

Rhein-Gymnasium Köln

Schulinterner Lehrplan

Informatik

Sekundarstufe II

(Stand: 2.2.2015)

1. Die Fachgruppe Informatik des Rhein-Gymnasiums Köln

Alle relevanten Informationen zum Rhein-Gymnasium sind auf unserer Homepage www.rhein-gymnasium-koeln.de zu finden.

Das Fach Informatik wird am Rheingymnasium ab der Jahrgangsstufe 8 im Wahlpflichtbereich II (WP II) zweistündig unterrichtet.

In den Jahrgangsstufen 5 und 6 wird ein für alle verpflichtender Kurs zur sogenannte Informations-technischen Grundausbildung (ITG) durchgeführt, der jedoch nicht unmittelbar dem Fach Informatik zuzuordnen ist. Der Kurs umfasst die Bereiche Textverarbeitung, Tabellenkalkulation und Präsentation mit dem Programmpaket OpenOffice.

Die Fachschaft Informatik sieht sich federführend für der Verwendung von virtuellen Lernumgebungen (z.B. www.matheprisma.de) und der Entwicklung von virtuellen Klassenraumangeboten über Moodle. In ITG wird in die Bedienung von Moodle eingeführt und bis zum Abitur in allen Informatikkursen verwendet.

In der Sekundarstufe II bietet das Rhein-Gymnasium für die eigenen Schülerinnen und Schüler in allen Jahrgangsstufen jeweils einen Grundkurs in Informatik an.

Um insbesondere Schülerinnen und Schülern gerecht zu werden, die in der Sekundarstufe I keinen Informatikunterricht besucht haben, wird in Kursen der Einführungsphase besonderer Wert darauf gelegt, dass keine Vorkenntnisse aus der Sekundarstufe I zum erfolgreichen Durchlaufen des Kurses erforderlich sind.

Der Unterricht der Sekundarstufe II wird mit Hilfe der Programmiersprache Java und der Entwicklungsumgebung BlueJ, sowie der (Spiele-)Entwicklungsumgebung Greenfoot durchgeführt. Bei hinreichender Zeit wird noch der JavaEditor von Röhner für die Gestaltung von graphischen Oberflächen angesprochen.

Die gemeinsame Entwicklung von Materialien und Unterrichtsvorhaben, die Evaluation von Lehr- und Lernprozessen sowie die stetige Überprüfung und eventuelle Modifikation des schulinternen Curriculums durch die Fachkonferenz Informatik stellen einen wichtigen Beitrag zur Qualitätssicherung und -entwicklung des Unterrichts dar.

2. Entscheidungen zum Unterricht

2.1 Unterrichtsvorhaben

Die Darstellung der Unterrichtsvorhaben im schulinternen Lehrplan besitzt den Anspruch, *sämtliche* im Kernlehrplan angeführten Kompetenzen abzudecken. Dies entspricht der Verpflichtung jeder Lehrkraft, Schülerinnen und Schülern Lerngelegenheiten zu ermöglichen, so dass *alle* Kompetenzerwartungen des Kernlehrplans von ihnen erfüllt werden können.

Die entsprechende Umsetzung erfolgt auf zwei Ebenen: der Übersichts- und der Konkretisierungsebene.

Im „Übersichtsraster Unterrichtsvorhaben“ (Kapitel 2.1.1) wird die für alle Lehrerinnen und Lehrer gemäß Fachkonferenzbeschluss vom 2.2.2015 *verbindliche* Verteilung der Unterrichtsvorhaben dargestellt. Das Übersichtsraster dient dazu, den Kolleginnen und Kollegen einen schnellen Überblick über die Zuordnung der Unterrichtsvorhaben zu den einzelnen Jahrgangsstufen sowie den im Kernlehrplan genannten Kompetenzen, Inhaltsfeldern und inhaltlichen Schwerpunkten zu verschaffen. Der ausgewiesene Zeitbedarf versteht sich als grobe Orientierungsgröße, die nach Bedarf über- oder unterschritten werden kann. Um Freiraum für Vertiefungen, besondere Schülerinteressen, aktuelle Themen bzw. die Erfordernisse anderer besonderer Ereignisse (z.B. Praktika, Kursfahrten o.ä.) zu erhalten, wurden im Rahmen dieses schulinternen Lehrplans ca. 75 Prozent der Bruttounterrichtszeit verplant.

Während der Fachkonferenzbeschluss zum „Übersichtsraster Unterrichtsvorhaben“ zur Gewährleistung vergleichbarer Standards sowie zur Absicherung von Lerngruppenübertritten und Lehrkraftwechseln für alle Mitglieder der Fachkonferenz Bindekraft entfalten soll, beinhaltet die Spalte „Beispiele, Medien, Materialien“ in der Sequenzierungsübersicht auch Elemente, die empfehlenden Charakter haben. Verbindliche Absprachen zu didaktisch-methodischen Zugängen, fächerübergreifenden Kooperationen, Lernmitteln und -orten in Bezug auf die Unterrichtsvorhaben finden sich unter dem Punkt „Vorhabenbezogene Konkretisierung“.

Um die Zusammenführung von ggf. zwei Grundkursen der EF zu einem Kurs in der Q1 und die Anschlussfähigkeit zur Verwendung des Schulbuches der Qualifikationsphase einigt sich die Fachschaft zudem für die Einführungsphase auf folgende Werkzeuge:

- Es werden die Grundlagen der Bedienung von Greenfoot erarbeitet.
- Die Verwendung der Java-Entwicklungsumgebung BlueJ ist den Schülern hinreichend bekannt und sie sind gehalten worden, diese zu Hause zu installieren.
- Die elektronischen Unterrichtsmaterialien werden in einem Moodle-Klassenraum gesammelt.

BlueJ bleibt auch in der Qualifikationsphase die zentrale Entwicklungsumgebung. Die weitere Verwendung von Greenfoot ergibt sich aus den Materialien der Neuauflage des Schulbuches für die Qualifikationsphase, das bisher nur in Auszügen veröffentlicht ist.

Da in den folgenden Unterrichtsvorhaben Inhalte in der Regel anhand von Problemstellungen in Anwendungskontexten bearbeitet werden, werden in einigen Unterrichtsvorhaben jeweils mehrere Inhaltsfelder angesprochen.

Das ab dem Schuljahr 2015/16 eingeführte Schulbuchreihe Informatik vom Schöningh-Verlag bildet die Grundlage der konkretisierten Unterrichtsvorhaben. Bis zur Lieferung des Buches werden die Unterrichtsvorhaben noch mit der Schulbuchreihe von Klett oder eigenen Materialien durchgeführt. Dieses Curriculum ist dann an das Buch anzupassen.

2.1.1 Übersichtsraster Unterrichtsvorhaben

Qualifikationsphase 1	
<p>Unterrichtsvorhaben Q1-I</p> <p>Thema: <i>Wiederholung der objektorientierten Modellierung und Programmierung anhand einer kontextbezogenen Problemstellung</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 • Informatiksysteme <p>Inhaltliche Schwerpunkte:</p> <ul style="list-style-type: none"> • Objekte und Klassen • Analyse, Entwurf und Implementierung von Algorithmen • Syntax und Semantik einer Programmiersprache • Nutzung von Informatiksystemen <p>Zeitbedarf: 8 Stunden</p>	<p>Unterrichtsvorhaben Q1-II</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 <p>Zeitbedarf: 20 Stunden</p>
<p>Unterrichtsvorhaben Q1-III</p> <p>Thema: <i>Suchen und Sortieren auf 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"> • Algorithmen • Formale Sprachen und Automaten <p>Inhaltliche Schwerpunkte:</p> <ul style="list-style-type: none"> • Analyse, Entwurf und Implementierung von Algorithmen • Algorithmen in ausgewählten informatischen Kontexten • Syntax und Semantik einer Programmiersprache <p>Zeitbedarf: 16 Stunden</p>	<p>Unterrichtsvorhaben Q1-IV</p> <p>Thema: <i>Modellierung und Nutzung von relationalen Datenbanken in Anwendungskontexten</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 • Informatik, Mensch und Gesellschaft <p>Inhaltliche Schwerpunkte:</p> <ul style="list-style-type: none"> • Datenbanken • Algorithmen in ausgewählten informatischen Kontexten • Syntax und Semantik einer Programmiersprache • Sicherheit <p>Zeitbedarf: 20 Stunden</p>

<p>Unterrichtsvorhaben Q1-V</p> <p>Thema: <i>Sicherheit und Datenschutz in Netzstrukturen</i></p> <p>Zentrale Kompetenzen:</p> <ul style="list-style-type: none"> • Argumentieren • Darstellen und Interpretieren • Kommunizieren und Kooperieren <p>Inhaltsfelder:</p> <ul style="list-style-type: none"> • Informatiksysteme • Informatik, Mensch und Gesellschaft <p>Inhaltliche Schwerpunkte:</p> <ul style="list-style-type: none"> • Einzelrechner und Rechnernetzwerke • Sicherheit • Nutzung von Informatiksystemen, Wirkungen der Automatisierung <p>Zeitbedarf: 10 Stunden</p>	
<p>Summe Qualifikationsphase 1: 74 Stunden</p>	

Qualifikationsphase 2	
<p>Unterrichtsvorhaben Q2-I</p> <p>Thema: <i>Modellierung und Implementierung von Anwendungen mit dynamischen, nichtlinearen 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 <p>Zeitbedarf: 20 Stunden</p>	<p>Unterrichtsvorhaben Q2-II</p> <p>Thema: <i>Endliche Automaten und formale Sprachen</i></p> <p>Zentrale Kompetenzen:</p> <ul style="list-style-type: none"> • Argumentieren • Modellieren • Darstellen und Interpretieren <p>Kommunizieren und Kooperieren</p> <p>Inhaltsfelder:</p> <ul style="list-style-type: none"> • Endliche Automaten und formale Sprachen <p>Inhaltliche Schwerpunkte:</p> <ul style="list-style-type: none"> • Endliche Automaten • Grammatiken regulärer Sprachen • Möglichkeiten und Grenzen von Automaten und formalen Sprachen <p>Zeitbedarf: 24 Stunden</p>
<p>Unterrichtsvorhaben Q2-III</p> <p>Thema: <i>Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit</i></p> <p>Zentrale Kompetenzen:</p> <ul style="list-style-type: none"> • Argumentieren • Kommunizieren und Kooperieren <p>Inhaltsfelder:</p> <ul style="list-style-type: none"> • Informatiksysteme • Informatik, Mensch und Gesellschaft <p>Inhaltliche Schwerpunkte:</p> <ul style="list-style-type: none"> • Einzelrechner und Rechnernetzwerke • Probleme und Grenzen der Automatisierung <p>Zeitbedarf: 12 Stunden</p>	<p>Unterrichtsvorhaben Q2-IV:</p> <p>Thema: <i>Zur Abiturvorbereitung: Wiederholung und Vertiefung ausgewählter Kompetenzen und Inhalte des ersten Jahrs der Qualifikationsphase</i></p>
Summe Qualifikationsphase 2: 56 Stunden	

2.1.2 Konkretisierte Unterrichtsvorhaben

Im Folgenden sollen die im *Unterkapitel 2.1.1* aufgeführten Unterrichtsvorhaben konkretisiert werden.

Hinweis: Verbindliche Festlegungen der Fachkonferenz:

Die Fachkonferenz der Beispielschule hat Themen, Leitfragen und die Ausführungen unter der Überschrift *Vorhabenbezogene Konkretisierung* verbindlich vereinbart, ebenso die Sequenzierung der Unterrichtsvorhaben (erste Tabellenspalte) und die ausgewiesenen Kompetenzen (zweite Tabellenspalte). 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.

Unterrichtliche Anregungen:

Darüber hinausgehende angeführten Beispiele, Medien und Materialien sind dagegen Vorschläge bzw. Hilfen für die Lehrkräfte. In diesen Bereichen sind Abweichungen von den vorgeschlagenen Vorgehensweisen möglich.

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

Die Schülerinnen und Schüler

- verwenden die Fachsprache bei der Kommunikation über informatische Sachverhalte (K),
- nutzen das verfügbare Informatiksystem zur strukturierten Verwaltung von Dateien unter Berücksichtigung der Rechteverwaltung (K),
- organisieren und koordinieren kooperatives und eigenverantwortliches Arbeiten (K),
- strukturieren den Arbeitsprozess, vereinbaren Schnittstellen und führen Ergebnisse zusammen (K),
- beurteilen Arbeitsorganisation, Arbeitsabläufe und Ergebnisse (K),
- präsentieren Arbeitsabläufe und -ergebnisse adressatengerecht (K).

Unterrichtsvorhaben Q1-I:

Thema: Wiederholung der objektorientierten Modellierung und Programmierung

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 mit BlueJ 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 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>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 Sichtbarkeitsbereiche zu (M), • 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><i>Beispiele:</i></p> <ol style="list-style-type: none"> 1. Fahrzeugflotte (→ Unterrichtsvorhaben der EF) mit dem Prinzip der Vererbung (Fahrzeug > Pkw, Lkw, Motorrad) 2. Vererbungsbaum für Tierarten

Unterrichtsvorhaben Q1-II:

Thema: 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 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 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 (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 Datensammlungen zu (M), • 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), 	<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 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>BlueJ-Projekt Patientenwarteschlange</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>	<ul style="list-style-type: none"> stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D). 	<p><i>Beispiel:</i> Heftstapel (eigenes Programmierprojekt ohne GUI)</p> <p>In einem Heftstapel soll das Heft einer Schülerin gefunden 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><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 Abfahrer müssen an jeder Stelle der Struktur, nicht nur am Ende oder Anfang eingefügt werden können.</p> <p><i>Materialien:</i></p> <p>BlueJ-Projekt Abfahrtslauf</p>
<p>4. Vertiefung - Anwendungen von Listen, Stapeln oder Schlangen in mindestens einem weiteren Kontext</p>		<p>Mindestens eines der Beispiele A bis C (ggf. in Gruppenarbeit mit Ergebnisvortrag)</p>
<p><i>Beispiel A:</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 B:</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 Nummern, wobei die Nummer jedoch erst eingesehen werden kann, wenn der Wagon der vorderste an der offenen Gleisseite ist. (Zwischen den Wagons herumzurnen, 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.</p>	<p><i>Beispiel C:</i> Autos an einer Ampel zur Zufahrtsregelung</p> <p>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.</p> <p><i>Materialien:</i></p> <p>BlueJ-Projekte</p>

Unterrichtsvorhaben Q1-III:

Thema: Suchen und Sortieren auf linearen Datenstrukturen

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 Implementierungen 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), 	<p><i>Beispiel:</i> Suchen in einer Liste von natürlichen Zahlen</p> <p><i>Beispiel:</i> Fahrzeugverwaltung</p> <p>Das existierende Projekt wird um die Suchmethoden erweitert.</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>	<ul style="list-style-type: none"> 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><i>Beispiele:</i> Zahlenliste und Fahrzeugverwaltung</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>(b) Untersuchung der Anzahl der Vergleichsoperationen und des Speicherbedarf bei beiden Sortierverfahren</p> <p>(c) Beurteilung der Effizienz der beiden Sortierverfahren</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 umgangssprachlich und grafisch dar (D). 	<p><i>Beispiele:</i> Zahlenliste und Fahrzeugverwaltung</p>

Unterrichtsvorhaben Q1-IV:

Thema: 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 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
<p>1. 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) <p>(c) Vertiefung an einem weiteren Datenbankbeispiel</p>	<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 (A), analysieren und erläutern die Syntax 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), 	<p><i>Beispiel:</i> Schulbuchausleihe</p> <p>Datenbank zur Schulbuch-Ausleihe für Open-Office Base (über 1000 Entleiher, 200 Bücher mit mehreren tausend Exemplaren und viele Ausleihvorgänge).</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 • Überprüfung von Datenbankschemata hinsichtlich der 1. bis 3. Normalform und Normalisierung (um Redundanzen zu vermeiden und Konsistenz zu gewährleisten) 	<ul style="list-style-type: none"> • bestimmen Primär- und Sekundärschlüssel (M), • überführen Datenbankschemata in vorgegebene Normalformen (M), • verwenden die Syntax und Semantik einer Datenbankabfragesprache, um Informationen aus einem Datenbanksystem zu extrahieren (I), • 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: 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>
--	---	--

Unterrichtsvorhaben Q1-V:

Thema: 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.

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-Verfahren) als Methoden Daten im Netz verschlüsselt zu übertragen</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 Informatiksystemen ergeben, hinsichtlich rechtlicher Vorgaben, ethischer Aspekte und gesellschaftlicher Werte unter Berücksichtigung unterschiedlicher Interessenlagen (A), • nutzen bereitgestellte Informatiksysteme und das Internet reflektiert zum Erschließen, zur Aufbereitung und Präsentation fachlicher Inhalte (D). 	<p><i>Beispiel:</i></p> <p>Client-Server-Programme und Lernmaterial des Gymnasiums Odenthal</p> <p><i>Beispiel:</i></p> <p>Programmieren der Cäsar-Verschlüsselung, Papierdurchgang mit der Vigenère-Verschlüsselung, Erarbeitung der RSA-Methode</p>
<p>2. Fallbeispiele zur Datenschutzproblematik und zum Urheberrecht</p>		<p><i>Beispiel:</i></p> <p>Datenschutz bei Facebook oder WhatsApp</p>

Unterrichtsvorhaben Q2-I:

Thema: 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 Materialien für das Zentralabitur in NRW) 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) implementiert werden. Unterschiede bezüglich der Möglichkeit, den Baum anhand der Ausgabe der Bauminhalte 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.

Zeitbedarf: 24 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<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 nichtlinearer) Datenstrukturen (A), • analysieren und erläutern Algorithmen und Programme (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M), • modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch 	<p><i>Primärmaterial:</i></p> <p>ETH-Zürich Leitprogramm Binäre Suchbäume (http://www.educ.ethz.ch/unt/um/inf/ad/baeume2/leitprogramm.pdf)</p> <p><i>Beispiel:</i> Termbaum</p> <p>Der Aufbau von Termen wird mit Hilfe von binären Baumstrukturen verdeutlicht.</p> <p><i>Beispiel:</i> Suchbäume (zur sortierten Speicherung von Daten)</p>
<p>2. Die Datenstruktur Binärbaum im Anwendungskontext unter Nutzung der Klasse BinaryTree</p> <p>(a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • erläutern Operationen dynamischer (linearer oder nichtlinearer) Datenstrukturen (A), • analysieren und erläutern Algorithmen und Programme (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M), • modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch 	<p><i>Beispiel:</i> Informatikerbaum als binärer Baum</p> <p>In einem <i>binären Baum</i> werden die Namen und die Geburtsdaten von Informatikern lexikographisch geordnet abgespeichert. Alle Namen, die nach dieser Ordnung</p>

<p>und Operationen im Anwendungskontext</p> <p>(b) Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramms</p> <p>(c) Erarbeitung der Klasse BinaryTree und beispielhafte Anwendung der Operationen</p> <p>(d) Implementierung der Anwendung oder von Teilen der Anwendung</p> <p>(e) Traversierung eines Binärbäums im Pre-, In- und Postorderdurchlauf</p>	<p>Spezialisieren und Generalisieren (M),</p> <ul style="list-style-type: none"> • verwenden bei der Modellierung geeigneter Problemstellungen die Möglichkeiten der Polymorphie (M), • entwickeln iterative und rekursive Algorithmen unter Nutzung der Konstruktionsstrategien „Modularisierung“ und „Teilen und Herrschen“ (M), • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I), • modifizieren Algorithmen und Programme (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), • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D). 	<p>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><i>Materialien:</i> BlueJ-Projekt Informatiker-Baum</p>
<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 Entwurfsdiagramms und Entwicklung eines Implementationsdiagramm</p> <p>(c) grafische Darstellung eines binären Suchbaums und Erarbeitung der Struktureigenschaften</p> <p>(d) Erarbeitung der Klasse BinarySearchTree und Einführung des Interface Item zur Realisierung einer geeigneten Ordnungsrelation</p> <p>(e) Implementierung der Anwendung oder von Teilen der Anwendung inklusive einer sortierten Ausgabe des Baums</p>	<p>(siehe obere Zelle)</p>	<p><i>Beispiel:</i> Informatikerbaum als Suchbaum</p> <p>In einem binären <i>Suchbaum</i> 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><i>Materialien:</i> BlueJ-Projekt Informatiker-Baum</p>
<p>4. Übung und Vertiefungen der Verwendung von Binärbäumen oder binären Suchbäumen anhand weiterer Problemstellungen</p>		<p><i>Beispiel:</i> Huffman-Codierung</p>

Unterrichtsvorhaben Q2-II:

Thema: 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 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 kontextbezogener Beispiele 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.

Zeitbedarf: 20 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien oder Materialien
<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>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> analysieren und erläutern die Eigenschaften endlicher Automaten einschließlich ihres Verhaltens auf bestimmte Eingaben (A), analysieren und erläutern Grammatiken regulärer Sprachen (A), zeigen die Grenzen endlicher Automaten und regulärer Grammatiken im Anwendungszusammenhang auf (A), ermitteln die formale Sprache, die durch eine Grammatik erzeugt wird (A), entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M), entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M), entwickeln zur akzeptierten Sprache eines Automaten die zugehörige Grammatik (M), entwickeln zur Grammatik einer regulären Sprache einen zugehörigen endlichen Automaten (M), modifizieren Grammatiken regulärer Sprachen (M), entwickeln zu einer regulären Sprache eine Grammatik, die die Sprache erzeugt (M), 	<p><i>Beispiele:</i></p> <p>Cola-Automat, Geldspielautomat,</p> <p>Roboter, Zustandsänderung eines Objekts „Auto“, Akzeptor für bestimmte Zahlen, Akzeptor für Teilwörter in längeren Zeichenketten, Akzeptor für Terme</p> <p><i>Werkzeug:</i> AutoEdit</p> <p><i>Materialien:</i> Skript RWTH Aachen</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>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M), entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M), entwickeln zur akzeptierten Sprache eines Automaten die zugehörige Grammatik (M), entwickeln zur Grammatik einer regulären Sprache einen zugehörigen endlichen Automaten (M), modifizieren Grammatiken regulärer Sprachen (M), entwickeln zu einer regulären Sprache eine Grammatik, die die Sprache erzeugt (M), 	<p><i>Beispiele:</i></p> <p>reguläre Grammatik für Wörter mit ungerader Parität, Grammatik für Wörter, die bestimmte Zahlen repräsentieren, Satzgliederungsgrammatik</p> <p><i>Werkzeug:</i> KfgEdit</p> <p><i>Materialien:</i> (s.o.)</p>

<p>3. Grenzen endlicher Automaten</p>	<ul style="list-style-type: none"> • stellen endliche Automaten in Tabellen oder Graphen dar und überführen sie in die jeweils andere Darstellungsform (D), • ermitteln die Sprache, die ein endlicher Automat akzeptiert (D). • beschreiben an Beispielen den Zusammenhang zwischen Automaten und Grammatiken (D). 	<p><i>Beispiele:</i> Klammerausdrücke, $a^n b^n$ im Vergleich zu $(ab)^n$</p>
--	--	---

Unterrichtsvorhaben Q2-III:

Thema: Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit

Leitfragen: Was sind die strukturellen Hauptbestandteile eines Computers und wie kann man sich die Ausführung eines maschinennahen Programms mit diesen Komponenten vorstellen? Welche Möglichkeiten bieten Informatiksysteme und wo liegen ihre Grenzen?

Vorhabenbezogene Konkretisierung:

Anhand einer von-Neumann-Architektur und einem maschinennahen Programm wird die prinzipielle Arbeitsweise von Computern verdeutlicht.

Ausgehend von den prinzipiellen Grenzen endlicher Automaten liegt die Frage nach den Grenzen von Computern bzw. nach Grenzen der Automatisierbarkeit nahe. Mit Hilfe einer entsprechenden Java-Methode 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 (Halteproblem). Anschließend werden Vor- und Nachteile der Grenzen der Automatisierbarkeit angesprochen und der Einsatz von Informatiksystemen hinsichtlich prinzipieller Möglichkeiten und prinzipieller Grenzen beurteilt.

Zeitbedarf: 12 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien oder Materialien
<p>1. Von-Neumann-Architektur und die Ausführung maschinennaher Programme</p> <p>(a) prinzipieller Aufbau einer von Neumann-Architektur mit CPU, Rechenwerk, Steuerwerk, Register und Hauptspeicher</p> <p>(b) einige maschinennahe Befehle und ihre Repräsentation in einem Binär-Code, der in einem Register gespeichert werden kann</p> <p>(c) Analyse und Erläuterung der Funktionsweise eines einfachen maschinennahen Programms</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> erläutern die Ausführung eines einfachen maschinennahen Programms sowie die Datenspeicherung auf einer „Von-Neumann-Architektur“ (A), untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen (A). 	<p><i>Beispiel:</i></p> <p>Addition von 4 zu einer eingegebenen Zahl mit einem Rechnermodell</p> <p><i>Werkzeug:</i></p> <p>Von-Neumann-Rechner Programm</p> <p><i>Materialien:</i></p> <p>Arbeitsblätter</p>
<p>2. Grenzen der Automatisierbarkeit</p> <p>(a) Vorstellung des Halteproblems</p> <p>(b) Unlösbarkeit des Halteproblems</p> <p>(c) Beurteilung des Einsatzes von Informatiksystemen hinsichtlich prinzipieller Möglichkeiten und prinzipieller Grenzen</p>		<p><i>Beispiel:</i> Halteproblem</p> <p><i>Material:</i></p> <p>Arbeitsblätter</p>

Unterrichtsvorhaben Q2-IV:

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

2.2 Grundsätze der fachmethodischen und fachdidaktischen Arbeit

In Absprache mit der Lehrerkonferenz sowie unter Berücksichtigung des Schulprogramms hat die Fachkonferenz Informatik des Rhein-Gymnasiums die folgenden fachmethodischen und fachdidaktischen Grundsätze beschlossen. In diesem Zusammenhang beziehen sich die Grundsätze 1 bis 14 auf fächerübergreifende Aspekte, die auch Gegenstand der Qualitätsanalyse sind, die Grundsätze 15 bis 21 sind fachspezifisch angelegt.

Überfachliche Grundsätze:

- 1) Geeignete Problemstellungen zeichnen die Ziele des Unterrichts vor und bestimmen die Struktur der Lernprozesse.
- 2) Inhalt und Anforderungsniveau des Unterrichts entsprechen dem Leistungsvermögen der Schüler/innen.
- 3) Die Unterrichtsgestaltung ist auf die Ziele und Inhalte abgestimmt.
- 4) Medien und Arbeitsmittel sind schülernah gewählt.
- 5) Die Schüler/innen erreichen einen Lernzuwachs.
- 6) Der Unterricht fördert eine aktive Teilnahme der Schüler/innen.
- 7) Der Unterricht fördert die Zusammenarbeit zwischen den Schülern/innen und bietet ihnen Möglichkeiten zu eigenen Lösungen.
- 8) Der Unterricht berücksichtigt die individuellen Lernwege der einzelnen Schüler/innen.
- 9) Die Schüler/innen erhalten Gelegenheit zu selbstständiger Arbeit und werden dabei unterstützt.
- 10) Der Unterricht fördert strukturierte und funktionale Partner- bzw. Gruppenarbeit.
- 11) Der Unterricht fördert strukturierte und funktionale Arbeit im Plenum.
- 12) Die Lernumgebung ist vorbereitet; der Ordnungsrahmen wird eingehalten.
- 13) Die Lehr- und Lernzeit wird intensiv für Unterrichtszwecke genutzt.
- 14) Es herrscht ein positives pädagogisches Klima im Unterricht.

Fachliche Grundsätze:

- 15) Der Unterricht unterliegt der Wissenschaftsorientierung und ist dementsprechend eng verzahnt mit seiner Bezugswissenschaft.
- 16) Der Unterricht ist problemorientiert und soll von realen Problemen ausgehen und sich auf solche rückbeziehen.
- 17) Der Unterricht folgt dem Prinzip der Exemplarizität und soll ermöglichen, informatische Strukturen und Gesetzmäßigkeiten in den ausgewählten Problemen und Projekten zu erkennen.
- 18) Der Unterricht ist anschaulich sowie gegenwarts- und zukunftsorientiert und gewinnt dadurch für die Schülerinnen und Schüler an Bedeutsamkeit.
- 19) Der Unterricht ist handlungsorientiert, d.h. projekt- und produktorientiert angelegt.
- 20) Im Unterricht werden sowohl für die Schule didaktisch reduzierte als auch reale Informatiksysteme aus der Wissenschafts-, Berufs- und Lebenswelt eingesetzt.
- 21) Der Unterricht beinhaltet reale Begegnung mit Informatiksystemen.

2.3 Grundsätze der Leistungsbewertung und Leistungsrückmeldung

Hinweis: Sowohl die Schaffung von **Transparenz bei Bewertungen** als auch die Vergleichbarkeit von Leistungen sind das Ziel, innerhalb der gegebenen Freiräume Vereinbarungen zu Bewertungskriterien und deren Gewichtung zu treffen.

Auf der Grundlage von §13 - §16 der APO-GOST sowie Kapitel 3 des Kernlehrplans Informatik für die gymnasiale Oberstufe hat die Fachkonferenz des Rhein-Gymnasiums die nachfolgenden Grundsätze zur Leistungsbewertung und Leistungsrückmeldung beschlossen. Die nachfolgenden Absprachen stellen die Minimalanforderungen an das lerngruppenübergreifende gemeinsame Handeln der Fachgruppenmitglieder dar. Bezogen auf die einzelne Lerngruppe kommen ergänzend weitere der in den Folgeabschnitten genannten Instrumente der Leistungsüberprüfung zum Einsatz.

2.3.1 Beurteilungsbereich Klausuren

Verbindliche Absprachen für die gesamte Oberstufe:

Bei der Formulierung von Aufgaben werden die für die Abiturprüfungen geltenden Operatoren des Faches Informatik schrittweise eingeführt, erläutert und dann im Rahmen der Aufgabenstellungen für die Klausuren benutzt.

Anzahl der Klausuren:

- Einführungsphase: 1 Klausur je Halbjahr
- Grundkurse Q 1: 2 Klausuren je Halbjahr
- Grundkurse Q 2.1: 2 Klausuren

Dauer der Klausur(en):

- Einführungsphase: 2 Unterrichtsstunden
- Grundkurse Q 1: 2 Unterrichtsstunden
- Grundkurse Q 2.1: 3 Unterrichtsstunden
- Grundkurse Q 2.2: 1 Klausur unter Abiturbedingungen

Anstelle einer Klausur kann gemäß dem Beschluss der Lehrerkonferenz in Q 1.2 eine Facharbeit geschrieben werden.

Die Aufgabentypen, sowie die Anforderungsbereiche I-III sind entsprechend den Vorgaben in Kapitel 3 des Kernlehrplans zu beachten.

Kriterien

Die Bewertung der schriftlichen Leistungen in Klausuren erfolgt über ein Raster mit Hilfspunkten, die im Erwartungshorizont den einzelnen Kriterien zugeordnet sind.

Spätestens ab der Qualifikationsphase orientiert sich die Zuordnung der Hilfspunktsumme zu den Notenstufen an dem Zuordnungsschema des Zentralabiturs.

Von diesem kann aber im Einzelfall begründet abgewichen werden, wenn sich z.B. besonders originelle Teillösungen nicht durch Hilfspunkte gemäß den Kriterien des Erwartungshorizontes abbilden lassen oder eine Abwertung wegen besonders schwacher Darstellung (APO-GOST §13 (2)) angemessen erscheint.

Die Note ausreichend (5 Punkte) soll bei Erreichen von 45 % der Hilfspunkte erteilt werden.

2.3.2 Beurteilungsbereich Sonstige Mitarbeit

Den Schülerinnen und Schülern werden die Kriterien zum Beurteilungsbereich „sonstige Mitarbeit“ zu Beginn des Schuljahres genannt.

Verbindliche Absprachen der Fachkonferenz: In der Qualifikationsphase erstellen, dokumentieren und präsentieren die Schülerinnen und Schüler in Kleingruppen anwendungsbezogene Softwareprojekte. Diese werden in die Note für die Sonstige Mitarbeit einbezogen.

Leistungsaspekte

Mündliche Leistungen

- Beteiligung am Unterrichtsgespräch
- Zusammenfassungen zur Vor- und Nachbereitung des Unterrichts
- Präsentation von Arbeitsergebnissen
- Referate
- Mitarbeit in Partner-/Gruppenarbeitsphasen

Praktische Leistungen am Computer

- Implementierung, Test und Anwendung von Informatiksystemen

Sonstige schriftliche Leistungen

- Sammlung der Projekte in Moodle
- Lernerfolgsüberprüfung durch kurze schriftliche Übungen
- finden mindestens einmal pro Quartal statt
- Bearbeitung von schriftlichen Aufgaben im Unterricht

Kriterien

Die folgenden allgemeinen Kriterien gelten sowohl für die mündlichen als auch für die schriftlichen Formen der sonstigen Mitarbeit. Zur Überprüfung der schriftlichen Formen wird die Plattform Moodle eingesetzt.

Die Bewertungskriterien stützen sich auf

- die Qualität der Beiträge,
- die Quantität der Beiträge und
- die Kontinuität der Beiträge.

Besonderes Augenmerk ist dabei auf

- die sachliche Richtigkeit,
- die angemessene Verwendung der Fachsprache,
- die Darstellungskompetenz,
- die Komplexität und den Grad der Abstraktion,
- die Selbstständigkeit im Arbeitsprozess,
- die Präzision und
- die Differenziertheit der Reflexion zu legen.

Bei Gruppenarbeiten auch auf

- das Einbringen in die Arbeit der Gruppe,
- die Durchführung fachlicher Arbeitsanteile und
- die Qualität des entwickelten Produktes.

Bei Projektarbeit darüber hinaus auf

- die Dokumentation des Arbeitsprozesses,
- den Grad der Selbstständigkeit,
- die Reflexion des eigenen Handelns und
- die Aufnahme von Beratung durch die Lehrkraft.

Grundsätze der Leistungsrückmeldung und Beratung

Die Grundsätze der Leistungsbewertung werden zu Beginn eines jeden Halbjahres den Schülerinnen und Schülern transparent gemacht. Leistungsrückmeldungen können erfolgen

- nach einer mündlichen Überprüfung,
- bei Rückgabe von schriftlichen Leistungsüberprüfungen,
- nach Abschluss eines Projektes,
- nach einem Vortrag oder einer Präsentation,
- bei auffälligen Leistungsveränderungen,
- auf Anfrage,
- als Quartalsfeedback oder als Quartalsnote (in der Sek II)
- zu Elternsprechtagen.

Die Leistungsrückmeldung kann

- durch ein Gespräch mit der Schülerin oder dem Schüler,
 - durch ein Feedback über Moodle,
 - durch eine individuelle Lern-/Förderempfehlung
- erfolgen.

Leistungsrückmeldungen erfolgen auch in der Einführungsphase im Rahmen der kollektiven und individuellen Beratung zur Wahl des Faches Informatik als fortgesetztes Grundkursfach in der Qualifikationsphase.

3. Entscheidungen zu fach- und unterrichtsübergreifenden Fragen

Fach- und aufgabenfeldbezogene sowie übergreifende Absprachen, z.B. zur Arbeitsteilung bei der Entwicklung crosscurricularer Kompetenzen (ggf. Methodentage, Facharbeitsvorbereitung, Schulprofil usw.)

Die Fachkonferenz Informatik hat sich im Rahmen des Schulprogramms für folgende zentrale Schwerpunkte entschieden:

Zusammenarbeit mit anderen Fächern

Im Informatikunterricht werden Kompetenzen anhand informatischer Inhalte in verschiedenen Anwendungskontexten erworben, in denen Schülerinnen und Schülern aus anderen Fächern Kenntnisse mitbringen können. Diese können insbesondere bei der Auswahl und Bearbeitung von Softwareprojekten berücksichtigt werden und in einem hinsichtlich der informatischen Problemstellung angemessenem Maß in den Unterricht Eingang finden.

Vorbereitung auf die Erstellung der Facharbeit

Möglichst schon im zweiten Halbjahr der Einführungsphase, spätestens jedoch im ersten Halbjahr des ersten Jahres der Qualifikationsphase werden im Unterricht an geeigneten Stellen Hinweise zur Erstellung von Facharbeiten gegeben. Das betrifft u. a. Themenvorschläge, Hinweise zu den Anforderungen und zur Bewertung. Es wird vereinbart, dass nur Facharbeiten vergeben werden, die mit der eigenständigen Entwicklung eines Softwareproduktes verbunden sind.

4. Qualitätssicherung und Evaluation

Das schulinterne Curriculum stellt keine starre Größe dar, sondern ist als „lebendes Dokument“ zu betrachten. Dementsprechend sind die Inhalte stetig zu überprüfen, um ggf. Modifikationen vornehmen zu können. Die Fachkonferenz (als professionelle Lerngemeinschaft) trägt durch diesen Prozess zur Qualitätsentwicklung und damit zur Qualitätssicherung des Faches bei.

Durch Diskussion der Aufgabenstellung von Klausuren in Fachdienstbesprechungen und eine regelmäßige Erörterung der Ergebnisse von Leistungsüberprüfungen wird ein hohes Maß an fachlicher Qualitätssicherung erreicht.

Das schulinterne Curriculum für die Qualifikationsphase (siehe 2.1) ist zunächst bis 2017 für den ersten Durchgang durch die gymnasiale Oberstufe nach Erlass des Kernlehrplanes verbindlich. Nach Abschluss des Abiturs 2017 wird die Fachkonferenz Informatik auf der Grundlage ihrer Unterrichtserfahrungen eine Gesamtsicht des schulinternen Curriculums vornehmen und ggf. eine Beschlussvorlage für die erste Fachkonferenz des folgenden Schuljahres erstellen.