/nutzersicht/materialeintrag.php?matId=4293>)

- JavaKara (<http://swisseduc.ch/informatik/karatojava/javakara/> - empfohlen wird, nach Greenfoot zu installieren, als Java-Pfad kann dann der USB-Stick gewählt werden)
- PSPad

Die folgenden Kompetenzen aus dem Bereich *Kommunizieren und Kooperieren* werden in allen Unterrichtsvorhaben der Einführungsphase vertieft und sollen aus Gründen der Lesbarkeit nicht in jedem Unterrichtsvorhaben separat aufgeführt werden:

Die Schülerinnen und Schüler

- verwenden Fachausdrücke bei der Kommunikation über informatische Sachverhalte (K),
- präsentieren Arbeitsabläufe und -ergebnisse (K),
- kommunizieren und kooperieren in Gruppen und in Partnerarbeit (K),
- nutzen das verfügbare Informatiksystem zur strukturierten Verwaltung und gemeinsamen Verwendung von Daten unter Berücksichtigung der Rechteverwaltung (K).

II) *Verlaufsplan Qualifikationsphase (Q1 und Q2) - GRUNDKURS*

Qualifikationsphase 1	
<p><u>Unterrichtsvorhaben Q1-I</u></p> <p>Thema: <i>Wiederholung und Ergänzung der Algorithmen durch rekursive Problemstellungen</i></p> <p>Zentrale Kompetenzen:</p> <ul style="list-style-type: none">• Argumentieren• Implementieren• Darstellen und Interpretieren• Kommunizieren und Kooperieren <p>Inhaltsfelder:</p> <ul style="list-style-type: none">• Algorithmen• Daten und Strukturen• Formale Sprachen und Automaten• Informatiksysteme <p>Inhaltliche Schwerpunkte:</p> <ul style="list-style-type: none">• Analyse, Entwurf und Implementierung von rekursiven Algorithmen• Syntax und Semantik einer Programmiersprache• Nutzung von Informatiksystemen	<p><u>Unterrichtsvorhaben Q1-II</u></p> <p>Thema: <i>Modellierung und Implementierung von Anwendungen mit dynamischen, linearen Datenstrukturen</i></p> <p>Zentrale Kompetenzen:</p> <ul style="list-style-type: none">• Argumentieren• Modellieren• Implementieren• Darstellen und Interpretieren• Kommunizieren und Kooperieren <p>Inhaltsfelder:</p> <ul style="list-style-type: none">• Daten und ihre Strukturierung• Algorithmen• Formale Sprachen und Automaten <p>Inhaltliche Schwerpunkte:</p> <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

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

Unterrichtsvorhaben Q1-IV

Thema:

Modellierung und Implementierung von Anwendungen mit dynamischen, nichtlinearen Datenstrukturen

Zentrale Kompetenzen:

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

Inhaltsfelder:

- Daten und ihre Strukturierung
- Algorithmen
- Formale Sprachen und Automaten

Inhaltliche Schwerpunkte:

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

Qualifikationsphase 1

Unterrichtsvorhaben Q1-V

Thema:

Endliche Automaten und formale Sprachen

Zentrale Kompetenzen:

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

Inhaltsfelder:

- Endliche Automaten und formale Sprachen

Inhaltliche Schwerpunkte:

- Endliche Automaten
- Grammatiken regulärer Sprachen
- Möglichkeiten und Grenzen von Automaten und formalen Sprachen

Unterrichtsvorhaben Q1-VI

Thema:

Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit

Zentrale Kompetenzen:

- Argumentieren
- Kommunizieren und Kooperieren

Inhaltsfelder:

- Informatiksysteme
- Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:

- Einzelrechner und Rechnernetzwerke
- Grenzen der Automatisierung

Qualifikationsphase 2

Unterrichtsvorhaben Q2-I

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

Unterrichtsvorhaben Q2-II

Thema:

Projekt - Modellierung, Entwicklung und Implementierung eines realen Auftrages

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:

- eigenständiges Arbeiten
- Erleben von Informatik als lebensnahes Fach

Qualifikationsphase 2

Unterrichtsvorhaben Q2-III

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
- Netzwerktechniken
 - Protokolle
 - OSI-Modell
 - Adressierung
 - Client-Server-Strukturen
 - Chat-Programme
 - Multithreading

Unterrichtsvorhaben Q2-IV

Thema:

Wiederholung und Vertiefung ausgewählter Kompetenzen und Inhalte des ersten Jahres der Qualifikationsphase

Zentrale Kompetenzen:

- Darstellen und Interpretieren
- Kommunizieren und Kooperieren

Inhaltliche Schwerpunkte:

- alle Themen des Abiturs

III) Konkretisierte Unterrichtsvorhaben

Im Folgenden sollen die oben aufgeführten Unterrichtsvorhaben konkretisiert werden.

Unterrichtsvorhaben Q1-I: Wiederholung und Ergänzung der Algorithmen durch rekursive Problemstellungen

Leitfragen: *Wie kann man unseren Objekten etwas beibringen? Welche Möglichkeiten zur Strukturierung habe ich? Wie kann man bestimmte, sich in sich wiederholende, mathematische Probleme effektiv lösen?*

Vorhabenbezogene Konkretisierung:

Diese Reihe dient der Wiederholung und Vertiefung des Themas Algorithmisierung. So werden verschiedene Methoden von den Schülerinnen und Schüler entwickelt und implementiert.

Beispielhaft beschäftigt sich dieses Unterrichtsvorhaben mit der Erarbeitung eines besonderen Taschenrechners. Es werden mehrere Funktionen wie die Fakultät, Modulorechnung sowie Fibonaccizahlen vorgestellt, von den Schülern untersucht und Algorithmen zur automatischen Berechnung entwickelt.

Anhand mehrerer Umsetzungsideen wird die Idee der Rekursion und der Unterschied zur Iteration erarbeitet, der allgemeine Aufbau einer rekursiven Funktion entwickelt und in mehreren Kontexten angewandt.

Im Anschluss folgen weitere klassische rekursive Problemstellungen, die gelöst werden.

Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien
<p>1. Wiederholung und Erweiterung der Kompetenz zur Algorithmisierung verschiedener Probleme</p> <p>(a) Analyse der Problemstellung (b) Entwicklung von Algorithmen (c) Analyse der Modellierung und Algorithmen (Flussdiagramm) (d) Implementierung der Methoden (e) Entwicklung von rekursiven Programmen</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • analysieren und erläutern Problemstellungen, • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen, • modellieren Methoden, • entwickeln Algorithmen zur Lösung der Probleme, • implementieren Methoden in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken, • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen, • 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 Methoden grafisch durch ein Fluss- oder Struktogramm dar. 	<p>Beispiel: <i>Der besondere Taschenrechner.</i> Es werden einem virtuellen Taschenrechner neue Funktionen beigebracht. So soll er z.B. Fakultäten und Fibonaccizahlen bestimmen. Nach der Analyse in Gruppen werden rekursive Lösungsansätze besprochen, dargestellt und vertieft. Diese werden an weiteren Aufgaben eingeübt.</p>

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

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 einem projektartigen Unterrichtsabschnitt wird die Verwaltung von Daten in Schlangen, Stapeln oder Listen vertieft. Modellierungen werden dabei in Entwurfs- und Implementationsdiagrammen dargestellt.

Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien
<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 mehrerer Objekte der Klasse Queue</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen, • analysieren und erläutern Algorithmen und Programme, • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen, • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu, • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen, • modifizieren Algorithmen und Programme, • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen, • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur 	<p>Beispiel: <i>Patientenwarteschlange</i> (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 bei Ankunft und das „Entfernen“ eines Patienten, wenn er zur Behandlung gerufen wird.</p> <p>Die Simulationsanwendung stellt eine GUI zur Verfügung, legt ein 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>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>	<ul style="list-style-type: none"> • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur 	<p>Beispiel: <i>Heftstapel</i></p> <p>In einem Heftstapel soll das Heft einer Schülerin gefunden werden.</p> <p>oder</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>Analyse von Programmen,</p> <ul style="list-style-type: none"> • interpretieren Fehlermeldungen und korrigieren den Quellcode, • testen Programme systematisch anhand von Beispielen, • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau. 	<p>Beispiel: Kartenstapel In einem Spiel „MauMau für Bluffer“ sollen Karten verdeckt auf einen Kartenstapel gelegt werden und können nachher überprüft werden.</p>
<p>3. Die Datenstruktur lineare Liste im Anwendungskontext unter Nutzung der Klasse List</p> <p>(a) Erarbeitung der Vorteile der Klasse List im Gegensatz zu den bereits bekannten linearen Strukturen</p> <p>(b) Modellierung und Implementierung einer kontextbezogenen Anwendung unter Verwendung der Klasse List.</p>		<p>Beispiel: Die Kartenhand bei einem Kartenspiel Die Karten der Spieler auf der Hand müssen verwaltet werden. Es können Karten aufgenommen werden und von beliebiger Stelle entfernt werden.</p>
<p>4. Vertiefung - Anwendungen von Listen, Stapeln oder Schlangen in mindestens einem weiteren Kontext</p>		<p>Beispiel: Skispringen 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>Beispiel: Terme in Postfix-Notation 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</p>

um den Wert des Terms zu berechnen, kann ein **Stack** verwendet werden.

Beispiel: Rangierbahnhof

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 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 „Heranfahren“ 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.

Unterrichtsvorhaben Q1-III: Suchen und Sortieren auf linearen Datenstrukturen

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

Vorhabenbezogene Konkretisierung:

In einem Anwendungskontext werden zunächst Informationen in einer linearen Liste bzw. einem Feld geordnet. 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 Suchverfahren entwickelt und implementiert (ebenfalls für lineare Listen oder Felder). Hierbei soll auch ein rekursives Sortierverfahren besprochen 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.

Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien
<p>1. 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 oder:</p> <p>(b) Implementierung eines einfachen Sortierverfahrens für ein Feld</p> <p>(c) Entwicklung eines rekursiven Sortierverfahrens für ein Feld oder eine Liste (z.B. Sortieren durch Mischen)</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 Nutzung der Strategien „Modularisierung“ und „Teilen und Herrschen“ (M), • 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), 	<p>Beispiel: Karteiverwaltung Für das bereits entwickelte Spiel müssen die Karten auf der Hand sortiert werden.</p> <p>oder</p> <p>Beispiel: Bundesjugendspiele 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 soll sich das Modell auf die drei Disziplinen „Lauf“, „Sprung“ und „Wurf“ beschränken. 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>2. Suchen von Daten in Listen und Arrays</p> <p>(a) Lineare Suche in Listen und 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>	<ul style="list-style-type: none"> • 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 	<p>Beispiel: Karteiverwaltung (s.o.)</p> <p>oder</p> <p>Beispiel: Bundesjugendspiele (s.o.)</p>

<p>3. Untersuchung der Effizienz der Sortierverfahren „Sortieren durch direktes Einfügen“ und „Quicksort“ auf linearen Listen</p> <ul style="list-style-type: none">(a) Grafische Veranschaulichung der Sortierverfahren(b) Untersuchung der Anzahl der Vergleichsoperationen und des Speicherbedarf bei beiden Sortierverfahren(c) Beurteilung der Effizienz der beiden Sortierverfahren	<p>umgangssprachlich und grafisch dar (D).</p>	<p>Beispiel: Karteiverwaltung (s.o.)</p> <p>oder</p> <p>Beispiel: Bundesjugendspiele (s.o.)</p>
--	--	---

Unterrichtsvorhaben Q1-IV: Modellierung und Implementierung von Anwendungen mit dynamischen, nichtlinearen Datenstrukturen

Leitfragen: *Wie können Daten im Anwendungskontext mit Hilfe binärer Baumstrukturen verwaltet werden? Wie kann dabei der rekursive Aufbau der Baumstruktur genutzt werden? Welche Vor- und Nachteile haben Suchbäume für die geordnete Verwaltung von Daten?*

Vorhabenbezogene Konkretisierung:

Anhand von Beispielen für Baumstrukturen werden grundlegende Begriffe eingeführt und der rekursive Aufbau binärer Bäume dargestellt.

Anschließend werden für eine Problemstellung in einem der Anwendungskontexte Klassen modelliert und implementiert. Dabei werden die Operationen der Datenstruktur Binärbaum thematisiert und die entsprechende Klasse `BinaryTree` der Vorgaben für das Zentralabitur NRW verwendet. Klassen und ihre Beziehungen werden in Entwurfs- und Implementationsdiagrammen dargestellt. Die Funktionsweise von Methoden wird anhand grafischer Darstellungen von Binärbäumen erläutert.

Unter anderem sollen die verschiedenen Baumtraversierungen (Pre-, Post- und Inorder) kennen gelernt werden. Unterschiede bezüglich der Möglichkeit, den Baum anhand der Ausgabe der Baum Inhalte via Pre-, In- oder Postorder-Traversierung zu rekonstruieren, werden dabei ebenfalls angesprochen, indem die fehlende Umkehrbarkeit der Zuordnung Binärbaum => Inorder-Ausgabe an einem Beispiel verdeutlicht wird.

Eine Tiefensuche wird verwendet, um einen in der Baumstruktur gespeicherten Inhalt zu suchen.

Zu einer Problemstellung in einem entsprechenden Anwendungskontext werden die Operationen der Datenstruktur „Suchbaum“ thematisiert und unter der Verwendung der Klasse `BinarySearchTree` (der Materialien für das Zentralabitur in NRW) weitere Klassen oder Methoden in diesem Anwendungskontext modelliert und implementiert. Auch in diesem Kontext werden grafische Darstellungen der Bäume verwendet.

Die Verwendung von binären Bäumen und Suchbäumen wird anhand weiterer Problemstellungen oder anderen Kontexten weiter geübt.

Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien
<p>1. Analyse von Baumstrukturen in verschiedenen Kontexten</p> <p>(a) Grundlegende Begriffe (Grad, Tiefe, Höhe, Blatt, Inhalt, Teilbaum, Ebene, Vollständigkeit)</p> <p>(b) Aufbau und Darstellung von binären Bäumen anhand von Baumstrukturen in verschiedenen Kontexten</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen, • analysieren und erläutern Algorithmen und Programme, • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen, • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen, • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu, • modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren, • verwenden bei der Modellierung geeigneter Problemstellungen die Möglichkeiten der Polymorphie, • entwickeln iterative und rekursive Algorithmen unter Nutzung der Konstruktionsstrategien „Modularisierung“ und „Teilen und Herrschen“, 	<p>Beispiel: Termbaum Der Aufbau von Termen wird mit Hilfe von binären Baumstrukturen verdeutlicht.</p> <p>oder</p> <p>Beispiel: Ahnenbaum Die binäre Baumstruktur ergibt sich daraus, dass jede Person genau einen Vater und eine Mutter hat.</p> <p><u>Weitere Beispiele für Anwendungskontexte für binäre Bäume:</u></p> <p>Beispiel: Suchbäume (zur sortierten Speicherung von Daten) Alle Inhalte, die nach einer Ordnung vor dem Inhalt im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Inhalt im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum (dies gilt für alle Teilbäume).</p> <p>oder</p> <p>Beispiel: Entscheidungsbäume Um eine Entscheidung zu treffen, werden mehrere Fragen mit ja oder nein beantwortet. Die Fragen, die möglich sind, wenn die</p>

	<ul style="list-style-type: none"> • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen, • modifizieren Algorithmen und Programme, • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen, • interpretieren Fehlermeldungen und korrigieren den Quellcode, • testen Programme systematisch anhand von Beispielen, • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau, • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar. 	<p>Antwort auf eine Frage mit „ja“ beantwortet wird, befinden sich im linken Teilbaum, die Fragen, die möglich sind, wenn die Antwort „nein“ lautet, stehen im rechten Teilbaum.</p> <p><i>oder</i></p> <p>Beispiel: <i>Codierungsbäume</i> für Codierungen, deren Alphabet aus genau zwei Zeichen besteht. Morse hat Buchstaben als Folge von Punkten und Strichen codiert. Diese Codierungen können in einem Binärbaum dargestellt werden, so dass ein Übergang zum linken Teilbaum einem Punkt und ein Übergang zum rechten Teilbaum einem Strich entspricht. Wenn man im Gesamtbaum startet und durch Übergänge zu linken oder rechten Teilbäumen einen Pfad zum gewünschten Buchstaben sucht, erhält man die Morsecodierung des Buchstabens.</p>
<p>2. Die Datenstruktur Binärbaum im Anwendungskontext unter Nutzung der Klasse BinaryTree</p> <p>(a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen im Anwendungskontext</p> <p>(b) Modellierung eines Entwurfdiagramms und Entwicklung eines Implementationsdiagramms</p> <p>(c) Erarbeitung der Klasse BinaryTree und beispielhafte Anwendung der Operationen</p>		<p>Beispiel: <i>Informatikerbaum als binärer Baum</i></p> <p>In einem binären Baum werden die Namen und die Geburtsdaten von Informatikern lexikographisch geordnet abgespeichert. Alle Namen, die nach dieser Ordnung vor dem Namen im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Namen im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum (dies gilt für alle Teilbäume).</p> <p>Folgende Funktionalitäten werden benötigt:</p> <ul style="list-style-type: none"> • Einfügen der Informatiker-Daten in den

<p>(d) Implementierung der Anwendung oder von Teilen der Anwendung</p> <p>(e) Traversierung eines Binärbaums im Pre-, In- und Postorderdurchlauf</p>		<p>Baum</p> <ul style="list-style-type: none"> • Suchen nach einem Informatiker über den Schlüssel „Name“ • Ausgabe des kompletten Datenbestands in nach Namen sortierter Reihenfolge
<p>3. Die Datenstruktur „binärer Suchbaum,, im Anwendungskontext unter Verwendung der Klasse BinarySearchTree</p> <p>(a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen</p> <p>(b) Modellierung eines Entwurfdiagramms und Entwicklung eines Implementationsdiagramms, grafische Darstellung eines binären Suchbaums und Erarbeitung der Struktureigenschaften</p> <p>(c) Erarbeitung der Klasse BinarySearchTree und Einführung des Interface „Item“ zur Realisierung einer geeigneten Ordnungsrelation</p> <p>(d) Implementierung der Anwendung oder von Teilen der Anwendung inklusive einer sortierten Ausgabe des Baums</p>		<p>Beispiel: Informatikerbaum als Suchbaum In einem binären Suchbaum werden die Namen und die Geburtsdaten von Informatikern lexikographisch geordnet abgespeichert. Alle Namen, die nach dieser Ordnung vor dem Namen im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Namen im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum (dies gilt für alle Teilbäume).</p> <p>Folgende Funktionalitäten werden benötigt:</p> <ul style="list-style-type: none"> • Einfügen der Informatiker-Daten in den Baum • Suchen nach einem Informatiker über den Schlüssel „Name“ • Ausgabe des kompletten Datenbestands in nach Namen sortierter Reihenfolge
<p>4. Übung und Vertiefung der Verwendung von Binärbäumen oder binären Suchbäumen anhand weiterer Problemstellungen</p>		<p>Beispiel: Codierungsbäume (s.o.) oder Huffman-Codierung</p> <p>oder</p>

Beispiel: *Buchindex*

Es soll eine Anwendung entwickelt werden, die anhand von Stichworten und zugehörigen Seitenzahlen ein Stichwortregister erstellt.

Da die Stichwörter bei der Analyse des Buches häufig gesucht werden müssen, werden sie in der Klasse *Buchindex* als Suchbaum (Objekt der Klasse `BinarySearchTree`) verwaltet. Alle Inhalte, die nach einer Ordnung vor dem Inhalt im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Inhalt im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum (dies gilt für alle Teilbäume).

oder

Beispiel: *Entscheidungsbäume (s.o.)*

oder

Beispiel: *Termbaum (s.o.)*

oder

Beispiel: *Ahnenbaum (s.o.)*

Unterrichtsvorhaben Q1-V: Endliche Automaten und formale Sprachen

Leitfragen: *Wie kann man (endliche) Automaten genau beschreiben? Wie können endliche Automaten (in alltäglichen Kontexten oder zu informatischen Problemstellungen) modelliert werden? Wie können Sprachen durch Grammatiken beschrieben werden? Welche Zusammenhänge gibt es zwischen formalen Sprachen, endlichen Automaten und regulären Grammatiken?*

Vorhabenbezogene Konkretisierung:

Anhand kontextbezogener sowie klassischer Beispiele werden endliche Automaten entwickelt, untersucht und modifiziert. Dabei werden verschiedene Darstellungsformen für endliche Automaten ineinander überführt und die akzeptierten Sprachen endlicher Automaten ermittelt. An einem Beispiel wird ein nichtdeterministischer Akzeptor eingeführt als Alternative gegenüber einem entsprechenden deterministischen Akzeptor.

Anhand von Beispielen werden Grammatiken regulärer Sprachen entwickelt, untersucht und modifiziert. Der Zusammenhang zwischen regulären Grammatiken und endlichen Automaten wird verdeutlicht durch die Entwicklung von allgemeinen Verfahren zur Erstellung einer regulären Grammatik für die Sprache eines gegebenen endlichen Automaten bzw. zur Entwicklung eines endlichen Automaten, der genau die Sprache einer gegebenen regulären Grammatik akzeptiert.

Auch andere Grammatiken werden untersucht, entwickelt oder modifiziert. An einem Beispiel werden die Grenzen endlicher Automaten ausgelotet. Die Schülerinnen und Schüler lernen die Funktionsweise des Kellerautomaten kennen und entwickeln mehrere Kellerautomaten.

Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien
<p>1. Endliche Automaten</p> <p>(a) Vom Automaten in den Schülerinnen und Schülern bekannten Kontexten zur formalen Beschreibung eines endlichen Automaten</p> <p>(b) Untersuchung, Darstellung und Entwicklung endlicher Automaten</p> <p>(c) Unterscheidung zwischen deterministischen und nicht-deterministischen Automaten</p> <p>(d) Entwickeln des Algorithmus zum Überführen eines nicht-deterministischen Automaten in einen deterministischen</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • analysieren und erläutern die Eigenschaften endlicher Automaten einschließlich ihres Verhaltens auf bestimmte Eingaben, • analysieren und erläutern Grammatiken regulärer Sprachen, • zeigen die Grenzen endlicher Automaten und regulärer Grammatiken im Anwendungszusammenhang auf, • ermitteln die formale Sprache, die durch eine Grammatik erzeugt wird, • entwickeln und modifizieren zu einer Problemstellung endliche Automaten, • entwickeln zur akzeptierten Sprache eines Automaten die zugehörige Grammatik, • entwickeln zur Grammatik einer regulären Sprache einen zugehörigen endlichen Automaten, • modifizieren Grammatiken regulärer Sprachen, • entwickeln zu einer regulären Sprache eine Grammatik, die die Sprache erzeugt, • stellen endliche Automaten in Tabellen oder Graphen dar und überführen sie in die jeweils andere Darstellungsform, • ermitteln die Sprache, die ein endlicher Automat akzeptiert, • beschreiben an Beispielen den Zusammenhang zwischen Automaten und Grammatiken, • entwickeln feste Algorithmen und Vorgehensweisen, • wenden bekannte Verfahren korrekt und schrittweise an. 	<p>Beispiele:</p> <p>Cola-Automat, Geldspielautomat, Roboter, Zustandsänderung eines Objekts „Auto“, Akzeptor für bestimmte Zahlen, Akzeptor für Teilwörter in längeren Zeichenketten, Akzeptor für Terme</p>
<p>2. Untersuchung und Entwicklung von Grammatiken regulärer Sprachen</p> <p>(a) Erarbeitung der formalen Darstellung regulärer Grammatiken</p> <p>(b) Untersuchung, Modifikation und Entwicklung von Grammatiken</p> <p>(c) Entwicklung von endlichen Automaten zum Erkennen regulärer Sprachen die durch Grammatiken gegeben werden</p> <p>(d) Entwicklung regulärer Grammatiken zu endlichen Automaten</p>	<p>(This cell is shared with the previous row and contains the same list of competencies.)</p>	<p>Beispiele:</p> <p>reguläre Grammatik für Wörter mit ungerader Parität, Grammatik für Wörter, die bestimmte Zahlen repräsentieren, Satzgliedergrammatik</p>

3. Grenzen endlicher Automaten

- (a) Untersuchung von nicht regulären Problemen
- (b) Untersuchung und Entwicklung von Kellerautomaten
- (c) Untersuchung von noch komplexeren Problemen

Beispiele:

Klammerausdrücke, $a^n b^n$ im Vergleich zu $(ab)^n$

Unterrichtsvorhaben Q1-VI: Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit

Leitfragen: *Welche Möglichkeiten bieten Informatiksysteme und wo liegen ihre Grenzen?*

Vorhabenbezogene Konkretisierung:

Anhand des Biberproblems wird das Halteproblem erkannt und die generelle Berechenbarkeit in Frage gestellt.

Ausgehend von den prinzipiellen Grenzen endlicher Automaten liegt die Frage nach den Grenzen von Computern bzw. nach Grenzen der Automatisierbarkeit nahe. Das Biberproblem wird anhand einer Simulation vertieft. So wird ein konkretes Halteproblem entdeckt. Es wird plausibel, dass es unmöglich ist, ein Informatiksystem zu entwickeln, das für jedes beliebige Computerprogramm und jede beliebige Eingabe entscheidet ob das Programm mit der Eingabe terminiert oder nicht. Anschließend werden Vor- und Nachteile der Grenzen der Automatisierbarkeit angesprochen und der Einsatz von Informatiksystemen hinsichtlich prinzipieller Möglichkeiten und prinzipieller Grenzen beurteilt.

Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien
<p>1. Die Turingmaschine als Repräsentant für moderne Computer sowie das Biberproblem</p> <ul style="list-style-type: none"> a) Aufbau einer Turingmaschine b) Bezug der Turingmaschine zur Von-Neumann-Architektur c) Analyse und Erläuterung der Funktionsweise eines Turingprogramms. 	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • erläutern die Ausführung eines einfachen Turingprogramms, • entwickeln einen Automaten nach den Vorgaben einer Turingmaschine • untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen. • Untersuchen Geschichtliches, aktuelle Gegebenheiten und nehmen an der aktuellen Diskussion zur Berechenbarkeit teil. 	<p>Beispiel: <i>Das Biberproblem</i> Ein Turingprogramm soll erstellt werden und möglichst viele Items legen. Mit Hilfe von Kara werden Automaten mit einer begrenzten Anzahl von Zuständen erstellt. In einem Wettbewerb wird versucht, den fleißigsten Automaten zu erstellen.</p>
<p>2. Grenzen der Automatisierbarkeit und Geschichtliches</p> <ul style="list-style-type: none"> a) Vorstellung des Halteproblems b) Unlösbarkeit des Halteproblems c) Beurteilung des Einsatzes von Informatiksystemen hinsichtlich prinzipieller Möglichkeiten und prinzipieller Grenzen d) Vorstellung der Komplexitätsklassen NP und P 		<p>Beispiel: Halteproblem, Film des Science-Slam zu NP und P</p>

Unterrichtsvorhaben Q2-I: Modellierung und Nutzung von relationalen Datenbanken in Anwendungskontexten

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

Vorhabenbezogene Konkretisierung:

Ausgehend von einer kontextbezogenen Problemstellung entwickeln Schülerinnen und Schüler für sie relevante Fragestellungen. Zur Beantwortung dieser Fragestellungen simulieren die Schülerinnen und Schüler die Problemstellung und entwickeln Lösungsideen. Die Lösungsideen werden gesammelt und zur Modellierung einer ersten Datenbank genutzt. Ein Entity-Relationship-Diagramm wird verwendet, um die Entitäten inklusive ihrer Attribute und Relationen im Datenbankschema darzustellen. Die entwickelte Datenbank wird am Computer umgesetzt.

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.

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.

Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien
<p>1. 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 • Überprüfung von Datenbankschemata hinsichtlich der 1. bis 3. Normalform und Normalisierung (um Redundanzen zu vermeiden und 	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • erläutern die Eigenschaften und den Aufbau von Datenbanksystemen unter dem Aspekt der sicheren Nutzung, • analysieren und erläutern die Syntax und Semantik einer Datenbankabfrage, • analysieren und erläutern eine Datenbankmodellierung, • erläutern die Eigenschaften normalisierter Datenbankschemata, • bestimmen Primär- und Sekundärschlüssel, • ermitteln für anwendungsbezogene Problemstellungen Entitäten, zugehörige Attribute, Relationen und Kardinalitäten, • modifizieren eine Datenbankmodellierung, • modellieren zu einem Entity-Relationship-Diagramm ein relationales Datenbankschema, • bestimmen Primär- und Sekundärschlüssel, • überführen Datenbankschemata in vorgegebene Normalformen, • verwenden die Syntax und Semantik einer Datenbankabfragesprache, um Informationen aus einem Datenbanksystem zu extrahieren, • ermitteln Ergebnisse von Datenbankabfragen über mehrere verknüpfte Tabellen, • stellen Entitäten mit ihren Attributen und die Beziehungen zwischen Entitäten in einem 	<p>Beispiel: <i>Projektwochen-Buchungssystem</i> Alle zwei Jahre können die Schülerinnen und Schüler vor den Projektwochen aus den angebotenen Kursen Projekte wählen.</p> <p>Beispiel: <i>Schulbuchausleihe</i></p> <p>Beispiel: <i>Schulmedienverwaltung</i></p>

<p>Konsistenz zu gewährleisten)</p>	<p>Entity-Relationship-Diagramm grafisch dar,</p> <ul style="list-style-type: none"> • überprüfen Datenbankschemata auf vorgegebene Normalisierungseigenschaften. 	
<p>2. Nutzung von relationalen Datenbanken</p> <p>(a) Aufbau von Datenbanken und Grundbegriffe</p> <ul style="list-style-type: none"> • Entwicklung von Fragestellungen zur vorhandenen Datenbank • Analyse der Struktur der vorgegebenen Datenbank und Erarbeitung der Begriffe Tabelle, Attribut, Datensatz, Datentyp, Primärschlüssel, Fremdschlüssel, Datenbankschema <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) • Vertiefung an einem weiteren Datenbankbeispiel 		<p>Beispiel: Buchungssystem</p> <p>Im 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>Zu den Beispielen ist die Datenbank zu modellieren, ggf. zu normalisieren und im Datenbanksystem umzusetzen. Weiter sollen sinnvolle Abfragen entwickelt werden.</p>

Unterrichtsvorhaben Q2-II: Projekt - Modellierung, Entwicklung und Implementierung eines realen Auftrages

Leitfragen: Wie setze ich eine konkrete Problemstellung um?

Vorhabenbezogene Konkretisierung:

Es wird eine Datenbank mit einem realen Auftraggeber entwickelt. Dieser stellt zunächst persönlich das Problem vor. Dabei kann es sich um einen beliebigen Auftrag handeln. Der Lehrer prüft zuvor die generelle Machbarkeit um eine Überforderung zu vermeiden. Diese Aufgabe soll in einer bestimmten Zeit von den Schülern gelöst werden. Dabei steht die Lehrkraft beratend zur Seite, ansonsten steht das Projekt aber komplett in Schülerhand. Eventuell müssen die Schülerinnen und Schüler sich auch selbstständig ein neues Werkzeug erarbeiten.

Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien
<p>Projekt</p> <ul style="list-style-type: none"> a) Vorstellen des Auftrages, evtl. Gespräch mit dem Auftraggeber b) Analyse des Auftrages, Unterteilung der Gruppen in Kompetenzbereiche c) Abstecken der Meilensteine d) Umsetzung des Problems in ein Modell, Erstellung einer Grafik sowie eines Pflichtenheftes, sofern noch nicht vorgegeben e) Erkundung der nötigen Werkzeuge f) Beginn der Implementation und Umsetzung des Modells g) Eventuell Verbesserung und Änderungen des Modells h) Präsentation der Ergebnisse 	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • diskutieren über die Machbarkeit des Auftrages, • analysieren die Anforderungen und die Elemente, • schätzen sich selbst und ihre Fähigkeiten zur Umsetzung ein und stellen dementsprechend Ziele auf, • erarbeiten sich weitestgehend selbständig ein für sie neues Werkzeug, • modellieren das Problem mit bekannten Mitteln und stellen es grafisch dar, • implementieren auf einem informatischen Werkzeug, • kommunizieren in der Gruppe über die Ergebnisse, • helfen sich gegenseitig bei Schwierigkeiten, • erkennen Schwächen im eigenen Modell und verbessern diese, • teilen die Aufgaben sinnvoll untereinander auf, • präsentieren ihre Ergebnisse. 	<p>Beispiel: Erstellung einer Datenbank</p> <p><u>Projektwochendatenbank:</u> Für die Schule muss ein Buchungssystem für die Projektwoche erstellt werden. Die Organisatoren der Projektwoche stellen ihre Ansprüche vor. Eine Umsetzung erfolgt auf dem Schulserver mit einer phpMyAdmin-Datenbankoberfläche sowie mit php. Es wird eine entsprechende Datenbank entwickelt sowie eine online-Oberfläche zum Buchen programmiert.</p> <p><u>Schulmaterialverwaltung:</u> Für die Schule muss eine Verwaltung der Schulmedien erstellt werden. Die Verantwortlichen für die Schulmedien stellen ihre Ansprüche vor. Eine Umsetzung erfolgt auf dem Schulserver mit einer phpMyAdmin-Datenbankoberfläche sowie mit php. Es wird eine entsprechende Datenbank entwickelt sowie eine online-Oberfläche zum Verwalten programmiert.</p> <p><u>Mitgliederverwaltung des Fördervereins:</u> Für den Förderverein der Schule muss eine Verwaltung der Mitglieder erstellt werden. Die Vorsitzende des Fördervereins stellt ihre Ansprüche vor. Eine Umsetzung erfolgt auf einem bereitgestellten Server mit einer Datenbankoberfläche sowie mit php. Es wird eine entsprechende Datenbank entwickelt sowie eine online-Oberfläche zum Verwalten programmiert.</p> <p><u>Handy-Programmierung:</u> Steuerung der Lego-Mindstorms-</p>

		<p>Roboter Die Schülerinnen und Schüler der Junior-Ingenieur-Akademie der Schule möchten die Lego-Mindstorms-Roboter gerne mit ihrem Handy steuern. Schülerinnen und Schüler sowie die Lehrkraft erklären ihre Vorstellungen. Eine Umsetzung erfolgt auf einer passenden Umgebung (z.B. Google AppInventor)</p> <p><u>Flash-Programmierung:</u> Die Homepage-AG möchte gerne eine interaktive Flashumgebung auf der Homepage der Schule installieren. Der Vorsitzende erklärt seine Vorstellungen. Eine Umsetzung erfolgt auf der für begrenzte Zeit verfügbaren Flash-Programmierungsumgebung.</p>
--	--	---

Unterrichtsvorhaben Q2-III: Sicherheit und Datenschutz in Netzstrukturen

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. Zu den Themen sollen Referate entwickelt werden.

Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien
<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) Protokolle zur Kommunikation im Netz</p> <p>(d) Vertraulichkeit, Integrität, Authentizität in Netzwerken sowie symmetrische und asymmetrische kryptografische Verfahren (Cäsar-, Vigenère-, RSA-Verfahren) als Methoden Daten im Netz verschlüsselt zu übertragen</p> <p>(e) Fallbeispiele zur Datenschutzproblematik und zum Urheberrecht</p> <p>(f) Adressierungstechniken in Netzwerken</p> <p>(g) Multithreading</p> <p>(h) Client-Server-Architektur mit Implementierung</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, • analysieren und erläutern Eigenschaften und Einsatzbereiche symmetrischer und asymmetrischer Verschlüsselungsverfahren, • untersuchen und bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen, die Sicherheit von Informatiksystemen sowie die Einhaltung der Datenschutzbestimmungen und des Urheberrechts, • untersuchen und bewerten Problemlagen, die sich aus dem Einsatz von Informatiksystemen ergeben, hinsichtlich rechtlicher Vorgaben, ethischer Aspekte und gesellschaftlicher Werte unter Berücksichtigung unterschiedlicher Interessenlagen, • analysieren und erläutern die Techniken der Adressierung im Netz, • nutzen bereitgestellte Informatiksysteme und das Internet reflektiert zum Erschließen, zur Aufbereitung und Präsentation fachlicher Inhalte. 	

Unterrichtsvorhaben Q2-IV: Wiederholung und Vertiefung ausgewählter Kompetenzen und Inhalte des ersten Jahres der Qualifikationsphase