

Schulinterner Lehrplan zum Kernlehrplan für die gymnasiale Oberstufe

im Fach Informatik

Stand: 03.06.2016 (Überarbeitung 01/2025)

1 Vorstellung des Informatikunterrichts am Ratsgymnasium Münster	3
1.1 Vorstellung der Schule und der Fachgruppe	3
1.2 Informatik in der Klasse 5 und 6	3
1.3 Informatikunterricht im Wahlpflichtbereich II (Klassen 9+10).....	4
1.4 Informatikunterrichts in der Gymnasialen Oberstufe	4
2 Entscheidungen zum Unterricht	5
2.1 Unterrichtsvorhaben	5
2.1.1 Einführungsphase	5
2.1.2 Qualifikationsphase	15
2.2 Grundsätze der Leistungsbewertung und Leistungsrückmeldung	43
2.2.1 Vorbemerkungen	43
2.2.2 Bewertungsbereich Sonstige Mitarbeit	44
2.2.3 Klausuren	45
3 Weitere Hinweise und Entscheidungen	46
4 Qualitätssicherung und Evaluation	47
4.1. Kollegialer Austausch.....	47
4.2. Evaluation durch die Schülerinnen und Schüler	47
5 Kompetenzerwartung und inhaltliche Schwerpunkte in der EF.....	48

1 Vorstellung des Informatikunterrichts am Ratsgymnasium Münster

1.1 Vorstellung der Schule und der Fachgruppe

Das im Jahr 1851 gegründete Ratsgymnasium ist das älteste städtische Gymnasium in Münster. Wir stehen damit in einer fast 160jährigen städtischen Bildungstradition. Im Schuljahr 2022/23 besuchen ca. 700 Schülerinnen und Schüler das Ratsgymnasium. Sie werden unterrichtet von 68 Lehrerinnen und Lehrern, die sich bewusst für den Ganzttag und das Schulprofil des Ratsgymnasiums entschieden haben. Zum Schulprofil gehören traditionell die Sprachen und Naturwissenschaften (sog. MINT-Fächer) als Schwerpunktfächer. Mit diesem Zwei-Säulen-Profil bereitet unser Unterricht in der Sekundarstufe I die Schülerinnen und Schüler hervorragend und

zielführend auf die Anforderungen in der gymnasialen Oberstufe vor. In drei voll ausgestatteten Rechnerräumen mit jeweils 16 Arbeitsplätzen kann der Unterricht durchgeführt werden. Ein Selbstlernzentrum ermöglicht es den Schülerinnen und Schülern in der GOST, auch außerhalb der Unterrichtszeiten in der gewohnten Umgebung zu arbeiten.

Informatik am Rats

Jgst 5	Informatik (ganzjährig mit einer Wochenstunde)	
Jgst 6	Informatik (ganzjährig mit einer Wochenstunde)	
Jgst 7		
Jgst 8		
Jgst 9	Informatik (WP II) (ganzjährig 4 Wochenstunden)	andere Fächer im WP II
Jgst 10	Informatik (WP II) (ganzjährig 4 Wochenstunden)	
EF	Grundkurs Informatik (3-stündig)	
Q1	Grundkurs Informatik (3-stündig)	
Q2	Grundkurs Informatik (3-stündig)	
Abitur	mdl. oder schriftlich im Grundkurs	

Informatische Inhalte und Fragestellungen werden am Ratsgymnasium bereits in den Jahrgangsstufen 5 und 6, im Wahlpflichtbereich II der Jahrgangsstufen 9 und 10 im Fach Informatik und in der Gymnasialen Oberstufe im Grundkurs behandelt. [siehe Abbildung].

1.2 Informatik in der Klasse 5 und 6

Am Ratsgymnasium erhalten unsere Schülerinnen und Schüler, klassenbezogen und damit für alle Schülerinnen und Schüler verpflichtend, in den Jahrgangsstufen 5 und 6 (jeweils 1 Wochenstunde) Unterricht in einem eigenständigen Fach mit dem Namen „Informatik“.

Die Informationstechnologie (IT) entwickelt sich rasend schnell, die massive Digitalisierung und Vernetzung (Internet) ermöglichen uns den Zugang zu Wissen, Erleichterung von Arbeit und Alltag und sind ein Teil unseres gesellschaftlichen Lebens geworden. Sie reicht in fast alle Bereiche: Vom Lernen über Textverarbeitungen zu Präsentationsmedien, Erstellung von Multimedia-Produkten, Einkaufen, Geldanlagen, Organisation, Kommunikation uvm.. Dadurch wird das Thema Information zu einem Teil der Allgemeinbildung wie die Energie im Fach Physik. Aber während die technischen Voraussetzungen heute in fast allen

Haushalten vorhanden sind, sind doch die Kenntnisse und Fertigkeiten nur eingeschränkt entwickelt, auf eine reine Bedienung fixiert. Der Ansatz im Fach IG geht über typische Produktschulungen zu spezieller Software hinaus, da diese in der Regel auf Bedienerfertigkeiten ausgerichtet sind, die selten übertragbar sind und schnell veralten. Die Folge ist Abhängigkeit von kommerziellen Produkten und kommerzieller Hilfe sowie Überschätzung der Möglichkeiten und Unterschätzung der Gefahren der IT, besonders im Zusammenhang mit dem Internet.

Ein kompetenter, verantwortungsvoller Umgang mit IT-Systemen setzt weitreichende, systematische Kenntnisse über ihren Aufbau, ihre Funktionsweise, ihre innere Struktur, ihre Möglichkeiten und Grenzen voraus, die weit über bloße Bedienerfertigkeiten hinausgeht. Im Fach Informatische Grundbildung werden diese langlebigen, übertragbaren Grundlagen der Informationstechnologie vermittelt. Der Unterricht ist praxisnah, schüler- und problemorientiert konzipiert und vermittelt die moderne objektorientierte Sichtweise auf IT-Systeme nach Möglichkeit spielerisch.

1.3 Informatikunterricht im Wahlpflichtbereich II (Klassen 9+10)

Das Wahlpflichtfach Informatik wird in der Jahrgangsstufe 9 und in der Jahrgangsstufe 10 jeweils vierstündig unterrichtet. Der Unterricht verfolgt in hohem Maße die Kriterien eines „offenen Unterrichts“. Das bedeutet, der Unterricht hat einen hohen Anteil an Projekt-, Gruppen- und Freiarbeit. Schülerinteressen, Beobachtungen aus dem Alltag und aktuelle Ereignisse werden im Unterricht angemessen berücksichtigt.

Für nahezu jeden Unterricht ist mittlerweile der Einsatz digitaler Hilfsmittel unentbehrlich geworden. Der Unterricht in Informatik soll dazu beitragen solche Hilfsmittel sachgerecht, zielgerichtet und verantwortungsvoll einzusetzen. Sie sind dabei stets Medium, Werkzeug und Inhalt des Lernens zugleich ist. Neben der Vermittlung fachlicher Inhalte hat das Fach Informatik aber auch die Aufgabe, alternative Lebenswege aufzuzeigen. Mit dem technisch-ingenieurwissenschaftlichen Bezug bietet gerade dieses Fach den Schülerinnen und Schülern einen Erprobungsraum für die technisch-konstruktive Arbeit und ermöglicht so eine Entscheidungsgrundlage für die weitere Lebensperspektive (z.B. die Berufswahl).

1.4 Informatikunterrichts in der Gymnasialen Oberstufe

Die Vorgaben für das Zentralabitur NRW regeln im Wesentlichen den Ablauf des Informatikunterrichts in der Gymnasialen Oberstufe.

Als Besonderheiten am Ratsgymnasium seien aber genannt:

- Als Entwicklungsumgebungen für den Einstieg in die objektorientierte Programmierung werden Greenfoot und BlueJ/JavaEditor genutzt.

2 Entscheidungen zum Unterricht

2.1 Unterrichtsvorhaben

2.1.1 Einführungsphase

Unterrichtsvorhaben EF-I: Einführung in die Informatik

Bemerkung: Thema und Leitfragen sind vom Unterrichtsvorhaben EF-II nicht trennbar.

Thema: Einführung in die Informatik

Leitfragen: Womit beschäftigt sich das Fach Informatik in der GOST? Was sind die Fragestellungen und Meilensteine der Informatik?

Vorhabenbezogene Konkretisierung:

Das erste Unterrichtsvorhaben stellt eine allgemeine Einführung in das Fach Informatik dar. Dabei sollen die Meilensteine in der Geschichte der Informatik und grundlegende Prinzipien wie das EVA-Prinzip und die Von-Neumann-Architektur erarbeitet werden.

Das Unterrichtsvorhaben ist so strukturiert, dass die Schülerinnen und Schüler anhand bekannter Alltagstechnik die Grundideen fundamentaler informatischer Konzepte (Inhaltsfelder) größtenteils selbstständig erarbeiten und nachvollziehen.

Ausgehend von dem bekannten Bedienungs- und Funktionalitätswissen eines Navigationsgerätes werden die Strukturierung von Daten, das Prinzip der Algorithmik, die Eigenheit formaler Sprachen, die Kommunikationsfähigkeit von Informatiksystemen und die positiven und negativen Auswirkungen auf Mensch und Gesellschaft thematisiert. Das am Navigationsgerät erworbene Wissen kann auf weitere den Schülerinnen und Schülern bekannte Informatiksysteme übertragen werden.

Mit Hilfe eines geschichtlichen Abrisses und populärer informatischer Fallbeispiele sollen die Wechselbeziehung zwischen der Informatik und der Gesellschaft thematisiert und die Auswirkungen des Einsatzes von Informatiksystemen auf die Gesellschaft deutlich werden.

Des Weiteren soll der grundlegende Aufbau eines Rechnersystems im Sinne der Von-Neumann-Architektur erarbeitet werden und mit dem grundlegenden Prinzip der Datenverarbeitung (Eingabe-Verarbeitung-Ausgabe) in Beziehung gesetzt werden.

Das gesamte Vorhaben wird durch eine eintägige Exkursion in das Heinz-Nixdorf-Forum nach Paderborn flankiert.

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien oder Materialien
<p>1. Geschichte und „Geschichtchen“ der Informatik und ihre Auswirkungen auf die heutige Gesellschaft</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • erläutern wesentliche Grundlagen der Geschichte der digitalen Datenverarbeitung (A) • bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen (A) 	<ul style="list-style-type: none"> • Film: „Als die Computer rechnen lernten“ • Fallbeispiele (berühmte Softwarefehler: z.B. Scud-Rakete im Irak-Krieg¹, Softwarefehler bei EC-Karten², Apollo-Mondlandung³, Ariane 5)
<p>2. Was ist Informatik?- Bereiche der Informatik</p> <ul style="list-style-type: none"> • Technische Informatik • Theoretische Informatik • Praktische Informatik • Informatik und Gesellschaft 	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • beschreiben und erläutern die Grundideen fundamentaler informatischer Konzepte (Inhaltsfelder) (A) • bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen (A) 	<p>Kapitel 1 „Was ist Informatik“</p> <ul style="list-style-type: none"> • Als Anschauungsmaterial bieten sich Navigationsgeräte an
<p>3. Aufbau informatischer Systeme</p> <ul style="list-style-type: none"> • Identifikation typischer Arbeitsweisen und Komponenten informatischer Systeme. • Herleitung der VN-Architektur • Identifikation des EVA-Prinzips als Prinzip der Verarbeitung von Daten und Grundlage der VN-Architektur. 	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • beschreiben und erläutern den strukturellen Aufbau und die Arbeitsweise singulärer Rechner am Beispiel der VN-Architektur (A) • nutzen das Internet zur Recherche, zum Datenaustausch und zur Kommunikation 	<ul style="list-style-type: none"> • Aus den vorherigen Beispielen und Diskussionen werden das EVA-Prinzip die in der VN-Architektur vorkommenden Komponenten herausgearbeitet. • Hier ist die Vergabe von Referaten oder eine arbeitsteilige Erarbeitung angedacht.

¹ u.a. http://de.wikipedia.org/wiki/MIM-104_Patriot

² Informatik 1 – Schöningh – Seite 125

³ u.a. <http://www.bernd-leitenberger.de/apollo-ueberfluessige-fakten.shtml>

Unterrichtsvorhaben EF-II: Einführung in die Informatik – Informatisches Problemlösen und Algorithmen als Beschreibung von Abläufen

Bemerkung: Thema und Leitfragen sind vom Unterrichtsvorhaben EF-I nicht trennbar.

Thema: Einführung in die Informatik – Informatisches Problemlösen und Algorithmen als Beschreibung von Abläufen

Leitfragen: Wie beschreibt man in der Informatik Abläufe computernah? Was zeichnet Algorithmen aus und wie kann man sie vergleichen?

Vorhabenbezogene Konkretisierung:

Anhand von alltäglichen Beispielen werden zunächst der Algorithmus-Begriff und seine Eigenschaften herausgearbeitet. Dabei sollen Algorithmen sowohl umgangssprachlich wie auch im PAP dargestellt. Bedeutsam ist dabei die Herausarbeitung der grundlegenden algorithmischen Strukturen (Sequenz, bedingte Verzweigung und kopfgesteuerte Schleife) und die Überprüfung der Eigenschaften von Algorithmen.

Im Anschluss werden anhand von konkreten Algorithmen die erarbeiteten Techniken geübt, Fragestellung zu Grenzen von Algorithmen thematisiert und die Idee der rekursiv formulierten Algorithmen erarbeitet.

Beim Labyrinth-Problem führt zum ersten Mal auf Modellentscheidungen (welche Sensoren hat der Akteur? Welche Aufträge und Anfragen liegen vor?) und dient als Übergang zur ersten Implementierung in Java (Unterrichtsvorhaben EF-III).

Zeitbedarf: 10 Stunden

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien oder Materialien
1. Der Algorithmus-Begriff	Die Schülerinnen und Schüler <ul style="list-style-type: none">kennen den Begriff „Algorithmus“ und seine Eigenschaften,stellen Algorithmen umgangssprachlich und grafisch (PAP) dar,analysieren und erläutern einfache Algorithmen (A)	<ul style="list-style-type: none">Ausgehend von den berühmten Softwarefehler⁴ und mit Hilfe alltäglicher Beispiele wird der Algorithmus-Begriff⁵ entwickelt und an Beispielen überprüft, geschärft und eingeübt.Mögliche Beispiele: Papierflieger (Sequenz), Kochrezept (Sequenz und bedingte Verzweigung) ...

⁴ Unterrichtsvorhaben EF-I

⁵ u.a. <http://www.netzmafia.de/skripten/buecher/perl/algorithmen.pdf>
oder http://intranet.lsg.musin.de/sj1213/images/e/e4/Ppt_EinstiegAlgorithmen_LB.pdf (Folien 1-7)

		<ul style="list-style-type: none"> • Material: PAPDesigner
<p>2. Beispiele für einfache Algorithmen und ihre Darstellung</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • analysieren und erläutern einfache Algorithmen und Programme (A), • entwerfen einfache Algorithmen und stellen sie umgangssprachlich und grafisch dar (M), • modifizieren einfache Algorithmen (A), • testen Programme schrittweise anhand von Beispielen (A) 	<ul style="list-style-type: none"> • Problem der Wegsuche (Dijkstra) → Navigationsgeräte: siehe Arbeitsmaterial in „Abenteuer Informatik“ (Gallenbacher, S.1-38) • Euklidischer Algorithmus⁶: Darstellung eines iterativen mathematischen Algorithmus mit Hilfe eines PAPs (Bedingte Verzweigung, kopfgesteuerte Schleife) und ugs. Beschreibung • Labyrinth-Problem: Die Rechte-Hand-Regel Darstellung in Umgangssprache und im PAP / Testen von Algorithmen / Übertragung auf vergleichbare Labyrinth / Grenze des Algorithmus (Problemklasse) • Türme von Hanoi⁷: Herausarbeiten von Regeln und Fragestellung aus der Geschichte, Reduzierung, praktisches Ausprobieren, Darstellung als Algorithmus, Zeitproblematik (Grenze der Umsetzung des Algorithmus) Rekursive Darstellung; Eigenschaften von Rekursionen (Selbstaufufruf mit Laufbedingung bei Verkleinerung des Problems)

⁶ z.B. http://de.wikipedia.org/wiki/Euklidischer_Algorithmus

⁷ Simulation und Geschichte unter

<http://www.blinde-kuh.de/spiele/hanoi/>

vertiefende Materialien: <http://www.mathematik.uni-muenchen.de/~hinz/turm.pdf>

Unterrichtsvorhaben EF-III: Einführung in die OOP mit Greenfoot

Thema: Einführung in die OOP mit Greenfoot

Leitfragen: Was sind Klassen, Objekte, Methoden, Attribute? Wie interagieren Objekte unterschiedlicher und gleicher Klassen miteinander? Wie werden Algorithmen entwickelt, dargestellt und in der Programmiersprache Java umgesetzt?

Vorhabenbezogene Konkretisierung:

Dieses Unterrichtsvorhaben beinhaltet eine Einführung in die OOP. Dabei werden Algorithmen erstellt und implementiert. In Kurzprojekten werden die Klassen und ihre Beziehungen in Form von UML-Diagrammen modelliert.

Begonnen wird mit einem Greenfoot-Szenario (i.d.R. Roboter, Ameisen, Krabben o.ä.). Nach der sukzessiven Einführung der zu behandelnden Kontrollstrukturen Verzweigung, Schleifen werden zunehmend komplexere Problemstellungen von den SuS erarbeitet und gelöst, wobei die Modellierung als Vorstufe der Implementation eingefordert wird.

In einem weiteren Schritt sollen Objekte miteinander interagieren können (z.B. Roboter und Akku, zwei Roboter). Hierbei wird der Unterschied zwischen zwei Beziehungsarten (Assoziation und Vererbung) erarbeitet und visualisiert.

Zeitbedarf: 25 Stunden

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien oder Materialien
<p>a) Grundlagen der OOP/Algorithmik</p> <ul style="list-style-type: none"> • Erarbeitung der Bedeutung von Klassen, Objekten und Methoden • Darstellung von Klassen und Objekten in UML-Notation • Benutzung von Objekten • Erlernen grundlegender Algorithmen mit Kontrollstrukturen • Erarbeitung des Unterschieds zwischen Aufträgen und Abfragen anhand von Anwendungsbeispielen <p>b) Objektbeziehungen modellieren und umsetzen</p> <ul style="list-style-type: none"> • Analyse von Problemstellungen, bei denen die Interaktion von Objekten notwendig ist • Darstellung der Kennt- und Ist-Beziehung in Form von Klassendiagrammen • Erarbeitung, Programmierung und Bewertung verschiedener Umsetzungsmöglichkeiten der Beziehungen <p>c) Einführung des Variablenkonzepts/Datentypen</p> <ul style="list-style-type: none"> • Problemorientierte Erarbeitung der Benutzung von Variablen als universellem Speicher. • Benutzung der Datentypen Integer, String, Char, Boolean. <p>d) Vertiefung der Inhalte anhand eines weiteren Anwendungsbeispiels</p>	<p>Die Schülerinnen und Schüler...</p> <ul style="list-style-type: none"> • ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M). • stellen den Zustand eines Objekts dar (D). • analysieren und erläutern einfache Algorithmen und Programme (A). • modifizieren einfache Algorithmen und Programme (I). • entwerfen einfache Algorithmen und stellen sie umgangssprachlich und grafisch dar (M). • dokumentieren Klassen durch Beschreibung der Funktionalität der Methoden (D). • analysieren und erläutern eine objektorientierte Modellierung (A) • modellieren Klassen mit ihren Attributen, ihren Methoden und Assoziationsbeziehungen (M). • modellieren Klassen unter Verwendung von Vererbung (M). • stellen die Kommunikation zwischen Objekten grafisch dar (M). • stellen Klassen, Assoziations- und Vererbungsbeziehungen in Diagrammen grafisch dar (D). • analysieren und erläutern eine objektorientierte Modellierung (A). • implementieren Algorithmen unter Verwendung von Variablen und 	<p>Greenfoot</p> <p>Szenarien für Greenfoot, z.B.:</p> <p>Wombats, Roboter, Krabben, Ameisen usw.</p> <p>PAP-Designer</p>

	<p>Wertzuweisungen, Kontrollstrukturen sowie Methodenaufrufen (I).</p> <ul style="list-style-type: none">• testen Programme schrittweise anhand von Beispielen (I).• nutzen das verfügbare Informatiksystem zur strukturierten Verwaltung und gemeinsamen Verwendung von Daten unter Berücksichtigung der Rechteverwaltung (K).	
--	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--

Unterrichtsvorhaben EF-IV: Softwareentwicklung

Thema: Softwareentwicklung

Leitfragen: Wie wird Software entwickelt? Welche Phasen durchläuft Software während der Entwicklung? Welche Bereiche werden strikt voneinander getrennt (MVC-Konzept)?

Vorhabenbezogene Konkretisierung:

Schülerinnen und Schüler durchlaufen in diesem UV einen vollständigen Softwareentwicklungsprozess am Beispiel des Spiels „Verflixte 7“.

Im Rahmen der Unterrichtsreihe erlernen sie die einzelnen Phasen OOA , OOD bis hin zur OOP. In der OOA Phase werden beteiligte Objekte mit ihren Fähigkeiten und Eigenschaften mit Hilfe der Substantiv-Verb Methode, Use-Case Diagramme und der CRC Methode identifiziert. Die OOD dient der Konkretisierung der OOA und Implementationsdiagramme werden erstellt.

Der letzte Schritt ist die Umsetzung des Projekts/ Modells mit Java einschließlich der Gestaltung einer grafischen Oberfläche. Sequenzdiagramme helfen das System zu testen und bei Bedarf zu verbessern.

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien oder Materialien
<p>a) Exkurs binäre Zahlen</p> <p>Im Rahmen eines kurzen Exkurses werden Fragestellungen rund um die Binärdarstellung besprochen.</p> <p>b) OOA</p> <p>Die OOA dient der Untersuchung der Aufgabenstellung und führt zu einem ersten UML-Modell. Die SuS benutzen dazu die Fachmethoden: USE-Case Diagramme, Substantiv Verb Methode, CRC Karten</p> <p>c) OOD</p> <p>Die OOD dient der Konkretisierung und führt zu einem Implementationsdiagramm</p> <p>d) OOP</p> <p>Mit BlueJ wird das Modell implementiert und unter anderem mit Hilfe des Debuggers und von Sequenzdiagrammen auf seine Lauffähigkeit getestet.</p> <p>e) MVC Konzept</p> <p>Das Modell wird mit Hilfe des Java-Editors um eine einfache grafische Oberfläche erweitert.</p>	<p>Die Schülerinnen und Schüler ...</p> <ul style="list-style-type: none"> • stellen ganze Zahlen und Zeichen in Binärcodes dar (D), • interpretieren Binärcodes als Zahlen und Zeichen (D). • ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • modellieren Klassen mit ihren Attributen, ihren Methoden und Assoziationsbeziehungen (M), • modellieren Klassen unter Verwendung von Vererbung (M), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen oder lineare Datensammlungen zu (M), • ordnen Klassen, Attributen und Methoden ihren Sichtbarkeitsbereich zu (M), • stellen den Zustand eines Objekts dar (D), • stellen die Kommunikation zwischen Objekten grafisch dar (M), • stellen Klassen, Assoziations- und Vererbungsbeziehungen in Diagrammen grafisch dar (D), • implementieren Klassen in einer Programmiersprache (I). • analysieren und erläutern einfache Algorithmen und Programme (A), • entwerfen einfache Algorithmen und stellen sie umgangssprachlich und grafisch dar (M), • implementieren Algorithmen unter Verwendung von Variablen und Wertzuweisungen, 	<ul style="list-style-type: none"> • Use Case Diagramme • Substantiv-Verb Methode • CRC Karten • Sequenzdiagramme • BlueJ • Java Editor

	<p>Kontrollstrukturen sowie Methodenaufrufen (I),</p> <ul style="list-style-type: none">• nutzen die im Unterricht eingesetzten Informatiksysteme selbstständig, sicher, zielführend und verantwortungsbewusst (D).• implementieren einfache Algorithmen unter Beachtung der Syntax und Semantik einer Programmiersprache (I),• interpretieren Fehlermeldungen und korrigieren den Quellcode (I).• dokumentieren Klassen durch Beschreibung der Funktionalität der Methoden (D).• analysieren und erläutern eine objektorientierte Modellierung (A)	
--	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--

2.1.2 Qualifikationsphase

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 durch Analyse und Erweiterung eines kontextbezogenen Beispiels</p> <p>(a) <i>Analyse der Problemstellung</i></p> <p>(b) <i>Analyse der Modellierung (Implementationsdiagramm)</i></p> <p>(c) <i>Erweiterung der Modellierung im Implementationsdiagramm (Vererbung, abstrakte Klasse)</i></p> <p>(d) <i>Kommunikation zwischen mindestens zwei Objekten (grafische Darstellung)</i></p> <p>(e) <i>Dokumentation von Klassen</i></p> <p>(f) <i>Implementierung der Anwendung oder von Teilen der Anwendung</i></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>Beispiel: Wetthuepfen</i></p> <p>Für ein Wetthüpfen zwischen einem Hasen, einem Hund und einem Vogel werden die Tiere gezeichnet. Alle Tiere springen wiederholt nach links. Die Höhe und Weite jedes Hüpfers ist zufällig. Evtl. marschieren sie anschließend hintereinander her.</p> <p>oder</p> <p><i>Beispiel: Tannenbaum</i></p> <p>Ein Tannenbaum soll mit verschiedenen Arten von Schmuckstücken versehen werden, die durch unterschiedliche geometrische Objekte dargestellt werden. Es gibt Kugeln, Päckchen in der Form von Würfeln und Zuckerringe in Form von Toren.</p> <p>Ein Prototyp, der bereits mit Kugeln geschmückt werden kann, kann zur Verfügung gestellt werden. Da alle Schmuckstücke über die Funktion des Auf- und Abschmückens verfügen sollen, liegt es nahe, dass entsprechende Methoden in einer gemeinsamen Oberklasse realisiert werden.</p> <p>oder</p> <p><i>Beispiel: Tic-Tac-Toe</i></p> <p><i>Materialien:</i></p> <p>Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.1-Wiederholung</p>

Unterrichtsvorhaben Q1-II

Thema: Such- und Sortieralgorithmen anhand kontextbezogener Beispiele

Leitfragen: *Wie können Objekte bzw. Daten effizient sortiert werden, so dass eine schnelle Suche möglich wird?*

Vorhabenbezogene Konkretisierung:

Dieses Unterrichtsvorhaben beschäftigt sich mit der Erarbeitung von Such- und Sortieralgorithmen. Der Schwerpunkt des Vorhabens liegt dabei auf den Algorithmen selbst und nicht auf deren Implementierung in einer Programmiersprache, auf die in diesem Vorhaben vollständig verzichtet werden soll.

Zunächst erarbeiten die Schülerinnen und Schüler mögliche Einsatzszenarien für Such- und Sortieralgorithmen, um sich der Bedeutung einer effizienten Lösung dieser Probleme bewusst zu werden. Anschließend werden Strategien zur Sortierung mit Hilfe eines explorativen Spiels von den Schülerinnen und Schülern selbst erarbeitet und hinsichtlich der Anzahl notwendiger Vergleiche auf ihre Effizienz untersucht.

Daran anschließend werden die erarbeiteten Strategien systematisiert und im Pseudocode notiert. Die Schülerinnen und Schüler sollen auf diese Weise das *Sortieren durch Vertauschen*, das *Sortieren durch Auswählen* und mindestens einen weiteren Sortieralgorithmus, kennen lernen.

Des Weiteren soll das Prinzip der *binären Suche* behandelt und nach Effizienzgesichtspunkten untersucht werden.

Zeitbedarf: 6 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Explorative Erarbeitung eines Sortierverfahrens</p> <p>(a) Sortierprobleme im Kontext informatischer Systeme und im Alltag (z.B. Dateisortierung, Tabellenkalkulation, Telefonbuch, Bundesligatabelle, usw.)</p> <p>(b) Vergleich zweier Elemente als Grundlage eines Sortieralgorithmus</p> <p>(c) Erarbeitung eines Sortieralgorithmus durch die Schülerinnen und Schüler</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none">• beurteilen die Effizienz von Algorithmen am Beispiel von Sortierverfahren hinsichtlich Zeit und Speicherplatzbedarf (A),• entwerfen einen weiteren Algorithmus zum Sortieren (M),• analysieren Such- und Sortieralgorithmen und wenden sie auf Beispiele an (D).	<p><i>Beispiel:</i> Sortieren mit Waage Die Schülerinnen und Schüler bekommen die Aufgabe, kleine, optisch identische Kunststoffbehälter aufsteigend nach ihrem Gewicht zu sortieren. Dazu steht ihnen eine Balkenwaage zur Verfügung, mit deren Hilfe sie das Gewicht zweier Behälter vergleichen können.</p> <p><i>Materialien:</i> Computer science unplugged – Sorting Algorithms, URL: www.csunplugged.org/sorting-algorithms abgerufen: 30. 03. 2014</p>

<p>2. Systematisierung von Algorithmen und Effizienzbetrachtungen</p> <ul style="list-style-type: none"> (a) Formulierung (falls selbst gefunden) oder Erläuterung von mehreren Algorithmen im Pseudocode (auf jeden Fall: Sortieren durch Vertauschen, Sortieren durch Auswählen) (b) Anwendung von Sortieralgorithmen auf verschiedene Beispiele (c) Bewertung von Algorithmen anhand der Anzahl der nötigen Vergleiche (d) Variante des Sortierens durch Auswählen (Nutzung eines einzigen oder zweier Felder bzw. lediglich eines einzigen zusätzlichen Ablageplatzes oder mehrerer neuer Ablageplätze) (e) Effizienzbetrachtungen an einem konkreten Beispiel bezüglich der Rechenzeit und des Speicherplatzbedarfs (f) Analyse des weiteren Sortieralgorithmus (sofern nicht in Sequenz 1 und 2 bereits geschehen) 		<p><i>Beispiele:</i> Sortieren durch Auswählen, Sortieren durch Vertauschen, Quicksort</p> <p>Quicksort ist als Beispiel für einen Algorithmus nach dem Prinzip <i>Teile und Herrsche</i> gut zu behandeln. Kenntnisse in rekursiver Programmierung sind nicht erforderlich, da eine Implementierung nicht angestrebt wird.</p> <p><i>Materialien:</i> Computer science unplugged – Sorting Algorithms, URL: www.csunplugged.org/sorting-algorithms abgerufen: 30. 03. 2014</p>
<p>3. Binäre Suche auf sortierten Daten</p> <ul style="list-style-type: none"> (a) Suchaufgaben im Alltag und im Kontext informatischer Systeme (b) Evtl. Simulationsspiel zum effizienten Suchen mit binärer Suche (c) Effizienzbetrachtungen zur binären Suche 		<p><i>Beispiel:</i> Simulationsspiel zur binären Suche nach Tischtennisbällen</p> <p>Mehrere Tischtennisbälle sind nummeriert, sortiert und unter Bechern verdeckt. Mit Hilfe der binären Suche kann sehr schnell ein bestimmter Tischtennisball gefunden werden.</p> <p><i>Materialien:</i> Computer science unplugged – Searching Algorithms, URL: www.csunplugged.org/searching-algorithms, abgerufen: 30. 03. 2014</p>

Unterrichtsvorhaben Q1-III:

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: 18 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) <i>Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen</i></p> <p>(b) <i>Erarbeitung der Funktionalität der Klasse Queue</i></p> <p>(c) <i>Modellierung und Implementierung der Anwendung unter Verwendung eines oder mehrerer Objekte der Klasse Queue</i></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,	<p><i>Beispiel:</i> Patientenwarteschlange (jeder kennt seinen Nachfolger bzw. alternativ: seinen Vorgänger)</p> <p>Sobald ein Patient in einer Arztpraxis eintrifft, werden sein Name und seine Krankenkasse erfasst. Die Verwaltung der Patientenwarteschlange geschieht über eine Klasse, die hier als Wartezimmer bezeichnet wird. Wesentliche Operationen sind das „Hinzufügen“ eines Patienten und das „Entfernen“ eines Patienten, wenn er zur Behandlung gerufen wird.</p> <p>Die Simulationsanwendung stellt eine GUI zur Verfügung, legt ein</p>

	<p>ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M),</p> <ul style="list-style-type: none"> • modifizieren Algorithmen und Programme (I), • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • testen Programme systematisch anhand von Beispielen (I), • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D). 	<p>Wartezimmer an und steuert die Abläufe. Wesentlicher Aspekt des Projektes ist die Modellierung des Wartezimmers mit Hilfe der Klasse Queue.</p> <p>Anschließend wird der Funktionsumfang der Anwendung erweitert: Patienten können sich zusätzlich in die Warteschlange zum Blutdruckmessen einreihen. Objekte werden von zwei Schlangen verwaltet.</p> <p><i>Materialien:</i></p> <p>Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.2 – Warteschlange</p> <p>(Download Q1-II.1)</p>
<p>2. Die Datenstruktur Stapel im Anwendungskontext unter Nutzung der Klasse Stack</p> <p>(a) <i>Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen</i></p> <p>(b) <i>Erarbeitung der Funktionalität der Klasse Stack</i></p> <p>(c) <i>Modellierung und Implementierung der Anwendung unter Verwendung eines oder mehrerer Objekte der Klasse Stack</i></p>		<p><i>Beispiel:</i> Heftstapel</p> <p>In einem Heftstapel soll das Heft einer Schülerin gefunden werden.</p> <p>oder</p> <p><i>Beispiel:</i> Kisten stapeln</p> <p>In einem Stapel nummerierter Kisten soll eine bestimmte Kiste gefunden und an einen Kunden geliefert werden. Dazu müssen Kisten auf verschiedene Stapel gestapelt und wieder zurückgestellt werden.</p>

<p>3. Die Datenstruktur lineare Liste im Anwendungskontext unter Nutzung der Klasse List</p> <p>(a) <i>Erarbeitung der Vorteile der Klasse List im Gegensatz zu den bereits bekannten linearen Strukturen</i></p> <p>(b) <i>Modellierung und Implementierung einer kontextbezogenen Anwendung unter Verwendung der Klasse List.</i></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>Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.2 - Listen</p>
<p>4. Vertiefung - Anwendungen von Listen, Stapeln oder Schlangen in mindestens einem weiteren Kontext</p>		<p><i>Beispiel:</i> Skispringen</p> <p>Ein Skispringen hat folgenden Ablauf: Nach dem Sprung erhält der Springer eine Punktzahl und wird nach dieser Punktzahl in eine Rangliste eingeordnet. Die besten 30 Springer qualifizieren sich für den zweiten Durchgang. Sie starten in umgekehrter Reihenfolge gegenüber der Platzierung auf der Rangliste. Nach dem Sprung erhält der Springer wiederum eine Punktzahl und wird nach der Gesamtpunktzahl aus beiden Durchgängen in die endgültige Rangliste eingeordnet.</p> <p><i>Beispiel:</i> Terme in Postfix-Notation</p> <p>Die sog. UPN (<i>Umgekehrt-Polnische-Notation</i>) bzw. <i>Postfix-Notation</i> eines Terms setzt den Operator hinter die Operanden. Um einen Term aus der gewohnten Infixschreibweise in einen Term in UPN umzuwandeln</p>

oder 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

		<p>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>Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1-II.3 – Anwendungen für lineare Datenstrukturen</p>
--	--	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Unterrichtsvorhaben Q1-IV:

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: 14 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Suchen von Daten in Listen und Arrays</p> <p>(a) <i>Lineare Suche in Listen und in Arrays</i></p> <p>(b) <i>Binäre Suche in Arrays als Beispiel für rekursives Problemlösen</i></p> <p>(c) <i>Untersuchung der beiden Suchverfahren hinsichtlich ihrer Effizienz (Laufzeitverhalten, Speicherbedarf)</i></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), • 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>Beispiel:</i> Karteiverwaltung</p> <p>Für ein Adressverwaltungsprogramm soll eine Methode zum Suchen einer Adresse geschrieben werden.</p> <p>oder</p> <p><i>Beispiel:</i> Bundesjugendspiele</p> <p>Die Teilnehmer an Bundesjugendspielen nehmen an drei Disziplinen teil und erreichen dort Punktzahlen. Diese werden in einer Wettkampfkarte eingetragen und an das Wettkampfbüro gegeben. Zur Vereinfachung sollte sich das Modell auf die drei Disziplinen „Lauf“, „Sprung“ und „Wurf“ beschränken.</p> <p>Im Wettkampfbüro wird das Ergebnis erstellt. Das Programm soll dafür zunächst den Besten einer Disziplin herausuchen können und später das gesamte Ergebnis nach gewissen Kriterien sortieren können.</p> <p><i>Materialien:</i></p> <p>Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.3 - Suchen und Sortieren</p>

<p>2. Sortieren in Listen und Arrays - Entwicklung und Implementierung von iterativen und rekursiven Sortierverfahren</p> <p>(a) Entwicklung und Implementierung eines einfachen Sortierverfahrens für eine Liste</p> <p>(b) Implementierung eines einfachen Sortierverfahrens für ein Feld</p> <p>(c) Entwicklung eines rekursiven Sortierverfahrens für ein Feld (z.B. Sortieren durch Mischen)</p>		<p><i>Beispiel:</i> Karteiverwaltung (s.o.)</p> <p>oder</p> <p><i>Beispiel:</i> Bundesjugendspiele (s.o.)</p> <p><i>Materialien:</i> (s.o.)</p>
<p>3. Untersuchung der Effizienz der Sortierverfahren „Sortieren durch direktes Einfügen“ und „Quicksort“ auf linearen Listen</p> <p>(a) Grafische Veranschaulichung der Sortierverfahren</p> <p>(b) Untersuchung der Anzahl der Vergleichsoperationen und des Speicherbedarf bei beiden Sortierverfahren</p> <p>(c) Beurteilung der Effizienz der beiden Sortierverfahren</p>		<p><i>Beispiel:</i> Karteiverwaltung (s.o.)</p> <p>oder</p> <p><i>Beispiel:</i> Bundesjugendspiele (s.o.)</p> <p><i>Materialien:</i> (s.o.)</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: 18 Stunden

Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Nutzung von relationalen Datenbanken</p> <p>(a) <i>Aufbau von Datenbanken und Grundbegriffe</i></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, Datenbank-schema <p>(b) <i>SQL-Abfragen</i></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) <i>Vertiefung an einem weiteren Datenbankbeispiel</i></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), • 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), 	<p><i>Beispiel: VideoCenter</i></p> <p>VideoCenter ist die Simulation einer Online-Videothek für den Informatik-Unterricht mit Webfrontends zur Verwaltung der Kunden, der Videos und der Ausleihe. Außerdem ist es möglich direkt SQL-Abfragen einzugeben. Es ist auch möglich, die Datenbank herunter zu laden und lokal zu installieren. Unter http://dokumentation.videocenter.schule.de/old/video/index.html (abgerufen: 30. 03. 2014) findet man den Link zu dem VideoCenter-System sowie nähere Informationen. Lesenswert ist auch die dort verlinkte „Dokumentation der Fallstudie“ mit didaktischem Material, welches alternativ bzw. ergänzend zu der im Folgenden beschriebenen Durchführung verwendet werden kann.</p> <p><i>Beispiel: Schulbuchausleihe</i></p> <p>Unter www.brd.nrw.de/lern-treffs/informatik/structure/material/sek2/datenbanken.php (abgerufen: 30. 03. 2014) wird eine Datenbank zur Verfügung gestellt, die Daten einer Schulbuch-Ausleihe enthält (über 1000 Entleiher, 200 Bücher mit mehreren tausend Exemplaren und viele Ausleihvorgänge). Die Datenbank kann in OpenOffice eingebunden werden.</p>

2. Modellierung von relationalen Datenbanken

(a) Entity-Relationship-Diagramm

- 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

(b) Entwicklung einer Datenbank aus einem Datenbankentwurf

- Modellierung eines relationalen Datenbankschemas zu einem Entity-Relationship-Diagramm inklusive der Bestimmung von Primär- und Sekundärschlüsseln

(c) Redundanz, Konsistenz und Normalformen

- 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)

- überprüfen Datenbankschemata auf vorgegebene Normalisierungseigenschaften (D).

Beispiel: Fahrradverleih

Der Fahrradverleih *BTR (BikesToRent)* verleiht unterschiedliche Typen von Fahrrädern diverser Firmen an seine Kunden. Die Kunden sind bei *BTR* registriert (Name, Adresse, Telefon). *BTR* kennt von den Fahrradfirmen den Namen und die Telefonnummer. Kunden von *BTR* können CityBikes, Treckingräder und Mountainbikes ausleihen.

Beispiel: Reederei

Die Datenverwaltung einer Reederei soll in einem Datenbanksystem umgesetzt werden. Ausgehend von der Modellierung soll mit Hilfe eines ER-Modells und eines Datenbankschemas dieser erste Entwurf normalisiert und in einem Datenbanksystem umgesetzt werden. Es schließen sich diverse SQL-Abfragen an, wobei auf die Relationenalgebra eingegangen wird.

Beispiel: Buchungssystem

In dem Online-Buchungssystem einer Schule können die Lehrer Medienräume, Beamer, Laptops, Kameras, usw. für einen bestimmten Zeitpunkt buchen, der durch Datum und die Schulstunde festgelegt ist. Dazu ist die Datenbank zu modellieren, ggf. zu normalisieren und im Datenbanksystem umzusetzen. Weiter sollen sinnvolle Abfragen entwickelt werden.

Unter <http://mrbs.sourceforge.net> (abgerufen: 30.03. 2014) findet man ein freies Online-Buchungssystem inklusive Demo, an Hand derer man erläutern kann, worum es in dem Projekt geht.

Beispiel: Schulverwaltung

		<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) <i>Beschreibung eines Datenbankzugriffs im Netz anhand eines Anwendungskontextes und einer Client-Server-Struktur zur Klärung der Funktionsweise eines Datenbankzugriffs</i></p> <p>(b) <i>Netztopologien als Grundlage von Client-Server-Strukturen und TCP/IP-Schichtenmodell als Beispiel für eine Paketübermittlung in einem Netz</i></p> <p>(c) <i>Vertraulichkeit, Integrität, Authentizität in Netzwerken sowie symmetrische und asymmetrische kryptografische Verfahren (Cäsar-, Vigenère-, RSA-Verfahren) als Methoden Daten im Netz verschlüsselt zu übertragen</i></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>Materialien:</i></p> <p>Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben, Verschlüsselung Q1.5 - Zugriff auf Daten in Netzwerken</p>
<p>2. Fallbeispiele zur Datenschutzproblematik und zum Urheberrecht</p>		<p><i>Materialien:</i></p> <p>Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1 5 - Datenschutz beim Videocenter, Materialblatt-Datenschutzgesetz</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) <i>Grundlegende Begriffe (Grad, Tiefe, Höhe, Blatt, Inhalt, Teilbaum, Ebene, Vollständigkeit)</i></p> <p>(b) <i>Aufbau und Darstellung von binären Bäumen anhand von Baumstrukturen in verschiedenen Kontexten</i></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), • 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 Spezialisieren und Generalisieren (M), • 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 	<p><i>Beispiel:</i> Termbaum Der Aufbau von Termen wird mit Hilfe von binären Baumstrukturen verdeutlicht.</p> <p>oder</p> <p><i>Beispiel:</i> Ahnenbaum Die binäre Baumstruktur ergibt sich daraus, dass jede Person genau einen Vater und eine Mutter hat.</p> <p><i>Weitere Beispiele für Anwendungskontexte für binäre Bäume:</i></p> <p><i>Beispiel:</i> 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><i>Beispiel:</i> Entscheidungsbäume Um eine Entscheidung zu treffen, werden mehrere Fragen mit ja oder nein beantwortet. Die Fragen, die möglich sind, wenn die 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>oder</p> <p><i>Beispiel:</i> Codierungsbäume 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</p>

	<p>Programmiersprache bei der Implementierung und zur Analyse von Programmen (I),</p> <ul style="list-style-type: none"> • 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>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><i>Materialien:</i> Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 – Binärbaum</p>
<p>2. Die Datenstruktur Binärbaum im Anwendungskontext unter Nutzung der Klasse Binary-Tree</p> <p>(a) <i>Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen im Anwendungskontext</i></p> <p>(b) <i>Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramms</i></p> <p>(c) <i>Erarbeitung der Klasse BinaryTree und beispielhafte Anwendung der Operationen</i></p> <p>(d) <i>Implementierung der Anwendung oder von Teilen der Anwendung</i></p> <p>(e) <i>Traversierung eines Binärbaums im Pre-, In- und Postorderdurchlauf</i></p>		<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 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> Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 – Binärbaum</p>
<p>3. Die Datenstruktur binärer Suchbaum im</p>		<p><i>Beispiel:</i> Informatikerbaum als Suchbaum</p>

<p>Anwendungskontext unter Verwendung der Klasse <i>BinarySearchTree</i></p> <p>(a) <i>Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen</i></p> <p>(b) <i>Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramm, grafische Darstellung eines binären Suchbaums und Erarbeitung der Struktureigenschaften</i></p> <p>(c) <i>Erarbeitung der Klasse <i>BinarySearchTree</i> und Einführung des Interface <i>Item</i> zur Realisierung einer geeigneten Ordnungsrelation</i></p> <p>(d) <i>Implementierung der Anwendung oder von Teilen der Anwendung inklusive einer sortierten Ausgabe des Baums</i></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> Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 – Binärer Suchbaum</p>
<p>4. Übung und Vertiefungen der Verwendung von Binärbäumen oder binären Suchbäumen anhand weiterer Problemstellungen</p>		<p><i>Beispiel:</i> Codierungsbäume (s.o.) oder Huffman-Codierung</p> <p>oder</p> <p><i>Beispiel:</i> 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 <i>BinarySearchTree</i>) 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</p>

		<p>Teilbaum. (Dies gilt für alle Teilbäume.)</p> <p><i>oder</i></p> <p><i>Beispiel:</i> Entscheidungsbäume (s.o.)</p> <p><i>oder</i></p> <p><i>Beispiel:</i> Termbaum (s.o.)</p> <p><i>oder</i></p> <p><i>Beispiel:</i> Ahnenbaum (s.o.)</p> <p><i>Materialien:</i> Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 – Anwendung Binärbaum</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 Schüleriinnen und Schülern bekannten Kontexten zur formalen Beschreibung eines endlichen Automaten</p> <p>(b) Untersuchung, Darstellung und Entwicklung endlicher Automaten</p>	<p>Die Schüleriinnen 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> Cola-Automat, Geldspielautomat, Roboter, Zustandsänderung eines Objekts „Auto“, Akzeptor für bestimmte Zahlen, Akzeptor für Teilwörter in längeren Zeichenketten, Akzeptor für Terme</p> <p><i>Materialien:</i> Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.2 – Endliche Automaten, Formale Sprachen</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>	<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> reguläre Grammatik für Wörter mit ungerader Parität, Grammatik für Wörter, die bestimmte Zahlen repräsentieren, Satzgliedergrammatik</p> <p><i>Materialien: (s.o.)</i></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> Addition von 4 zu einer eingegebenen Zahl mit einem Rechnermodell</p> <p><i>Materialien:</i> Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.3 –Von-Neumann-Architektur und maschinennahe Programmierung</p>

<p>2. Grenzen der Automatisierbarkeit</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 		<p><i>Beispiel:</i> Halteproblem</p> <p><i>Materialien:</i> Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.3 - Halteproblem (Download Q2-III.2)</p>
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Unterrichtsvorhaben Q2-IV:

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

2.2 Grundsätze der Leistungsbewertung und Leistungsrückmeldung

2.2.1 Vorbemerkungen

Die rechtlich verbindlichen Vorgaben zur Leistungsbewertung finden sich in § 48 SchulG, in § 6 der APO-SI sowie den §§ 13-17 der APO-GOST. Danach ist eine rein rechnerische Bildung von Abschlussnoten unzulässig. Die Fachkonferenz legt nach § 70 SchulG Grundsätze zu Verfahren und Kriterien der Leistungsbewertung fest. Sie orientiert sich dabei an den in den Lehrplänen für die Sekundarstufen I und II ausgewiesenen Kompetenzen.

Das fachbezogene Leistungskonzept ist für alle Mitglieder einer Fachschaft verbindlich. Es soll für ein möglichst hohes Maß an Transparenz und Vergleichbarkeit von Leistungsbeurteilungen sorgen. Rückfragen zum Leistungsstand richten Schülerinnen und Schüler sowie Eltern bitte immer zunächst an die unterrichtenden Fachlehrerinnen und Fachlehrer.

Die Grundsätze der Leistungsbewertung werden den Schülerinnen und Schülern immer zum Schuljahresbeginn, bei Lehrerwechsel auch zum Halbjahresbeginn mitgeteilt. Ein Hinweis darauf wird im Klassenbuch / Kursheft vermerkt.

Kriterien der Leistungsbewertung im Zusammenhang mit konkreten, insbesondere offenen Arbeitsformen werden den Schülerinnen und Schülern grundsätzlich vor deren Beginn transparent gemacht.

Jede Lehrerin / jeder Lehrer dokumentiert regelmäßig und kontinuierlich die von den Schülerinnen und Schülern erbrachten Leistungen.

Die Leistungsrückmeldung erfolgt in regelmäßigen Abständen (zumindest zum Quartalsende) differenziert und individuell in schriftlicher oder mündlicher Form.

Bei Minderleistungen erhalten die Schülerinnen und Schüler sowie ihre Erziehungsberechtigten im Zusammenhang mit den Halbjahreszeugnissen individuelle Lern- und Förderempfehlungen, die die Lernenden - ihrem jeweiligen Leistungsstand entsprechend - zum Weiterlernen ermutigen, indem sie Hinweise zu Erfolg versprechenden individuellen Lernstrategien geben. Den Eltern werden im Rahmen der Lern- und Förderempfehlungen Wege aufgezeigt, wie sie das Lernen ihrer Kinder unterstützen können.

Bei Elternsprechtagen und im Rahmen regelmäßiger Sprechstunden erhalten die Erziehungsberechtigten Gelegenheit, sich über den Leistungsstand ihrer Kinder zu informieren und dabei Perspektiven für die weitere Lernentwicklung zu besprechen.

2.2.2 Bewertungsbereich Sonstige Mitarbeit

Die Beurteilung der sonstigen Mitarbeit erfolgt gemäß dem Lehrplan Informatik. Sie erfasst die Qualität, die Quantität und die Kontinuität verschiedener Beiträge. Für die Bewertung der Leistungen sind sowohl Inhalts- als auch Darstellungsleistungen zu berücksichtigen. Die sonstige Mitarbeit wird dabei in einem kontinuierlichen Prozess während des Schulhalbjahres festgestellt.

Formen der sonstigen Mitarbeit

- Mündliche Beteiligung am Unterrichtsgespräch
 - Regelmäßige Beteiligung (quantitativ und qualitativ)
 - Wiedergabe von Kenntnissen und Prozessen
 - Darstellung von Problemsituationen
 - Beiträge zur Entwicklung von Problemlösungen und Bewertung von Arbeitsständen und –ergebnissen
 - Vorstellung von Arbeitsergebnissen aus Gruppen- oder Partnerarbeit
 - Nutzung adäquater Fachsprache
- Selbstständige Arbeit zu zweit oder in Gruppen.
 - Mitarbeit bei der Organisation (Absprachen, Arbeitseinteilung, Schnittstellenvereinbarung, gegenseitige Hilfen) der gemeinsamen Arbeit
 - Kooperatives Verhalten und gemeinsame konstruktive Suche nach Lösungswegen bei den gestellten Problemen
 - Erreichen des Arbeitszieles unter Einsatz der zur Verfügung stehenden Materialien und der erlernten Arbeitstechniken
- Praktische Arbeit am Computer
 - Vor- und Nachbereitung der Rechnerarbeitsphasen (u.a. Dokumentationen)
 - Kooperatives Arbeiten am Rechner und Beratung anderer Arbeitsgruppen
 - Sinnstiftender Umgang mit den vorhandenen Ressourcen (Nutzung von Hardware, Werkzeugen, insbesondere Umgang mit der Programmierumgebung, angemessene Reaktion auf Fehlermeldungen usw.)
- Anfertigung und Präsentation von Hausaufgaben
 - Vor- und Nachbereitung von Unterrichtsstunden
 - Intensive und möglichst selbstständige Auseinandersetzung mit Materialien und schriftlichen Informationen

- Selbstständige Weiterarbeit und Vervollständigung von Programmierlösungen (auch mit Hilfe des SLZ).
- Bis zu eine schriftlichen Übung pro Halbjahr
- Evtl. Erstellen von Protokollen
- Evtl. Referate

Die Reihenfolge der oben angegebenen Punkte spiegelt die Bedeutung bei der Beurteilung im Sinne einer Rangfolge wider, wobei nach konkretem Unterrichtsverlauf die Gewichtungen abweichen können.

2.2.3 Klausuren

Es ist Beschluss der Fachschaft, im 1. Halbjahr der Einführungsphase nur eine Klausur (2. Quartal) zu schreiben. Die Dauer der Klausuren ist laut APO-GOST vorgegeben.

Kompetenzen: Analyse, Algorithmik, Programmiertechnik, Modellierung, Umgang mit den vorhandenen Ressourcen

Materialgrundlage: vorhandene Ressourcen

Aufgabenformate: Reproduktive Aufgabenteile, Modellierung und Realisierung von Programmen

zugelassene Hilfsmittel: keine

- Einführungsphase: 1 Klausur im 1. Halbjahr, 2 Klausuren im 2. Halbjahr, Dauer der Klausur: 2 Unterrichtsstunden
- Grundkurse Q 1: 2 Klausuren je Halbjahr, Dauer der Klausuren: 2 Unterrichtsstunden
- Grundkurse Q 2.1: 2 Klausuren, Dauer der Klausuren: 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.

Übersicht über Gewichtung/Punkteverteilung - Bewertungskriterien

Jede Aufgabe wird nach Bedeutung der überprüften Lernziele, Anteil der erwarteten Arbeitszeit, Umfang und Komplexität der einzubringenden Teilleistungen, Grad der geforderten Selbstständigkeit und Art und Form der Darstellung entsprechend bepunktet und dieses den Schülerinnen und Schülern offengelegt. Der Hilfscharakter eines solchen Bewertungsverfahrens ist zu betonen und die Gesamtnote ist nicht unbedingt und unflexibel aus der erreichten Prozentzahl zu bestimmen.

Notentabelle:

Durch Leistungen mit vorwiegend wiederholendem Charakter ist eine ausreichende Bewertung erzielen zu können. In der Regel reichen 40% der Maximalpunktzahl zu einem ausreichenden Ergebnis. Die Notenstufen ergeben sich i.a. daraus aus äquidistanter Einteilung.

Die Dokumentation der Leistungsbewertung enthält Kriterienraster und Förderhinweise.

3 Weitere Hinweise und Entscheidungen

Innerhalb der EF soll eine Exkursion zum Heinz-Nixdorf-Forum nach Paderborn unternommen werden, bei dem die im Lehrplan thematisierten Bereiche aus der Geschichte der Informatik und dem Bereich „Informatik, Mensch und Gesellschaft“ aufgegriffen werden.

Die Fachschaft Informatik strebt an, das Buch „Informatik 2“ aus dem Schöningh-Verlag nach Erscheinen (ca. August 2015) als Lehrbuch für die Q1 und Q2 einzuführen.

4 Qualitätssicherung und Evaluation

4.1. Kollegialer Austausch

Durch einen regelmäßigen Austausch innerhalb der Fachschaft im Hinblick auf unterrichtlich eingesetzte Materialien und Methoden wird eine Homogenisierung des Unterrichts angestrebt. Zudem ist der Austausch von gestellten Klausuren obligatorisch.

Ein kollegialer Austausch findet auch zwischen unserer Fachschaft und der Informatik-Fachschaft des kooperierenden Schlaungymnasiums statt.

Das schulinterne Curriculum ist zunächst bis 2017 für den ersten Durchgang durch die gymnasiale Oberstufe nach Erlass des Kernlehrplanes verbindlich. Erstmalig nach Ende der Einführungsphase im Sommer 2015, werden in einer Sitzung der Fachkonferenz Erfahrungen ausgetauscht und ggf. Änderungen für den nächsten Durchgang der Einführungsphase beschlossen, um erkannten ungünstigen Entscheidungen schnellstmöglich entgegenwirken zu können.

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.

4.2. Evaluation durch die Schülerinnen und Schüler

Eine Evaluation hinsichtlich in der Fachgruppe strittig diskutierter Entscheidungen im neuen schulinternen Lehrplan soll den Diskussions- und Entscheidungsprozess in der Fachschaft zum Ende des Sj 2016/2017 unterstützen.

Entsprechend sollen die Leistungen der Schülerinnen und Schüler in der Qualifikationsphase kritisch auf die Planungsentscheidungen der EF reflektiert werden.

5 Kompetenzerwartung und inhaltliche Schwerpunkte in der EF

Inhaltsfeld 1: Daten und ihr Strukturierung

Inhaltsfelder (Gegenstände) Kompetenzbereiche (Prozesse)	Kompetenzerwartungen	Didaktische Hinweise und Vereinbarungen	Hinweise und Vereinbarungen zu geeigneten Kontexten
Die SuS ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M)	<p>Die SuS können anhand alltäglicher Beispiele Klassen identifizieren, wesentliche Eigenschaften und Fähigkeiten benennen, sowie Aufträge und Anfragen mit Art des Rückgabewerts unterscheiden und die Klasse grafisch darstellen.</p> <p>Sie unterscheiden zwischen Klassen und konkreten Exemplaren.</p>	<p>Bei der Identifizierung der Rückgabewerte genügt die Einteilung in Wahrheitswert, Zahl und Text. Eine Unterscheidung zwischen Ganzzahl und Kommazahl wird nicht erwartet.</p> <p>Als Darstellungsform wird das UML-Diagramm genutzt.</p> <p>Fachsprachlich wird die Vorstellung von Klassen als Baupläne und von Objekten als Exemplare vermittelt, wobei das Wort „Objekt“ aufgrund seiner Doppeldeutigkeit zunächst vermieden wird.</p>	Alltägliche Beispiele können Schüler (Mitschüler), Lehrer, Klassenräume, Tiere ...sein.
modellieren Klassen mit ihren Attributen, ihren Methoden und Assoziationsbeziehungen (M)			
modellieren Klassen unter Verwendung von Vererbung (M)			

Inhaltsfelder (Gegenstände) Kompetenzbereiche (Prozesse)	Kompetenzerwartungen	Didaktische Hinweise und Vereinbarungen	Hinweise und Vereinbarungen zu geeigneten Kontexten
ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen oder lineare Datensammlungen zu (M)			
ordnen Klassen, Attributen und Methoden ihren Sichtbarkeitsbereich zu (M)			
stellen den Zustand eines Objekts dar (D)			
stellen die Kommunikation zwischen Objekten grafisch dar (M)			
stellen Klassen, Assoziations- und Vererbungsbeziehungen in Diagrammen grafisch dar (D)			
dokumentieren Klassen durch Beschreibung der Funktionalität der Methoden (D)			
analysieren und erläutern eine objektorientierte Modellierung (A)			
implementieren Klassen in einer Programmiersprache auch unter Nutzung			

Inhaltsfelder (Gegenstände) Kompetenzbereiche (Prozesse)	Kompetenzerwartungen	Didaktische Hinweise und Vereinbarungen	Hinweise und Vereinbarungen zu geeigneten Kontexten
dokumentierter Klassenbibliotheken (I).			

Inhaltsfeld 2: Algorithmen

Inhaltliche Schwerpunkte:

- Analyse, Entwurf und Implementierung einfacher Algorithmen

- Algorithmen zum Suchen und Sortieren

Analyse, Entwurf und Implementierung einfacher Algorithmen:

Inhaltsfelder (Gegenstände) Kompetenzbereiche (Prozesse)	Kompetenzerwartungen	Didaktische Hinweise und Vereinbarungen	Hinweise und Vereinbarungen zu geeigneten Kontexten
Die SuS analysieren und erläutern einfache Algorithmen und Programme (A)	Die SuS können im Rahmen des Programmierprojekts Methoden mit einfachen Kontrollstrukturen verstehen und die Funktionsweise erklären.	Verzweigungen und Schleifen werden von den SuS erkannt und in der Wirkung beschrieben. Als Darstellungsform wird das PAP benutzt.	Problemorientierte Beispiele wie z.B. Mauern oder das Ende der Spielwelt des betrachteten Szenarios dienen als Gegenstand zur Erarbeitung. Werkzeuge sind Greenfoot, BlueJ, StruktED und PAP-Designer.
Die SuS modifizieren einfache Algorithmen und Programme (I),	Die SuS können Methoden je nach Aufgabe verändern und erweitern.	Bedingungen bei den Verzweigungen und Abbruchbedingungen bei Schleifen werden so verändert, dass weiterführende Aufgaben gelöst werden. Die Manipulation von Attributen auch durch Parameter wird ebenfalls benutzt.	Z.B. das Ablaufen eines Labyrinths wird in verschiedenen Schwierigkeitsgraden entwickelt. In einem weiterführenden Projekt werden die algorithmischen Fachkenntnisse vertieft.

Inhaltsfelder (Gegenstände) Kompetenzbereiche (Prozesse)	Kompetenzerwartungen	Didaktische Hinweise und Vereinbarungen	Hinweise und Vereinbarungen zu geeigneten Kontexten
Die SuS entwerfen einfache Algorithmen und stellen sie umgangssprachlich und grafisch dar (M),	Die SuS können eindeutige Handlungsvorschriften umgangssprachlich beschreiben und als Pseudocode formulieren. Abläufe werden durch geeignete Diagramme dargestellt	Programmabläufe, insbesondere Schleifen und Verzweigungen werden in Form von Struktogrammen dargestellt und als Pseudocode formuliert.	Werkzeug ist der PAP-Designer.
Die SuS implementieren Algorithmen unter Verwendung von Variablen und Wertzuweisungen, Kontrollstrukturen sowie Methodenaufrufen (I),	Die SuS können Algorithmen im Methodenkonzept programmieren. Dabei werden Variablen verändert und Parameter genutzt. Einfache Kontrollstrukturen werden von den SuS beherrscht.	Die Anwendung des Variablenkonzepts wird von den SuS beherrscht und bei der Umsetzung von Algorithmen angewendet. Verzweigungen sowie Schleifen werden auch in komplexeren Zusammenhängen geschachtelt benutzt.	Werkzeuge sind Greenfoot, BlueJ und der Java-Editor
Die SuS testen Programme schrittweise anhand von Beispielen (I).	Die SuS erschließen sich die Funktionsweise von Methoden systematisch in verschiedenen Problemsituationen. Sie können durch gezielte Wertebelegung von Attributen den Ablauf nachvollziehen.	Ausgewählte oder von den SuS selbst implementierte Programme/Methoden können sowohl auf Fehler, als auch auf ihre Funktionsweise hin untersucht werden.	Werkzeuge: BlueJ-Debugger

Inhaltsfeld 3: Formale Sprachen und Automaten

Inhaltsfelder (Gegenstände) Kompetenzbereiche (Prozesse)	Kompetenzerwartungen	Didaktische Hinweise und Vereinbarungen	Hinweise und Vereinbarungen zu geeigneten Kontexten
Die SuS implementieren einfache Algorithmen unter Beachtung der Syntax und Semantik einer Programmiersprache (I),	Die SuS können Quelltext in korrekter Syntax und Semantik programmieren. Typische Fehlermeldungen werden von den SuS richtig eingeschätzt. Fehler im Programm können weitgehend eigenständig beseitigt werden.	Die richtige Syntax der Sprache wird kontinuierlich im Unterricht aufgegriffen und geübt.	Durch die übersichtliche Strukturierung von Quelltext wird die fehlerarme Programmierung gefördert. Die farbliche Hervorhebung von Quelltextelementen in BlueJ wird benutzt.
interpretieren Fehlermeldungen und korrigieren den Quellcode (I)	Die SuS können Fehlermeldungen richtig einschätzen und dazu benutzen, um Fehler im Quelltext gezielt zu beseitigen.	Mit Hilfe von Fehlermeldung und Debugger werden Fehler im Programm systematisch behoben. Typische Fehlermeldungen werden im Unterricht thematisiert.	Häufige Fehler wie Klammerung, Groß- und Kleinschreibung werden unterrichtsbegleitend thematisiert. Als Hilfsmittel wird der BlueJ-Debugger benutzt.

Inhaltsfeld 4: Informatiksysteme

Inhaltsfelder (Gegenstände) Kompetenzbereiche (Prozesse)	Kompetenzerwartungen	Didaktische Hinweise und Vereinbarungen	Hinweise und Vereinbarungen zu geeigneten Kontexten
<p><i>Digitalisierung</i> Die SuS stellen ganze Zahlen und Zeichen in Binärcodes dar (D). Die SuS interpretieren Binärcodes als Zahlen und Zeichen (D).</p>	<p>Die SuS wissen, dass die Verarbeitung und die Speicherung von Daten letztlich immer auf zwei Zustände basieren.</p> <p>Die SuS begründen den Zahlenüberlauf bei Integer-Zahlen anhand der binären Zahldarstellung im Rechner.</p> <p>Die SuS können Buchstaben mit Hilfe der ASCII-Code-Tabelle in dezimale und binäre Zahlen umsetzen und so Buchstaben im Alphabet unter Berücksichtigung von Wertebereichen verschieben.</p>	<p>SuS aus dem IF8-Kurs kennen die Binärdarstellung aus dem WP11-Bereich (logische Schaltungen).</p> <hr/> <p>SuS aus dem IF_EF-Kurs erarbeiten in einem Exkurs Fragestellungen rund um die Binärdarstellung, die Speicherung von Daten und der logischen Verknüpfung von Aussagen.</p>	<p>Im Zusammenhang mit der ASCII-Code-Tabelle und der binären Zahldarstellung (ggf. Bitmapdarstellung) wird deutlich, dass alle Darstellungsweisen binär reduzierbar sind.</p> <p>Material aus „Informatik 2“ – Seite 38-45</p> <p>Im Kontext von Caesar-Verschlüsselung werden diese Erkenntnisse programmieretechnisch umgesetzt.</p>
<p><i>Einzelrechner</i> Die SuS beschreiben und erläutern den strukturellen Aufbau und die Arbeitsweise singulärer Rechner am Beispiel der „Von-Neumann-Architektur“ (A).</p>	<p>Die SuS kennen den Grundaufbau eines VN-R. und können die entscheidende Idee nennen.</p>	<p>Die entscheidende Idee des VN-R. (Programm wird behandelt wie Daten und damit wird der Rechner zur universellen Maschine) und das EVA-Prinzip werden herausgearbeitet. Eine geschichtliche Einordnung der Idee geschieht.</p> <p>Bearbeitung dieses Aspektes ggf. als Schülerreferat.</p>	<p>Als Material wird das Leitprogramm der RWTH Aachen zum VN-R. (Kapitel 1) und das Informatik-Duden-Buch genutzt.</p>

Inhaltsfelder (Gegenstände) Kompetenzbereiche (Prozesse)	Kompetenzerwartungen	Didaktische Hinweise und Vereinbarungen	Hinweise und Vereinbarungen zu geeigneten Kontexten
<p><i>Dateisystem</i> Die SuS nutzen das verfügbare Informatiksystem zur strukturierten Verwaltung und gemeinsamen Verwendung von Daten unter Berücksichtigung der Rechteverwaltung (K).</p>	<p>Die SuS können ihren Dateien sinnvoll benennen, in Ordner sinnvoll organisiert ablegen.</p> <p>Sie können die verschiedenen Dateitypen, die in einem Java-Projekt (Greenfoot und BlueJ) auftreten unterscheiden.</p>	<p>Es soll im Zusammenhang mit den .java- und .class-Dateien die Begriffe Quellcode und Bytecode thematisiert und damit die Plattformunabhängigkeit von Java hinterfragt werden.</p>	<p>Auf dem Laufwerk N wird für jeden Kurs ein gemeinsamer Ordner eingerichtet.</p>
<p><i>Internet</i> Die Schülerinnen und Schüler nutzen das Internet zur Recherche, zum Datenaustausch und zur Kommunikation (K).</p>	<p>Die SuS nutzen Programmiererergebnisse ihrer Mitschülerinnen und –schüler und des Lehrers und aus weiteren Quellen und stellen auch eigenen Quellcode anderen bereit. Übernommener oder zur Verfügung gestellter Programmcode soll adäquat debattiert werden.</p>	<p>Die Übernahme von Quellcode anderer Mits, des Lehrers und aus anderen Quellen wird thematisiert und in diesem Zusammenhang der Schutz des geistigen Eigentums besprochen.</p>	

Inhaltsfeld 5: Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:

- Einsatz von Informatiksystemen
- Wirkungen der Automatisierung
- Geschichte der automatischen Datenverarbeitung

Inhaltsfelder (Gegenstände) Kompetenzbereiche (Prozesse)	Kompetenzerwartungen	Didaktische Hinweise und Vereinbarungen	Hinweise und Vereinbarungen zu geeigneten Kontexten
Einsatz von Informatiksystemen: Die Schülerinnen und Schüler nutzen die im Unterricht eingesetzten Informatiksysteme selbstständig, sicher, zielführend und verantwortungsbewusst (D).	Die SuS können die zur Verfügung gestellte Hardware und die für den Unterricht eingeführte Software sicher und reflektiert einsetzen.		In der EF werden folgende Werkzeuge eingesetzt: - Greenfoot - BlueJ - PAPDesigner - Java-Editor
Wirkungen der Automatisierung Die Schülerinnen und Schüler bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen (A).			- VN-Architektur - Ggf. Auswirkung der Kryptologie im 2. Weltkrieg auf die Entwicklung der Informatik
Geschichte der automatischen Datenverarbeitung Die Schülerinnen und Schüler erläutern wesentliche Grundlagen der Geschichte der digitalen Datenverarbeitung (A).	Die SuS kennen die Meilensteine der automatisierten Datenverarbeitung und die dabei auftretenden entscheidenden Ideen.		