Heinrich-Heine-Gymnasium
Bottrop

Schulinterner Lehrplan

zum Kernlehrplan für die

Sekundarstufe II

des Faches

Informatik

Stand: **29.05.2015**

geplante Überarbeitung (z.B. wg. Abiturvorgaben): Ende des Schuljahres 2015/16

1. Das Heinrich Heine Gymnasium

Das Heinrich-Heine Gymnasium liegt am Rande der Bottroper Innenstadt. Es ist fünfzügig und hat im Schuljahr 2014/15 ca. 1.000 Schülerinnen und Schüler.

Das großzügige Schulgelände in ruhiger Umgebung und die modernisierten Gebäude sind die Rahmenbedingungen für eine Schule mit einem offenen, freundlichen und schülerzentrierten Schulklima.

Das Schulprogramm und die Schulordnung („WIR“) formulieren unseren Anspruch, eine leistungsorientierte und an den individuellen Fähigkeiten und Bedürfnissen der Schülerinnen und Schüler ausgerichtete Schule zu sein, die den Schülerinnen und Schülern das bestmögliche Rüstzeug für ihren weiteren Lebensweg gibt, sowohl hinsichtlich ihrer fachlichen und sozialen Kompetenzen als auch hinsichtlich einer breit angelegten Bildung der Persönlichkeiten.

Die fachliche Profilierung der Schule stellt sich wie folgt dar:

Sprachliches Profil:
Neben Englisch und Latein Plus als Eingangssprache besteht in der Jahrgangsstufe 6 die Wahl zwischen Latein und Französisch. In der Differenzierung in Klasse 8 kann Französisch oder Spanisch als dritte Fremdsprache gewählt werden und das Fach Italienisch wird am Heinrich-Heine-Gymnasium Bottrop als spät einsetzende Fremdsprache in der gymnasialen Oberstufe angeboten.

Naturwissenschaftliches Profil:
Die Naturwissenschaften und die Informatik sind über das Fach MINT in der Erprobungsstufe und der Klasse 7, die Informatik und die NW (Naturwissenschaften)-Kurse im Differenzierungsbereich sowie die Leistungskurse in Biologie, Physik und Chemie in der Sekundarstufe II fest verankert.

Künstlerisch-musisches Profil:
Die Orientierungsstufe bietet in diesem Bereich den Orchesterkurs als Alternative zu dem herkömmlichen Musikunterricht; im Bereich Musik besteht eine enge Kooperation mit der Musikschule der Stadt Bottrop, die den Schülerinnen und Schülern die Möglichkeit der Instrumentalausbildung im Rahmen der Schule bietet und Grundlage für die breit angelegte Orchesterarbeit darstellt; im Differenzierungsbereich wird die Kombination „Kunst und Geschichte“ angeboten; der Leistungskurs Kunst ist festes Angebot in der Sekundarstufe II.

1.1 Das Fach Informatik am Heinrich Heine Gymnasium

Speziell das Fach Informatik ist organisatorisch in der Sekundarstufe I in den MINT-Zweig der Schule eingebunden, den Schülerinnen und Schüler als Ergänzung zum obligatorischen Unterricht anwählen können. Dort werden unter anderem in altersgerechter Weise mit einer geeigneten didaktischen Lernumgebung Computerspiele programmiert sowie Lego Mindstorms-Roboter konstruiert und programmiert, um gestellte Aufgaben autonom zu bewältigen.

Das Fach Informatik wird am Heinrich Heine Gymnasium ab der Jahrgangsstufe 8 im Wahlpflichtbereich II (WP II) vierstündig unterrichtet und von etwa einem Viertel der Schülerinnen und Schüler besucht. In der zweijährigen Laufzeit dieser Kurse wird in altersstufengerechter Weise unter anderem auf Grundlagen der Algorithmik am Beispiel einer didaktischen Lernumgebung und auf die technische Informatik am Beispiel von Schaltwerken und Schaltnetzen eingegangen, die praxisnah auch experimentell untersucht werden. Es werden aber auch einige Inhalte der theoretischen Informatik angesprochen wie z.B. die Effizienz von Algorithmen am Beispiel der Suche nach dem kürzesten Weg auf einer Landkarte oder bei der Sicherheit von Verschlüsselungsverfahren. Der Unterricht erfolgt dabei in enger Verzahnung mit Inhalten der Mathematik und Physik.

In der Sekundarstufe II bietet das Heinrich Heine Gymnasium für die eigenen Schülerinnen und Schüler in allen Jahrgangsstufen jeweils ein bis zwei Grundkurse 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. Das Fach Informatik kann dabei am Ende der Qualifikationsphase insbesondere als drittes oder viertes Abiturfach gewählt werden.

Der Unterricht erfolgt im 45-Minuten-Takt. Die Kursblockung sieht grundsätzlich für Grundkurse eine Doppelstunde und eine Einzelstunde vor, wobei letztere ebenfalls zu einer Doppelstunde im Zwei-Wochen-Takt zusammengefasst werden kann.

2. Die Fachgruppe Informatik

Zurzeit, im Schuljahr 2014/15, besteht die Fachschaft Informatik des Heinrich Heine Gymnasiums aus vier Lehrkräften. In der gymnasialen Oberstufe gibt es in der Einführungsphase 2 Grundkurse und in den Jahrgangsstufen Q1 und Q2 jeweils einen Grundkurs.

2.1 Funktionen und Aufgaben der Fachgruppe vor dem Hintergrund des Schulprogramms

Die Fachgruppe Informatik sieht sich besonders dem Leitziel der Handlungsfähigkeit in einer zunehmend technisch-informatisch geprägten Umwelt verpflichtet. Neben den fachlichen Zielen werden durch die Auseinandersetzung mit Themen wie Urheberrecht, Datenschutz und Sicherheit Werte und Normen im Umgang mit digitalen Medien, den eigenen Daten und den Daten Dritter vermittelt.

In Übereinstimmung mit dem Schulprogramm setzt sich die Fachgruppe auch das Ziel, Schülerinnen und Schüler zu unterstützen, selbstständige, eigenverantwortliche, selbstbewusste, sozial kompetente und engagierte Persönlichkeiten zu werden. In der Sekundarstufe II sollen die Schülerinnen und Schüler darüber hinaus auf die zukünftigen Herausforderungen in Studium und Beruf vorbereitet werden. Dazu gehört auch der sichere und verantwortungsvolle Umgang mit informatischen Systemen.

2.2 Verfügbare Ressourcen

Der Fachgruppe Informatik stehen vier Computerräume mit jeweils ca. 30 Computerarbeitsplätzen zur Verfügung. Alle Arbeitsplätze sind an das schulinterne Rechnernetz angeschlossen, so dass Schülerinnen und Schüler über einen individuell gestaltbaren Zugang zum zentralen Server der Schule alle Arbeitsplätze der vier Räume zum Zugriff auf ihre eigenen Daten, zur Recherche im Internet oder zur Bearbeitung schulischer Aufgaben verwenden können.

An allen Rechnern sind die gängigen Programme zur Textverarbeitung, Tabellenkalkulation und Präsentationserstellung installiert. Darüber hinaus wird ausschließlich kostenlose Software verwendet, die somit von den Schülerinnen und Schülern auch zuhause z.B. zur Vor- oder Nachbereitung des Unterrichts verwendet werden kann.

In der Schülerbibliothek stehen zahlreiche Fachbücher zur Verfügung.

2.3 Funktionsinhaber der Fachgruppe

- Fachkonferenzvorsitzernder: Sven Biermann, OStR
- Stellvertreter: N.N. (ehem. Frau Wiebke Thiel)

3. Entscheidungen zum Unterricht

Der Unterricht der Sekundarstufe II wird mit Hilfe der Programmiersprache Java durchgeführt. In der Einführungsphase kommt dabei die kostenlose Entwicklungsumgebung Greenfoot für den Einstieg in die Java-Programmierung zum Einsatz, welche das Erstellen von grafischen Programmen erleichtert. Spätestens in der Qualifikationsphase werden andere kostenlose, didaktische Entwicklungsumgebungen wie BlueJ oder der JavaEditor verwendet.

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.

3.1 Unterrichtsvorhaben

Die Darstellung der Unterrichtsvorhaben im schulinternen Lehrplan besitzt den Anspruch,sämtliche im Kernlehrplan Informatik für die Gymnasiale Oberstufe angeführten Kompetenzen abzudecken. Dies entspricht der Verpflichtung jeder Lehrkraft, Schülerinnen und Schülern Lerngelegenheiten zu ermöglichen, so dass alleKompetenzerwartungen des Kernlehrplans von ihnen erfüllt werden können. Die entsprechende Umsetzung erfolgt auf zwei Ebenen: der Übersichts- und der Konkretisierungsebene.

Im „Übersichtsraster Unterrichtsvorhaben“ (Abschnitt 3.1.1) 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“ (Abschnitt 3.1.2) 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 Abschnitten 3.2 bis 3.3 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.

3.1.1 Übersichtsraster Unterrichtsvorhaben

Übersichtsraster EF

Einführungsphase	
Unterrichtsvorhaben EF-I Thema: *Was ist Informatik? Informatik damals, heute und morgen…* Zentrale Kompetenzen: – Argumentieren – Darstellen und Interpretieren – Kommunizieren und Kooperieren Inhaltsfelder: – Informatiksysteme – Informatik, Mensch und Gesellschaft Inhaltliche Schwerpunkte: – Geschichte der automatischen Datenverarbeitung – Einsatz von Informatiksystemen – Auswirkungen der Automatisierung auf Mensch und Gesellschaft – Internet und Datenschutz Zeitbedarf: 10 Stunden	*Unterrichtsvorhaben EF-II* Thema: *Wie funktioniert ein Computer?* Zentrale Kompetenzen: – Modellieren – Argumentieren – Darstellen und Interpretieren – Kommunizieren und Kooperieren Inhaltsfelder: – Informatiksysteme – Algorithmen Inhaltliche Schwerpunkte: – Digitalisierung – Information und Daten – Darstellung im Binärsystem – EVA-Prinzip – von-Neumann-Architektur – Analyse einfacher maschinennaher Programme Zeitbedarf: 12 Stunden

Einführungsphase	
Unterrichtsvorhaben EF-III Thema: *Such- und Sortieralgorithmen anhand kontextbezogener Beispiele* Zentrale Kompetenzen: – Argumentieren – Modellieren – Darstellen und Interpretieren – Kommunizieren und Kooperieren Inhaltsfelder: – Algorithmen Inhaltliche Schwerpunkte: – Algorithmen zum Suchen und Sortieren – Analyse, Entwurf und Implementation einfacher Algorithmen Zeitbedarf: 9 Stunden	*Unterrichtsvorhaben EF-IV* Thema: *Grundlagen der objektorientierten Modellierung und Implementation einfacher Algorithmen am Beispiel eines Marsroboters in Greenfoot* 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 – Syntax und Semantik einer Programmiersprache – Analyse, Entwurf und Implementierung einfacher Algorithmen Zeitbedarf: 18 Stunden

Einführungsphase	
Unterrichtsvorhaben EF-V Thema: *Modellierung und Implementierung von Klassen- und Objektbeziehungen anhand von grafischen Spielen und Simulationen* 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 – Syntax und Semantik einer Programmiersprache – Analyse, Entwurf und Implementierung einfacher Algorithmen Zeitbedarf: 18 Stunden	*Unterrichtsvorhaben EF-VI* Thema: *Entwicklung eines eigenen Softwareprojekts – Planung, Durchführung, Dokumentation und Präsentation* 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 – Syntax und Semantik einer Programmiersprache – Analyse, Entwurf und Implementierung einfacher Algorithmen Zeitbedarf: 18 Stunden
Summe Einführungsphase: 74 Stunden	

Übersichtsraster Q1

Qualifikationsphase 1	
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

Qualifikationsphase 1	
Unterrichtsvorhaben Q1-III Thema: *Suchen und Sortieren auf linearen Datenstrukturen* Zentrale Kompetenzen: – Argumentieren – Modellieren – Implementieren – Darstellen und Interpretieren – Kommunizieren und Kooperieren Inhaltsfelder: – Algorithmen – Formale Sprachen und Automaten Inhaltliche Schwerpunkte: – Analyse, Entwurf und Implementierung von Algorithmen – Algorithmen in ausgewählten informatischen Kontexten – Syntax und Semantik einer Programmiersprache Zeitbedarf: 16 Stunden	*Unterrichtsvorhaben Q1-IV* Thema: *Modellierung und Nutzung vonrelationalen Datenbanken in Anwendungskontexten* Zentrale Kompetenzen: – Argumentieren – Modellieren – Implementieren – Darstellen und Interpretieren – Kommunizieren und Kooperieren Inhaltsfelder: – Daten und ihre Strukturierung – Algorithmen – Formale Sprachen und Automaten – Informatik, Mensch und Gesellschaft Inhaltliche Schwerpunkte: – Datenbanken – Algorithmen in ausgewählten informatischen Kontexten – Syntax und Semantik einer Programmiersprache – Sicherheit Zeitbedarf: 20 Stunden

Qualifikationsphase 1	
Unterrichtsvorhaben Q1-V Thema: *Sicherheit und Datenschutz in Netzstrukturen* Zentrale Kompetenzen: – Argumentieren – Darstellen und Interpretieren – Kommunizieren und Kooperieren Inhaltsfelder: – Informatiksysteme – Informatik, Mensch und Gesellschaft Inhaltliche Schwerpunkte: – Einzelrechner und Rechnernetzwerke – Sicherheit – Nutzung von Informatiksystemen, Wirkungen der Automatisierung Zeitbedarf: 10 Stunden	
Summe Qualifikationsphase 1: 74 Stunden	

Übersichtsraster Q2

Qualifikationsphase 2	
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	*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

Qualifikationsphase 2	
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	
Summe Qualifikationsphase 2: 56 Stunden	

3.1.2 Konkretisierte Unterrichtsvorhaben
Im Folgenden werden die im Abschnitt 3.1.1 aufgeführten Unterrichtsvorhaben konkretisiert.

Verbindliche Festlegungen der Fachkonferenz:
Die Fachkonferenz des Heinrich Heine Gymnasiums hat Themen, Leitfragen und die Ausführungen unter der Überschrift Vorhabenbezogene Konkretisierung verbindlich vereinbart, ebenso die Sequenzierung der Unterrichtsvorhaben (erste Tabellenspalte) und die ausgewiesenen Kompetenzen (zweite Tabellenspalte). Alle Mitglieder der Fachkonferenz haben sich darauf verständigt, in ihrem Unterricht Lerngelegenheiten anzubieten, so dass Schülerinnen und Schüler diese Kompetenzen im Rahmen der festgelegten Unterrichtssequenzen erwerben oder vertiefen können.

Unterrichtliche Anregungen:
Die angeführten Beispiele, Medien und Materialien (dritte Tabellenspalte) sind dagegen Vorschläge bzw. Hilfen für die Lehrkräfte des Heinrich Heine Gymnasiums. In diesen Bereichen sind Abweichungen von den vorgeschlagenen Vorgehensweisen möglich.

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

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).

In der Einführungsphase wird die didaktische Entwicklungsumgebung Greenfoot verwendet, die unter www.greenfoot.org kostenlos heruntergeladen werden kann. Auf dieser Seite befinden sich auch zahlreiche Beispielprojekte und (i.d.R. englischsprachige) Tutorials und Dokumentationen. Die deutschsprachige Seite www.greenfoot-center.de bietet weitere Dokumentationen und Tutorials.

Unterrichtsvorhaben EF-I

Thema: Was ist Informatik? Informatik damals, heute und morgen…

Leitfragen: *Womit beschäftigt sich die Wissenschaft Informatik? Wie hat sich die Wissenschaft Informatik entwickelt? Welche gesellschaftlichen Auswirkungen gingen und gehen damit einher? Welche Anforderungen an den Datenschutz ergeben sich daraus?*

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. Aufgrund ihrer teils sehr unterschiedlichen Vorerfahrungen und Erwartungen an das Fach Informatik ist es sinnvoll, dessen unterschiedlichen Facetten und Ziele zu erarbeiten.

Die im Vergleich zu den meisten anderen Wissenschaften eher kurze und spannende Geschichte der Informatik ist gut geeignet, um einen von den teils sehr unterschiedlichen Vorerfahrungen der Schülerinnen und Schüler unabhängigen ersten Zugang zum Fach Informatik zu erhalten. Hierbei sind insbesondere die wesentlichen Kernideen, welche die Entwicklung des Computers bzw. des Faches Informatik maßgeblich vorangetrieben haben und sich auch heute noch in modernen Rechnern als wesentliche Konzepte wiederfinden, besonders herauszuarbeiten, um die Geschichte der Informatik mit der Lebenswelt der Schülerinnen und Schüler zu verknüpfen.

Anschließend wird verstärkt auf den Wandel der Gesellschaft und der Arbeitswelt durch die Möglichkeit der automatischen Datenverarbeitung durch den Computer eingegangen. Dabei wird insbesondere der Aspekt des Datenschutzes thematisiert. Dazu wird das Bundesdatenschutzgesetz in Auszügen behandelt und auf schülernahe Beispielsituationen zur Anwendung gebracht. Dabei steht keine formale juristische Bewertung der Beispielsituationen im Vordergrund, die im Rahmen eines Informatikunterrichts auch nicht geleistet werden kann, sondern vielmehr eine persönliche Einschätzung von Fällen im Geiste des Datenschutzgesetzes.

Zeitbedarf: 10 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Was ist Informatik? (a) Informatik als Wissenschaft der Verarbeitung von Informationen (b) Teilgebiete des Faches Informatik (c) Ziele des Unterrichtsfaches Informatik	Die Schülerinnen und Schüler – erläutern wesentliche Grundlagen der Geschichte der digitalen Datenverarbeitung (A), – nutzen das Internet zur Recherche, zum Datenaustausch und zur Kommunikation (K), – bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen (A).	*Beispiel:* Internetrecherche Die Schülerinnen und Schüler recherchieren das Zitat „In der Informatik geht es …“ von E. Dijkstra und anschließend, wie dies zu verstehen ist. Sie identifizieren dabei Teilgebiete und Aufgaben des Faches Informatik und fassen diese in einem gemeinsamen Informatik-Wiki zusammen.
2. Eine kurze Geschichte der Informatik		*Beispiel:* Gruppenpuzzle Die Schülerinnen und Schüler fassen zu vorgegebenem

(a) Beispiele für historische Rechenmaschinen (b) Von Babbages Analytical Engine zum Computer als Massenware (c) Miniaturisierung und Mooresches Gesetz (d) Programmiersprachen und Benutzeroberflächen erleichtern die Interaktion zwischen Mensch und Maschine		Material (z.B. „Eine Maschine verändert die Welt“ Teil 1 - Wie die Computer rechnen lernten und Teil 2 - Die Computer-Industrie entsteht) arbeitsteilig zu bestimmten Teilgebieten die Kernideen in der Entwicklung der Computer zusammen und recherchieren ggf. zusätzlich. In Expertengruppen (zu gleichen Teilgebieten) sammeln und ergänzen sie anschließend ihre Ergebnisse und erläutern sich dabei gegenseitig, inwiefern es sich dabei um Kernideen der Informatik handelt. In Basisgruppen erläutern die Schülerinnen und Schüler ihre unterschiedlichen Teilgebiete, erstellen einen gemeinsamen Zeitstrahl auf einem Plakat und bereiten eine Präsentation im Plenum vor.
3. Gesellschaftliche Auswirkungen (a) Informatische Systeme im Alltag (b) Veränderungen der Arbeitswelt, insbesondere im Ruhrgebiet (c) Erarbeitung grundlegender Begriffe des Datenschutzes in Anknüpfung an die Lebenswelt der Schülerinnen und Schüler (d) Diskussion und Bewertung von Fallbeispielen aus dem Themenbereich „Datenschutz“		*Beispiel:* Änderungen im Alltag und in der Arbeitswelt Die Schülerinnen und Schüler befragen Eltern und Verwandte in Bezug auf ihre heutigen Gewohnheiten im Umgang mit informatischen Systemen im Vergleich zu früher. Beispiele: Wie hat man früher kommuniziert? Was hat sich im Beruf geändert? Welche Kompetenzen werden zukünftig besonders gefragt sein? *Beispiel:* Änderungen im Alltag und in der Arbeitswelt Die Schülerinnen und Schüler bearbeiten Fallbeispiele aus ihrer eigenen Erfahrungswelt oder der aktuellen Medienberichterstattung. *Materialien:* Materialblatt zum Bundesdatenschutzgesetz (Einführungsphase UV VI.1)

Unterrichtsvorhaben EF-II

Thema: Wie funktioniert ein Computer?

Leitfragen: *Wie werden Informationen in einem Computer dargestellt? Wie verarbeitet dieser die Informationen? Wie kann die in der Schule vorhandene informatische Ausstattung genutzt werden?*

Vorhabenbezogene Konkretisierung:

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: 12 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Information, deren Kodierung und Speicherung (a) Information und Daten: Darstellung von Informationen in Schrift, Bild und Ton (b) Digitalisierung am Beispiel von Text und Bild: ASCII und RGB (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), – 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).	*Beispiel:* Textkodierung Kodierung und Dekodierung von Texten mit unbekannten Zeichensätzen (z.B. Wingdings) *Beispiel:* Bildkodierung Kodierung von Bildinformationen in Raster- und Vektorgrafiken
2. Zahlensysteme (a) Darstellen ganzer Zahlen im Dualsystem, Umrechnung vom und zum Dezimalsystem (b) Rechnen mit Binärzahlen (Addition) (c) Nibbles, Bytes und Words (e) Das Hexadezimalsystem zur einfacheren Darstellung von Binärzahlen (f) Interpretation eines Hexcodes als Klartextdatei (ASCII) und umgekehrt		*Beispiel:* Zahlenraten Eine Schülerin oder ein Schüler denkt sich eine Zahl von 0 bis 63 aus und verrät diese nicht. Anschließend werden ein paar Karten mit mehreren Zahlen darauf gezeigt. Die Schülerin bzw. der Schüler teilt nur kurz mit, ob die ausgedachte Zahl dabei ist oder nicht. Nach wenigen Karten kann man schnell sagen, was die gedachte Zahl war.

		Über eine Analyse des Spiels lässt sich gut das Binärsystem entwickeln. *Material:* Präsentation „Zahlenraten“
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-Prinzips (Eingabe-Verarbeitung-Ausgabe) als Prinzip der Verarbeitung von Daten und Grundlage der „Von-Neumann-Architektur“		*Material:* Demonstrations-hardware Durch Demontage eines Demonstrationsrechners entdecken Schülerinnen und Schüler die verschiedenen Hardwarekomponenten eines Informatiksystems. Als Demonstrationsrechner bietet sich ein ausrangierter Schulrechner an.
4. Arbeitsweise eines „Von-Neumann-Rechners“ mit einem Modellrechner (a) Vereinfachungen des verwendeten Modellrechners (b) Einfache Rechnungen mit dem Modellrechner (c) Sprungbefehle und Schleifen (d) Analyse einfacher, maschinennaher Programme		*Beispiel:* Einfache Rechnungen mit einem simulierten Modellrechner Mit einem einfachen Modellrechner werden einfache Rechnungen durchgeführt und maschinennahe Programme analysiert. *Material:* – JOHNNY-Modellrechner – Unterrichtsreihe zu JOHNNY – MOPS-Modellrechner

Unterrichtsvorhaben EF-III

Thema: Such- und Sortieralgorithmen anhand kontextbezogener Beispiele

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

Vorhabenbezogene Konkretisierung:

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

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

Daran anschließend werden die erarbeiteten Strategien systematisiert und im Pseudocode notiert oder als Programmablaufplan (PAP) visualisiert. Die Schülerinnen und Schüler sollen auf diese Weise das Sortieren durch Vertauschen, das Sortieren durch Auswählen und mindestens einen weiteren Sortieralgorithmus kennen lernen.

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

Zeitbedarf: 9 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterichtssequenzen	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Explorative Erarbeitung eines Sortierverfahrens (a) Sortierprobleme im Kontext informatischer Systeme und im Alltag (z.B. Dateisortierung, Tabellenkalkulation, Telefonbuch, Bundesligatabelle, usw.) (b) Vergleich zweier Elemente als Grundlage eines Sortieralgorithmus (c) Erarbeitung eines Sortieralgorithmus durch die Schülerinnen und Schüler	Die Schülerinnen und Schüler – beurteilen die Effizienz von Algorithmen am Beispiel von Sortierverfahren hinsichtlich Zeit und Speicherplatzbedarf (A), – entwerfen einen weiteren Algorithmus zum Sortieren (M), – analysieren Such- und Sortieralgorithmen und wenden sie auf Beispiele an (D).	*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. *Materialien:* Computer science unplugged – Sorting Algorithms, URL: www.csunplugged.org/searching-algorithms (abgerufen: 30.03.2014)
2. Systematisierung von Algorithmen und Effizienzbetrachtungen (a) Formulierung (falls selbst gefunden) oder Erläuterung von mehreren Algorithmen im Pseudocode (auf jeden Fall: Sortieren durch Vertauschen, Sortieren durch Auswählen) (b) Anwendung von Sortieralgorithmen auf verschiedene Beispiele (c) Bewertung von Algorithmen anhand der Anzahl der nötigen Vergleiche (d) Variante des Sortierens durch Auswählen (Nutzung eines einzigen oder zweier Felder bzw. lediglich eines einzigen zusätzlichen Ablageplatzes oder mehrerer neuer Ablageplätze) (e) Effizienzbetrachtungen an einem konkreten Beispiel bezüglich der Rechenzeit und des Speicherplatzbedarfs (f) Analyse des weiteren Sortieralgorithmus (sofern nicht		*Beispiel:* Sortieren durch Auswählen, Sortieren durch Vertauschen, Quicksort Quicksort ist als Beispiel für einen Algorithmus nach dem Prinzip Teile und Herrsche gut zu behandeln. Kenntnisse in rekursiver Programmierung sind nicht erforderlich, da eine Implementierung nicht angestrebt wird. Materialien: Computer science unplugged – Sorting Algorithms, URL: www.csunplugged.org/searching-algorithms (abgerufen: 30.03.2014)

in Sequenz 1 und 2 bereits geschehen)		
3. Binäre Suche auf sortierten Daten (a) Suchaufgaben im Alltag und im Kontext informatischer Systeme (b) Evtl. Simulationsspiel zum effizienten Suchen mit binärer Suche (c) Effizienzbetrachtungen zur binären Suche		*Beispiel:* Simulationsspiel zur binären Suche nach Tischtennisbällen Mehrere Tischtennisbälle sind nummeriert, sortiert und unter Bechern verdeckt. Mit Hilfe der binären Suche kann sehr schnell ein bestimmter Tischtennisball gefunden werden. *Materialien:* Computer science unplugged – Searching Algorithms, URL: www.csunplugged.org/searching-algorithms (abgerufen: 30.03.2014)

Unterrichtsvorhaben EF-IV

Thema: Grundlagen der objektorientierten Modellierung und Implementation einfacher Algorithmen am Beispiel eines Marsroboters in Greenfoot

Leitfragen: *Wie lassen sich Gegenstandsbereiche informatisch modellieren und in einem Greenfoot-Szenario informatisch realisieren? Wie lassen sich Aktionen von Objekten flexibel 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 wird die objektorientierte Analyse für das Greenfoot-Szenario Planetenerkundung durchgeführt. Die vom Szenario vorgegebenen Klassen werden von Schülerinnen und Schülern in Teilen analysiert und entsprechende Objekte anhand einfacher Problemstellungen erprobt. Die Ler-nenden implementieren und testen einfache Programme. Die Greenfoot-Umgebung ermöglicht es, Beziehungen zwischen Klassen zu einem späteren Zeitpunkt zu thematisieren. So kann der Fokus hier auf Grundlagen wie der Unterscheidung zwischen Klasse und Objekt, Attribute, Me-thoden, Objektidentität und Objektzustand gelegt werden.

Damit zunächst eine Fokussierung auf die Grundlagen der Objektorientierung möglich ist, ohne dass algorithmische Probleme ablenken, wird in den ersten drei Sequenzen dieses Unterrichtsvorhabens zunächst auf die Verwendung von Kontrollstrukturen verzichtet, so dass der Quellcode aus einer rein linearen Sequenz besteht.

Die Möglichkeiten, komplexere Probleme mit einer rein linearen Befehlssequenz zu lösen, sind natürlich begrenzt. Die Arbeit an diesen Projekten kann fließend zur vierten Sequenz führen, bei der Kontrollstrukturen im Mittelpunkt stehen, sobald die grundlegenden Konzepte der Objektorientierung sicher verinnerlicht wurden.

Das Ziel der vierten Sequenz besteht darin, das Verhalten von Objekten flexibel zu programmieren. Der Schwerpunkt liegt dabei auf der Erarbeitung von Kontrollstrukturen. Die Strukturen „bedingte Anweisung“ und „Wiederholung“ 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.

Der Schwerpunkt der letzten Sequenz dieses 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:

Unterichtssequenzen	**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). – analysieren und erläutern einfache Algorithmen und Programme (A),	*Beispiel:* Beispiele aus dem Alltag
2. Analyse von Objekten und Klassen im Greenfoot-Szenario (a) Schritte der objektorientierten Analyse, Modellierung und Implementation (b) Analyse und Erprobung der Objekte im Greenfoot-Szenario		*Beispiel:* 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 in Greenfoot (a) Quelltext einer Java-Klasse (b) Implementation eigener Methoden, Dokumentation mit JavaDoc		*Material:* Programmierung einfacher Aktionen in Greenfoot Das Szenario „Planetenerkundung“ wird gezielt um eigene Methoden zur Lösung einfacher Aufgaben ergänzt, das Programm

(c) Programme übersetzen (Aufgabe des Compilers) und testen	– entwerfen einfache Algorithmen und stellen sie umgangssprachlich und grafisch dar (M), – ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen zu (M), – implementieren Algorithmen unter Verwendung von Variablen und Wertzuweisungen, Kontrollstrukturen sowie Methodenaufrufen (I), – testen Programme schrittweise anhand von Beispielen (I), – interpretieren Fehlermeldungen und korrigieren den Quellcode (I).	übersetzt und schrittweise getestet.
4. Algorithmische Grundstrukturen in Java (a) Bedingte Anweisungen (if) (b) Verknüpfungen von Bedingungen durch die logischen Funktionen UND, ODER und NICHT (c) Wiederholungen (Schleifen mit while und for) (d) Systematisierung des Vorgehens zur Entwicklung von Algorithmen zur Lösung komplexerer Probleme		*Beispiel:* Programmierung komplexerer Aktionen in Greenfoot Für das Szenario „Planetenerkundung“ werden gezielt Lösungen implementiert, so dass vorgegebene, einfache Probleme gelöst werden, die bedingte Anweisungen und Schleifen benötigen.
5. 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		*Beispiel:* Erweiterung des Marsrovers um weitere Eigenschaften und Fähigkeiten Für das Szenario „Planetenerkundung“ werden die Eigenschaften und Fähigkeiten des Marsrovers erweitert, z.B. um eine zusätzliche Energieverwaltung.

Unterrichtsvorhaben EF-V

Thema: Modellierung und Implementierung von Klassen- und Objektbeziehungen anhand von grafischen Spielen und Simulationen

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 Schwer-punkt. 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.

Zeitbedarf: 18 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterichtssequenzen	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 Programmier-sprache auch unter Nutzung dokumentierter Klassenbibliotheken (I), – testen Programme schrittweise anhand von Beispielen (I),	*Beispiel:* Kapitel 6.1 bis 6.3
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) Erstellung von Sequenzdiagrammen als Vorbereitung für die Programmierung		*Beispiel:* Kapitel 6.4 bis 6.5
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		*Material:* Kapitel 6.4 bis 6.5
4. Vererbungsbeziehungen (a) Das Grundprinzip der Vererbung wird erarbeitet (b) Die Vorteile der Vererbungsbeziehungen		*Beispiel:* Kapitel 6.5

(c) Vererbung wird implementiert	– 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 Vererbungsbeziehungen in Diagrammen grafisch dar (D), – dokumentieren Klassen durch Beschreibung der Funktionalität der Methoden (D).	
5. Softwareprojekt (a) Analyse und Dekonstruktion eines Spiels (Modelle, Quelltexte) (b) Erweiterung des Spiels um weitere Funktionalitäten (c) Modellierung eines Spiels aufgrund einer Anforderungsbeschreibung		*Beispiel:* Space Shooter Ein einfacher Weltraum-Shooter wird anhand eines Leitprogrammes erstellt und um eigene Funktionalität erweitert.

Unterrichtsvorhaben EF-VI

Thema: Entwicklung eines eigenen Softwareprojekts – Planung, Durchführung, Dokumentation und Präsentation

Leitfragen: *Wie plane ich ein eigenes Softwareprojekt von der Idee bis zur fertigen Umsetzung? Wie wird ein Projekt dokumentiert und präsentiert?*

Vorhabenbezogene Konkretisierung:

In diesem Unterrichtsvorhaben entwickeln die Schülerinnen und Schüler vorzugsweise in Partnerarbeit ein eigenes Programmierprojekt - z.B. ein Spiel. Ziel ist es zunächst, eigene Ideen für ein Programmierprojekt mit angemessenem Schwierigkeitsgrad zu finden und mit der Lehrkraft zu vereinbaren.

Anschließend wird der Arbeitsprozess geplant (Analyse und Dekonstruktion). Ziel ist ein Entwurfsdiagramm aller benötigten Klassen, das u.a. auch die Beziehungen zwischen den einzelnen Klassen wiedergibt.

Bei der anschließenden Implementierung des Projektes soll ggf. möglichst eigenständig recherchiert werden, der Lehrer steht jedoch beratend zur Verfügung. Der Arbeitsprozess sowie verwendete Quellen (z.B. Grafiken, Literatur) sollen dabei angemessen dokumentiert werden.

In einer abschließenden Präsentation stellen die Schülerinnen und Schüler ihr Projekt vor. Dabei steht die Erläuterung der wesentlichen eigenen Ideen und Leistung bei der Umsetzung des Projektes im Vordergrund.

Zeitbedarf: 18 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Zielsetzung (a) Ideenfindung für ein eigenes Projekt und Einschätzung der Schwierigkeit (b) Erarbeitung eines Pflichtenheftes und Absegnung durch die Lehrkraft	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),	*Beispiel:* Beispiele für verschiedene Projekte findet man im Informatik-Wiki des HHG, URL: www.hhg-informatik.de
2. Modellierung und Planung des Arbeitsprozesses (a) Analyse und Dekonstruktion (b) Erstellung eines Entwurfsdiagrammes		
3. Implementierung des Projektes (a) Implementierung in Java (b) Backups an entscheidenden Stellen, Versionierung (c) Dokumentation der Arbeit		
4. Präsentation des Projektes (a) Erstellen eines Projektpräsentation (b) Halten eines angemessenen Vortrages (c) Gegenseitige Beurteilung der Projektidee, der Umsetzung und der Präsentation		

	– modifizieren einfache Algorithmen und Programme (I), – entwerfen einfache Algorithmen und stellen sie umgangssprachlich und grafisch dar (M), – stellen Klassen, Assoziations- und Vererbungsbeziehungen in Diagrammen grafisch dar (D), – dokumentieren Klassen durch Beschreibung der Funktionalität der Methoden (D).	

Unterrichtsvorhaben in der Qualifikationsphase
In der Qualifikationsphase werden die Unterrichtsvorhaben unter Berücksichtigung derVorgaben für das Zentralabitur Informatik in NRW konkretisiert.

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

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 anhand einer kontextbezogenen Problemstellung

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?*

Vorhabenbezogene Konkretisierung:

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

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

Zeitbedarf: 8 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterichtssequenzen	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	*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. oder *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

	Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M), – implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I), – nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), – wenden eine didaktisch orientierte Entwicklungsumgebung zur Demonstration, zum Entwurf, zur Implementierung und zum Test von Informatiksystemen an (I), – interpretieren Fehlermeldungen und korrigieren den Quellcode (I), – stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D), – dokumentieren Klassen (D), – stellen die Kommunikation zwischen Objekten grafisch dar (D).	Form von Würfeln und Zuckerringe in Form von Toren.

Unterrichtsvorhaben Q1-II

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

Leitfragen: *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:

Unterichtssequenzen	**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 (c) Modellierung und Implementierung der Anwendung unter Verwendung eines oder mehrerer Objekte der Klasse Queue	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), – 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	*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 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.
2. Die Datenstruktur Stapel im Anwendungskontext unter Nutzung der Klasse Stack (a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen		*Beispiel:* Heftstapel In einem Heftstapel soll das Heft einer Schülerin gefunden werden. oder *Beispiel:* Kisten stapeln

(b) Erarbeitung der Funktionalität der Klasse Stack (c) Modellierung und Implementierung der Anwendung unter Verwendung eines oder mehrerer Objekte der Klasse Stack	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).	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.		*Material:* 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.
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.

Unterrichtsvorhaben Q1-III

Thema: Suchen und Sortieren auf linearen Datenstrukturen

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

Vorhabenbezogene Konkretisierung:

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

Die verschiedenen Verfahren werden hinsichtlich Speicherbedarf und Zahl der Vergleichsoperationen miteinander verglichen.

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

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

Zeitbedarf: 16 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterichtssequenzen	**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), – 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	*Beispiel:* Karteiverwaltung Für ein Adressverwaltungsprogramm soll eine Methode zum Suchen einer Adresse geschrieben werden. oder *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 können.
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		*Beispiel:* Karteiverwaltung (s.o.) oder *Beispiel:* Bundesjugendspiele (s.o.)

(c) Entwicklung eines rekursiven Sortierverfahren für ein Feld (z.B. Sortieren durch Mischen)	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).	
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		*Beispiel:* Karteiverwaltung (s.o.) oder *Beispiel:* Bundesjugendspiele (s.o.)

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:

Unterichtssequenzen	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	*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 http://dokumentation.videocenter.schule.de/old/video/index.html (abgerufen: 30. 03. 2014) findet man den Link zu dem VideoCenter-System sowie nähere Informationen. Lesenswert ist auch die dort verlinkte „Dokumentation der Fallstudie“ mit didaktischem Material, welches alternativ bzw. ergänzend zu der im Folgenden beschriebenen Durchführung verwendet werden kann. *Beispiel*: Schulbuchausleihe Unter http://www.brd.nrw.de/lerntreffs/informatik/structure/material/sek2/datenbanken.php (abgerufen: 30. 03. 2014) wird eine Datenbank zur Verfügung gestellt, die Daten einer Schulbuch-Ausleihe enthält (über 1000 Entleiher, 200 Bücher mit mehreren tausend Exemplaren und viele Ausleihvorgänge). Die Datenbank kann in OpenOffice eingebunden werden.
2. Modellierung von relationalen Datenbanken (a) Entity-Relationship-Diagramm		*Beispiel*: Fahrradverleih Der Fahrradverleih BTR (BikesToRent) verleiht unterschiedliche Typen von

– Ermittlung von Entitäten, zugehörigen Attributen, Relationen und Kardina¬litäten in Anwendungssituationen und Modellierung eines Datenbankentwurfs in Form eines Entity-Relationship-Diagramms – Erläuterung und Modifizierung einer Datenbankmodellierung (b) Entwicklung einer Datenbank aus einem Datenbankentwurf – Modellierung eines relationalen Datenbankschemas zu einem Entity-Relationship-Diagramm inklusive der Bestimmung von Primär- und Sekundärschlüsseln (c) Redundanz, Konsistenz und Normalformen – Untersuchung einer Datenbank hinsichtlich Konsistenz und Redundanz in einer Anwendungssituation – Überprüfung von Datenbankschemata hinsichtlich der 1. bis 3. Normalform und Normalisierung (um Redundanzen zu vermeiden und Konsistenz zu gewährleisten)	Normalisierungseigenschaften (D).	Fahrrädern diverser Firmen an seine Kunden. Die Kunden sind bei BTR registriert (Name, Adresse, Telefon). BTR kennt von den Fahrradfirmen den Namen und die Telefonnummer. Kunden von BTR können CityBikes, Treckingräder und Mountainbikes ausleihen. *Beispiel*: Reederei Die Datenverwaltung einer Reederei soll in einem Datenbanksystem umgesetzt werden. Ausgehend von der Modellierung soll mit Hilfe eines ER-Modells und eines Datenbankschemas dieser erste Entwurf normalisiert und in einem Datenbanksystem umgesetzt werden. Es schließen sich diverse SQL-Abfragen an, wobei auf die Relationenalgebra eingegangen wird. *Beispiel*: Buchungssystem In dem Online-Buchungssystem einer Schule können die Lehrer Medienräume, Beamer, Laptops, Kameras, usw. für einen bestimmten Zeitpunkt buchen, der durch Datum und die Schulstunde festgelegt ist. Dazu ist die Datenbank zu modellieren, ggf. zu normalisieren und im Datenbanksystem umzusetzen. Weiter sollen sinnvolle Abfragen entwickelt werden. Unter http://mrbs.sourceforge.net (abgerufen: 30.03. 2014) findet man ein freies Online-Buchungssystem inklusive Demo, an Hand derer man erläutern kann, worum es in dem Projekt geht. *Beispiel*: Schulverwaltung 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:

Unterichtssequenzen	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-	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	*Beispiel:* Verschlüsselung, Zugriff auf Daten in Netzwerken

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	Verschlüsselungsverfahren (A), – untersuchen und bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen, die Sicherheit von Informatiksystemen sowie die Einhaltung der Datenschutzbestimmungen und des Urheberrechts (A), – untersuchen und bewerten Problemlagen, die sich aus dem Einsatz von Informatiksystemen ergeben, hinsichtlich rechtlicher Vorgaben, ethischer Aspekte und gesellschaftlicher Werte unter Berücksichtigung unterschiedlicher Interessenlagen (A), – nutzen bereitgestellte Informatiksysteme und das Internet reflektiert zum Erschließen, zur Aufbereitung und Präsentation fachlicher Inhalte (D).	
2. Fallbeispiele zur Datenschutzproblematik und zum Urheberrecht		*Beispiel*: 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 à 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:

Unterichtssequenzen	**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 ab¬strakte Klassen unter	*Beispiel:* Termbaum Der Aufbau von Termen wird mit Hilfe von binären Baumstrukturen verdeutlicht. oder *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.) oder *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

	Verwendung von Vererbung durch Spezialisieren und Generalisieren (M), – verwenden bei der Modellierung geeigneter Problemstellungen die Möglichkeiten der Polymorphie (M), – entwickeln iterative und rekursive Algorithmen unter Nutzung der Konstruktionsstrategien „Modularisierung“ und „Teilen und Herrschen“ (M), – implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I), – modifizieren Algorithmen und Programme (I), – nutzen die Syntax und Semantik einer 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).	sich im linken Teilbaum, die Fragen, die möglich sind, wenn die Antwort „nein“ lautet, stehen im rechten Teilbaum. oder *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.
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 (d) Implementierung der Anwendung oder von Teilen der Anwendung (e) Traversierung eines Binärbaums im Pre-, In- und Postorderdurchlauf		*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: – Einfügen der Informatiker-Daten in den Baum – Suchen nach einem Informatiker über den Schlüssel Name – Ausgabe des kompletten Datenbestands in nach Namen sortierter Reihenfolge
3. Die Datenstruktur binärer Suchbaum im Anwendungskontext unter Verwendung der Klasse BinarySearchTree		*Beispiel:* Informatikerbaum als Suchbaum In einem binären Suchbaum werden die Namen und die Geburtsdaten von Informatikern lexikographisch geordnet

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

Unterrichtsvorhaben 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:

Unterichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, 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
2. Untersuchung und Entwicklung von Grammatiken regulärer Sprachen (a) Erarbeitung der formalen Darstellung regulärer Grammatiken		*Beispiele:* – reguläre Grammatik für Wörter mit ungerader Parität, – Grammatik für Wörter, die bestimmte Zahlen repräsentieren, – Satzgliederungs-Grammatik

(b) Untersuchung, Modifikation und Entwicklung von Grammatiken (c) Entwicklung von endlichen Automaten zum Erkennen regulärer Sprachen die durch Grammatiken gegeben werden (d) Entwicklung regulärer Grammatiken zu endlichen Automaten	– entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M), – entwickeln zur akzeptierten Sprache eines Automaten die zugehörige Grammatik (M), – entwickeln zur Grammatik einer regulären Sprache einen zugehörigen endlichen Automaten (M), – modifizieren Grammatiken regulärer Sprachen (M), – entwickeln zu einer regulären Sprache eine Grammatik, die die Sprache erzeugt (M), – 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).	
3. Grenzen endlicher Automaten		*Beispiele:* – Klammerausdrücke, – anbn 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:

Unterichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, 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
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

3.2 Grundsätze der fachmethodischen und fachdidaktischen Arbeit

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

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.
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.

3.3 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 Konrad-Zuse-Gymnasiums 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.

3.3.1 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.

Instrumente

- Einführungsphase: 1 Klausur je Halbjahr, Dauer der Klausur: 2 Unterrichtsstunden
- Grundkurse Q1: 2 Klausuren je Halbjahr, Dauer der Klausuren: 2 Unterrichtsstunden
- Grundkurse Q2.1: 2 Klausuren, Dauer der Klausuren: 3 Unterrichtsstunden
- Grundkurse Q2.2: 1 Klausur unter Abiturbedingungen
- Anstelle einer Klausur kann gemäß dem Beschluss der Lehrerkonferenz im zweiten Halbjahr der Jahrgangsstufe 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.

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.

3.3.2 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
Mündliche Leistungen

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

Praktische Leistungen am Computer

- Implementierung, Test und Anwendung von Informatiksystemen

Sonstige schriftliche Leistungen

- Arbeitsmappe und Arbeitstagebuch zu einem durchgeführten Unterrichtsvorhaben
- Lernerfolgsüberprüfung durch kurze schriftliche Übungen in Kursen, in denen höchstens 50% der Kursmitglieder eine Klausur schreiben, finden schriftliche Übungen mindestens einmal pro Kurshalbjahr statt, in anderen Kursen entscheidet über die Durchführung die Lehrkraft.Schriftliche Übungen 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.

4. Entscheidungen zu fach- und unterrichtsübergreifenden Fragen

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

4.1 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. Da im Inhaltsfeld Informatik, Mensch und Gesellschaft auch gesellschaftliche und ethische Fragen im Unterricht angesprochen werden, soll eine mögliche Zusammenarbeit mit den Fächern Sozialwissenschaften und Philosophie in einer gemeinsamen Fachkonferenz ausgelotet werden.

4.2 Projekttage

Sofern am Heinrich Heine Gymnasium im jeweils laufenden Schuljahr Projekttage angeboten werden, versucht die Fachkonferenz Informatik in diesem Zusammenhang mindestens ein Projekt für Schülerinnen und Schüler der gymnasialen Oberstufe anzubieten.

4.3 Vorbereitung auf die Erstellung der Facharbeit

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

4.4 Exkursionen

In der Einführungsphase bietet sich im Rahmen des Unterrichtsvorhabens „Was ist Informatik? Informatik damals, heute und morgen…“ eine Exkursion zum Heinz Nixdorf Museums Forum an. Diese außerunterrichtliche Veranstaltung wird in diesem Fall im Unterricht vor- und nachbereitet.

In der Qualifikationsphase kann im Rahmen einer Exkursion zur Hochschule Ruhr West der Studiengang Informatik an der HRW und mögliche spätere Berufsfelder vorgestellt werden.

4.5 Spitzenförderung durch Wettbewerbe und Schülerstudium

Als Beitrag zur individuellen Förderung bietet die Fachschaft Informatik für interessierte Schülerinnen und Schüler die Teilnahme an verschiedenen Wettbewerben wie dem

Informatik-Biber-Wettbewerb, dem Bundeswettbewerb Informatik, der Robocom oder Invent a Chip und die beratende Begleitung bei diesen Wettbewerben an.

Schülerinnen und Schüler, die eine besondere Begabung und ein besonderes Interesse am Studium der Informatik haben, haben in Absprache mit der Lehrkraft und der HRW die Möglichkeit, sich für ein Jungstudium an der Hochschule Ruhr West einzuschreiben.

5. 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 (Abschnitt 3.1) ist zunächst bis 2017 für den ersten Durchgang durch die gymnasiale Oberstufe nach Erlass des Kernlehrplanes verbindlich. Erstmalig nach Ende der Einführungsphase im Sommer 2015, werden in einer Sitzung der Fachkonferenz Erfahrungen ausgetauscht und ggf. Änderungen für den nächsten Durchgang der Einführungsphase beschlossen, um erkannten ungünstigen Entscheidungen schnellstmöglich entgegenwirken zu können.

Nach Abschluss des Abiturs 2017 wir die Fachkonferenz Informatik auf der Grundlage ihrer Unterrichtserfahrungen eine Gesamtsicht des schulinternen Curriculums vornehmen und ggf. eine Beschlussvorlage für die erste Fachkonferenz des folgenden Schuljahres erstellen.