Europaschule
Gymnasium der Stadt Kerpen
Gemeinsam lernen und leben - grenzüberschreitend denken und handeln

Schulinterner Lehrplan
zum Kernlehrplan für die gymnasiale Oberstufe

Informatik

(Stand: 03.11.2022)

Inhalt

1 Die Fachgruppe Informatik des Gymnasiums der Stadt Kerpen

Beim Europagymnasium Kerpen handelt es sich um Europaschule mit einem halbtags-, ganztags- und einem bilingualem Zweig. Das Gymnasium ist eins der Größten und umfasst zurzeit ca. 1900 Schülerinnen und Schülern und 170 Lehrerinnen und Lehrer. Das Einzugsgebiet der Schule umfasst Kerpen sowie für den bilingualen Zweig umliegende Städte und Gemeinde.
Das Fach Informatik wird am Gymnasium der Stadt Kerpen in den Jahrgangsstufen 5 und 6 einstündig und in den Jahrgangsstufen 9 und 10 (Differenzierungsfächer: MNI und PTI) dreistündig unterrichtet. In den Jahrgangsstufen 5 und 6 wird in altersstufengerechter Weise eine informatischen Grundbildung vermittelt. Die Differenzierungsfächer MNI und PTI werden von etwa 40-50 Schülerinnen und Schüler pro Jahrgang besucht. In der zweijährigen Laufzeit dieser Kurse wird in altersstufengerechter Weise unter anderem auf Grundlagen der Algorithmik am Beispiel einer didaktischen Lernumgebung, auf die technische Informatik am Beispiel von Schaltwerken und Schaltnetzen und auf Robotik eingegangen. Der Unterricht erfolgt dabei in enger Verzahnung mit Inhalten der Mathematik, den Naturwissenschaften, insbesondere der Physik, und der Technik.

Für die Jahrgangsstufen 5-7 wird zusätzlich zum Pflichtangebot eine Informatik-AG angeboten. In der Sekundarstufe I wird darüber hinaus eine ECDL-AG für alle Schülerinnen und Schüler angeboten, in der sie den Europäischen Computerführerschein erwerben können. Für die Schülerinnen und Schülern bestehen ebenfalls die Möglichkeiten an Wettbewerben, wie dem für Einsteiger konzipierten Informatik Biber und dem anspruchsvolleren Bundeswettbwerb Informatik, teilzunehmen. Im Jahr 2013 wurde das Gymnasium der Stadt Kerpen auch aufgrund dieser vielfältigen Angebote als MINT-freundliche Schule ausgezeichnet und am 03.11.2017 offiziell in das nationale Excellence-Schulnetzwerk MINT-EC aufgenommen.

Wir bemühen uns ebenso um einen engen Anschluss an mögliche zukünftige Berufsfelder, beispielsweise durch eine Lernpartnerschaft mit dem Computacenter, einer ortsnah in Sindorf ansässigen IT-Firma zu deren Leistungsumfang ein breites, kundenorientiertes Spektrum gehört (u.a. Hardwarelösungen, Systemadministration inklusive online Lösungen, Webanwendungen). Auch ehemalige Schülerinnen und Schüler, die nach ihrer Schullaufbahn ein/e (duales) informatikassoziiertes Studium oder Ausbildung aufgenommen haben, geben regelmäßig Feedback im Informatikunterricht der Sekundarstufe II.

In der Sekundarstufe II bietet das Gymnasium der Stadt Kerpen für die eigenen Schülerinnen und Schüler in allen Jahrgangsstufen jeweils ein bis drei Grundkurse und in der Qualifikationsphase einen Leistungskurs in Informatik an.

Um insbesondere Schülerinnen und Schülern gerecht zu werden, die in der Sekundarstufe I keinen Informatikunterricht besucht haben, wird in Kursen der Einführungsphase besonderer Wert darauf gelegt, dass keine Vorkenntnisse aus der Sekundarstufe I zum erfolgreichen Durchlaufen des Kurses erforderlich sind. Den Schülerinnen und Schülern steht für die außerunterrichtliche Bearbeitung der Aufgaben das Selbstlernzentrum zur Verfügung.

Der Unterricht der Sekundarstufe II wird mit Hilfe der Programmiersprache Java durchgeführt. In der Einführungsphase wird dabei in die integrierte Entwicklungsumgebung BlueJ eingeführt, um objektorientierte Programmierung zu lernen.

Durch projektartiges Vorgehen, offene Aufgaben und Möglichkeiten, Problemlösungen zu verfeinern oder zu optimieren, entspricht der Informatikunterricht der Oberstufe in besonderem Maße den Erziehungszielen, Leistungsbereitschaft zu fördern, ohne zu überfordern.

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

Zurzeit besteht die Fachschaft Informatik des Gymnasiums der Stadt Kerpen aus sechs Lehrkräften, denen sechs Computerräume mit 16 bis 24 Computerarbeitsplätzen und ein Selbstlernzentrum mit 20 Plätzen zur Verfügung stehen. Alle Arbeitsplätze sind an das schulinterne Rechnernetz angeschlossen, so dass Schülerinnen und Schüler über einen Zugang zum zentralen Server der Schule Zugriff auf ihre eigenen Daten, zur Recherche im Internet oder zur Bearbeitung schulischer Aufgaben verwenden können.

2 Entscheidungen zum Unterricht

2.1 Unterrichtsvorhaben

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

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

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

Während der Fachkonferenzbeschluss zum „Übersichtsraster Unterrichtsvorhaben“ zur Gewährleistung vergleichbarer Standards sowie zur Absicherung von Lerngruppenübertritten und Lehrkraftwechseln für alle Mitglieder der Fachkonferenz Bindekraft entfalten soll, beinhaltet die Ausweisung „konkretisierter Unterrichtsvorhaben“ (Kapitel 2.1.2) Beispiele und Materialien, die empfehlenden Charakter haben. Referendarinnen und Referendaren sowie neuen Kolleginnen und Kollegen dienen diese vor allem zur standardbezogenen Orientierung in der neuen Schule, aber auch zur Verdeutlichung von unterrichtsbezogenen fachgruppeninternen Absprachen zu didaktisch-methodischen Zugängen, fächerübergreifenden Kooperationen, Lernmitteln und -orten sowie vorgesehenen Leistungsüberprüfungen, die im Einzelnen auch den Kapiteln 2.2 bis 2.3 zu entnehmen sind.

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

2.1.1 Übersichtsraster Unterrichtsvorhaben

<table>
<tr><th colspan="2">Einführungsphase</th></tr>
<tr><td><u>Unterrichtsvorhaben E-I</u>

Thema:
Was macht Informatik? - Einführung in die Inhaltsfelder der Informatik

Zentrale Kompetenzen:
• Kommunizieren und Kooperieren
• Darstellen und Interpretieren
• Argumentieren

Inhaltsfelder:
• Informatiksysteme
• Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:
• Einsatz, Nutzung und Aufbau von Informatiksystemen
• Wirkung der Automatisierung

Zeitbedarf: 6 Stunden</td><td><u>Unterrichtsvorhaben E-II</u>

Thema:
Grundlagen der objektorientierten Analyse, Modellierung und Implementierung

Zentrale Kompetenzen:
• Modellieren
• Implementieren
• Darstellen und Interpretieren
• Kommunizieren und Kooperieren

Inhaltsfelder:
• Daten und ihre Strukturierung
• Formale Sprachen und Automaten

Inhaltliche Schwerpunkte:
• Objekte und Klassen
• Syntax und Semantik einer Programmiersprache

Zeitbedarf: 8 Stunden</td></tr>
<tr><td><u>Unterrichtsvorhaben E-III</u>

Thema:
Algorithmische Grundstrukturen in Java

Zentrale Kompetenzen:
• Argumentieren
• Modellieren
• Implementieren
• Kommunizieren und Kooperieren

Inhaltsfelder:
• Daten und ihre Strukturierung
• Algorithmen
• Formale Sprachen und Automaten

Inhaltliche Schwerpunkte:
• Objekte und Klassen
• Syntax & Semantik einer Programmiersprache
• Analyse, Entwurf und Implementierung einfacher Algorithmen

Zeitbedarf: 18 Stunden</td><td><u>Unterrichtsvorhaben E-IV</u>

Thema:
Das ist die digitale Welt! – Einführung in die Grundlagen, Anwendungsgebiete und Verarbeitung binärer Codierung

Zentrale Kompetenzen:
• Kommunizieren und Kooperieren
• Darstellen und Interpretieren
• Argumentieren

Inhaltsfelder:
• Informatiksysteme
• Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:
• Binäre Codierung und Verarbeitung
• Besondere Eigenschaften der digitalen Speicherung und Verarbeitung von Daten

Zeitbedarf: 6 Stunden</td></tr>
</table>

<table>
<tr><th colspan="2">Einführungsphase</th></tr>
<tr><td><u>Unterrichtsvorhaben E-V</u>

Thema:
Modellierung und Implementierung von Klassen- und Objektbeziehungen anhand lebensnaher Anforderungsbeispiele

Zentrale Kompetenzen:
• Kommunizieren und Kooperieren
• Darstellen und Interpretieren
• Argumentieren
• Modellieren
• Implementieren

Inhaltsfelder:
• Daten und ihre Strukturierung
• Algorithmen
• Formale Sprachen und Automaten

Inhaltliche Schwerpunkte:
• Objekte und Klassen
• Syntax und Semantik einer Programmiersprache
• Analyse, Entwurf und Implementierung einfacher Algorithmen

Zeitbedarf: 18 Stunden</td><td><u>Unterrichtsvorhaben E-VI</u>

Thema:
Such- und Sortieralgorithmen

Zentrale Kompetenzen:
• Argumentieren
• Modellieren
• Darstellen und Interpretieren
• Kommunizieren und Kooperieren

Inhaltsfelder:
• Algorithmen
• Daten und ihre Strukturierung

Inhaltliche Schwerpunkte:
• Algorithmen zum Suchen und Sortieren
• Analyse, Entwurf und Implementierung einfacher Algorithmen
• Objekte und Klassen

Zeitbedarf: 10 Stunden</td></tr>
<tr><td><u>Unterrichtsvorhaben E-VII</u>

Thema:
Leben in der digitalen Welt – Immer mehr Möglichkeiten und immer mehr Gefahren!?

Zentrale Kompetenzen:
• Kommunizieren und Kooperieren
• Darstellen und Interpretieren
• Argumentieren

Inhaltsfelder:
• Informatiksysteme
• Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:
• Geschichte der automatischen Datenverarbeitung
• Wirkungen der Automatisierung
• Dateisystem

Zeitbedarf: 8 Stunden</td><td>Summe Einführungsphase:
74 Stunden</td></tr>
</table>

Qualifikationsphase 1 - Grundkurs	
Unterrichtsvorhaben Q1-I **Thema:** *Wiederholung der objektorientierten Modellierung und Programmierung anhand einer kontextbezogenen Problemstellung* **Zentrale Kompetenzen:** • Argumentieren • Modellieren • Implementieren • Darstellen und Interpretieren • Kommunizieren und Kooperieren **Inhaltsfelder:** • Daten und ihre Strukturierung • Algorithmen • Formale Sprachen und Automaten • Informatiksysteme **Inhaltliche Schwerpunkte:** • Objekte und Klassen • Analyse, Entwurf und Implementierung von Algorithmen • Syntax und Semantik einer Programmiersprache • Nutzung von Informatiksystemen **Zeitbedarf**: 8 Stunden	*Unterrichtsvorhaben Q1-II* **Thema:** *Modellierung und Implementierung von Anwendungen mit dynamischen, linearen Datenstrukturen* **Zentrale Kompetenzen:** • Argumentieren • Modellieren • Implementieren • Darstellen und Interpretieren • Kommunizieren und Kooperieren **Inhaltsfelder:** • Daten und ihre Strukturierung • Algorithmen • Formale Sprachen und Automaten **Inhaltliche Schwerpunkte:** • Objekte und Klassen • Analyse, Entwurf und Implementierung von Algorithmen • Algorithmen in ausgewählten informatischen Kontexten • Syntax und Semantik einer Programmiersprache **Zeitbedarf**: 20 Stunden

<table>
<tr><th colspan="2">Qualifikationsphase 1 - Grundkurs</th></tr>
<tr><td><u>Unterrichtsvorhaben Q1-III</u>

Thema:
Suchen und Sortieren auf linearen Datenstrukturen

Zentrale Kompetenzen:
• Argumentieren
• Modellieren
• Implementieren
• Darstellen und Interpretieren
• Kommunizieren und Kooperieren

Inhaltsfelder:
• Algorithmen
• Formale Sprachen und Automaten

Inhaltliche Schwerpunkte:
• Analyse, Entwurf und Implementierung von Algorithmen
• Algorithmen in ausgewählten informatischen Kontexten
• Syntax und Semantik einer Programmiersprache

Zeitbedarf: 16 Stunden</td><td><u>Unterrichtsvorhaben Q1-IV</u>

Thema:
Modellierung und Nutzung von relationalen Datenbanken in Anwendungskontexten

Zentrale Kompetenzen:
• Argumentieren
• Modellieren
• Implementieren
• Darstellen und Interpretieren
• Kommunizieren und Kooperieren

Inhaltsfelder:
• Daten und ihre Strukturierung
• Algorithmen
• Formale Sprachen und Automaten
• Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:
• Datenbanken
• Algorithmen in ausgewählten informatischen Kontexten
• Syntax und Semantik einer Programmiersprache
• Sicherheit

Zeitbedarf: 20 Stunden</td></tr>
<tr><td><u>Unterrichtsvorhaben Q1-V</u>

Thema:
Sicherheit und Datenschutz in Netzstrukturen

Zentrale Kompetenzen:
• Argumentieren
• Darstellen und Interpretieren
• Kommunizieren und Kooperieren

Inhaltsfelder:
• Informatiksysteme
• Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:
• Einzelrechner und Rechnernetzwerke
• Sicherheit
• Nutzung von Informatiksystemen, Wirkungen der Automatisierung

Zeitbedarf: 10 Stunden</td><td>Summe Qualifikationsphase 1 Grundkurs

74 Stunden</td></tr>
</table>

<table>
<tr><th colspan="2">Qualifikationsphase 2 - Grundkurs</th></tr>
<tr><td>

Unterrichtsvorhaben Q2-I

Thema:
Modellierung und Implementierung von Anwendungen mit dynamischen, nichtlinearen Datenstrukturen

Zentrale Kompetenzen:
- Argumentieren
- Modellieren
- Implementieren
- Darstellen und Interpretieren
- Kommunizieren und Kooperieren

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

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

Zeitbedarf: 24 Stunden

</td><td>

Unterrichtsvorhaben Q2-II

Thema:
Endliche Automaten und formale Sprachen

Zentrale Kompetenzen:
- Argumentieren
- Modellieren
- Darstellen und Interpretieren
- Kommunizieren und Kooperieren

Inhaltsfelder:
- Endliche Automaten und formale Sprachen

Inhaltliche Schwerpunkte:
- Endliche Automaten
- Grammatiken regulärer Sprachen
- Möglichkeiten und Grenzen von Automaten und formalen Sprachen

Zeitbedarf: 20 Stunden

</td></tr>
<tr><td>

Unterrichtsvorhaben Q2-III

Thema:
Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit

Zentrale Kompetenzen:
- Argumentieren
- Kommunizieren und Kooperieren

Inhaltsfelder:
- Informatiksysteme
- Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:
- Einzelrechner und Rechnernetzwerke
- Grenzen der Automatisierung

Zeitbedarf: 12 Stunden

</td><td>

Summe Qualifikationsphase 2 Grundkurs

56 Stunden

</td></tr>
</table>

Qualifikationsphase 1 - Leistungskurs	
Unterrichtsvorhaben Q1-I **Thema:** *Wiederholung der objektorientierten Modellierung und Programmierung anhand einer kontextbezogenen Problemstellung.* **Zentrale Kompetenzen:** • Argumentieren • Modellieren • Implementieren • Darstellen und Interpretieren • Kommunizieren und Kooperieren **Inhaltsfelder:** • Daten und ihre Strukturierung • Algorithmen • Formale Sprachen und Automaten • Informatiksysteme **Inhaltliche Schwerpunkte:** • Objekte und Klassen, Beziehungen von Klassen, eventuell ergänzt um Interfaces • Analyse, Entwurf und Implementierung von Algorithmen • Syntax und Semantik einer Programmiersprache • Nutzung von Informatiksystemen **Zeitbedarf**: 20 Stunden	*Unterrichtsvorhaben Q1-II* **Thema:** *Modellierung und Implementierung von Anwendungen mit dynamischen, linearen Datenstrukturen* **Zentrale Kompetenzen:** • Argumentieren • Modellieren • Implementieren • Darstellen und Interpretieren • Kommunizieren und Kooperieren **Inhaltsfelder:** • Daten und ihre Strukturierung • Algorithmen • Formale Sprachen und Automaten **Inhaltliche Schwerpunkte:** • Objekte und Klassen • Analyse, Entwurf und Implementierung von Algorithmen • Algorithmen in ausgewählten informatischen Kontexten • Syntax und Semantik einer Programmiersprache **Zeitbedarf**: 22 Stunden

Qualifikationsphase 1 - Leistungskurs	
Unterrichtsvorhaben Q1-III **Thema:** *Modellierung und Implementierung von Anwendungen mit dynamischen nicht-linearen Datenstrukturen* **Zentrale Kompetenzen:** • Argumentieren • Modellieren • Implementieren • Darstellen und Interpretieren • Kommunizieren und Kooperieren **Inhaltsfelder:** • Daten und ihre Strukturierung • Algorithmen • Formale Sprachen und Automaten **Inhaltliche Schwerpunkte:** • Objekte und Klassen • Analyse, Entwurf und Implementierung von Algorithmen • Algorithmen in ausgewählten informatischen Kontexten • Syntax und Semantik einer Programmiersprache **Zeitbedarf**: 34 Stunden	*Unterrichtsvorhaben Q1-IV* **Thema:** *Suchen und Sortieren auf linearen Datenstrukturen und mit Hilfe von Baumstrukturen* **Zentrale Kompetenzen:** • Argumentieren • Modellieren • Implementieren • Darstellen und Interpretieren • Kommunizieren und Kooperieren **Inhaltsfelder:** • Daten und ihre Strukturierung • Algorithmen • Formale Sprachen und Automaten • Informatiksysteme **Inhaltliche Schwerpunkte:** • Objekte und Klassen • Analyse, Entwurf und Implementierung von Algorithmen • Algorithmen in ausgewählten informatischen Kontexten • Syntax und Semantik einer Programmiersprache **Zeitbedarf**: 22 Stunden

Qualifikationsphase 1 - Leistungskurs	
Unterrichtsvorhaben Q1-V **Thema:** *Modellierung und Nutzung von relationalen Datenbanken in Anwendungskontexten* **Zentrale Kompetenzen:** • Argumentieren • Modellieren • Implementieren • Darstellen und Interpretieren • Kommunizieren und Kooperieren **Inhaltsfelder:** • Daten und ihre Strukturierung • Algorithmen • Formale Sprachen und Automaten • Informatiksysteme • Informatik, Mensch und Gesellschaft **Inhaltliche Schwerpunkte:** • Datenbanken • Algorithmen in ausgewählten informatischen Kontexten • Syntax und Semantik einer Programmiersprache • Sicherheit **Zeitbedarf**: 22 Stunden	**Summe Qualifikationsphase 1 Leistungskurs** **120 Stunden**

<table>
<tr><th colspan="2">Qualifikationsphase 2 - Leistungskurs</th></tr>
<tr><td><u>Unterrichtsvorhaben Q2-I</u>

Thema:
Endliche Automaten und formale Sprachen

Zentrale Kompetenzen:
• Argumentieren
• Modellieren
• Darstellen und Interpretieren
• Kommunizieren und Kooperieren

Inhaltsfelder:
• Formale Sprachen und Automaten
• Algorithmen
• Informatiksysteme

Inhaltliche Schwerpunkte:
• Endliche Automaten, Kellerautomaten
• Grammatiken regulärer und kontextfreier Sprachen
• Möglichkeiten und Grenzen von Automaten und formalen Sprachen

Zeitbedarf: 26 Stunden</td><td><u>Unterrichtsvorhaben Q2-II</u>

Thema:
Kommunikation in Netzwerken, Client-Server-Anwendungen, Sicherheit und Datenschutz in Netzstrukturen

Zentrale Kompetenzen:
• Argumentieren
• Modellieren
• Darstellen und Interpretieren
• Kommunizieren und Kooperieren

Inhaltsfelder:
• Daten und ihre Strukturierung
• Algorithmen
• Informatiksysteme
• Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:
• Einzelrechner und Rechnernetzwerke
• Sicherheit
• Nutzung von Informatiksystemen, Wirkungen der Automatisierung

Zeitbedarf: 36 Stunden</td></tr>
<tr><td><u>Unterrichtsvorhaben Q2-III</u>

Thema:
Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit

Zentrale Kompetenzen:
• Argumentieren
• Modellieren
• Darstellen und Interpretieren
• Kommunizieren und Kooperieren

Inhaltsfelder:
• Formale Sprachen und Automaten
• Informatiksysteme
• Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:
• Einzelrechner und Rechnernetzwerke
• Grenzen der Automatisierung

Zeitbedarf: 16 Stunden</td><td>Summe Qualifikationsphase 2
Leistungskurs

78 Stunden</td></tr>
</table>

2.1.2 Konkretisierte Unterrichtsvorhaben

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

In der Einführungs- und Qualifikationsphase wird die integrierte Entwicklungsumgebung BlueJ verwendet. Die folgenden Installationspakete und Dokumentationen stehen zur Verfügung:

- Installationspaket (Windows, MacOS, Linux und andere Betriebssysteme): http://www.bluej.org
- Dokumentationen (englisch): http://www.bluej.org/doc/documentation.html
- Tutorial (deutsch): http://www.bluej.org/tutorial/blueJ-tutorial-deutsch.pdf

In der Qualifikationsphase wird zusätzlich die Entwicklungsumgebung NetBeans verwendet. Die folgenden Installationspakete und Dokumentationen stehen zur Verfügung:

- Installationspaket (Windows, MacOS und Linux): https://netbeans.org/downloads/index.html
- Dokumentationen (englisch): https://netbeans.org/kb/index.html

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

http://www.standardsicherung.schulministerium.nrw.de/abitur-gost/fach.php?fach=15 (abgerufen: 30.04.2014)

-

I) Einführungsphase

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

Die Schülerinnen und Schüler

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

Unterrichtsvorhaben EF-I

Thema: Einführung in die Nutzung von Informatiksystemen und in grundlegende Begrifflichkeiten

Leitfragen: Womit beschäftigt sich die Wissenschaft der Informatik? Wie kann die in der Schule vorhandene informatische Ausstattung genutzt werden?

Vorhabenbezogene Konkretisierung:
Das erste Unterrichtsvorhaben stellt eine allgemeine Einführung in das Fach Informatik dar. Dabei ist zu berücksichtigen, dass für manche Schülerinnen und Schüler in der Einführungsphase der erste Kontakt mit dem Unterrichtsfach Informatik stattfindet, so dass zu Beginn Grundlagen des Fachs behandelt werden müssen.

Zunächst wird auf den Begriff der Information eingegangen und die Möglichkeit der Kodierung in Form von Daten thematisiert. Anschließend wird auf die Übertragung von Daten im Sinne des Sender-Empfänger-Modells eingegangen. Dabei wird eine überblickartige Vorstellung der Kommunikation von Rechnern in Netzwerken erarbeitet.

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.

Bei der Beschäftigung mit Datenkodierung, Datenübermittlung und Datenverarbeitung ist jeweils ein Bezug zur konkreten Nutzung der informatischen Ausstattung der Schule herzustellen. So wird in die verantwortungsvolle Nutzung dieser Systeme eingeführt.

Zeitbedarf: 6 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Information, deren Kodierung und Speicherung (a) Informatik als Wissenschaft der Verarbeitung von Informationen (b) Darstellung von Informationen in Schrift, Bild und Ton (c) Speichern von Daten mit informatischen Systemen am Beispiel der Schulrechner (d) Vereinbarung von Richtlinien zur Datenspeicherung auf den Schulrechnern (z.B. Ordnerstruktur, Dateibezeichner usw.)	Die Schülerinnen und Schüler • beschreiben und erläutern den Aufbau und die Arbeitsweise singulärer Rechner am Beispiel der „Von-Neumann-Architektur“ (A), • nutzen die im Unterricht eingesetzten Informatiksysteme selbstständig, sicher, zielführend und verantwortungsbewusst (D), • nutzen das Internet zur Recherche, zum Datenaustausch und zur Kommunikation (K).	*Beispiel*: Textkodierung Kodierung und Dekodierung von Texten mit unbekannten Zeichensätzen (z.B. Wingdings) *Beispiel*: Bildkodierung Kodierung von Bildinformationen in Raster- und Vektorgrafiken
2. Informations- und Datenübermittlung in Netzen (a) „Sender-Empfänger-Modell“ und seine Bedeutung für die Eindeutigkeit von Kommunikation (b) Informatische Kommunikation in Rechnernetzen am Beispiel des Schulnetzwerks (z.B. Benutzeranmeldung, Netzwerkordner, Zugriffsrechte, Client-Server) (c) Grundlagen der technischen Umsetzung von Rechnerkommunikation am Beispiel des Internets (z.B. Netzwerkadresse, Paketvermittlung, Protokoll)		*Beispiel*: Rollenspiel zur Paketvermittlung im Internet Schülerinnen und Schüler übernehmen die Rollen von Clients und Routern. Sie schicken spielerisch Informationen auf Karten von einem Schüler-Client zum anderen. Jede Schülerin und jeder Schüler hat eine Adresse, jeder Router darüber hinaus eine Routingtabelle. Mit Hilfe der Tabelle und einem Würfel wird entschieden, wie ein Paket weiter vermittelt wird.

(d) Richtlinien zum verantwortungsvollen Umgang mit dem Internet		
3. Aufbau informatischer Systeme (a) Identifikation typischer Komponenten informatischer Systeme und anschließende Beschränkung auf das Wesentliche, Herleitung der „Von-Neumann-Architektur“ (b) Identifikation des EVA (Eingabe-Verarbeitung-Ausgabe) als Prinzip der Verarbeitung von Daten und Grundlage der „Von-Neumann-Architektur“		*Material*: Demonstrationshardware Durch Demontage eines Demonstrationsrechners entdecken Schülerinnen und Schüler die verschiedenen Hardwarekomponenten eines Informatiksystems.

Unterrichtsvorhaben EF-II

Thema: Grundlagen der objektorientierten Analyse, Modellierung und Implementierung

Leitfragen: *Wie lassen sich Gegenstandsbereiche informatisch modellieren und in einem Szenario informatisch realisieren?*

Vorhabenbezogene Konkretisierung:
Ein zentraler Bestandteil des Informatikunterrichts der Einführungsphase ist die Objektorientierte Programmierung. Dieses Unterrichtsvorhaben führt in die Grundlagen der Analyse, Modellierung und Implementierung in diesem Kontext ein.

Dazu werden zunächst konkrete Gegenstandsbereiche aus der Lebenswelt der Schülerinnen und Schüler analysiert und im Sinne des objektorientierten Paradigmas strukturiert. Dabei werden die grundlegenden Begriffe der Objektorientierung und Modellierungswerkzeuge wie Objektdiagramme und Klassendiagramme eingeführt.

Im Anschluss kann zum Beispiel die objektorientierte Analyse für das Greenfoot-Szenario Planetenerkundung durchgeführt werden. Die vom Szenario vorgegebenen Klassen werden von Schülerinnen und Schülern in Teilen analysiert und entsprechende Objekte anhand einfacher Problemstellungen erprobt. Die Lernenden implementieren und testen einfache Programme. Die Greenfoot-Umgebung ermöglicht es, Beziehungen zwischen Klassen zu einem späteren Zeitpunkt (Kapitel 4) zu thematisieren. So kann der Fokus hier auf Grundlagen wie der Unterscheidung zwischen Klasse und Objekt, Attribute, Methoden, Objektidentität und Objektzustand gelegt werden.
Alternativ kann mit der Realisierung erster Projekte mit Hilfe der didaktischen Programmierumgebung GLOOP begonnen werden. Die von der Bibliothek vorgegebenen Klassen werden von Schülerinnen und Schülern in Teilen analysiert und entsprechende Objekte anhand einfacher Problemstellungen erprobt. Dazu muss der grundlegende Aufbau einer Java-Klasse thematisiert und zwischen Deklaration, Initialisierung und Methodenaufrufen unterschieden werden.

Da bei der Umsetzung dieser ersten Projekte konsequent auf die Verwendung von Kontrollstrukturen verzichtet wird und der Quellcode aus einer rein linearen Sequenz besteht, ist auf diese Weise eine Fokussierung auf die Grundlagen der Objektorientierung möglich, ohne dass algorithmische Probleme ablenken. Natürlich kann die Arbeit an diesen Projekten unmittelbar zum nächsten Unterrichtsvorhaben führen. Dort stehen unter anderem Kontrollstrukturen im Mittelpunkt.

Zeitbedarf: 8 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Identifikation von Objekten und Klassen (a) An einem lebensweltnahen Beispiel werden Objekte und Klassen im Sinne der objektorientierten Modellierung eingeführt. (b) Objekte werden durch Objektdiagramme, Klassen durch Klassendiagramme dargestellt. (c) Die Modellierungen werden einem konkreten Anwendungsfall entsprechend angepasst.	Die Schülerinnen und Schüler • ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften und ihre Operationen (M), • stellen den Zustand eines Objekts dar (D), • modellieren Klassen mit ihren Attributen und ihren Methoden (M), • implementieren einfache Algorithmen unter Beachtung der Syntax und Semantik einer Programmiersprache (I), • implementieren Klassen in einer Programmiersprache, auch unter Nutzung dokumentierter Klassenbibliotheken (I).	*Beispiel*: Vogelschwarm Schülerinnen und Schüler betrachten einen Vo-gelschwarm als Menge gleichartiger Objekte, die in einer Klasse mit Attributen und Methoden zu-sammengefasst werden können. *Materialien*: Ergänzungsmaterialien zum Lehrplannavigator - Allgemeine Objektorientierung
2. Analyse von Objekten und Klassen (a) Schritte der objektorientierten Analyse, Modellierung und Implementation (b) Analyse und Erprobung der Objekte		Das Greenfoot-Szenario „Planetenerkundung“ Von der Realität zu Objekten Von den Objekten zu Klassen, Klassendokumentation Objekte inspizieren Methoden aufrufen Objektidentität und Objektzustand
3. Implementierung einfacher Aktionen (a) Quelltext einer Java-Klasse (b) Implementation eigener Methoden, Dokumentation mit JavaDoc (c) Programme übersetzen (Aufgabe des Compilers) und testen		Programmierung in Greenfoot, BlueJ oder GLOOP Methoden schreiben Programme übersetzen und testen *Beispiel*: Olympische Ringe Die Schülerinnen und Schüler bilden das Emblem der olympischen Spiele *Beispiel*: Skulpturengarten Schülerinnen und Schüler erstellen ein Programm, das mit Hilfe von geometrischen Objekten der GLOOP-Umgebung einen Skulpturengaten auf den Bildschirm bringt.

Unterrichtsvorhaben EF-III

Thema: Algorithmische Grundstrukturen in Java

Leitfragen: *Wie lassen sich Aktionen von Objekten flexibel realisieren?*

Vorhabenbezogene Konkretisierung:

Das Ziel dieses Unterrichtsvorhabens besteht darin, das Verhalten von Objekten flexibel zu programmieren. Ein erster Schwerpunkt liegt dabei auf der Erarbeitung von Kontrollstrukturen. Die Strukturen Wiederholung und bedingte Anweisung werden an einfachen Beispielen eingeführt und anschließend anhand komplexerer Problemstellungen erprobt. Da die zu entwickelnden Algorithmen zunehmend umfangreicher werden, werden systematische Vorgehensweisen zur Entwicklung von Algorithmen thematisiert.

Ein zweiter Schwerpunkt des Unterrichtsvorhabens liegt auf dem Einsatz von Variablen. Beginnend mit lokalen Variablen, die in Methoden und Zählschleifen zum Einsatz kommen, über Variablen in Form von Parametern und Rückgabewerten von Methoden, bis hin zu Variablen, die die Attribute einer Klasse realisieren, lernen die Schülerinnen und Schüler die unterschiedlichen Einsatzmöglichkeiten des Variablenkonzepts anzuwenden.

Zeitbedarf: 18 Stunden

Sequenzierung des Unterrichtsvorhabens:

<table>
<tr><th>Unterrichtssequenzen</th><th>Zu entwickelnde Kompetenzen</th><th>Beispiele, Medien, Materialien</th></tr>
<tr><td rowspan="2">1. Algorithmen
(a) Wiederholungen (FOR- / While-Schleife)
(b) Verwaltung von Objekten in eindimensionalen Feldern (Arrays)
(c) bedingte Anweisungen
(d) Verknüpfung von Bedingungen durch die logischen Funktionen UND, ODER und NICHT
(e) Systematisierung des Vorgehens zur Entwicklung von Algorithmen zur Lösung komplexerer Probleme</td><td rowspan="2">Die Schülerinnen und Schüler
• analysieren und erläutern einfache Algorithmen und Programme (A),
• entwerfen einfache Algorithmen und stellen sie umgangssprachlich und grafisch dar (M),
• ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen zu (M),
• modifizieren einfache Algorithmen und Programme (I),
• implementieren Algorithmen unter Verwendung von Variablen und Wertzuweisungen, Kontrollstrukturen sowie Methodenaufrufen (I),
• implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I),
• implementieren einfache Algorithmen unter Beachtung der Syntax und Semantik einer Programmiersprache (I),
• testen Programme schrittweise anhand von Beispielen (I),
• interpretieren Fehlermeldungen und korrigieren den Quellcode (I).</td><td>Lerneinheiten
- Wiederholungen
- Bedingte Anweisungen
- Logische Funktionen
- Algorithmen entwickeln

Beispiele aus der Mathematik:
Zahlen aufsummieren, Zinseszinsen, Fakultät uvm.

Beispiele:
Zufallszahlen und einfache Würfelspiele</td></tr>
<tr><td>2. Variablen und Methoden
(a) Implementierung eigener Methoden mit lokalen Variablen, auch zur Realisierung einer Zählschleife
(b) Implementierung eigener Methoden mit Parameterübergabe und/oder Rückgabewert
(c) Implementierung von Konstruktoren
(d) Realisierung von Attributen</td><td>Lerneinheiten
- lokale Variablen
- Methoden
- Attribute

Beispiel: Ampeln
Die Schülerinnen und Schüler erstellen eine Ampelkreuzung mit mehreren Ampelanlagen an einem Bahnübergang.

Beispiele:
Komplexe Würfelspiele</td></tr>
</table>

Unterrichtsvorhaben EF-IV

Thema: Das ist die digitale Welt! - Einführung in die Grundlagen, Anwendungsgebiete und Verarbeitung binärer Codierung

Leitfragen: Wie werden binäre Informationen gespeichert und wie können sie davon ausgehend weiter verarbeitet werden? Wie unterscheiden sich analoge Medien und Geräte von digitalen Medien und Geräten? Wie ist der Grundaufbau einer digitalen Rechenmaschine?

Vorhabenbezogene Konkretisierung:
Das Unterrichtsvorhaben hat die binäre Speicherung und Verarbeitung sowie deren Besonderheiten zum Inhalt.

Im ersten Schritt erarbeiten die Schülerinnen und Schüler anhand ihnen bekannter technischer Gegenstände die Gemeinsamkeiten, Unterschiede und Besonderheiten der jeweiligen analogen und digitalen Version. Nach dieser ersten grundlegenden Einordnung des digitalen Prinzips wenden die Schülerinnen und Schüler das Binäre als Zahlensystem mit arithmetischen und logischen Operationen an und codieren Zeichen binär.

Zeitbedarf: 6 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Analoge und digitale Aufbereitung und Verarbeitung von Daten (a) Erarbeitung der Unterschiede von analog und digital (b) Zusammenfassung und Bewertung der technischen Möglichkeiten von analog und digital	Die Schülerinnen und Schüler • bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen (A) • stellen ganze Zahlen und Zeichen in Binärcodes dar (D), • interpretieren Binärcodes als Zahlen und Zeichen (D) • nutzen das Internet zur Recherche, zum Datenaustausch und zur Kommunikation (K) • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I)	***Exkurs „Analog und Digital“***
2. Der Umgang mit binärer Codierung von Informationen (a) Das binäre (und hexadezimale) Zahlensystem (b) Binäre Informationsspeicherung (c) Binäre Verschlüsselung (d) Implementation eines Binärumrechners		***Exkurs „Binäre Welt“***

Unterrichtsvorhaben EF-V

Thema: Modellierung und Implementierung von Klassen- und Objektbeziehungen anhand lebensnaher Anforderungsbeispiele

Leitfragen: Wie werden realistische Systeme anforderungsspezifisch reduziert, als Entwurf modelliert und implementiert? Wie kommunizieren Objekte und wie wird dieses dargestellt und realisiert?

Vorhabenbezogene Konkretisierung:
Das Unterrichtsvorhaben hat die Entwicklung von Objekt -und Klassenbeziehungen zum Schwerpunkt. Dazu werden, ausgehend von der Realität, über Objektidentifizierung und Entwurf bis hin zur Implementation kleine Softwareprodukte in Teilen oder ganzheitlich erstellt.

Zuerst identifizieren die Schülerinnen und Schüler Objekte und stellen diese dar. Aus diesen Objekten werden Klassen und ihre Beziehungen in Entwurfsdiagrammen erstellt.

Nach diesem ersten Modellierungsschritt werden über Klassendokumentationen und der Darstellung von Objektkommunikationen anhand von Sequenzdiagrammen Implementationsdiagramme entwickelt. Danach werden die Implementationsdiagramme unter Berücksichtigung der Klassendokumentationen in Javaklassen programmiert. In einem letzten Schritt wird das Konzept der Vererbung sowie seiner Vorteile erarbeitet.

Schließlich sind die Schülerinnen und Schüler in der Lage, eigene kleine Softwareprojekte zu entwickeln. Ausgehend von der Dekonstruktion und Erweiterung eines Spiels wird ein weiteres Projekt von Grund auf modelliert und implementiert. Dabei können arbeitsteilige Vorgehensweisen zum Einsatz kommen. In diesem Zusammenhang wird auch das Erstellen von graphischen Benutzeroberflächen eingeführt.

Zeitbedarf: 18 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Umsetzung von Anforderungen in Entwurfsdiagramme (a) Aus Anforderungsbeschreibungen werden Objekte mit ihren Eigenschaften identifiziert (b) Gleichartige Objekte werden in Klassen (Entwurf) zusammengefasst und um Datentypen und Methoden erweitert	Die Schülerinnen und Schüler • analysieren und erläutern eine objektorientierte Modellierung (A), • stellen die Kommunikation zwischen Objekten grafisch dar (M), • 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), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen oder lineare Datensammlungen zu (M), • ordnen Klassen, Attributen und Methoden ihren Sichtbarkeitsbereich zu (M), • modellieren Klassen unter Verwendung von Vererbung (M), • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I), • testen Programme schrittweise anhand von Beispielen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • 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). • stellen Klassen, Assoziations- und Vererbungs-beziehungen in Diagrammen grafisch dar (D), • dokumentieren Klassen durch Beschreibung der Funktionalität der Methoden (D)	Lerneinheiten - Von der Realität zum Programm - Objekte identifizieren - Klassen und Beziehungen entwerfen
2. Implementationsdiagramme als erster Schritt der Programmierung (a) Erweiterung des Entwurfsdiagramms um Konstruktoren und get- und set-Methoden (b) Festelegung von Datentypen in Java, sowie von Rückgaben und Parametern (c) Entwicklung von Klassendokumentationen (d) (d) Erstellung von Sequenzdiagrammen als Vorbereitung Vorbereitung für die Programmierung		Lerneinheiten - Klassen und Beziehungen implementieren - Vererbung
3. Programmierung anhand der Dokumentation und des Implementations- und Sequenzdiagrammes (a) Klassen werden in Java-Quellcode umgesetzt (b) Das Geheimnisprinzip wird umgesetzt (c) Einzelne Klassen und das Gesamtsystem werden anhand der Anforderungen und Dokumentationen auf ihre Korrektheit überprüft.		Lerneinheiten - Klassen und Beziehungen implementieren - Vererbung
4. Vererbungsbeziehungen (a) Das Grundprinzip der Vererbung wird erarbeitet (b) Die Vorteile der Vererbungsbeziehungen (c) Vererbung wird implementiert		Lerneinheit Vererbung
5. Softwareprojekt (a) Analyse und Dekonstruktion eines Spiels (Modelle, Quelltexte) (b) Erweiterung des Spiels um weitere Funktionalitäten (c) Modellierung eines Spiels aufgrund einer Anforderungsbeschreibung, inklusive einer grafischen Benutzeroberfläche (d) (arbeitsteilige) Implementation des Spiels		Lerneinheiten - Softwareentwicklung - Oberflächen (GUI)

Unterrichtsvorhaben EF-VI

Thema: Such- und Sortieralgorithmen

Leitfragen: *Wie können Objekte bzw. Daten effizient gesucht und sortiert werden?*

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 lernen die Schülerinnen und Schüler das Feld als eine erste Datensammlung kennen. Optional können nun zunächst die wesentlichen Eigenschaften von Algorithmen wie z.B. Korrektheit, Terminiertheit, Effizienz und Verständlichkeit sowie die Schritte einer Algorithmenentwicklung erarbeitet werden (Klärung der Anforderung, Visualisierung, Zerlegung in Teilprobleme).

Daran anschließend lernen die Schülerinnen und Schüler zunächst Strategien des Suchens (lineare Suche, binäre Suche, Hashing) und dann des Sortierens (Selection Sort, Insertion Sort, Bubble Sort) kennen. Die Projekteinstiege dienen dazu, die jeweiligen Strategien handlungsorientiert zu erkunden und intuitive Effizienzbetrachtungen der Suchalgorithmen vorzunehmen.

Schließlich wird die Effizienz unterschiedlicher Sortierverfahren beurteilt.

Zeitbedarf: 10 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Kapitel und Materialien
1. Modellierung und Implementation von Datenansammlungen (a) Modellierung von Attributen als Felder (b) Deklaration, Instanziierung und Zugriffe auf ein Feld	Die Schülerinnen und Schüler • analysieren Such- und Sortieralgorithmen und wenden sie auf Beispiele an (D) • entwerfen einen weiteren Algorithmus zum Sortieren (M) • beurteilen die Effizienz von Algorithmen am Beispiel von Sortierverfahren hinsichtlich Zeit und Speicherplatzbedarf (A) • ordnen Attributen lineare Datenansammlungen zu (M)	Wiederholung: Arrays und Schleifen
2. Explorative Erarbeitung von Suchverfahren (a) Erkundung von Strategien für das Suchen auf unsortierten Daten, auf sortierten Daten und mithilfe einer Berechnungsfunktion. (b) Vergleich der drei Verfahren durch intuitive Effizienzbetrachtungen.		Projekteinstieg: Suchen Mit Hilfe eines Spiels von der lineare Suche zur binären Suche *Material*: Einfuehrungsspiel (Quelle: Gymnasium-Odenthal.de)
3. Systematisierung von Algorithmen und Effizienzbetrachtungen (a) Formulierung (falls selbst gefunden) oder Erläuterung von mehreren Algorithmen im Pseudocode (b) Anwendung von Sortieralgorithmen auf verschiedene Beispiele (c) Bewertung von Algorithmen anhand der Anzahl der nötigen Vergleiche (d) Effizienzbetrachtungen an einem konkreten Beispiel bezüglich der Rechenzeit und des Speicherplatzbedarfs (e) Analyse eines weiteren Sortieralgorithmus (sofern nicht in (a) bereits geschehen)		Projekteinstieg: Sortieren Selection Sort, Insertion Sort, Bubble Sort *Beispiel*: 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. *Beispiele*: Sortieren durch Auswählen, Sortieren durch Vertauschen, Quicksort Quicksort ist als Beispiel für einen Algorithmus nach dem Prinzip Teile und Herrsche gut zu behandeln. *Materialien*: Computer science unplugged – Sorting Algorithms, URL: http://www.csunplugged.org/sorting-algorithms abgerufen: 30. 03. 2014

Unterrichtsvorhaben EF-VII

Thema: Leben in der digitalen Welt – Immer mehr Möglichkeiten und immer mehr Gefahren!?

Leitfragen: Welche Entwicklungen, Ideen und Erfindungen haben zur heutigen Informatik geführt? Welche Auswirkungen hat die Informatik für das Leben des modernen Menschen?

Vorhabenbezogene Konkretisierung:
Das Unterrichtsvorhaben stellt die verschiedenen Entwicklungsstränge der Informatik in den Fokus. Darüber hinaus wird beispielhaft analysiert und bewertet, welche Möglichkeiten und Gefahren die moderne Informationsverarbeitung mit sich bringt.

Im ersten Schritt des Unterrichtsvorhabens wird anhand von Themenkomplexen entscheidende Entwicklungen der Informatik erarbeitet. Dabei werden auch übergeordnete Tendenzen identifiziert.

Ausgehend von dieser Betrachtung kann die aktuelle Informatik hinsichtlich ihrer Leistungsfähigkeit analysiert werden. Dabei soll herausgestellt werden, welche positiven und negativen Folgen Informatiksysteme mit sich bringen können.

Zeitbedarf: 12 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Kapitel und Materialien
1. Schriftzeichen, Rechenmaschine, Computer (a) Anhand von Schwerpunkten, wie z.B. Datenspeicherung, Maschinen, Vernetzung sollen wichtige Entwicklungen der Informatik vorgestellt werden. (b) (b) Anhand der unterschiedlichen Schwerpunkte sollen universelle Tendenzen der Entwicklung der Informationsverarbeitung erarbeitet werden.	Die Schülerinnen und Schüler • bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen (A), • erläutern wesentliche Grundlagen der Geschichte der digitalen Datenverarbeitung (A) • nutzen das Internet zur Recherche, zum Datenaustausch und zur Kommunikation (K)	***Exkurs „Geschichte der Informatik“***
2. Die Informationsverarbeitung und ihre Möglichkeiten und Gefahren (a) Ausgehend von 1. werden Tendenzen der Entwicklung der Informatik erarbeitet (b) Informatik wird als Hilfswissenschaft klassifiziert, die weit über ihren originären Bereich hinaus Effizienz- und Leistungssteigerungen erzeugt (c) (c) Anhand von Fallbeispielen werden technische und organisatorische Vorteile, sowie deren datenschutzrechtlichen Nachteile betrachtet.		***Exkurs „Informatik und Gesellschaft“***

II) Qualifikationsphase Grundkurs

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.

Das Aufrufen von Methoden auf Objekten kann dabei als Nachrichtenaustausch zwischen verschiedenen Objekten interpretiert werden und wird als solcher verdeutlicht, indem die Kommunikation zwischen zwei ausgewählten Objekten grafisch dargestellt wird (Zeit-Sequenz-Diagramm). In diesem Zusammenhang wird das Nachrichtenkonzept der objektorientierten Programmierung (Methodenaufruf mit Punktnotation auf Objekten/Klassen – statische versus dynamische Methoden) wiederholt.

Zeitbedarf: 8 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Wiederholung und Erweiterung der objektorientierten Modellierung und Programmierung durch Analyse und Erweiterung eines kontextbezogenen Beispiels (a) Analyse der Problemstellung (b) Analyse der Modellierung (Implementationsdiagramm) (c) Erweiterung der Modellierung im Implementationsdiagramm (Vererbung, abstrakte Klasse) (d) Kommunikation zwischen mindestens zwei Objekten (grafische Darstellung) (e) Dokumentation von Klassen (f) Implementierung der Anwendung oder von Teilen der Anwendung	Die Schülerinnen und Schüler • 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),	*Beispiel:* Nim-Spiel Das Nim-Spiel ist ein Spiel für zwei Personen, bei dem abwechselnd eine Anzahl von Gegenständen, etwa Streichhölzer, weggenommen werden. Hier kann eine Interface-Klasse bei einer Klasse Computerspieler und Spieler hilfreich sein. *Beispiel:* Wetthuepfen 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. *Beispiel*: Tannenbaum 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. 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. *Materialien:*

	• stellen die Kommunikation zwischen Objekten grafisch dar (D).	Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.1-Wiederholung (Download Q1-I.1)

Unterrichtsvorhaben Q1-II

Thema: Modellierung und Implementierung von Anwendungen mit dynamischen, linearen Datenstrukturen

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

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

Zeitbedarf: 20 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Die Datenstruktur Schlange im Anwendungskontext unter Nutzung der Klasse Queue (a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen (b) Erarbeitung der Funktionalität der Klasse Queue	Die Schülerinnen und Schüler • erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (A), • analysieren und erläutern Algorithmen und Programme (A),	*Beispiel:* Patientenwarteschlange (jeder kennt seinen Nachfolger bzw. alternativ: seinen Vorgänger) 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. Die Simulationsanwendung stellt eine GUI zur Verfügung, legt ein Wartezimmer

(c) Modellierung und Implementierung der Anwendung unter Verwendung eines oder mehrerer Objekte der Klasse Queue	• beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M), • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • modifizieren Algorithmen und Programme (I), • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • testen Programme systematisch anhand von Beispielen (I), • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D).	an und steuert die Abläufe. Wesentlicher Aspekt des Projektes ist die Modellierung des Wartezimmers mit Hilfe der Klasse Queue. Anschließend wird der Funktionsumfang der Anwendung erweitert: Patienten können sich zusätzlich in die Warteschlange zum Blutdruckmessen einreihen. Objekte werden von zwei Schlangen verwaltet. *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.2 – Warteschlange
2. Die Datenstruktur Stapel im Anwendungskontext unter Nutzung der Klasse Stack (a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen (b) Erarbeitung der Funktionalität der Klasse Stack (c) Modellierung und Implementierung der Anwendung unter Verwendung eines oder mehrerer Objekte der Klasse Stack		*Beispiel:* Heftstapel In einem Heftstapel soll das Heft einer Schülerin gefunden werden. *Beispiel*: Kisten stapeln 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.
3. Die Datenstruktur lineare Liste im Anwendungskontext unter Nutzung der Klasse List (a) Erarbeitung der Vorteile der Klasse List im Gegensatz zu den bereits bekannten linearen Strukturen (b) Modellierung und Implementierung einer kontextbezogenen Anwendung unter Verwendung der Klasse List.		*Beispiel:* Nim-Spiel Das im Unterrichtsvorgabe Q1-I implementierte Nim-Spiel wird um eine Liste, die dem Computerspieler als Gedächtnis dienen soll, erweitert. Im Laufe des Spiels, soll sich der Computerspieler über die Liste ungünstige Spielverläufe merken und somit lernen. (Heuristik) *Beispiel*: Abfahrtslauf 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. *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.2 – Listen
4. Vertiefung - Anwendungen von Listen, Stapeln oder Schlangen in mindestens einem weiteren Kontext		*Beispiel:* Skispringen Ein Skispringen hat folgenden Ablauf: Nach dem Sprung erhält der Springer eine Punktzahl und wird nach dieser Punktzahl in eine Rangliste eingeordnet. Die besten 30 Springer qualifizieren sich für den zweiten Durchgang. Sie starten in umgekehrter Reihenfolge gegenüber der Platzierung auf der Rangliste. Nach

		dem Sprung erhält der Springer wiederum eine Punktzahl und wird nach der Gesamtpunktzahl aus beiden Durchgängen in die endgültige Rangliste eingeordnet. *Beispiel*: Terme in Postfix-Notation Die sog. UPN (*Umgekehrt-Polnische-Notation*) bzw. *Postfix-Notation* eines Terms setzt den Operator hinter die Operanden. Um einen Term aus der gewohnten Infixschreibweise in einen Term in UPN umzuwandeln oder um den Wert des Terms zu berechnen, kann ein Stack verwendet werden. *Beispiel*: Rangierbahnhof Auf einem Güterbahnhof gibt es drei Gleise, die nur zu einer Seite offen sind. Wagons können also von einer Seite auf das Gleis fahren und nur rückwärts wieder hinausfahren. Die Wagons tragen Nummern, wobei die Nummer jedoch erst eingesehen werden kann, wenn der Wagon der vorderste an der offenen Gleisseite ist. (Zwischen den Wagons herumzuturnen, um die anderen Wagonnummern zu lesen, wäre zu gefährlich.) Zunächst stehen alle Wagons unsortiert auf einem Gleis. Ziel ist es, alle Wagons in ein anderes Gleis zu fahren, so dass dort die Nummern der Wagons vom Gleisende aus aufsteigend in richtiger Reihenfolge sind. Zusätzlich zu diesen beiden Gleisen gibt es ein Abstellgleis, das zum Rangieren benutzt werden kann. *Beispiel*: Autos an einer Ampel zur Zufahrtsregelung Es soll eine Ampel zur Zufahrtsregelung in Java simuliert werden. An einem geradlinigen, senkrecht von unten nach oben verlaufenden Straßenstück, das von Autos nur einspurig in eine Richtung befahren werden kann, ist ein Haltepunkt markiert, an dem die Ampel steht. Bei einem Klick auf eine Schaltfläche mit der Aufschrift „Heranfahren“ soll ein neues Auto an den Haltepunkt heranfahren bzw. bis an das letzte Auto, das vor dem Haltepunkt wartet. Grünphasen der Ampel werden durch einen Klick auf eine Schaltfläche mit der Aufschrift „Weiterfahren“ simuliert. In jeder Grünphase darf jeweils nur ein Auto weiterfahren. Die anderen Autos rücken nach. *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1-II.3 – Anwendungen für lineare Datenstrukturen

Unterrichtsvorhaben Q1-III

Thema: Suchen und Sortieren auf linearen Datenstrukturen

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

Vorhabenbezogene Konkretisierung:
In einem Anwendungskontext werden zunächst Informationen in einer linearen Liste bzw. einem Feld gesucht. Hierzu werden Verfahren entwickelt und implementiert bzw. analysiert und erläutert, wobei neben einem iterativen auch ein rekursives Verfahren thematisiert wird und mindestens ein Verfahren selbst entwickelt und implementiert wird.

Anschließend wird die Implementationen von Quicksort sowie dem Sortieren durch Einfügen analysiert und erläutert. Falls diese Verfahren vorher schon entdeckt wurden, sollen sie hier wiedererkannt werden. Die rekursive Abarbeitung eines Methodenaufrufs von Quicksort wird grafisch dargestellt.

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

Zeitbedarf: 16 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Suchen von Daten in Listen und Arrays (a) Lineare Suche in Listen und in Arrays (b) Binäre Suche in Arrays als Beispiel für rekursives Problemlösen (c) Untersuchung der beiden Suchverfahren hinsichtlich ihrer Effizienz (Laufzeitverhalten, Speicherbedarf)	Die Schülerinnen und Schüler • 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),	*Beispiel:* Karteiverwaltung Für ein Adressverwaltungsprogramm soll eine Methode zum Suchen einer Adresse geschrieben werden. *Beispiel*: Bundesjugendspiele Die Teilnehmer an Bundesjugendspielen nehmen an drei Disziplinen teil und erreichen dort Punktzahlen. Diese werden in einer Wettkampfkarte eingetragen und an das Wettkampfbüro gegeben. Zur Vereinfachung sollte sich das Modell auf die drei Disziplinen „Lauf“, „Sprung“ und „Wurf“ beschränken. Im Wettkampfbüro wird das Ergebnis erstellt. Das Programm soll dafür zunächst den Besten einer Disziplin heraussuchen können und später das gesamte Ergebnis nach gewissen Kriterien sortieren
2. Sortieren in Listen und Arrays - Entwicklung und Implementierung von iterativen und rekursiven Sortierverfahren (a) Entwicklung und Implementierung eines einfachen Sortierverfahrens für eine Liste (b) Implementierung eines einfachen Sortierverfah-		

rens für ein Feld (c) Entwicklung eines rekursiven Sortierverfahren für ein Feld (z.B. Sortieren durch Mischen)	• 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).	können. *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.3 - Suchen
3. Untersuchung der Effizienz der Sortierverfahren „Sortieren durch direktes Einfügen“ und „Quicksort“ auf linearen Listen (a) Grafische Veranschaulichung der Sortierverfahren (b) Untersuchung der Anzahl der Vergleichsoperationen und des Speicherbedarf bei beiden Sortierverfahren (c) Beurteilung der Effizienz der beiden Sortierverfahren		

Unterrichtsvorhaben Q1-IV

Thema: Modellierung und Nutzung von relationalen Datenbanken in Anwendungskontexten

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

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

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

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

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

Zeitbedarf: 20 Stunden

Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Nutzung von relationalen Datenbanken (a) Aufbau von Datenbanken und Grundbegriffe • Entwicklung von Fragestellungen zur vorhandenen Datenbank • Analyse der Struktur der vorgegebenen Datenbank und Erarbeitung der Begriffe Tabelle, Attribut, Datensatz, Datentyp, Primärschlüssel, Fremdschlüssel, Datenbankschema (b) SQL-Abfragen • 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) (c) Vertiefung an einem weiteren Datenbankbeispiel	Die Schülerinnen und Schüler • 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 einen Datenbanksystem zu extrahieren (I), • ermitteln Ergebnisse von Datenbankabfragen über mehrere verknüpfte Tabellen (D), • stellen Entitäten mit ihren Attributen und die Beziehungen zwischen Entitäten in einem Entity-Relationship-Diagramm grafisch dar (D), • überprüfen Datenbankschemata auf vorgegebene Normalisierungseigenschaften (D).	*Beispiel:* VideoCenter VideoCenter ist die Simulation einer Online-Videothek für den Informatik-Unterricht mit Webfrontends zur Verwaltung der Kunden, der Videos und der Ausleihe. Außerdem ist es möglich direkt SQL-Abfragen einzugeben. Es ist auch möglich, die Datenbank herunter zu laden und lokal zu installieren. Unter schule.de (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. *Beispiel*: Schulbuchausleihe Unter brd.nrw.de (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.
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		*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

• Modellierung eines relationalen Datenbankschematas 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)		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* 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.

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
1. Daten in Netzwerken und Sicherheitsaspekte in Netzen sowie beim Zugriff auf Datenbanken (a) Beschreibung eines Datenbankzugriffs im Netz anhand eines Anwendungskontextes und einer Client-Server-Struktur zur Klärung der Funktionsweise eines Datenbankzugriffs (b) Netztopologien als Grundlage von Client-Server-Strukturen und TCP/IP-Schichtenmodell als Beispiel für eine Paketübermittlung in einem Netz (c) Vertraulichkeit, Integrität, Authentizität in Netzwerken sowie symmetrische und asymmetrische kryptografische Verfahren (Cäsar-, Vigenère-, RSA-Verfahren) als Methoden Daten im Netz verschlüsselt zu übertragen	Die Schülerinnen und Schüler • 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	*Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben, Verschlüsselung Q1.5 - Zugriff auf Daten in Netzwerken
2. Fallbeispiele zur Datenschutzproblematik und zum Urheberrecht		*Materialien:* *Ergänzungsmaterialien zum Lehrplannavigator*

	reflektiert zum Erschließen, zur Aufbereitung und Präsentation fachlicher Inhalte (D).	*Unterrichtsvorhaben Q1.5* - Datenschutz beim Videocenter, Materialblatt-Datenschutzgesetz

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 Umkehrbarbeit 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
1. Analyse von Baumstrukturen in verschiedenen Kontexten (a) Grundlegende Begriffe (Grad, Tiefe, Höhe, Blatt, Inhalt, Teilbaum, Ebene, Vollständigkeit) (b) Aufbau und Darstellung von binären Bäumen anhand von Baumstrukturen in verschiedenen Kontexten	Die Schülerinnen und Schüler • 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),	*Beispiel:* Termbaum Der Aufbau von Termen wird mit Hilfe von binären Baumstrukturen verdeutlicht. *Beispiel*: Ahnenbaum Die binäre Baumstruktur ergibt sich daraus, dass jede Person genau einen Vater und eine Mutter hat. <u>*Weitere Beispiele für Anwendungskontexte für binäre Bäume*</u>*:* *Beispiel*: Suchbäume (zur sortierten Speicherung von Daten) Alle Inhalte, die nach einer Ordnung vor dem Inhalt im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Inhalt im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.) *Beispiel*: 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

	• entwickeln iterative und rekursive Algorithmen unter Nutzung der Konstruktionsstrategien „Modularisierung“ und „Teilen und Herrschen“ (M), • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I), • modifizieren Algorithmen und Programme (I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • testen Programme systematisch anhand von Beispielen (I), • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D), • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D).	im linken Teilbaum, die Fragen, die möglich sind, wenn die Antwort „nein“ lautet, stehen im rechten Teilbaum. *Beispiel*: 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 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. *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 – Binärbaum
2. Die Datenstruktur Binärbaum im Anwendungskontext unter Nutzung der Klasse BinaryTree (a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen im Anwendungskontext (b) Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramms (c) Erarbeitung der Klasse BinaryTree und beispielhafte Anwendung der Operationen		*Beispiel*: Informatikerbaum *als binärer Baum* In einem *binären Baum* werden die Namen und die Geburtsdaten von Informatikern lexikographisch geordnet abgespeichert. Alle Namen, die nach dieser Ordnung vor dem Namen im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Namen im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.) Folgende Funktionalitäten werden benötigt:

(d) Implementierung der Anwendung oder von Teilen der Anwendung (e) Traversierung eines Binärbaums im Pre-, In- und Postorderdurchlauf		• Einfügen der Informatiker-Daten in den Baum • Suchen nach einem Informatiker über den Schlüssel Name • Ausgabe des kompletten Datenbestands in nach Namen sortierter Reihenfolge *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 – Binärbaum
3. Die Datenstruktur binärer Suchbaum im Anwendungskontext unter Verwendung der Klasse BinarySearchTree (a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen (b) Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramm, grafische Darstellung eines binären Suchbaums und Erarbeitung der Struktureigenschaften (c) Erarbeitung der Klasse BinarySearchTree und Einführung des Interface Item zur Realisierung einer geeigneten Ordnungsrelation (d) Implementierung der Anwendung oder von Teilen der Anwendung inklusive einer sortierten Ausgabe des Baums		*Beispiel*: Informatikerbaum als *Suchbaum* In einem binären *Suchbaum* werden die Namen und die Geburtsdaten von Informatikern lexikographisch geordnet abgespeichert. Alle Namen, die nach dieser Ordnung vor dem Namen im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Namen im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.) Folgende Funktionalitäten werden benötigt: • Einfügen der Informatiker-Daten in den Baum • Suchen nach einem Informatiker über den Schlüssel Name • Ausgabe des kompletten Datenbestands in nach Namen sortierter Reihenfolge

		Materialien: Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 – Binärer Suchbaum
4. Übung und Vertiefungen der Verwendung von Binärbäumen oder binären Suchbäumen anhand weiterer Problemstellungen		*Beispiel:* Codierungsbäume (s.o.) oder Huffman-Codierung *Beispiel*: Buchindex Es soll eine Anwendung entwickelt werden, die anhand von Stichworten und zugehörigen Seitenzahlen ein Stichwortregister erstellt. Da die Stichwörter bei der Analyse des Buches häufig gesucht werden müssen, werden sie in der Klasse Buchindex als Suchbaum (Objekt der Klasse BinarySearchTree) verwaltet. Alle Inhalte, die nach einer Ordnung vor dem Inhalt im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Inhalt im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.) *Beispiel*: Entscheidungsbäume (s.o.) *Beispiel*: Termbaum (s.o.) *Beispiel:* Ahnenbaum (s.o.)

		Materialien: Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 – Anwendung Binärbaum

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
1. Endliche Automaten (a) Vom Automaten in den Schülerinnen und Schülern bekannten Kontexten zur formalen Beschreibung eines endlichen Automaten (b) Untersuchung, Darstellung und Entwicklung endlicher Automaten	Die Schülerinnen und Schüler • 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),	*Beispiele:* Cola-Automat, Geldspielautomat, Roboter, Zustandsänderung eines Objekts „Auto“, Akzeptor für bestimmte Zahlen, Akzeptor für Teilwörter in längeren Zeichenketten, Akzeptor für Terme *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.2 – Endliche Automaten, Formale Sprachen
2. Untersuchung und Entwicklung von Grammatiken regulärer Sprachen (a) Erarbeitung der formalen Darstellung regulärer Grammatiken (b) Untersuchung, Modifikation und Entwicklung von Grammatiken (c) Entwicklung von endlichen Automaten zum Erkennen regulärer Sprachen die durch Grammatiken gegeben werden	• 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	*Beispiele:* reguläre Grammatik für Wörter mit ungerader Parität, Grammatik für Wörter, die bestimmte Zahlen repräsentieren, Satzgliederungsgrammatik *Materialien: (s.o.)*

(d) Entwicklung regulärer Grammatiken zu endlichen Automaten	die Sprache erzeugt (M),	
3. Grenzen endlicher Automaten	• 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).	*Beispiele:* Klammerausdrücke, a^nb^n im Vergleich zu $(ab)^n$

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 maschinenahen 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, dass 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**
1. Von-Neumann-Architektur und die Ausführung maschinennaher Programme a) prinzipieller Aufbau einer von Neumann-Architektur mit CPU, Rechenwerk, Steuerwerk, Register und Hauptspeicher b) einige maschinennahe Befehlen und ihre Repräsentation in einem Binär-Code, der in einem Register gespeichert werden kann c) Analyse und Erläuterung der Funktionsweise eines einfachen	Die Schülerinnen und Schüler • 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).	*Beispiel:* Addition von 4 zu einer eingegeben Zahl mit einem Rechnermodell *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichts-

maschinennahen Programms		vorhaben Q2.3 –Von-Neumann-Architektur und maschinen-nahe Programmierung
2. Grenzen der Automatisierbarkeit a) Vorstellung des Halteproblems b) Unlösbarkeit des Halteproblems c) Beurteilung des Einsatzes von Informatiksystemen hinsichtlich prinzipieller Möglichkeiten und prinzipieller Grenzen		*Beispiel:* Halteproblem *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.3 - Halteproblem

Unterrichtsvorhaben Q2-IV

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

II) Qualifikationsphase ***Leistungskurs***
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 Daten unter Berücksichtigung der Rechteverwaltung und zur Organisation von Arbeitsabläufen sowie zur Verteilung und Zusammenführung von Arbeitsanteilen (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 und Ergänzung ausgewählter Aspekte der objektorientierten Modellierung und Programmierung unter Verwendung einer neuen Entwicklungsumgebung.

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.

Das Aufrufen von Methoden auf Objekten kann dabei als Nachrichtenaustausch zwischen verschiedenen Objekten interpretiert werden und wird als solcher verdeutlicht, indem die Kommunikation zwischen zwei ausgewählten Objekten grafisch dargestellt wird (Zeit-Sequenz-Diagramm). In diesem Zusammenhang wird das Nachrichtenkonzept der objektorientierten Programmierung (Methodenaufruf mit Punktnotation auf Objekten/Klassen – statische versus dynamische Methoden) wiederholt.

Zeitbedarf: 20 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Wiederholung und Erweiterung der objektorientierten Modellierung und Programmierung durch Analyse und Erweiterung eines kontextbezogenen Beispiels (a) Analyse der Problemstellung (b) Analyse der Modellierung (Implementationsdiagramm) (c) Erweiterung der Modellierung im Implementationsdiagramm (Vererbung, abstrakte Klasse, Interface) (d) Kommunikation zwischen mindestens zwei Objekten (grafische Darstellung) (e) Dokumentation von Klassen (f) Implementierung der Anwendung oder von Teilen der Anwendung unter Entwurf einer GUI	Die Schülerinnen und Schüler • 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 professionelle 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),	*Beispiel:* Nim-Spiel Das Nim-Spiel ist ein Spiel für zwei Personen, bei dem abwechselnd eine Anzahl von Gegenständen, etwa Streichhölzer, weggenommen werden. Hier kann eine Interface-Klasse bei einer Klasse Computerspieler und Spieler hilfreich sein. *Beispiel:* Wetthüpfen 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. *Beispiel*: Tannenbaum 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. 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. Weitere Beispiele: Taschenrechner, Geometrische Objekte

	• stellen die Kommunikation zwischen Objekten grafisch dar (D).	*Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.1-Wiederholung (Download Q1-I.1)

Unterrichtsvorhaben Q1-II

Thema: Modellierung und Implementierung von Anwendungen mit dynamischen, linearen Datenstrukturen

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

Vorhabenbezogene Konkretisierung:
Nach Analyse einer Problemstellung in einem geeigneten Anwendungskontext, in dem Daten nach dem First-In-First-Out-Prinzip verwaltet werden, werden der Aufbau von Schlangen am Beispiel dargestellt und die Operationen der Klasse Queue erläutert. Anschließend werden für die Anwendung notwendige Klassen modelliert und implementiert. Eine Klasse für eine den Anforderungen der Anwendung entsprechende Oberfläche wird dabei eventuell 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: 22 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Die Datenstruktur Schlange im Anwendungskontext unter Nutzung der Klasse Queue (a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen (b) Erarbeitung der Funktionalität der Klasse	Die Schülerinnen und Schüler • erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (A), • analysieren und erläutern Algorithmen	*Beispiel: W*arteschlange im Supermarkt (jeder kennt seinen Nachfolger bzw. alternativ: seinen Vorgänger) An der Kasse muss man sich immer hinten anstellen und wird vorne abkassiert. Die Simulationsanwendung stellt eine GUI zur Verfügung, die sich zu Übungszwecken erweitern lässt. *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben

Queue (c) Modellierung und Implementierung der Anwendung unter Verwendung eines oder mehrerer Objekte der Klasse Queue	und Programme (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M), • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • modifizieren Algorithmen und Programme (I), • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • testen Programme systematisch anhand von Beispielen (I), • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D).	Q1.2 – Warteschlange (Achtung: Patientenbeispiel – eher geeignet für Prioritätswarteschlange, zu realisieren als Liste!)
2. Die Datenstruktur Stapel im Anwendungskontext unter Nutzung der Klasse Stack (a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen (b) Erarbeitung der Funktionalität der Klasse Stack (c) Modellierung und Implementierung der Anwendung unter Verwendung eines oder mehrerer Objekte der Klasse Stack		*Beispiel:* Heftstapel In einem Heftstapel soll das Heft einer Schülerin gefunden werden. *Beispiel*: Kisten stapeln 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. Weitere Beispiele: Einkaufswagenstack, Rangierbahnhof, Methodenstack
3. Die Datenstruktur lineare Liste im Anwendungskontext unter Nutzung der Klasse List (a) Erarbeitung der Vorteile der Klasse List im Gegensatz zu den bereits bekannten linearen Strukturen (b) Modellierung und Implementierung einer kontextbezogenen Anwendung unter Verwendung der Klasse List.		*Beispiel:* Nim-Spiel Das im Unterrichtsvorgabe Q1-I implementierte Nim-Spiel wird um eine Liste, die dem Computerspieler als Gedächtnis dienen soll, erweitert. Im Laufe des Spiels, soll sich der Computerspieler über die Liste ungünstige Spielverläufe merken und somit lernen. (Heuristik) *Beispiel*: Abfahrtslauf 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. *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.2 – Listen
4. Vertiefung - Anwendungen von Listen, Stapeln oder Schlangen in mindestens einem weiteren Kontext		*Beispiel:* Skispringen Ein Skispringen hat folgenden Ablauf: Nach dem Sprung erhält der Springer eine Punktzahl und wird nach dieser Punktzahl in eine Rangliste eingeordnet. Die besten 30 Springer qualifizieren sich für den zweiten Durchgang. Sie starten in umgekehrter Reihenfolge gegenüber der Platzierung auf der Rangliste. Nach dem Sprung erhält der Springer wiederum eine Punktzahl und wird nach der Gesamtpunktzahl aus beiden Durchgängen in die endgültige Rangliste eingeordnet. *Beispiel*: Terme in Postfix-Notation

		Die sog. UPN (*Umgekehrt-Polnische-Notation*) bzw. *Postfix-Notation* eines Terms setzt den Operator hinter die Operanden. Um einen Term aus der gewohnten Infixschreibweise in einen Term in UPN umzuwandeln oder um den Wert des Terms zu berechnen, kann ein Stack verwendet werden. *Beispiel*: Rangierbahnhof Auf einem Güterbahnhof gibt es drei Gleise, die nur zu einer Seite offen sind. Wagons können also von einer Seite auf das Gleis fahren und nur rückwärts wieder hinausfahren. Die Wagons tragen Nummern, wobei die Nummer jedoch erst eingesehen werden kann, wenn der Wagon der vorderste an der offenen Gleisseite ist. (Zwischen den Wagons herumzuturnen, um die anderen Wagonnummern zu lesen, wäre zu gefährlich.) Zunächst stehen alle Wagons unsortiert auf einem Gleis. Ziel ist es, alle Wagons in ein anderes Gleis zu fahren, so dass dort die Nummern der Wagons vom Gleisende aus aufsteigend in richtiger Reihenfolge sind. Zusätzlich zu diesen beiden Gleisen gibt es ein Abstellgleis, das zum Rangieren benutzt werden kann. *Beispiel*: Autos an einer Ampel zur Zufahrtsregelung Es soll eine Ampel zur Zufahrtsregelung in Java simuliert werden. An einem geradlinigen, senkrecht von unten nach oben verlaufenden Straßenstück, das von Autos nur einspurig in eine Richtung befahren werden kann, ist ein Haltepunkt markiert, an dem die Ampel steht. Bei einem Klick auf eine Schaltfläche mit der Aufschrift „Heranfahren“ soll ein neues Auto an den Haltepunkt heranfahren bzw. bis an das letzte Auto, das vor dem Haltepunkt wartet. Grünphasen der Ampel werden durch einen Klick auf eine Schaltfläche mit der Aufschrift „Weiterfahren“ simuliert. In jeder Grünphase darf jeweils nur ein Auto weiterfahren. Die anderen Autos rücken nach. *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1-II.3 – Anwendungen für lineare Datenstrukturen

Unterrichtsvorhaben Q1-III

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? Wie können Daten im Anwendungskontext als Graph verwaltet werden? Welche Vor- und Nachteile hat die Modellierung als Graph gegenüber der Verwaltung von Daten als Baum?*

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 Umkehrbarbeit der Zuordnung Binärbaum → Inorder-Ausgabe an einem Beispiel verdeutlicht wird.

Eine Tiefensuche (welche nach dem Programmierparadigma „Backtracking“ arbeitet) wird verwendet, um einen in der Baumstruktur gespeicherten Inhalt zu suchen. Die Breitensuche wird analog erarbeitet und kann als Kürzeste-wege-Algorithmus auf ungewichtete Graphen übertragen werden.

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. Ebenfalls wird die Datenstruktur Graph als starke Verallgemeinerung erarbeitet, modelliert und implementiert.

Zeitbedarf: 34 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Analyse von Baumstrukturen in verschiedenen Kontexten (a) Grundlegende Begriffe (Grad, Tiefe, Höhe, Blatt, Inhalt, Teilbaum, Ebene, Vollständigkeit) (b) Aufbau und Darstellung von binären Bäumen anhand von Baumstrukturen in verschiedenen Kontexten (c) Aufbau und Darstellung von Graphen anhand von Baumstrukturen in verschiedenen Kontexten	Die Schülerinnen und Schüler • 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),	*Beispiel:* Termbaum Der Aufbau von Termen wird mit Hilfe von binären Baumstrukturen verdeutlicht. *Beispiel*: Ahnenbaum Die binäre Baumstruktur ergibt sich daraus, dass jede Person genau einen Vater und eine Mutter hat. *Weitere Beispiele für Anwendungskontexte für binäre Bäume:* *Beispiel*: Suchbäume (zur sortierten Speicherung von Daten) Alle Inhalte, die nach einer Ordnung vor dem Inhalt im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Inhalt im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.) *Beispiel*: 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

	• entwickeln iterative und rekursive Algorithmen unter Nutzung der Konstruktionsstrategien „Modularisierung“ und „Teilen und Herrschen“, „Backtracking“ (M), • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I), • modifizieren Algorithmen und Programme (I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • testen Programme systematisch anhand von Beispielen (I), • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D), • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D).	im linken Teilbaum, die Fragen, die möglich sind, wenn die Antwort „nein“ lautet, stehen im rechten Teilbaum. *Beispiel*: 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 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. *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 – Binärbaum
2. Die Datenstruktur Binärbaum im Anwendungskontext unter Nutzung der Klasse BinaryTree (a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen im Anwendungskontext (b) Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramms (c) Erarbeitung der Klasse BinaryTree und beispielhafte Anwendung der Operationen		*Beispiel*: Informatikerbaum *als binärer Baum* In einem *binären Baum* werden die Namen und die Geburtsdaten von Informatikern lexikographisch geordnet abgespeichert. Alle Namen, die nach dieser Ordnung vor dem Namen im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Namen im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.) Folgende Funktionalitäten werden benötigt:

(d) Implementierung der Anwendung oder von Teilen der Anwendung (e) Traversierung eines Binärbaums im Pre-, In- und Postorderdurchlauf		• Einfügen der Informatiker-Daten in den Baum • Suchen nach einem Informatiker über den Schlüssel Name • Ausgabe des kompletten Datenbestands in nach Namen sortierter Reihenfolge <u>*Weitere Beispiele für Anwendungskontexte Graph:*</u> *Modellierung von Strassenkarten, Stadtplänen, Gebäudeplänen, U-Bahnplänen als Graphen und Bestimmung kürzester Wege (Algorithmus von Dijkstra), Modellierung in Optimierungskontexten z.B. Aufbau eines Strom-/Rechnernetzwerks mit möglichst wenigen Kanten (Minimaler aufspannender Baum: Algorithmus von Kruskal).* *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 – Binärbaum
3. Die Datenstruktur binärer Suchbaum im Anwendungskontext unter Verwendung der Klasse BinarySearchTree (a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen (b) Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramm,		*Beispiel*: Informatikerbaum als *Suchbaum* In einem binären *Suchbaum* werden die Namen und die Geburtsdaten von Informatikern lexikographisch geordnet abgespeichert. Alle Namen, die nach dieser Ordnung vor dem Namen im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Namen im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.)

grafische Darstellung eines binären Suchbaums und Erarbeitung der Struktureigenschaften (c) Erarbeitung der Klasse BinarySearchTree und Einführung des Interface Item zur Realisierung einer geeigneten Ordnungsrelation (d) Implementierung der Anwendung oder von Teilen der Anwendung inklusive einer sortierten Ausgabe des Baums		Folgende Funktionalitäten werden benötigt: • Einfügen der Informatiker-Daten in den Baum • Suchen nach einem Informatiker über den Schlüssel Name • Ausgabe des kompletten Datenbestands in nach Namen sortierter Reihenfolge *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 – Binärer Suchbaum
4. Übung und Vertiefungen der Verwendung von Binärbäumen oder binären Suchbäumen und Graphen anhand weiterer Problemstellungen		*Beispiel:* Codierungsbäume (s.o.) oder Huffman-Codierung *Beispiel*: Buchindex Es soll eine Anwendung entwickelt werden, die anhand von Stichworten und zugehörigen Seitenzahlen ein Stichwortregister erstellt. Da die Stichwörter bei der Analyse des Buches häufig gesucht werden müssen, werden sie in der Klasse Buchindex als Suchbaum (Objekt der Klasse BinarySearchTree) verwaltet. Alle Inhalte, die nach einer Ordnung vor dem Inhalt im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Inhalt im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.)

		Beispiel: Entscheidungsbäume (s.o.) *Beispiel*: Termbaum (s.o.) *Beispiel:* Ahnenbaum (s.o.) *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 – Anwendung Binärbaum

Unterrichtsvorhaben Q1-IV

Thema: Suchen und Sortieren auf linearen Datenstrukturen mit Hilfe von Baumstrukturen

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 und der Vorteil einer sortiert vorgehaltenen Datenstruktur erarbeitet. 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.

Anschließend wird die Implementationen von Quicksort sowie dem Sortieren durch Einfügen 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 analysiert und beurteilt. Dabei werden weitere (schlüsselbasierte und spezialisierte) Sortierverfahren betrachtet, analysiert und eventuell implementiert. Als Verfahren mit worst-case Laufzeit O(n log n), welche adressatengerecht analysiert werden kann eignet sich MergeSort, welches ebenfalls ein Beispiel des Programmierparadigmas „Divide-And-Conquer“ darstellt.

Zeitbedarf: 22 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Suchen von Daten in Listen und Arrays (a) Lineare Suche in Listen und in Arrays (b) Binäre Suche in Arrays als Beispiel für rekursives Problemlösen (c) Untersuchung der beiden Suchverfahren hinsichtlich ihrer Effizienz (Laufzeitverhalten, Speicherbedarf)	Die Schülerinnen und Schüler • 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),	*Beispiel:* Karteiverwaltung Für ein Adressverwaltungsprogramm soll eine Methode zum Suchen einer Adresse geschrieben werden. *Beispiel*: Bundesjugendspiele Die Teilnehmer an Bundesjugendspielen nehmen an drei Disziplinen teil und erreichen dort Punktzahlen. Diese werden in einer Wettkampfkarte eingetragen und an das Wettkampfbüro gegeben. Zur Vereinfachung sollte sich das Modell auf die drei Disziplinen
2. Sortieren in Listen und Arrays - Entwicklung und Implementierung von iterativen und rekursiven		

Sortierverfahren (a) Entwicklung und Implementierung eines einfachen Sortierverfahrens für eine Liste (b) Implementierung eines einfachen Sortierverfahrens für ein Feld (c) Entwicklung eines rekursiven Sortierverfahren für ein Feld (z.B. Sortieren durch Mischen)	• 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).	„Lauf”, „Sprung“ und „Wurf“ beschränken. Im Wettkampfbüro wird das Ergebnis erstellt. Das Programm soll dafür zunächst den Besten einer Disziplin heraussuchen können und später das gesamte Ergebnis nach gewissen Kriterien sortieren können. Beispiel: Rate die Zahl Denke Dir eine Zahl zwischen 0 und 100. Wie viele Versuche braucht der andere, um nur mit den Hinweisen > oder < die Zahl herauszufinden? – Hieran können sowohl die Vorteile der Sortierung (Ordnung der natürlichen Zahlen) als auch der binären Suchbäume demonstriert werden. *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.3 - Suchen
3. Untersuchung der Effizienz der Sortierverfahren „Sortieren durch direktes Einfügen“ und „Quick-Sort“ auf linearen Listen (a) Grafische Veranschaulichung der Sortierverfahren (b) Untersuchung der Anzahl der Vergleichsoperationen und des Speicherbedarf bei beiden Sortierverfahren (c) Beurteilung der Effizienz der beiden Sortierverfahren		

Unterrichtsvorhaben Q1-V

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: 22 Stunden

Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Nutzung von relationalen Datenbanken (a) Aufbau von Datenbanken und Grundbegriffe • Entwicklung von Fragestellungen zur vorhandenen Datenbank • Analyse der Struktur der vorgegebenen Datenbank und Erarbeitung der Begriffe Tabelle, Attribut, Datensatz, Datentyp, Primärschlüssel, Fremdschlüssel, Datenbankschema (b) SQL-Abfragen • 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) (c) Vertiefung an einem weiteren Datenbankbeispiel	Die Schülerinnen und Schüler • 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 einen Datenbanksystem zu extrahieren (I), • ermitteln Ergebnisse von Datenbankabfragen über mehrere verknüpfte Tabellen (D), • stellen Entitäten mit ihren Attributen und die Beziehungen zwischen Entitäten in einem Entity-Relationship-Diagramm grafisch dar (D), • überprüfen Datenbankschemata auf vorgegebene Normalisierungseigenschaften (D).	*Beispiel:* VideoCenter VideoCenter ist die Simulation einer Online-Videothek für den Informatik-Unterricht mit Webfrontends zur Verwaltung der Kunden, der Videos und der Ausleihe. Außerdem ist es möglich direkt SQL-Abfragen einzugeben. Es ist auch möglich, die Datenbank herunter zu laden und lokal zu installieren. Unter schule.de (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. *Beispiel*: Schulbuchausleihe Unter brd.nrw.de (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.
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		*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

• Modellierung eines relationalen Datenbankschematas 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)		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* 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.

Unterrichtsvorhaben Q2-I

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 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, insbesondere kontextfreie, werden untersucht, entwickelt oder modifiziert. An einem Beispiel werden die Grenzen endlicher Automaten ausgelotet und durch die Anwendung von Kellerautomaten erweitert.

Zeitbedarf: 26 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien oder Materialien
1. Endliche Automaten (a) Vom Automaten in den Schülerinnen und Schülern bekannten Kontexten zur formalen Beschreibung eines endlichen Automaten (b) Untersuchung, Darstellung und Entwicklung endlicher Automaten	Die Schülerinnen und Schüler • analysieren und erläutern die Eigenschaften endlicher Automaten einschließlich ihres Verhaltens auf bestimmte Eingaben (A), • analysieren und erläutern Grammatiken (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),	*Beispiele:* Cola-Automat, Geldspielautomat, Roboter, Zustandsänderung eines Objekts „Auto“, Akzeptor für bestimmte Zahlen, Akzeptor für Teilwörter in längeren Zeichenketten, Akzeptor für Terme *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.2 – Endliche Automaten, Formale Sprachen
2. Untersuchung und Entwicklung von Grammatiken Sprachen (a) Erarbeitung der formalen Darstellung Grammatiken (b) Untersuchung, Modifikation und Entwicklung von Grammatiken (c) Entwicklung von endlichen Automaten zum Erkennen Sprachen die durch Grammatiken gegeben werden	• entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M), • entwickeln zur akzeptierten Sprache eines Automaten die zugehörige Grammatik (M), • entwickeln zur Grammatik einer regulären/kontextfreien Sprache einen zugehörigen endlichen Automaten/Kellerautomaten (M), • modifizieren Grammatiken regulärer/kontextfreier Sprachen (M),	*Beispiele:* reguläre Grammatik für Wörter mit ungerader Parität, Grammatik für Wörter, die bestimmte Zahlen repräsentieren, Satzgliederungsgrammatik *Materialien: (s.o.)*

(d) Entwicklung Grammatiken zu endlichen Automaten	• entwickeln zu einer regulären/kontextfreien Sprache eine Grammatik, die die Sprache erzeugt (M),	
3. Grenzen endlicher Automaten (a) Zählproblem (b) Lösung durch die Einführung von Kellerautomaten	• 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).	*Beispiele:* Klammerausdrücke, a^nb^n im Vergleich zu $(ab)^n$

Unterrichtsvorhaben Q2-II

Thema: Kommunikation in Netzwerken, Client-Server-Anwendungen, Sicherheit und Datenschutz in Netzstrukturen

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

Vorhabenbezogene Konkretisierung:
Ausgehend von realen Rechnernetzen (Schulnetzwerk, Internet) oder im historischen Kontext werden die Grundlagen und Voraussetzungen (Kommunikationsprotokolle, OSI-Schichtenmodell) für Datenübertragung erarbeitet. Anschließend werden 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: 36 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Kommunikation in Rechnernetzen (a) Beschreibung der Kommunikation im Netz anhand eines Anwendungskontextes und einer Client-Server-Struktur zur Klärung der Funktionsweise der Datenübertragung (b) Netztopologien als Grundlage von Client-Server-Strukturen und TCP/IP-Schichtenmodell als Beispiel für eine Paketübermittlung in einem Netz (c) Vertraulichkeit, Integrität, Authentizität in Netzwerken sowie symmetrische und asymmetrische kryptografische Verfahren (Cäsar-, Vigenère-, RSA-Verfahren) als Methoden Daten im Netz verschlüsselt zu übertragen	Die Schülerinnen und Schüler • entwickeln und erweitern Protokolle zur Kommunikation in einem Client-Server-Netzwerk (M) • analysieren und erläutern Protokolle zur Kommunikation in einem Client-Server-Netzwerk (A) • beschreiben und erläutern Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken (A), • analysieren und erläutern Funktionsweisen, 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),	*Beispiel: Indianer* Entwicklung der Grundlagen der elektronischen Datenübertragung (Kodierung) am Beispiel von Rauchzeichen zwischen Indianerdörfern, dabei Wiederholung von Binärkodierung als Basis des Rechneraufbaus. Diskussion von fehlerhafter Übertragung und Prüfverfahren *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben, Verschlüsselung Q1.5 - Zugriff auf Daten in Netzwerken Simulation eines realen Rechnernetzes mit Filius: http://www.lernsoftware-filius.de/ Nach einem Skript können komplexer werdende

	• 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).	Rechnernetze (einzelne, mehrere Subnetze) aufgebaut werden und der Datenaustausch kann schichtweise mitverfolgt werden.
2. Fallbeispiele zur Datenschutzproblematik und zum Urheberrecht		*Materialien:* *Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.5* - Datenschutz beim Videocenter, Materialblatt-Datenschutzgesetz

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 maschinenahen Programms mit diesen Komponenten vorstellen? Welche Möglichkeiten bieten Informatiksysteme und wo liegen ihre Grenzen? Welche theoretischen Überlegungen existieren zur begrenzten Leistungsfähigkeit von Informatiksystemen?*

Vorhabenbezogene Konkretisierung:
Anhand einer von-Neumann-Architektur und einem maschinennahen Programm wird die prinzipielle Arbeitsweise von Computern verdeutlicht. Diese kann mit Hilfe des Modells der Turingmaschine weiter abstrahiert werden.

Ausgehend von den prinzipiellen Grenzen endlicher Automaten bzw. Kellerautomaten 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, dass 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
1. Von-Neumann-Architektur und die Ausführung maschinennaher Programme a) prinzipieller Aufbau einer von Neumann-Architektur mit CPU, Rechenwerk, Steuerwerk, Register und Hauptspeicher b) einige maschinennahe Befehlen und ihre Repräsentation in einem Binär-Code, der in einem Register gespeichert werden kann c) Analyse und Erläuterung der Funktionsweise eines einfachen maschinennahen Programms	Die Schülerinnen und Schüler • 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).	*Beispiel:* Addition von 4 zu einer eingegeben Zahl mit einem Rechnermodell *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.3 –Von-Neumann-Architektur und maschinennahe Programmierung
2. Grenzen der Automatisierbarkeit a) Vorstellung des Halteproblems b) Unlösbarkeit des Halteproblems c) Beurteilung des Einsatzes von Informatiksystemen hinsichtlich prinzipieller Möglichkeiten und prinzipieller Grenzen		*Beispiel:* Halteproblem *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.3 – Halteproblem *Beispiel:* Das Affenpuzzle Ein Puzzle mit vielen Freiheitsgraden in der Anordnung der Teile gibt den SuS ein Gefühl für schwer lösbare Probleme

		(Hinführung zu Komplexitätsklassen P, NP) Jens Gallenbacher – Abenteuer Informatik

Unterrichtsvorhaben Q2-IV

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

2.2 Grundsätze der fachmethodischen und fachdidaktischen Arbeit

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

Überfachliche Grundsätze:

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

Fachliche Grundsätze:

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

2.3 Grundsätze der Leistungsbewertung und Leistungsrückmeldung

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

2.3.1 Beurteilungsbereich Klausuren

Verbindliche Absprachen:
Bei der Formulierung von Aufgaben werden die für die Abiturprüfungen geltenden Operatoren des Faches Informatik schrittweise eingeführt, erläutert und dann im Rahmen der Aufgabenstellungen für die Klausuren benutzt. Es werden die Korrekturzeichen nach Vorgaben des Schulministeriums verwendet.

Instrumente:

- Einführungsphase: 1 Klausur je 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 im 1. Halbjahr der Q1 eine Facharbeit geschrieben werden.

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

Kriterien

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

Die zweite Klausur in der Einführungsphase soll von den parallel unterrichtenden Kollegen stichprobenartig kreuzkorrigiert werden.

Spätestens ab der Qualifikationsphase orientiert sich die Zuordnung der Hilfspunktsumme zu den Notenstufen an dem Zuordnungsschema des Zentralabiturs.
Von diesem kann aber im Einzelfall begründet abgewichen werden, wenn sich z.B. besonders originelle Teillösungen nicht durch Hilfspunkte gemäß den Kriterien des Erwartungshorizontes abbilden lassen oder eine Abwertung wegen besonders schwacher Darstellung (APO-GOSt §13 (2)) angemessen erscheint.

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

2.3.2 Beurteilungsbereich Sonstige Mitarbeit

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

Verbindliche Absprachen der Fachkonferenz

- Alle Schülerinnen und Schüler führen in der Einführungsphase in Kleingruppen ein Kurzprojekt durch und fertigen dazu eine Arbeitsmappe mit Arbeitstagebuch an. Dies wird in die Note für die Sonstige Mitarbeit einbezogen.

- In der Qualifikationsphase erstellen, dokumentieren und präsentieren die Schülerinnen und Schüler in Kleingruppen ein anwendungsbezogenes Softwareprodukt. Dies wird in die Note für die Sonstige Mitarbeit einbezogen.

Leistungsaspekte

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

<u>Praktische Leistungen am Computer</u>
- Implementierung, Test und Anwendung von Informatiksystemen

<u>Sonstige schriftliche Leistungen</u>
- Arbeitsmappe und Arbeitstagebuch zu einem durchgeführten Unterrichtsvorhaben
- Lernerfolgsüberprüfung durch kurze schriftliche Übungen
  Schriftliche Übung dauern ca. 20 Minuten und umfassen den Stoff der letzten ca. 4–6 Stunden.
- Bearbeitung von schriftlichen Aufgaben im Unterricht

Kriterien

Die folgenden allgemeinen Kriterien gelten sowohl für die mündlichen als auch für die schriftlichen Formen der sonstigen Mitarbeit.

Die Bewertungskriterien stützen sich auf
- die Qualität der Beiträge,
- die Quantität der Beiträge und
- die Kontinuität der Beiträge.

Besonderes Augenmerk ist dabei auf

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

Bei Gruppenarbeiten auch auf

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

Bei Projektarbeit darüber hinaus auf

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

Grundsätze der Leistungsrückmeldung und Beratung

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

- nach einer mündlichen Überprüfung,
- bei Rückgabe von schriftlichen Leistungsüberprüfungen,
- nach Abschluss eines Projektes,
- nach einem Vortrag oder einer Präsentation,
- bei auffälligen Leistungsveränderungen,
- auf Anfrage,
- als Quartalsfeedback und
- zu Eltern- oder Schülersprechtagen.

Die Leistungsrückmeldung kann

- durch ein Gespräch mit der Schülerin oder dem Schüler,
- durch einen Feedbackbogen,
- durch die schriftliche Begründung einer Note oder
- durch eine individuelle Lern-/Förderempfehlung

erfolgen.

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

3 Entscheidungen zu fach- und unterrichtsübergreifenden Fragen

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

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

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

4 Qualitätssicherung und Evaluation

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

Das schulinterne Curriculum ist zunächst bis 2025 verbindlich.