Schulinternen Lehrplan zum Kernlehrplan für die gymnasiale Oberstufe

Informatik

Version: v8

Seite

Inhaltsverzeichnis

Kapitel 1: Die Fachgruppe Informatik des Alexander-von-Humboldt-Gymnasiums Neuss.....3

Kapitel 2: Entscheidungen zum Unterricht.....4

Kapitel 2.1: Unterrichtsvorhaben.....4
Kapitel 2.2: Übersichtsraster Unterrichtsvorhaben.....5
Kapitel 2.2.1: Konkretisierte Unterrichtsvorhaben.....12

Kapitel 3: Grundsätze der fachmethodischen und fachdidaktischen Arbeit.....57

Kapitel 4: Grundsätze der Leistungsbewertung und Leistungsrückmeldung.....58

Kapitel 4.1: Beurteilungsbereich Klausuren.....59
Kapitel 4.2: Beurteilungsbereich Sonstige Mitarbeit.....60

Kapitel 5: Lehr- und Lernmittel.....63

Kapitel 6: Entscheidungen zu fach- und -unterrichtsübergreifenden Fragen.....64

Kapitel 7: Qualitätssicherung und Evaluation.....64

Kapitel 1: Die Fachgruppe Informatik des Alexander-von-Humboldt-Gymnasiums Neuss

Beim Alexander-von-Humboldt-Gymnasium handelt es sich um eine vierzügige Schule im Zentrum von Neuss mit zurzeit ca. 1000 Schülerinnen und Schülern. Im Bereich der Sekundarstufe II kooperiert das Alexander-von-Humboldt-Gymnasium mit seiner Nachbarschule, dem Nelly-Sachs-Gymnasiums und bietet mit ihnen zahlreiche gemeinsame Kurse an, unter anderem in Informatik.
Das Fach Informatik wird am Alexander-von-Humboldt-Gymnasium ab der Jahrgangsstufe 8 im Wahlpflichtbereich II (WP II) zweistündig unterrichtet.

Organisatorisch ist das Fach Informatik in der Sekundarstufe I in den MINT-Zweig der Schule eingebunden, den Schülerinnen und Schüler als Alternative zu einem bilingualen Zweig anwählen können.

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

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

Der Unterricht der Sekundarstufe II wird mit Hilfe der Programmiersprache Java durchgeführt. In der Einführungsphase kommt dabei zusätzlich eine didaktische Lernumgebung zum einsatz.

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

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

Zurzeit besteht die Fachschaft Informatik des Alexander-von-Humboldt-Gymnasiums aus fünf Lehrkräften, denen zwei Computerräume mit 17 bzw. 16 Computerarbeitsplätzen, zwei Laptopwagen und zwei Ipadkoffer zur Verfügung stehen. Alle Arbeitsplätze sind an das schulinterne Rechnernetz angeschlossen, so dass Schülerinnen und Schüler über einen individuell gestaltbaren Zugang zum zentralen Server der Schule haben. SchülerInnen haben Zugriff auf ihre eigenen Daten, sie können im Internet recherchieren und ihre schulischer Aufgaben bearbeiten.

Der Unterricht erfolgt im 45 bzw. 90-Minuten-Takt. Die Kursblockung sieht grundsätzlich für Grundkurse eine Doppelstunde und je nach Anzahl der Schüler im Grundkurs zusätzlich eine Einzelstunde vor.

Kapitel 2: Entscheidungen zum Unterricht

Kapitel 2.1: Unterrichtsvorhaben

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

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

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

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

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

Kapitel 2.2: Übersichtsraster Unterrichtsvorhaben

I) Einführungsphase

<table>
<tr><th colspan="2">Einführungsphase</th></tr>
<tr>
<td>

Unterrichtsvorhaben E-I

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

Zentrale Kompetenzen:
- Werkzeuge beherrschen
- Argumentieren
- Darstellen und Interpretieren
- Kommunizieren und Kooperieren

Inhaltsfelder:
- Informatiksysteme
- Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:
- Einzelrechner
- Umgang mit ausgewählter Programmierumgebung
- Dateisystem
- Internet
- Einsatz von Informatiksystemen

Zeitbedarf: 6 Stunden

</td>
<td>

Unterrichtsvorhaben E-II

Thema:
Grundlagen der objektorientierten Analyse, Modellierung und Implementierung anhand von statischen Grafikszenen

Zentrale Kompetenzen:
- Werkzeuge beherrschen
- Modellieren
- Implementieren
- Darstellen und Interpretieren
- Kommunizieren und Kooperieren

Inhaltsfelder:
- Daten und ihre Strukturierung

Inhaltliche Schwerpunkte:
- Umgang mit ausgewählter Programmierumgebung
- Objekte und Klassen
- Syntax und Semantik einer Programmiersprache

Zeitbedarf: 14 Stunden

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

Einführungsphase	
Unterrichtsvorhaben E-III **Thema**: *Grundlagen der objektorientierten Programmierung und algorithmischer Grundstrukturen in Java* **Zentrale Kompetenzen:** • Argumentieren • Modellieren • Implementieren • Kommunizieren und Kooperieren **Inhaltsfelder:** • Daten und ihre Strukturierung • Algorithmen • Formale Sprachen und Automaten **Inhaltliche Schwerpunkte:** • Objekte und Klassen • Syntax und Semantik einer Programmiersprache • Analyse, Entwurf und Implementierung einfacher Algorithmen **Zeitbedarf**: 20 Stunden	*Unterrichtsvorhaben E-IV* **Thema**: *Modellierung und Implementierung von Klassen- und Objektbeziehungen.* **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**: 20 Stunden

Summe Einführungsphase: 60

II) Qualifikationsphase (Q1 und Q2) - GRUNDKURS

<table>
<tr><th colspan="2">Qualifikationsphase 1</th></tr>
<tr>
<td>

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: 10 Stunden

</td>
<td>

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

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

Qualifikationsphase 1	
Unterrichtsvorhaben Q1-III **Thema:** *Grundsätzlicher Überblick zu Such- und Sortieralgorithmen auf linearen Datenstrukturen (Array, lineare Liste, …* **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**: 12 Stunden	*Unterrichtsvorhaben Q1-IV* **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**: 12 Stunden

Qualifikationsphase 1	
Unterrichtsvorhaben Q1-V **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:** 10 Stunden	
Summe Qualifikationsphase 1: 64 Stunden	

Qualifikationsphase 2	
Unterrichtsvorhaben Q2-I **Thema:** *Modellierung und Implementierung von Anwendungen mit dynamischen, linearen und 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**: 30 Stunden	*Unterrichtsvorhaben Q2-II* **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: 8 Stunden**

Qualifikationsphase 2	
Unterrichtsvorhaben Q2-III **Thema:** *Modellierung und Nutzung von relationalen Datenbanken in Anwendungskontexten* **Zentrale Kompetenzen:** • Argumentieren • Modellieren • Implementieren • Darstellen und Interpretieren • Kommunizieren und Kooperieren **Inhaltsfelder:** • Daten und ihre Strukturierung • Algorithmen • Formale Sprachen und Automaten • Informatik, Mensch und Gesellschaft **Inhaltliche Schwerpunkte:** • Datenbanken • Algorithmen in ausgewählten informatischen Kontexten • Syntax und Semantik einer Programmiersprache • Sicherheit ***Zeitbedarf**: 14 Stunden*	
Summe Qualifikationsphase 2: 52 Stunden	

Kapitel 2.2.1: Konkretisierte Unterrichtsvorhaben

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

Die Schülerinnen und Schüler

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

Unterrichtsvorhaben EF-I

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

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

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

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

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

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

Zeitbedarf: 6 Stunden

Sequenzierung des Unterrichtsvorhabens:

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

c) Grundlagen der technischen Umsetzung von Rechnerkommunikation am Beispiel des Internets (z.B. Netzwerkadresse, Paketvermittlung, Protokoll) *d)* Richtlinien zum verantwortungsvollen Umgang mit dem Internet		wie ein Paket weiter vermittelt wird.
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*: Demonstrationshardware Durch Demontage eines Demonstrationsrechners entdecken Schülerinnen und Schüler die verschiedenen Hardwarekompo-nenten eines Informatiksystems. Als Demonstrationsrechner bietet sich ein ausrangierter Schulrechner an.

Unterrichtsvorhaben EF-II

Thema: Grundlagen der objektorientierten Analyse, Modellierung und Implementierung anhand von statischen Grafikszenen

Leitfrage: *Wie lassen sich Gegenstandsbereiche informatisch modellieren und im Sinne einer Simulation informatisch realisieren?*

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

Dazu werden zunächst konkrete Gegenstandsbereiche aus der Lebenswelt der Schülerinnen und Schüler analysiert und im Sinne des Objektorientierten Paradigmas strukturiert. Dabei werden die grundlegenden Begriffe der Objektorientierung und Modellierungswerkzeuge wie Objektkarten, Klassenkarten oder Beziehungsdiagramme eingeführt.

Im Anschluss wird mit der Realisierung erster Projekte mit Hilfe der Programmierumgebung Eclipse begonnen. Hier muss der grundlegende Aufbau einer Java-Klasse thematisiert und zwischen Deklaration, Initialisierung und Methodenaufrufen unterschieden werden.

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

Zeitbedarf: 14 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. **Identifikation *von Objekten*** *a)* Am Beispiel eines lebensweltnahen Beispiels werden Objekte im Sinne der Objektorientierten Modellierung eingeführt. *b)* Objekte werden mit Objektkarten visualisiert und mit sinnvollen Attributen und „Fähigkeiten“, d.h. Methoden versehen. *c)* Manche Objekte sind prinzipiell typgleich und werden so zu einer Objektsorte bzw. Objektklasse zusammengefasst. *d)* Vertiefung: Modellierung weiterer Beispiele ähnlichen Musters *e)*	Die Schülerinnen und Schüler • 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), • stellen die Kommunikation zwischen Objekten grafisch dar (M), • implementieren einfache Algorithmen unter Beachtung der Syntax und Semantik einer Programmiersprache (I), • stellen den Zustand eines Objekts dar (D).	*Beispiel:* Vogelschwarm Schülerinnen und Schüler betrachten einen Vogelschwarm als Menge gleichartiger Objekte, die in einer Klasse mit Attributen und Methoden zusammengefasst werden können.
2. **Implementierung dreidimensionaler, statischer Szenen**		*Beispiel:* Skulpturengarten Schülerinnen und Schüler erstellen ein

a) Grundaufbau *einer Java-Klasse* *b) Konzeption einer Szene mit* Kamera*, Licht und sichtbaren Objekten* *c)* Deklaration *und Initialisierung von Objekten* *d) Methodenaufrufe mit* Parameterübergabe *zur Manipulation von Objekteigenschaften (z.B. Farbe, Position, Drehung)*		Programm, das mit Hilfe von geometrischen Objekten einen Skulpturengarten auf den Bildschirm bringt. *Beispiel*: Olympische Ringe Die Schülerinnen und Schüler bilden das Emblem der olympischen Spiele nach.

Unterrichtsvorhaben EF-III

Thema: Grundlagen der objektorientierten Programmierung und algorithmischer Grundstrukturen in Java.

Leitfragen: *Wie lassen sich Animationen und Simulationen optischer Gegenstandsbereiche unter Berücksichtigung von Tastatureingaben realisieren?*

Vorhabenbezogene Konkretisierung:
Der Schwerpunkt dieses Unterrichtsvorhabens liegt auf der Entwicklung mehrerer Projekte, die durch Eingaben des Benutzers gesteuerte Animationen aufweisen. Zunächst wird ein Projekt bearbeitet, das in Anlehnung an das vorangegangene Unterrichtsvorhaben eine Szene darstellt, die lediglich aus Objekten besteht, zu denen das didaktische System Klassen vorgibt. Einzelne Objekte der Szene werden animiert, um ein einfaches Spiel zu realisieren oder die Szene optisch aufzuwerten. Für die Umsetzung dieses Projekts werden Kontrollstrukturen in Form von Schleifen und Verzweigungen benötigt und eingeführt.

Sind an einem solchen Beispiel im Schwerpunkt Schleifen und Verzweigungen eingeführt worden, sollen diese Konzepte an weiteren Beispielprojekten eingeübt werden. Dabei muss es sich nicht zwangsläufig um solche handeln, bei denen Kontrollstrukturen lediglich zur Animation verwendet werden. Auch die Erzeugung größerer Mengen grafischer Objekte und deren Verwaltung in einem Feld kann ein Anlass zur Verwendung von Kontrollstrukturen sein.

Das Unterrichtsvorhaben schließt mit einem Projekt, das komplexere grafische Elemente beinhaltet, so dass die Schülerinnen und Schüler mehr als nur die Klasse erstellen müssen, welche die Szene als Ganzes darstellt. Elemente der Szene müssen zu sinnhaften eigenen Klassen zusammengefasst werden, die dann ihre eigenen Attribute und Dienste besitzen. Auch dieses Projekt soll eine Animation, ggf. im Sinne einer Simulation, sein, bei der Attributwerte von Objekten eigener Klassen verändert werden und diese Veränderungen optisch sichtbar gemacht werden.

Komplexere Assoziationsbeziehungen zwischen Klassen werden in diesem Unterrichtsvorhaben zunächst nicht behandelt. Sie stellen den Schwerpunkt des folgenden Vorhabens dar.

Zeitbedarf: 20 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Bewegungsanimationen am Beispiel einfacher grafischer Objekte *a) Kontinuierliche Verschiebung eines Objekts mit Hilfe einer Schleife (While-Schleife)* *b) Tastaturabfrage zur Realisierung einer* Schleifenbedingung *für eine Animationsschleife* *c) Mehrstufige Animationen mit* mehreren *sequenziellen Schleifen* *d) Berechnung von Abständen* zwischen *Objekten mit Hilfsvariablen* *e)* Meldungen zur Kollision zweier Objekte mit Hilfe von Abstandsberechnungen und Verzweigungen (IF-Anweisungen*)*	Die Schülerinnen und Schüler • analysieren und erläutern einfache Algorithmen und Programme (A), • entwerfen einfache Algorithmen und stellen sie umgangssprachlich und 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), • modifizieren einfache Algorithmen und Programme (I), • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I),	*Beispiel*: Wurfspiel Die Schülerinnen und Schüler realisieren mit Objekten ein Spiel, bei dem ein Ball über den Bildschirm bewegt und auf eine runde Zielscheibe geworfen werden soll.
2. Erstellen und Verwalten größerer Mengen einfacher grafischer Objekte *a) Erzeugung von Objekten mit Hilfe von Zählschleifen (FOR-Schleife)*		*Beispiel:* Hubschrauberlandeplatz Die Schülerinnen und Schüler realisieren einen runden Hubschrauberlandeplatz und eine Reihe von Landemarkierungen, die in einem Feld verwaltet werden. Mit Hilfe der Landemarkierungen werden verschiedene

b) Verwaltung von Objekten in eindimensionalen Feldern (Arrays) *c) Animation von Objekten, die in eindimensionalen Feldern (Arrays) verwaltet werden* *d) Vertiefung: Verschiedene* Feldbeispiele	• implementieren Algorithmen unter Verwendung von Variablen und Wertzuweisungen, Kontrollstrukturen sowie Methodenaufrufen (I), • implementieren einfache Algorithmen unter Beachtung der Syntax und Semantik einer Programmiersprache (I), • testen Programme schrittweise anhand von Beispielen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I).	Lauflichter realisiert. *Beispiel*: Schachbrett Die Schülerinnen und Schüler realisieren mit Hilfe mehrerer Quader ein Schachbrett. *Beispiel*: Magischer Würfel Die Schülerinnen und Schüler erstellen einen großen Würfel, der aus mehreren kleineren, farbigen Würfeln besteht.
3. Modellierung und Animation komplexerer grafisch repräsentierbarer Objekte *a) Modellierung eines Simulationsprogramms mit eigenen Klassen, die sich selbst mit Hilfe von einfachen Objekten zeigen mit Hilfe eines* Implementationsdiagramms *b) Implementierung eigener* Methoden *mit und ohne Parameterübergabe* *c)* Realisierung *von Zustandsvariablen* *d) Thematisierung des Geheimnisprinzips und des* Autonomitätsprinzips *von Objekten*		*Beispiel: Kerzensimulation* Die Schülerinnen und Schüler modellieren und erstellen eine Klasse, mit deren Hilfe Kerzen simuliert werden können. Eine Kerze kann angezündet und gelöscht werden. Abgesehen davon brennen Kerzen abhängig von ihrer Dicke unterschiedlich schnell ab. *Beispiel*: Uhren Die Schülerinnen und Schüler erstellen eine Simulation mehrerer Uhren für verschiedene Zeitzonen. *Beispiel:* Ampeln Die Schülerinnen und Schüler erstellen eine

e) Animation mit Hilfe des Aufrufs von selbstimplementierten *Methoden* *f) Vertiefung: Weitere Projekte*		Ampelkreuzung mit mehreren Ampelanlagen an einem Bahnübergang.

Unterrichtsvorhaben EF-IV

Thema: Modellierung und Implementierung von Klassen- und Objektbeziehungen.

Leitfrage: *Wie lassen sich komplexere Datenflüsse und Beziehungen zwischen Objekten und Klassen realisieren?*

Vorhabenbezogene Konkretisierung:
Dieses Unterrichtsvorhaben beschäftigt sich im Schwerpunkt mit dem Aufbau komplexerer Objektbeziehungen. Während in vorangegangenen Unterrichtsvorhaben Objekte nur jeweils solchen Objekten Nachrichten schicken konnten, die sie selbst erstellt haben, soll in diesem Unterrichtsvorhaben diese hierarchische Struktur aufgebrochen werden.

Dazu bedarf es zunächst einer präzisen Unterscheidung zwischen Objektreferenzen und Objekten, so dass klar wird, dass Dienste eines Objektes von unterschiedlichen Objekten über unterschiedliche Referenzen in Anspruch genommen werden können. Auch der Aufbau solcher Objektbeziehungen muss thematisiert werden. Des Weiteren wird das Prinzip der Vererbung im objektorientierten Sinne angesprochen. Dazu werden die wichtigsten Varianten der Vererbung anhand von verschiedenen Projekten vorgestellt. Zunächst wird die Vererbung als Spezialisierung im Sinne einer einfachen Erweiterung einer Oberklasse vorgestellt. Darauf folgt ein Projekt, welches das Verständnis von Vererbung um den Aspekt der späten Bindung erweitert, indem Dienste einer Oberklasse überschrieben werden. Modellierungen sollen in Form von Implementationsdiagrammen erstellt werden.

Zum Abschluss kann kurz auf das Prinzip der abstrakten Klasse eingegangen werden. Dieser Inhalt ist aber nicht obligatorisch für die Einführungsphase.

Zeitbedarf: 20 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Vertiefung des Referenzbegriffs und Einführung des Prinzips der dynamischen Referenzierung *a) Steuerung einfacher grafischer Objekte über eine Referenz aktuell, die jeweils durch eine Klickselektion mit der Maus auf ein neues* Objekt *gesetzt werden kann.*	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	*Beispiel*: Seifenblasen Die Schülerinnen und Schüler entwickeln ein Spiel, bei dem Seifenblasen über den Bildschirm schweben und durch anklicken mit der Maus zum Zerplatzen gebracht werden können. *Beispiel*: Sonnensystem Die Schülerinnen und Schüler entwickeln eine Simulation des Sonnensystems bei der Daten zum angeklickten Planeten ausgegeben werden.
2. Entwicklung eines Spiels mit der Notwendigkeit von Kollisionskontrollen zwischen zwei oder mehr grafischen Objekten *a) Modellierung des Spiels ohne* Berücksichtigung *der Kollision mit Hilfe eines Implementationsdiagramms* *b) Dokumentation der Klassen des Projekts* *c) Implementierung eines Prototypens*		*Beispiel:* Ufospiel Die Schülerinnen und Schüler entwickeln die Simulation eines Ufos, das Asteroiden ausweichen soll mit denen eine Kollision möglich ist. *Beispiel*: Billardkugeln Die Schülerinnen und Schüler entwickeln ein Spiel, bei dem tickende Billardkugeln mit einer beweglichen Box eingefangen werden sollen. *Beispiel*: Autospiel

ohne Kollision *d)* Ergänzung *einer Kollisionsabfrage durch zusätzliche Assoziationsbeziehungen in Diagramm, Dokumentation und Quellcode* *e)* Verallgemeinerung *der neuen Verwendung von Objektreferenzen* *f) Vertiefung: Entwicklung weiterer* Spiele *und Simulationen mit vergleichbarer Grundmodellierung*	Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I), • testen Programme schrittweise anhand von Beispielen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • modifizieren einfache Algorithmen und Programme (I), • stellen Klassen, Assoziations- und Vererbungsbeziehungen in Diagrammen grafisch dar (D), • dokumentieren Klassen durch Beschreibung der Funktionalität der Methoden (D).	Die Schülerinnen und Schüler entwickeln ein Autospiel, bei dem ein Auto durch einen Wald fahren und mit Bäumen kollidieren kann.
3. Erarbeitung einer Simulation mit grafischen Objekten, die sich durch* unterschiedliche *Ergänzungen voneinander unterscheiden (Vererbung durch Spezialisierung ohne Überschreiben von Methoden) *a) Analyse und Erläuterung einer Basisversion der grafischen Klasse* *b) Realisierung von grafischen Erweiterungen zur Basisklasse mit und ohne Vererbung (Implementationsdiagramm und Quellcode)* *c) Verallgemeinerung und* Reflexion *des Prinzips der Vererbung am Beispiel*		*Beispiel*: Schneemann Die Schülerinnen und Schüler erstellen eine Simulation von Schneemännern, die unterschiedliche Kopfbedeckungen tragen.

der Spezialisierung		
4. Entwicklung einer komplexeren Simulation mit grafischen Elementen, die unterschiedliche Animationen durchführen (Vererbung mit Überschreiben von Methoden) *a) Analyse und Erläuterung einer einfachen grafischen Animationsklasse* *b) Spezialisierung der Klasse zu* Unterklassen *mit verschiedenen Animationen durch Überschreiben der entsprechenden Animationsmethode* *c) Reflexion des Prinzips der späten* Bindung *d) Vertiefung: Entwicklung eines* vergleichbaren *Projekts mit einer (abstrakten) Oberklasse*		*Beispiel*: Flummibälle Die Schülerinnen und Schüler entwickeln eine Simulation von Flummibällen, bei der unterschiedliche Bälle unterschiedliche Bewegungen durchführen. *Beispiel*: Weihnachtsbaum Die Schülerinnen und Schüler entwickeln eine Simulation eines Weihnachtsbaums mit Hilfe einer abstrakten Klasse Schmuck.

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

Die Schülerinnen und Schüler

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

Unterrichtsvorhaben Q1-I:

Thema: Wiederholung der objektorientierten Modellierung und Programmierung 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?*

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

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

Zeitbedarf: 10 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Wiederholung und Erweiterung der objektorientierten Modellierung und Programmierung durch Analyse und Erweiterung eines kontextbezogenen Beispiels *a) Analyse* der *Problemstellung* *b) Analyse der Modellierung (Implementationsdiagramm)* *c)* Erweiterung *der Modellierung im Implementationsdiagramm (Vererbung, abstrakte Klasse)* *d) Kommunikation zwischen* mindestens *zwei Objekten (grafische Darstellung)* *e) Dokumentation von Klassen* *f)* Implementierung *der* Anwendung *oder von Teilen der Anwendung*	Die Schülerinnen und Schüler • analysieren und erläutern objektorientierte Modellierungen (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M), • ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M), • modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M), • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • wenden eine didaktisch orientierte	*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 Form von Würfeln und Zuckerringe in Form von Toren. Ein Prototyp, der bereits mit Kugeln geschmückt werden kann, kann zur Verfügung gestellt werden. Da alle Schmuckstücke über die Funktion des Auf- und Abschmückens verfügen sollen, liegt es nahe, dass entsprechende Methoden in einer gemeinsamen Oberklasse realisiert werden.

	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).	

Unterrichtsvorhaben Q1-II:

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

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

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

Zeitbedarf: 20 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Die Datenstruktur Schlange im Anwendungskontext unter Nutzung der Klasse Queue *a) Analyse der Problemstellung,* Ermittlung *von Objekten, ihren Eigenschaften und Operationen* *b) Erarbeitung der Funktionalität der* Klasse *Queue* *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 dynamischen Datenstrukturen (I),	*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. *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.2 – Warteschlange
2. Die Datenstruktur Stapel im Anwendungskontext unter Nutzung der Klasse Stack	• nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • interpretieren Fehlermeldungen und	*Beispiel:* Heftstapel In einem Heftstapel soll das Heft einer Schülerin gefunden werden. oder

a) Analyse der Problemstellung, Ermittlung *von Objekten, ihren Eigenschaften und Operationen* *b)* Erarbeitung *der Funktionalität der Klasse Stack* *c) Modellierung und* Implementierung *der Anwendung unter Verwendung eines oder mehrerer Objekte der Klasse Stack*	korrigieren den Quellcode (I), • testen Programme systematisch anhand von Beispielen (I), • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D).	*Beispiel*: Kisten stapeln In einem Stapel nummerierter Kisten soll eine bestimmte Kiste gefunden und an einen Kunden geliefert werden. Dazu müssen Kisten auf verschiedene Stapel gestapelt und wieder zurückgestellt werden.
3. Die Datenstruktur lineare Liste im Anwendungskontext unter Nutzung der Klasse List *a) Erarbeitung der Vorteile der Klasse List im Gegensatz zu den bereits bekannten linearen Strukturen* *b) Modellierung und Implementierung einer* kontextbezogenen *Anwendung unter Verwendung der Klasse List.*		*Beispiel*: 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: Grundsätzlicher Überblick zu Such- und Sortieralgorithmen auf linearen Datenstrukturen (Array, lineare Listen)

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

Vorhabenbezogene Konkretisierung:
In einem Anwendungskontext werden zunächst Informationen in 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 Felder und lineare Listen). 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: 12 Stunden

Sequenzierung des Unterrichtsvorhabens:

<table>
<tr><th>Unterrichtssequenzen</th><th>Zu entwickelnde Kompetenzen</th><th>Beispiele, Medien, Materialien</th></tr>
<tr><td>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)</td><td rowspan="2">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 Programmiersprache bei der Implementierung und zur Analyse von Programmen (I),</td><td>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.</td></tr>
<tr><td>2. Sortieren in Listen und Arrays - Entwicklung und Implementierung von iterativen und rekursiven Sortierverfahren

a) Entwicklung und Implementierung eines einfachen Sortierverfahrens für eine Liste</td><td>Beispiel: Karteiverwaltung
(s.o.)

oder

Beispiel: Bundesjugendspiele</td></tr>
</table>

b) Implementierung eines einfachen Sortierverfahrens für ein Feld *c) Entwicklung eines rekursiven* Sortierverfahren *für ein Feld (z.B. Sortieren durch Mischen)*	• 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* Speicherbedarfs *bei beiden Sortierverfahren* *c)* Beurteilung *der Effizienz der beiden Sortierverfahren*		*Beispiel:* Karteiverwaltung (s.o.) oder *Beispiel*: Bundesjugendspiele (s.o.)

Unterrichtsvorhaben Q1-IV: 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: 12 Stunden

Sequenzierung des Unterrichtsvorhabens:

<table>
<tr><th>Unterrichtssequenzen</th><th>Zu entwickelnde Kompetenzen</th><th>Beispiele, Medien oder Materialien</th></tr>
<tr><td>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</td><td rowspan="3">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),

• 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</td><td>Beispiele:

Cola-Automat, Geldspielautomat,

Roboter, Zustandsänderung eines Objekts „Auto“, Akzeptor für bestimmte Zahlen, Akzeptor für Teilwörter in längeren Zeichenketten, Akzeptor für Terme</td></tr>
<tr><td>2. Untersuchung und Entwicklung von Grammatiken regulärer Sprachen

a) Erarbeitung der formalen Darstellung regulärer Grammatiken

b) Untersuchung, Modifikation und Entwicklung von Grammatiken

c) Entwicklung von endlichen Automaten zum Erkennen regulärer Sprachen die durch Grammatiken gegeben werden

d) Entwicklung regulärer Grammatiken zu endlichen Automaten</td><td>Beispiele:

reguläre Grammatik für Wörter mit ungerader Parität, Grammatik für Wörter, die bestimmte Zahlen repräsentieren, Satzgliederungsgrammatik</td></tr>
<tr><td>3. Grenzen endlicher Automaten</td><td>Beispiele:</td></tr>
</table>

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

Unterrichtsvorhaben Q1-V:
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: 10 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien oder Materialien**
1. Von-Neumann-Architektur und die Ausführung maschinennaher Programme *a) prinzipieller Aufbau einer von Neumann-Architektur mit CPU,* Rechenwerk*, Steuerwerk, Register und Hauptspeicher* *b) einige maschinennahe Befehlen und ihre Repräsentation in einem Binär-Code, der in einem Register gespeichert werden kann* *c) Analyse und Erläuterung der Funktionsweise eines* einfachen *maschinennahen Programms*	Die Schülerinnen und Schüler • erläutern die Ausführung eines einfachen maschinennahen Programms sowie die Datenspeicherung auf einer „Von-Neumann-Architektur“ (A), • untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen (A).	*Beispiel:* Addition von 4 zu einer eingegeben Zahl mit einem Rechnermodell
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

Unterrichtsvorhaben Q2-I:

Thema: *Modellierung und Implementierung von Anwendungen mit dynamischen linearen und nichtlinearen Datenstrukturen.*

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

Vorhabenbezogene Konkretisierung:
Anhand von Beispielen für Baumstrukturen werden grundlegende Begriffe eingeführt und der rekursive Aufbau binärer Bäume dargestellt.
Anschließend werden für eine Problemstellung in einem der Anwendungskontexte Klassen modelliert und implementiert. Dabei werden die Operationen der Datenstruktur Binärbaum thematisiert und die entsprechende Klasse BinaryTree (der Materialien für das Zentralabitur in NRW) der Vorgaben für das Zentralabitur NRW verwendet. Klassen und ihre Beziehungen werden in Entwurfs- und Implementationsdiagrammen dargestellt. Die Funktionsweise von Methoden wird anhand grafischer Darstellungen von Binärbäumen erläutert.

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

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

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

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

Zeitbedarf: 30 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Analyse von Baumstrukturen in verschiedenen Kontexten *a) Grundlegende Begriffe (Grad, Tiefe, Höhe, Blatt, Inhalt, Teilbaum, Ebene, Vollständigkeit)* *b) Aufbau und Darstellung von binären* Bäumen *anhand von Baumstrukturen in verschiedenen Kontexten*	Die Schülerinnen und Schüler • erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (A), • analysieren und erläutern Algorithmen und Programme (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M), • modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M), • verwenden bei der Modellierung geeigneter Problemstellungen die Möglichkeiten der Polymorphie (M), • entwickeln iterative und rekursive	*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

	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).	mehrere Fragen mit ja oder nein beantwortet. Die Fragen, die möglich sind, wenn die Antwort auf eine Frage mit „ja“ beantwortet wird, befinden sich im linken Teilbaum, die Fragen, die möglich sind, wenn die Antwort „nein“ lautet, stehen im rechten Teilbaum. *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*		*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

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*		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 *a) Analyse der Problemstellung, Ermittlung von Objekten, ihren* Eigenschaften *und Operationen* *b) Modellierung eines Entwurfsdiagramms und* Entwicklung *eines Implementationsdiagramm,* *c) grafische Darstellung eines binären Suchbaums und Erarbeitung der Struktureigenschaften* *d) Erarbeitung der Klasse*		*Beispiel*: Informatikerbaum als *Suchbaum* In einem binären *Suchbaum* werden die Namen und die Geburtsdaten von Informatikern lexikographisch geordnet abgespeichert. Alle Namen, die nach dieser Ordnung vor dem Namen im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Namen im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.) Folgende Funktionalitäten werden benötigt: • Einfügen der Informatiker-Daten in den Baum

BinarySearchTree und Einführung des Interface Item zur Realisierung einer geeigneten *Ordnungsrelation* *e) Implementierung der Anwendung oder von Teilen der* Anwendung *inklusive einer sortierten Ausgabe des Baums*		• 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: *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: 8 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Daten in Netzwerken und Sicherheitsaspekte in Netzen sowie beim Zugriff auf Datenbanken *a) Beschreibung eines Datenbankzugriffs im Netz anhand eines Anwendungskontextes und einer Client-Server-Struktur zur Klärung der Funktionsweise eines Datenbankzugriffs* *b) Netztopologien als Grundlage von Client-Server-Strukturen und TCP/IP-Schichtenmodell als* Beispiel *für eine Paketübermittlung in einem Netz* *c) Vertraulichkeit, Integrität, Authentizität in Netzwerken sowie symmetrische und asymmetrische kryptografische Verfahren (Cäsar-, Vigenère-, RSA-*Verfahren*) als Methoden Daten im Netz verschlüsselt zu übertragen*	Die Schülerinnen und Schüler • beschreiben und erläutern Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken (A), • analysieren und erläutern Eigenschaften und Einsatzbereiche symmetrischer und asymmetrischer Verschlüsselungsverfahren (A), • untersuchen und bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen, die Sicherheit von Informatiksystemen sowie die Einhaltung der Datenschutzbestimmungen und des Urheberrechts (A), • untersuchen und bewerten Problemlagen, die sich aus dem Einsatz von Informatiksystemen ergeben, hinsichtlich rechtlicher Vorgaben, ethischer Aspekte und gesellschaftlicher Werte unter Berücksichtigung unterschiedlicher Interessenlagen (A), • nutzen bereitgestellte Informatiksysteme und das Internet reflektiert zum Erschließen, zur Aufbereitung und Präsentation fachlicher Inhalte (D).	
2. Fallbeispiele zur Datenschutzproblematik und zum Urheberrecht		

Unterrichtsvorhaben Q2-III: 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: 14 Stunden

Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Nutzung von relationalen Datenbanken *a) Aufbau von Datenbanken und* Grundbegriffe • Entwicklung von Fragestellungen zur vorhandenen Datenbank • Analyse der Struktur der vorgegebenen Datenbank und Erarbeitung der Begriffe Tabelle, Attribut, Datensatz, Datentyp, Primärschlüssel, Fremdschlüssel, Datenbankschema *b) SQL*-Abfragen • Analyse vorgegebener SQL-Abfragen und Erarbeitung der Sprachelemente von SQL (SELECT (DISTINCT) … FROM, WHERE, AND, OR, NOT) auf einer Tabelle • Analyse und Erarbeitung von SQL-Abfragen auf einer und mehrerer Tabelle zur Beantwortung der Fragestellungen (JOIN, UNION, AS, GROUP BY,ORDER BY, ASC, DESC, COUNT, MAX, MIN, SUM, Arithmetische Operatoren: +, -, *, /, (…), Vergleichsoperatoren: =, <>, >, <, >=, <=, LIKE, BETWEEN, IN, IS NULL)	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	*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 (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 ww.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.

c) Vertiefung an einem weiteren Datenbankbeispiel	vorgegebene Normalformen (M), • verwenden die Syntax und Semantik einer Datenbankabfragesprache, um Informationen aus einen Datenbanksystem zu extrahieren (I), • ermitteln Ergebnisse von Datenbankabfragen über mehrere verknüpfte Tabellen (D), • stellen Entitäten mit ihren Attributen und die Beziehungen zwischen Entitäten in einem Entity-Relationship-Diagramm grafisch dar (D), • überprüfen Datenbankschemata auf vorgegebene Normalisierungseigenschaften (D).	
2. Modellierung von relationalen Datenbanken *a) Entity*-Relationship-*Diagramm* • Ermittlung von Entitäten, zugehörigen Attributen, Relationen und Kardinalitäten in Anwendungssituationen und Modellierung eines Datenbankentwurfs in Form eines Entity-Relationship-Diagramms • Erläuterung und Modifizierung einer Datenbankmodellierung *b)* Entwicklung *einer Datenbank aus einem Datenbankentwurf* • Modellierung eines relationalen Datenbankschematas zu einem Entity-Relationship-Diagramm inklusive der Bestimmung von Primär- und Sekundärschlüsseln *c) Redundanz, Konsistenz und* Normalformen • Untersuchung einer Datenbank hinsichtlich Konsistenz und Redundanz in einer Anwendungssituation		*Beispiel:* Fahrradverleih Der Fahrradverleih *BTR (BikesToRent)* verleiht unterschiedliche Typen von Fahrrädern diverser Firmen an seine Kunden. Die Kunden sind bei *BTR* registriert (Name, Adresse, Telefon). *BTR* kennt von den Fahrradfirmen den Namen und die Telefonnummer. Kunden von *BTR* können CityBikes, Treckingräder und Mountainbikes ausleihen. *Beispiel*: Reederei Die Datenverwaltung einer Reederei soll in einem Datenbanksystem umgesetzt werden. Ausgehend von der Modellierung soll mit Hilfe eines ER-Modells und eines Datenbankschemas dieser erste Entwurf normalisiert und in einem Datenbanksystem umgesetzt werden. Es schließen sich diverse SQL-Abfragen an, wobei auf die Relationenalgebra eingegangen wird. *Beispiel:* Buchungssystem In dem Online-Buchungssystem einer Schule können die Lehrer Medienräume, Beamer, Laptops, Kameras, usw. für einen bestimmten Zeitpunkt buchen, der durch Datum und die Schulstunde festgelegt ist. Dazu ist die Datenbank zu modellieren, ggf. zu normalisieren und im Datenbanksystem umzusetzen. Weiter sollen sinnvolle Abfragen

• Überprüfung von Datenbankschemata hinsichtlich der 1. bis 3. Normalform und Normalisierung (um Redundanzen zu vermeiden und Konsistenz zu gewährleisten)		entwickelt werden. Unter (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.

Kapitel 3: Grundsätze der fachmethodischen und fachdidaktischen Arbeit

Unter Berücksichtigung des Schulprogramms hat die Fachkonferenz Informatik des Alexander-von-Humboldt-Gymnasiums die folgenden fachmethodischen und fachdidaktischen Grundsätze beschlossen. In diesem Zusammenhang beziehen sich die Grundsätze 1 bis 14 auf fächerübergreifende Aspekte, die auch Gegenstand der Qualitätsanalyse sind, die Grundsätze 15 bis 21 sind fachspezifisch angelegt.

Überfachliche Grundsätze:

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

Fachliche Grundsätze:

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

Kapitel 4:Grundsätze der Leistungsbewertung und Leistungsrückmeldung

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

Auf der Grundlage von §13 - §16 der APO-GOSt sowie Kapitel 3 des Kernlehrplans Informatik für die gymnasiale Oberstufe hat die Fachkonferenz des Alexander-von-Humboldt-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.

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..
- Die Korrektur und Bewertung der Klausuren erfolgt anhand eines kriterienorientierten Bewertungsbogens, den die Schülerinnen und Schüler als Rückmeldung erhalten.

Nachteilsausgleich:
Bei Lernbeeinträchtigungen wird im Rahmen der rechtlichen Vorgaben individuell geprüft, inwiefern ein Nachteilsausgleich gewährt werden kann (vgl. APO_GOSt, §13, 7; z.B. verlängerte Arbeits- oder Auswahlzeiten).

Hinweise zur Berücksichtigung von Verstößen gegen den Sprachgebrauch:
Sprachliche Richtigkeit und korrekte Verwendung der Fachsprache werden sowohl bei Klausuren als auch bei der "Sonstigen Mitarbeit" eingefordert. Grobe Verstöße diesbezüglich sollen bei der Bewertung mit berücksichtigt werden (z.B. im Rahmen der Darstellungsleistung) und können in Anlehnung an die Bestimmungen des Zentralabiturs zu einer Herabsetzung der Notenpunkte führen.

Verbindliche Instrumente:

Im Fach Informatik setzt sich die Kursabschlussnote gleichwertig aus den Beurteilungsbereichen "Klausuren" und "Sonstige Mitarbeit" zusammen.
In beiden Beurteilungsbereichen werden sowohl prozess- als auch inhaltsbezogenen Kompetenzen berücksichtigt.

Kapitel 4.1: Beurteilungsbereich Klausuren

Instrumente:

- Einführungsphase: 1 Klausur im 1. Halbjahr und 2 Klausuren im 2. Halbjahr
  Dauer der Klausur: 2 Unterrichtsstunden

- Grundkurse Q 1: 2 Klausuren je Halbjahr
  Dauer der Klausuren: 2 Unterrichtsstunden im 1. Halbjahr und 3 Unterrichts- stunden im 2. Halbjahr

- Grundkurse Q 2.1: 2 Klausuren
  Dauer der Klausuren: 3 Unterrichtsstunden
- Grundkurse Q 2.2: 1 Klausur unter Abiturbedingungen

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.

Kapitel 4.2: Beurteilungsbereich Sonstige Mitarbeit

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

Verbindliche Absprachen der Fachkonferenz

In der Einführungs- bzw. 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
- Mitarbeit in Partner-/Gruppenarbeitsphasen
- Referate

Praktische Leistungen am Computer
- Implementierung, Test und Anwendung von Informatiksystemen

Sonstige schriftliche Leistungen
- Arbeitsmappe und Arbeitstagebuch (Portfolio) 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 Übung dauern ca. 20 Minuten und umfassen den Stoff der letzten ca. 4–6 Stunden.
- Bearbeitung von schriftlichen Aufgaben im Unterricht

Kriterien

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

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

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

Bei Gruppenarbeiten auch auf

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

Bei Projektarbeit darüber hinaus auf

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

Im Folgenden werden Kriterien für die Bewertung der sonstigen Leistungen jeweils für eine gute bzw. eine ausreichende Leistung dargestellt. Dabei ist bei der Bildung der Quartals- und Abschlussnote jeweils die Gesamtentwicklung der Schülerin bzw. des Schülers zu berücksichtigen, eine arithmetische Bildung aus punktuell erteilten Einzelnoten erfolgt nicht:

Leistungsaspekt	**Anforderungen für eine**	
	gute Leistung	**ausreichende Leistung**
	Die Schülerin, der Schüler	
Qualität der Unterrichtsbeiträge **(Kommunizieren, Argumentiern, Präsentieren)**	nennt richtige Lösungen und begründet sie nachvollziehbar im Zusammenhang der Aufgabenstellung	nennt teilweise richtige Lösungen, in der Regel jedoch ohne nachvollziehbare Begründungen
	geht selbstständig auf andere Lösungen ein, findet Argumente und Begründungen für ihre/seine eigenen Beiträge	geht selten auf andere Lösungen ein, nennt Argumente, kann sie aber nicht begründen
	kann ihre/seine Ergebnisse auf unterschiedliche Art und mit unterschiedlichen Medien darstellen	kann ihre/seine Ergebnisse nur auf eine Art darstellen
Kontinuität/Quantität **(Kommunizieren)**	beteiligt sich regelmäßig am Unterrichtsgespräch	nimmt eher selten am Unterrichtsgespräch teil
Selbstständigkeit **(Problemlösen, Kommunizieren, Modellieren, Implementieren)**	bringt sich von sich aus in den Unterricht ein	beteiligt sich gelegentlich eigenständig am Unterricht
	ist selbstständig ausdauernd bei der Sache und erledigt Aufgaben gründlich und zuverlässig	benötigt oft eine Aufforderung, um mit der Arbeit zu beginnen; arbeitet Rückstände nur teilweise auf
	strukturiert und erarbeitet neue Lerninhalte weitgehend selbstständig, stellt selbstständig Nachfragen	erarbeitet neue Lerninhalte mit umfangreicher Hilfestellung, fragt diese aber nur selten nach
	erarbeitet bereitgestellte Materialien selbstständig	erarbeitet bereitgestellte Materialen eher lückenhaft
Kooperation **(Kommunizieren)**	bringt sich ergebnisorientiert in die Gruppen-/Partnerarbeit ein	bringt sich nur wenig in die Gruppen-/Partnerarbeit ein
	arbeitet kooperativ und respektiert die Beiträge Anderer	unterstützt die Gruppenarbeit nur wenig, stört aber nicht
Gebrauch der Fachsprache **(Kommunizieren)**	wendet Fachbegriffe sachangemessen an und kann ihre Bedeutung erklären	versteht Fachbegriffe nicht immer, kann sie teilweise nicht sachangemessen anwenden
Werkzeuggebrauch **(Werkzeuge, Modellieren, Implementieren)**	setzt Werkzeuge im Unterricht sicher bei der Bearbeitung von Aufgaben und zur Visualisierung von Ergebnissen ein	benötigt häufig Hilfe beim Einsatz von Werkzeugen zur Bearbeitung von Aufgaben
Präsentation/Referat **(Kommunizieren, Präsentieren)**	präsentiert vollständig, strukturiert und gut nachvollziehbar	präsentiert an mehreren Stellen eher oberflächlich, die Präsentation weist Verständnislücken auf
Schriftliche Übung	ca. 75% der erreichbaren Punkte	ca. 50% der erreichbaren Punkte

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.

Kapitel 5: Lehr- und Lernmittel

In der Qualifikationsphase wird das Lehrbuch Informatik 2 - Neubearbeitung - für die gymnasiale Oberstufe, Schöningh, 2016 sowie ein von den unterrichtenden Lehrkräften erstelltes Script – benutzt.

Kapitel 6: Entscheidungen zu fach- und -unterrichtsübergreifenden Fragen

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

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

Projekttage
Alle zwei Jahre werden am Alexander-von-Humboldt-Gymnasium Projekttage angeboten. Die Fachkonferenz Informatik bietet in diesem Zusammenhang mindestens ein Projekt für Schülerinnen und Schüler der gymnasialen Oberstufe an.
Jedes Jahr wird am Alexander-von-Humboldt-Gymnasium MINT-Tag durchgeführt. Die Fachschaft Informatik bietet in diesem Zusammenhang Präsentation von Projekten (8. bis 12. Klassen), die aus den Unterrichtsreihen stammen.

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.

Um die Verknüpfung mit den **Ganztagsangeboten** der Schule zu stärken, wird je nach personellen Ressourcen eine bzw. zwei informatikspezifische Arbeitsgemeinschaft angeboten. Hierbei werden Themenwünsche der Schülerinnen und Schülern berücksichtigt (z.B. Computer, Robotik, 3D-Design).

Zu dem Thema **Verkehrserziehung** trägt die Informatik in sofern bei, als dass sie im Bereich der Logikschaltungen die Ampel thematisiert und so die SuS für die Problematik des geregelten Verkehrsflusses sensibilisiert. Um dem Aspekt der **Berufsorientierung** im Fach Informatik gerecht zu werden, wird in naher Zukunft eine Kooperation mit außerschulischen Partnern angestrebt. Die realitätsnahe Vermittlung der übergeordneten Handlungskompetenz trägt natürlich ohnehin dazu bei, dass das Fach nie komplett isoliert betrachtet wird und dadurch Einblicke in verschiedene Berufswelten speziell der Ingenieurswissenschaften vermittelt. Medien spielen als Fachgegenstand in der Informatik eine große Rolle. Auch der Umgang mit Medien bzw. **Medienerziehung** und damit verbundenen Bereiche wie bspw. Datenschutz sind Inhaltsfelder der Informatik.

Kapitel 7: Qualitätssicherung und Evaluation

Grundsätzlich werden fachbezogenen Unterrichtsaktivitäten von allen am Unterrichtbeteiligten reflektiert und auf Erfolg und Effizienz hin überprüft. Durch Evaluation der Ergebnisse der Klausuren und sonstigen schriftlichen bzw. mündlichen Leistungsergebnisse der SchülerInnen wird ein hohes Maß an fachlicher Qualitätssicherung erreicht.

Nach Abschluss des Abiturs 2021 wird die Fachkonferenz Informatik auf der Grundlage ihrer Unterrichtserfahrungen eine Gesamtsicht des schulinternen Curriculums vornehmen und ggf. eine Beschlussvorlage für die erste Fachkonferenz des folgenden Schuljahres erstellen.

Der folgende Abschnitt dient dazu, den Ist-Zustand in der fachlichen Arbeit festzustellen und zu dokumentieren, Beschlüsse der Fachkonferenz zur Fachgruppenarbeit in übersichtlicher Form festzuhalten sowie die Durchführung der Beschlüsse zu kontrollieren und zu reflektieren. Die Liste wird regelmäßig überarbeitet und angepasst. Sie dient auch dazu, Handlungsschwerpunkte für die Fachgruppe zu identifizieren und abzusprechen.

Fachgruppenarbeit im Überblick	
Funktionen	
Fachvorsitz	J. Martin
Stellvertretung	M. Kirschner
Sammlungsleitung	O. Waldrat, J. Martin
Fachlehrkräfte	O. Waldrat, J. Martin
Lerngruppen	Jgst. EF-Q2

Räumliche / materielle Ressourcen	
Fachräume	R. 0.016, R. 1.018
Lehrwerke	EF: Script, Q1-Q2: Script + Informatik 2 - Neubearbeitung - für die gymnasiale Oberstufe, Schöningh, 2016