Schulinterner Lehrplan für das Fach Informatik am Max-Planck-Gymnasium Gelsenkirchen in der Sekundarstufe II

Inhaltsverzeichnis

1. Das Fach Informatik an der Schule

Das Max-Planck-Gymnasium (MPG) ist eine vierzügige Schule mit ca. 840 Schülerinnen und Schülern (Stand: Januar 2014), etwa 60 Lehrerinnen und Lehrern. Das Einzugsgebiet der Schule umfasst zum größten Teil den Gelsenkirchener Stadtteil Buer. Die Schule kooperiert in der Oberstufe mit der Nachbarschule, dem Annette-von-Droste-Hülshoff-Gymnasium, unter anderem im Fach Informatik.

Das Fach Informatik wird am MPG ab der Jahrgangsstufe 8 im Wahlpflichtbereich II (WP II) dreistündig unterrichtet. In der Regel kommt dabei bei ca. 100 Schülerinnen und Schülern ein Kurs in der Kursstärke von etwa 20 Lernenden zustande. In diesem Kurs wird in altersangemessener Weise unter anderem auf die Grundlagen der Algorithmik am Beispiel einer didaktischen Lernumgebung, auf Funktionsweise und gesellschaftliche Wirkung der globalen Vernetzung und auf Datensicherheit eingegangen.

Darüber hinaus ist das Fach im MINT-Zweig der Schule eingebunden, der von den Schülerinnen und Schülern zu Beginn der Jahrgangsstufe 5 angewählt werden kann und mit einer zusätzlichen Wochenstunde bis zur Jahrgangsstufe 7 unterrichtet wird. Inhaltlich geht es dabei um den Einsatz des Computers als Werkzeug beim Verfassen von Texten oder Präsentationen, die grundsätzlichen Funktionen eines Betriebssystems wie etwa des Dateisystems und den Einsatz im Mathematik- und NW-Unterricht.

Im Unterricht in der Sekundarstufe II wird Java als Vertreter einer Programmiersprache Paradigmas der Objektorientierung eingesetzt, um den derzeitigen Anforderungen der Abiturvorgaben gerecht zu werden. In der Jahrgangsstufe 10 (Einführungsphase) kommt dabei zu didaktischen Reduzierung der Komplexität der Programmerstellung für Anfänger die Bibliothek GLOOP zum Einsatz, mit der

auf einfache Weise dreidimensionale Grundkörper wie Kugeln, Quader oder Prismen dargestellt und manipuliert werden können.

In der Einführungsphase (EPh) geht es in erster Linie um das Erlernen der grundsätzlichen Elemente von Java. Der Unterricht ist geprägt von abwechselnden Arbeitsphasen mündlicher oder schriftlicher Natur und der Arbeit an den PCs, um erarbeitete Verfahren, Strukturen oder Ideen praktisch zu erproben. Dabei ist der Unterricht im wesentlichen projektartig orientiert an Beispielszenarien, diem anfangs noch enger geführt zunehmend offeneren Charakter besitzen, um die Schülerinnen und Schüler zu motivieren, eigene Lösungen zu entwerfen, vorgegebene Lösungen durch eigene Ideen zu variieren oder zu erweitern.

Die Schule ist mit drei Computerräumen mit jeweils 28 Computerarbeitsplätzen sowie einigen mobilen Laptopwagen ausgestattet, die alle an das schulinterne Netzwerk angeschlossen sind, der es den Benutzern ermöglicht, mit einer einheitliche Umgebung zu arbeiten, gleich welchen PC sie gerade nutzen.

2. Entscheidungen zum Unterricht

Hinweis: Die nachfolgend dargestellte Umsetzung der verbindlichen Kompetenzerwartungen des Kernlehrplans findet auf zwei Ebenen statt. Das Übersichtsraster gibt den Lehrkräften einen raschen Überblick über die laut Fachkonferenz verbindlichen Unterrichtsvorhaben pro Schuljahr. In dem Raster sind außer dem Thema des jeweiligen Vorhabens das schwerpunktmäßig damit verknüpfte Inhaltsfeld bzw. die Inhaltsfelder, inhaltliche Schwerpunkte des Vorhabens sowie Schwerpunktkompetenzbereiche ausgewiesen. Die Konkretisierung von Unterrichtsvorhaben führt weitere Kompetenzerwartungen auf und verdeutlicht vorhabenbezogene Absprachen, z.B. zur Festlegung auf einen Aufgabentyp bei der Lernerfolgsüberprüfung durch eine Klausur.

2.1. Unterrichtsvorhaben

Die Darstellung der Unterrichtsvorhaben in diesem Dokument deckt sämtliche im Kernlehrplan Informatik [http://www.standardsicherung.schulministerium.nrw.de/lehrplaene/upload/klp_SII/if/GOSt_Informatik_Endfassung.pdf] angeführte Kompetenzen.

Zur Darstellung der Unterrichtsvorhaben werden zunächst die Unterrichtsvorhaben benannt und die angesprochenen Inhaltsfelder und Kompetenzen benannt. Außerdem wird eine Einschätzung der benötigten Unterrichtszeit benannt. Die benannten Vorhaben und ihre Kompetenzerwartungen sind für die Lehrer der Schule verbindlich, allerdings ist der Zeitbedarf nur als grober Richtwert zu verstehen, der je nach Bedarf unter- oder überschritten werden kann. Darüber hinaus lässt der schulinterne Lehrplan Platz für individuelle Schwerpunktsetzung der Fachlehrer, der zur Vertiefung, Eingehen auf Schülerinteressen, dem Besuch außerschulischer Lernorte o.ä. verwendet werden kann.

Die Darstellung der Unterrichtsvorhaben im Übersichtsraster sind verbindlich. Dies ermöglicht Schülerinnen und Schülern den Wechsel der Lerngruppe, z.B. im Falle einer Wiederholung einer Jahrgangsstufe, und garantiert einheitliche Standards. Demgegenüber haben die konkretisierten Unterrichtsvorhaben keine direkte bindende Wirkung, sondern hegen den Anspruch, neuen Kolleginnen oder Kollegen, Referendarinnen oder Referendaren Empfehlungen zu Inhalten, didaktiken Zugängen, Methoden, Kooperationen, Lernmitteln aber auch möglichen Leistungsüberprüfungen zu geben. Dabei ist es grundsätzlich erwünscht, dass die konkretisierten Sequenzen von den Lehrkräften überarbeitet und ihrer eigenen Vorstellungen und Erfahrung angepasst werden, solange die dargelegten Kompetenzen berücksichtigt werden.

2.1.1. Übersichtsraster Unterrichtsvorhaben

Einführungsphase (EPh)

EPh-UV 1

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

Zentrale Kompetenzen

- Argumentieren
- Darstellen und Interpretieren
- Kommunizieren und Kooperieren

Inhaltsfelder

- Informatiksysteme
- Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte

- Einzelrechner
- Dateisystem
- Einsatz von Informatiksystemen

Zeitbedarf: 5 Stunden

EPh-UV 2

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

Zentrale Kompetenzen

- Modellieren
- Implementieren
- Darstellen und Interpretieren
- Kommunizieren und Kooperieren

Inhaltsfelder

- Daten und ihre Strukturierung
- Formale Sprachen und Automaten

Inhaltliche Schwerpunkte

- Objekte und Klassen
- Syntax und Semantik einer Programmiersprache

Zeitbedarf: 8 Stunden

EPh-UV 3

Thema: Grundlagen der objektorientierten Programmierung und algorithmischer Grundstrukturen in Java anhand von einfachen Animationen

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 Implementation einfacher Algorithmen

Zeitbedarf: 18 Stunden

EPh-UV 4

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 Implementation einfacher Algorithmen

Zeitbedarf: 18 Stunden

EPh-UV 5

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

EPh-UV 6

Thema: Geschichte der digitalen Datenverarbeitung und die Grundlagen des Datenschutzes

Zentrale Kompetenzen

- Argumentieren
- Darstellen und Interpretieren
- Kommunizieren und Kooperieren

Inhaltsfelder

- Informatik, Mensch und Gesellschaft
- Informatiksysteme

Inhaltliche Schwerpunkte

- Wirkungen der Automatisierung
- Geschichte der automatischen Datenverarbeitung
- Digitalisierung

Zeitbedarf: 9 Stunden

Qualifikationsphase 1 (Q1)

Q1-UV 1

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 von Programmiersprachen
- Nutzung von Informatiksystemen

Zeitbedarf: 9 Stunden

Q1-UV 2

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 von Programmiersprachen

Zeitbedarf: 20 Stunden

Q1-UV 3

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 von Programmiersprachen

Zeitbedarf: 15 Stunden

Q1-UV 4

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: 24 Stunden

Q1-UV 5

Thema: Sicherheit und Datenschutz in Netzen

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: 12 Stunden

Qualifikationsphase 2 (Q2)

Q2-UV 1

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: 18 Stunden

Q2-UV 2

Thema: Endliche Automaten und formale Sprachen

Zentrale Kompetenzen

- Argumentieren
- Modellieren
- Darstellen und Interpretieren
- Kommunizieren und Kooperieren

Inhaltsfelder

- Formale Sprachen und Automaten

Inhaltliche Schwerpunkte

- Endliche Automaten
- Grammatiken regulärer Sprachen
- Möglichkeiten und Grenzen von Automaten und formalen Sprachen

Zeitbedarf: 21 Stunden

Q2-UV 3

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

2.1.2. Konkretisierte Unterrichtsvorhaben

Im Folgenden sollen die im Unterkapitel Abschnitt 2.1.1, „Übersichtsraster Unterrichtsvorhaben“ aufgeführten Unterrichtsvorhaben konkretisiert werden.

Hinweis

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

Unterrichtliche Anregungen: Die angeführten Beispiele, Medien und Materialien sind dagegen Vorschläge bzw. Hilfen für die Lehrkräfte der Beispielschule. In diesen Bereichen sind Abweichungen von den vorgeschlagenen Vorgehensweisen möglich.

Einführungsphase (EPh)

Einführung in die Nutzung von Informatiksystemen und in grundlegende Begrifflichkeiten (Unterrichtsvorhaben EF-I)

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.

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.

Zeitbedarf: 5 Stunden

Tabelle 1. Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Information, deren Kodierung und Speicherung a. Informatik als Wissenschaft der Verarbeitung von Informationen b. Darstellung von Informationen in Schrift, Bild und Ton c. Speichern von Daten mit informatischen Systemen am Beispiel der Schulrechner d. Vereinbarung von Richtlinien zur Datenspeicherung auf den Schulrechnern (z.B. Ordnerstruktur, Dateibezeichner usw.)	Die Schülerinnen und Schüler • beschreiben und erläutern den Aufbau und die Arbeitsweise singulärer Rechner am Beispiel der „Von-Neumann-Architektur“ (A), • nutzen die im Unterricht eingesetzten Informatiksysteme selbstständig, sicher, zielführend und verantwortungsbewusst (D), • nutzen das Internet zur Recherche, zum Datenaustausch und zur Kommunikation (K).	*Beispiel:* Textkodierung Kodierung und Dekodierung von Texten mit unbekannten Zeichensätzen (z.B. Wingdings) *Beispiel:* Bildkodierung Kodierung von Bildinformationen in Raster- und Vektorgrafiken
2. Aufbau informatischer Systeme a. Identifikation typischer Komponenten informatischer Systeme und anschließende Beschränkung auf das		*Material:* Demonstrationshardware Durch Demontage eines Demonstrationsrechners entdecken Schülerinnen und Schüler die verschiedenen Hardwarekomponenten eines Informatiksystems. Als Demonstrations-

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
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“		rechner bietet sich ein ausrangierter Schulrechner an.

Grundlagen der objektorientierten Analyse, Modellierung und Implementierung anhand von statischen Grafikszenen (Unterrichtsvorhaben EF-II)

Leitfragen: 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 didaktischen Programmierumgebung GLOOP begonnen. Die von der Bibliothek vorgegebenen Klassen werden von Schülerinnen und Schülern in Teilen analysiert und entsprechende Objekte anhand einfacher Problemstellungen erprobt. Dazu muss der grundlegende Aufbau einer Java-Klasse thematisiert und zwischen Deklaration, Initialisierung und Methodenaufrufen unterschieden werden.

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

Zeitbedarf: 8 Stunden

Tabelle 2. 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	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),	*Beispiel:* Vogelschwarm Schülerinnen und Schüler betrachten einen Vogelschwarm als Menge gleichartiger Objekte, die in einer Klasse mit Attributen und Methoden zusammengefasst werden können.

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
„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	• 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).	
2. Analyse von Klassen didaktischer Lernumgebungen a. Objektorientierte Programmierung als modularisiertes Vorgehen (Entwicklung von Problemlösungen auf Grundlage vorhandener Klassen) b. Teilanalyse der Klassen der didaktischen Lernumgebungen GLOOP		*Materialien:* Dokumentation der Bibliothek GLOOP
3. Implementierung dreidimensionaler, statischer Szenen 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)		*Beispiel:* Skulpturengarten Schülerinnen und Schüler erstellen ein Programm, das mit Hilfe von geometrischen Objekten der GLOOP-Umgebung einen Skulpturengarten auf den Bildschirm bringt. *Beispiel:* Olympische Ringe Die Schülerinnen und Schüler bilden das Emblem der olympischen Spiele mit Hilfe von GLOOP-Objekten nach. *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator — Sequenzielle Programmierung

Grundlagen der objektorientierten Programmierung und algorithmischer Grundstrukturen in Java anhand von einfachen Animationen (Unterrichtsvorhaben EF-III)

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 Umset-

zung 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: 18 Stunden

Tabelle 3. Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Bewegungsanimationen am Beispiel einfacher grafischer Objekte (GLObjekte) a. Kontinuierliche Verschiebung eines GLObjekts 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 GLObjekten mit Hilfsvariablen e. Meldungen zur Kollision zweier GLObjekte 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),	*Beispiel:* Wurfspiel Die Schülerinnen und Schüler realisieren mit Objekten der GLOOP-Umgebung ein Spiel, bei dem ein Ball über den Bildschirm bewegt und auf eine runde Zielscheibe geworfen werden soll. *Material:* Ergänzungsmaterialien zum Lehrplannavigator – Kontrollstrukturen
2. Erstellen und Verwalten größerer Mengen einfacher grafischer Objekte (GLObjekte) a. Erzeugung von Objekten mit Hilfe von Zählschleifen (FOR-Schleife)	• ordnen Klassen, Attributen und Methoden ihren Sichtbarkeitsbereich zu (M), • modifizieren einfache Algorithmen und Programme (I),	*Beispiel:* Hubschrauberlandeplatz Die Schülerinnen und Schüler realisieren einen runden Hubschrauberlandeplatz und eine Reihe von Landemarkierungen, die in einem Feld verwaltet

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
b. Verwaltung von Objekten in eindimensionalen Feldern (Arrays) c. Animation von Objekten, die in eindimensionalen Feldern (Arrays) verwaltet werden d. Vertiefung: Verschiedene Feldbeispiele	• implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I), • 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).	werden. Mit Hilfe der Landemarkierungen werden verschiedene 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. *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator — Kontrollstrukturen
3. Modellierung und Animation komplexerer grafisch repräsentierbarer Objekte a. Modellierung eines Simulationsprogramms mit eigenen Klassen, die sich selbst mit Hilfe von einfachen GLObjekten 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 e. Animation mit Hilfe des Aufrufs von selbstimplementierten Methoden f. Vertiefung: Weitere Projekte		*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 Ampelkreuzung mit mehreren Ampelanlagen an einem Bahnübergang. *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator — Eigene Klassen

Modellierung und Implementierung von Klassen- und Objektbeziehungen anhand von grafischen Spielen und Simulationen (Unterrichtsvorhaben EF-IV)

Leitfragen: 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: 18 Stunden

Tabelle 4. Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Vertiefung des Referenzbegriffs und Einführung des Prinzips der dynamischen Referenzierung a. Einführung der GLOOP-Objektselektion mit der Maus b. Einführung der Klasse GLObjekt als Oberklasse aller sichtbaren Objekte in GLOOP c. 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),	*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 Prototypen ohne Kollision d. Ergänzung einer Kollisionsabfrage durch zusätzliche Assoziationsbeziehungen in	• 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),	*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 Die Schülerinnen und Schüler entwickeln ein Autospiel, bei dem ein Auto durch einen Wald fahren und mit Bäumen kollidieren kann.

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
Diagramm, Dokumentation und Quellcode e. Verallgemeinerung der neuen Verwendung von Objektreferenzen f. Vertiefung: Entwicklung weiterer Spiele und Simulationen mit vergleichbarer Grundmodellierung	• 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).	*Materialien:* Ergänzungsmaterialien zum Lehrplannavigator — Assoziationen *Informationsblatt:* Ergänzungsmaterialien zum Lehrplannavigator — Implementationsdiagramme
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 der Spezialisierung		*Beispiel:* Schneemann Die Schülerinnen und Schüler erstellen eine Simulation von Schneemännern, die unterschiedliche Kopfbedeckungen tragen. *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator — Vererbung
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. *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator — Vererbung

Such- und Sortieralgorithmen anhand kontextbezogener Beispiele (Unterrichtsvorhaben EF-V)

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

Vorhabenbezogene Konkretisierung

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

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

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

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

Zeitbedarf: 9 Stunden

Tabelle 5. Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	**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 [http://www.csunplugged.org/sorting-algorithms]
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		*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 [http://

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
c. Bewertung von Algorithmen anhand der Anzahl der nötigen Vergleiche d. Variante des Sortierens durch Auswählen (Nutzung eines einzigen oder zweier Felder bzw. lediglich eines einzigen zusätzlichen Ablageplatzes oder mehrerer neuer Ablageplätze) e. Effizienzbetrachtungen an einem konkreten Beispiel bezüglich der Rechenzeit und des Speicherplatzbedarfs f. Analyse des weiteren Sortieralgorithmus (sofern nicht in Sequenz 1 und 2 bereits geschehen)		www.csunplugged.org/sorting-algorithms]
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 — Sorting Algorithms [http://www.csunplugged.org/sorting-algorithms]

Geschichte der digitalen Datenverarbeitung und die Grundlagen des Datenschutzes (Unterrichtsvorhaben EF-VI)

Leitfragen: Welche Entwicklung durchlief die moderne Datenverarbeitung und welche Auswirkungen ergeben sich insbesondere hinsichtlich neuer Anforderungen an den Datenschutz daraus?

Vorhabenbezogene Konkretisierung

Das folgende Unterrichtsvorhaben stellt den Abschluss der Einführungsphase dar. Schülerinnen und Schüler sollen selbstständig informatische Themenbereiche aus dem Kontext der Geschichte der Datenverarbeitung und insbesondere den daraus sich ergebenen Fragen des Datenschutzes bearbeiten. Diese Themenbereiche werden in Kleingruppen bearbeitet und in Form von Plakatpräsentationen vorgestellt. Schülerinnen und Schüler sollen dabei mit Unterstützung des Lehrenden selbstständige Recherchen zu ihren Themen anstellen und auch eine sinnvolle Eingrenzung ihres Themas vornehmen.

Anschließend wird verstärkt auf den Aspekt des Datenschutzes eingegangen. 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: 15 Stunden

Tabelle 6. Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Selbstständige Erarbeitung von Themen durch die Schülerinnen und Schüler a. Mögliche Themen zur Erarbeitung in Kleingruppen: • „Eine kleine Geschichte der Digitalisierung: vom Morsen zum modernen Digitalcomputer“ • „Eine kleine Geschichte der Kryptographie: von Caesar zur Enigma“ • „Von Nullen, Einsen und mehr: Stellenwertsysteme und wie man mit ihnen rechnet“ • „Kodieren von Texten und Bildern: ASCII, RGB und mehr“ • „Auswirkungen der Digitalisierung: Veränderungen der Arbeitswelt und Datenschutz“ b. Vorstellung und Diskussion durch Schülerinnen und Schüler	Die Schülerinnen und Schüler • bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen (A), • erläutern wesentliche Grundlagen der Geschichte der digitalen Datenverarbeitung (A), • stellen ganze Zahlen und Zeichen in Binärcodes dar (D), • interpretieren Binärcodes als Zahlen und Zeichen (D), • nutzen das Internet zur Recherche, zum Datenaustausch und zur Kommunikation. (K).	*Beispiel:* Ausstellung zu informatischen Themen Die Schülerinnen und Schüler bereiten eine Ausstellung zu informatischen Themen vor. Dazu werden Stellwände und Plakate vorbereitet, die ggf. auch außerhalb des Informatikunterrichts in der Schule ausgestellt werden können. *Materialien:* Schülerinnen und Schüler recherchieren selbstständig im Internet, in der Schulbibliothek, in öffentlichen Bibliotheken, usw.
2. Vertiefung des Themas Datenschutz a. Erarbeitung grundlegender Begriffe des Datenschutzen b. Problematisierung und Anknüpfung an die Lebenswelt der Schülerinnen und Schüler c. Diskussion und Bewertung von Fallbeispielen aus dem Themenbereich „Datenschutz“		*Beispiel:* Fallbeispiele aus dem aktuellen Tagesgeschehen Vertauschen, Quicksort Die Schülerinnen und Schüler bearbeiten Fallbeispiele aus ihrer eigenen Erfahrungswelt oder der aktuellen Medienberichterstattung. *Materialien:* Materialblatt zum Bundesdatenschutzgesetz

Qualifikationsphase, 1. Jahr (Q1)

Wiederholung der objektorientierten Modellierung und Programmierung anhand einer kontextbezogenen Problemstellung (Unterrichtsvorhaben Q1-I)

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 (möglicherweise 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: 9 Stunden

Tabelle 7. Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
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	*Material:* 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 Tori. 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.

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
	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).	

Modellierung und Implementierung von Anwendungen mit dynamischen, linearen Datenstrukturen (Unterrichtsvorhaben Q1-II)

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.

Alle Fachklassen werden mindestens an einem Beispiel behandelt. Erlaubt es der Zeitplan, so kann in einem weiteren Anwendungskontext beispielsweise die Kombination zweier linearer Datenstrukturen oder ein besonders interessanter informatischer Kontext wie ein Warteschlangensystem oder die Terminterpretation klammerfreier Rechenterme betrachtet werden.

Zeitbedarf: 20 Stunden

Tabelle 8. 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 nichtlinearer) Datenstrukturen (A), • analysieren und erläutern Algorithmen und Programme (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M), • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • modifizieren Algorithmen und Programme (I), • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • testen Programme systematisch anhand von Beispielen (I), • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D).	*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 b. Erarbeitung der Funktionalität der Klasse `Stack` c. Modellierung und Implementierung der Anwendung unter Verwendung eines oder mehrerer Objekte der Klasse `Stack`		*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`		*Beispiel:* Abfahrtslauf Bei einem Abfahrtslauf kommen die Skifahrer nacheinander an und werden nach ihrer

Unterrichtssequenzen	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
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`.		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

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
		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.

Suchen und Sortieren auf linearen Datenstrukturen (Unterrichtsvorhaben Q1-III)

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: 15 Stunden

Tabelle 9. Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Suchen von Daten in Listen und Arrays a. Lineare Suche in Listen und in Arrays b. Binäre Suche in Arrays als Beispiel für rekursives Problemlösen c. Untersuchung der beiden Suchverfahren hinsichtlich ihrer Effizienz (Laufzeitverhalten, Speicherbedarf)	Die Schülerinnen und Schüler • analysieren und erläutern Algorithmen und Programme (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • beurteilen die Effizienz von Algorithmen unter Berücksichtigung des Speicherbedarfs und der Zahl der Operationen (A), • entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“ und „Teilen und Herrschen“ (M), • modifizieren Algorithmen und Programme (I), • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I), • implementieren und erläutern iterative und rekursive Such- und Sortierverfahren (I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • testen Programme systematisch anhand von Beispielen (I), • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D).	*Beispiel:* Karteiverwaltung Für ein Adressverwaltungsprogramm soll eine Methode zum Suchen einer Adresse geschrieben werden.
2. Sortieren in Listen und Arrays - Entwicklung und Implementierung von iterativen und rekursiven Sortierverfahren a. Entwicklung und Implementierung eines einfachen Sortierverfahrens für eine Liste b. Implementierung eines einfachen Sortierverfahrens für ein Feld c. Entwicklung eines rekursiven Sortierverfahren für ein Feld (z.B. Sortieren durch Mischen)		
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		

Modellierung und Nutzung von relationalen Datenbanken in Anwendungskontexten (Unterrichtsvorhaben Q1-IV)

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: 24 Stunden

Tabelle 10. 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`	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	*Beispiel:* Video-Center Video-Center [http://dokumentation.videocenter.schule.de/old/video/index.html] 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. Der obige Link führt zur Datenbank sowie näheren Informationen. Lesenswert ist auch die dort verlinkte „Dokumentation der Fallstudie” mit didaktischem Material, welches alternativ bzw. ergänzend zu der im Folgenden beschriebenen

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
(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	Entitäten, zugehörige Attribute, Relationen und Kardinalitäten (M), • modifizieren eine Datenbankmodellierung (M), • modellieren zu einem Entity-Relationship-Diagramm ein relationales Datenbankschema (M), • bestimmen Primär- und Sekundärschlüssel (M), • überführen Datenbankschemata in vorgegebene Normalformen (M), • verwenden die Syntax und Semantik einer Datenbankabfragesprache, um Informationen aus einen Datenbanksystem zu extrahieren (I), • ermitteln Ergebnisse von Datenbankabfragen über mehrere verknüpfte Tabellen (D), • stellen Entitäten mit ihren Attributen und die Beziehungen zwischen Entitäten in einem Entity-Relationship-Diagramm grafisch dar (D), • überprüfen Datenbankschemata auf vorgegebene Normalisierungseigenschaften (D).	Durchführung verwendet werden kann. *Beispiel:* Schulbuchausleihe [http://www.brd.nrw.de/lerntreffs/informatik/structure/material/sek2/datenbanken.php] Unter obigem Link wird eine Datenbank zur Verfügung gestellt, die Daten einer Schulbuch-Ausleihe enthält (über 1000 Entleiher, 200 Bücher mit mehreren tausend Exemplaren und viele Ausleihvorgänge). Die Datenbank kann in OpenOffice eingebunden werden.
2. Modellierung von relationalen Datenbanken a. Entity-Relationship-Diagramm • Ermittlung von Entitäten, zugehörigen Attributen, Relationen und Kardinalitäten in Anwendungssituationen und Modellierung eines Datenbankentwurfs in Form eines Entity-Relationship-Diagramms • Erläuterung und Modifizierung einer Datenbankmodellierung b. Entwicklung einer Datenbank aus einem Datenbankentwurf • Modellierung eines relationalen Datenbankschemas zu einem Entity-Relationship-Diagramm inklusive der Bestimmung von Primär- und Sekundärschlüsseln c. Redundanz, Konsistenz und Normalformen		*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,

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
• 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)		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. Das freie Buchungssystem MRBS (meeting room booking system) [http://www.brd.nrw.de/lerntreffs/informatik/structure/material/sek2/datenbanken.php] bietet neben der Software eine 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.

Sicherheit und Datenschutz in Netzen (Unterrichtsvorhaben Q1-V)

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: 12 Stunden

Tabelle 11. 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).	*Materialien:* Ergänzungsmaterialien zum Lehrplannavigator — Verschlüsselung, Zugriff auf Daten in Netzwerken
2. Fallbeispiele zur Datenschutzproblematik und zum Urheberrecht		*Material:* Ergänzungsmaterialien zum Lehrplannavigator — Datenschutz beim Videocenter, Materialblatt-Datenschutzgesetz

Qualifikationsphase, 2. Jahr (Q2)

Modellierung und Implementierung von Anwendungen mit dynamischen, nichtlinearen Datenstrukturen (Unterrichtsvorhaben Q2-I)

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 $\rightarrow$ 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: 18 Stunden

Tabelle 12. 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 nichtlinearer) Datenstrukturen (A), analysieren und erläutern Algorithmen und Programme (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M), • modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch 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),	*Beispiel:* Termbaum Der Aufbau von Termen wird mit Hilfe von binären Baumstrukturen verdeutlicht. *Beispiel:* Ahnenbaum Die binäre Baumstruktur ergibt sich daraus, dass jede Person genau einen Vater und eine Mutter hat. **Weitere Beispiele für Anwendungskontexte binärer Bäume:** *Beispiel:* Suchbäume (zur sortierten Speicherung von Daten) Alle Inhalte, die nach einer Ordnung vor dem Inhalt im aktuellen Teilbaum stehen, sind in dessen

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
	• 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).	linkem Teilbaum, alle die nach dem Inhalt im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.) *Beispiel:* Entscheidungsbäume Um eine Entscheidung zu treffen, werden mehrere Fragen mit ja oder nein beantwortet. Die Fragen, die möglich sind, wenn die Antwort auf eine Frage mit „ja“ beantwortet wird, befinden sich im linken Teilbaum, die Fragen, die möglich sind, wenn die Antwort „nein“ lautet, stehen im rechten Teilbaum. *Beispiel:* Codierungsbäume für Codierungen, deren Alphabet aus genau zwei Zeichen besteht Morse hat Buchstaben als Folge von Punkten und Strichen codiert. Diese Codierungen können in einem Binärbaum dargestellt werden, so dass ein Übergang zum linken Teilbaum einem Punkt und ein Übergang zum rechten Teilbaum einem Strich entspricht. Wenn man im Gesamtbaum startet und durch Übergänge zu linken oder

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
		rechten Teilbäumen einen Pfad zum gewünschten Buchstaben sucht, erhält man die Morsecodierung des Buchstabens. *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 — Binärbaum
2. Die Datenstruktur Binärbaum im Anwendungskontext unter Nutzung der Klasse `BinaryTree` a. Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen im Anwendungskontext b. Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramms c. Erarbeitung der Klasse BinaryTree und beispielhafte Anwendung der Operationen d. Implementierung der Anwendung oder von Teilen der Anwendung e. Traversierung eines Binär-		*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: • Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 — Binärbaum • Einfügen der Informatiker-Daten in den Baum

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
baums im Pre-, In- und Postorderdurchlauf		• Suchen nach einem Informatiker über den Schlüssel Name • Ausgabe des kompletten Datenbestands in nach Namen sortierter Reihenfolge *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 — Binärbaum
3. Die Datenstruktur binärer Suchbaum im Anwendungskontext unter Verwendung der Klasse `BinarySearchTree` a. Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen b. Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramm, grafische Darstellung eines binären Suchbaums und Erarbeitung der Struktureigenschaften c. Erarbeitung der Klasse `BinarySearchTree` und Einführung des Interface `Item` zur Realisierung einer geeigneten Ordnungsrelation d. Implementierung der Anwendung oder von Teilen der Anwendung inklusive einer sortierten		

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

Endliche Automaten und formale Sprachen (Unterrichtsvorhaben Q2-II)

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.

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: 21 Stunden

Tabelle 13. Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	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), • 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 zwi-	*Beispiel:* Cola-Automat, Geldspielautomat Roboter, Zustandsänderung eines Objekts „Auto“, Akzeptor für bestimmte Zahlen, Akzeptor für Teilwörter in längeren Zeichenketten, Akzeptor für Terme *Material:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.2 — Endliche Automaten, Formale Sprachen
2. Untersuchung und Entwicklung von Grammatiken regulärer Sprachen a. Erarbeitung der formalen Darstellung regulärer Grammatiken b. Untersuchung, Modifikation und Entwicklung von Grammatiken c. Entwicklung von endlichen Automaten zum Erkennen regulärer Sprachen die durch Grammatiken gegeben werden d. Entwicklung regulärer Grammatiken zu endlichen Automaten		*Beispiel:* Reguläre Grammatik für Wörter mit ungerader Anzahl eines Buchstabens, Satzgliederungsgrammatik, Grammatik für Wörter, die bestimmte Zahlen repräsentieren, Akzeptor und Grammatik für Sprache aller durch zwei, drei, vier, ... teilbare Zahlen *Beispiel:* Reguläre Ausdrücke in einem Texteditor, Darstellung der Verknüpfungsoperationen regulärer Ausdrücke als Grammatik/Automat
3. Grenzen endlicher Automaten		*Beispiel:* Klammerausdrücke, $a^n b^n$ im Vergleich zu $(ab)^n$

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
	schen Automaten und Grammatiken (D).	

Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit (Unterrichtsvorhaben Q2-III)

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

Tabelle 14. Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	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 *Material:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.3 — Von-Neumann-Architektur und maschinennahe Programmierung
2. Grenzen der Automatisierbarkeit a. Vorstellung des Halteproblems b. Unlösbarkeit des Halteproblems		*Beispiel:* Halteproblem *Material:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.3 — Halteproblem

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
c. Beurteilung des Einsatzes von Informatiksystemen hinsichtlich prinzipieller Möglichkeiten und prinzipieller Grenzen		

2.2. Grundsätze der fachmethodischen und fachdidaktischen Arbeit

Unter Berücksichtigung des Schulprogramms und des Sozialcurriculums hat die Fachkonferenz Informatik des Max-Planck-Gymnasiums die folgenden fachmethodischen und fachdidaktischen Grundsätze beschlossen. In diesem Zusammenhang beziehen sich die Grundsätze 1 bis 14 auf fächerübergreifende Aspekte, die Grundsätze 15 bis 20 sind fachspezifisch angelegt.

Überfachliche Grundsätze

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

Fachliche Grundsätze

15. Der Unterricht unterliegt der Wissenschaftsorientierung und ist dementsprechend eng verzahnt mit seiner Bezugswissenschaft.
16. Der Unterricht ist problemorientiert und soll von realen Problemen ausgehen und sich auf solche rückbeziehen.
17. Der Unterricht folgt dem Prinzip der Exemplarizität und soll ermöglichen, informatische Strukturen und Gesetzmäßigkeiten in den ausgewählten Problemen und Projekten zu erkennen.

18.Der Unterricht ist handlungsorientiert, d.h. projekt- und produktorientiert angelegt.

19.Im Unterricht werden sowohl für die Schule didaktisch reduzierte als auch reale Informatiksysteme aus der Wissenschafts-, Berufs- und Lebenswelt eingesetzt.

20.Der Unterricht beinhaltet reale Begegnung mit Informatiksystemen.

2.3. Grundsätze der Leistungsbewertung und Leistungsrückmeldung

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

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

2.3.1. Beurteilungsbereich Klausuren

Verbindliche Absprachen

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

Instrumente

- Einführungsphase: 1 Klausur je Halbjahr, Dauer der Klausur: 2 Unterrichtsstunden
- Grundkurse Q 1: 2 Klausuren je Halbjahr, Dauer der Klausuren: 2 Unterrichtsstunden
- Grundkurse Q 2.1: 2 Klausuren, Dauer der Klausuren: 3 Unterrichtsstunden
- Grundkurse Q 2.2: 1 Klausur unter Abiturbedingungen
- Anstelle einer Klausur kann gemäß dem Beschluss der Lehrerkonferenz in Q 1.2 eine Facharbeit geschrieben werden.

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

Kriterien

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

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

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

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

2.3.2. Beurteilungsbereich Sonstige Mitarbeit

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

Leistungsaspekte

Mündliche Leistungen

- Beteiligung am Unterrichtsgespräch
- Zusammenfassung 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

- Arbeitsmappen und Arbeistagebuch 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

Leistungsaspekte

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.

3. Entscheidungen zu fach- und unterrichts-übergreifenden 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 Kennt-

nisse mitbringen können. Diese können insbesondere bei der Auswahl und Bearbeitung von Softwareprojekten berücksichtigt werden und in einem hinsichtlich der informatischen Problemstellung angemessenem Maß in den Unterricht Eingang finden.

Vorbereitung auf die Erstellung der Facharbeit

Möglichst schon 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.

Exkursionen

In der Einführungsphase wird im Rahmen des Unterrichtsvorhabens „Geschichte der digitalen Datenverarbeitung und die Grundlagen des Datenschutzes“ eine Exkursion zum Heinz Nixdorf MuseumsForum durchgeführt. Die außerunterrichtliche Veranstaltung wird im Unterricht vor- und nachbereitet.

4. Qualitätssicherung und Evaluation

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

Das schulinterne Curriculum (siehe 2.1) 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.