



**KONRAD
HERESBACH
GYMNASIUM**



RESPEKT
PERSÖNLICHKEIT
ERFOLG
LERNEN IN SOZIALER
VERANTWORTUNG.
LERN- UND LEBENSORT KHG.

Schulinterner Lehrplan
Des Konrad-Heresbach-Gymnasiums
zum Kernlehrplan für die gymnasiale Oberstufe

Informatik - Sekundarstufe II

(Entwurf Stand: 15.04.2024)

Inhaltsverzeichnis

1 Die Fachgruppe Informatik	2
2 Entscheidungen zum Unterricht	4
2.1 Unterrichtsvorhaben	4
2.1.1 Übersichtsraster Unterrichtsvorhaben	6
I) Einführungsphase	6
2.1.2 Konkretisierte Unterrichtsvorhaben.....	9
I) Einführungsphase	14
2.2 Grundsätze der fachmethodischen und fachdidaktischen Arbeit.....	55
2.3 Grundsätze der Leistungsbewertung und Leistungsrückmeldung.....	56
I) Beurteilungsbereich Klausuren	56
II) Beurteilungsbereich Sonstige Mitarbeit	57
3 Qualitätssicherung und Evaluation	57
4 Querschnittsthema Verbraucherbildung	58

1 Die Fachgruppe Informatik

Das Konrad-Heresbach-Gymnasium ist eine drei- bis vierzügige Schule im Zentrum von Mettmann mit zurzeit ca. 730 Schülerinnen und Schülern. Oberstufenunterricht im Fach Informatik hat zuletzt vor ca. 25 Jahren am KHG stattgefunden.

Das Fach Informatik wird 2021/2022 letztmalig ab der Jahrgangsstufe 8 im Wahlpflichtbereich II (WP II) zweistündig unterrichtet. Ab dem Jahre 2022/23 ist es im Rahmen von G9 wieder dreistündig und beginnt in Klasse 9.

In der zweijährigen Laufzeit dieser Kurse wird in altersstufengerechter Weise unter anderem auf Grundlagen der Algorithmik, auf Robotik, die Erstellung von Homepages und Künstliche Intelligenz eingegangen.

In der Jahrgangsstufe 6 wird ab dem Jahrgang 2022/2023 Informatik zweistündig unterrichtet. In der Sekundarstufe II ist geplant, Informatik nur als Grundkurs einzurichten.

Alle Schülerinnen und Schüler erhalten im Rahmen des Profulfaches *Informationstechnische Grundbildung (ITG)* in der Klasse 5 und 8 eine Grundlagenbildung, deren Inhalte aber keine kerninformatischen Themen

sind, sondern dessen Fokus eher auf der Schulung und Nutzung von Software liegt.

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

Der Unterricht der Sekundarstufe II wird mit Hilfe der Programmiersprache Java durchgeführt. In der Einführungsphase kommt dabei eine didaktische Bibliothek zum Einsatz, welche das Erstellen von Benutzungsoberflächen und grafischen Programmen erleichtert.

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. Zurzeit besteht die Fachschaft Informatik des KHGs aus drei Lehrkräften (eine für die SII), denen drei Computerräume mit jeweils 17 Computerarbeitsplätzen 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 alle Arbeitsplätze der drei Räume zum Zugriff auf ihre eigenen Daten, zur Recherche im Internet oder zur Bearbeitung schulischer Aufgaben verwenden können.

Der Unterricht erfolgt im 45-Minuten-Takt. Die Kursblockung sieht grundsätzlich für Grundkurse eine Doppelstunde und eine Einzelstunde vor.

Präventiv wurde das Fach als Kooperationskurs mit dem Heinrich-Heine-Gymnasium gedacht, um es langfristig auch in der Q1 und Q2 weiterzuführen zu können und ggf. sogar als Abiturfach anzubieten.

Als Unterrichtswerk für die EF wurde gemeinsam mit dem Heinrich-Heine-Gymnasium das Buch *Informatik - Lehrwerk für die gymnasiale Oberstufe Ausgabe 2021* (ISBN 978-3-14-123370-4 für die EF und ISBN 9783141233780 für die Qualifizierungsphase) ausgewählt.

Die in den Büchern separat ausgewiesenen Fachmethoden bilden die Grundlage für die weitere Arbeit in der Oberstufe und werden nach und nach systematisch in den Unterricht integriert.

2 Entscheidungen zum Unterricht

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 deckt sämtliche im schulinternen Lehrplan im Kernlehrplan angeführten Kompetenzen ab. 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. Aufgrund der geringen Erfahrung mit einigen Vorhaben und der Umsetzung des neuen Lehrplans, muss eine gewisse Flexibilität in der Ablaufplanung Vorrang vor dem Beharren auf der festgelegten Sequenzierung haben. Um diesen Vorgang transparent zu gestalten, sollen Änderungen mit den Fachkollegen möglich sein und nach einer ersten Erprobung evaluiert werden. Das schulinterne Curriculum wird bei erfolgreicher Erprobung der Änderungen entsprechend angepasst.

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

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

2.1.1 Übersichtsraster Unterrichtsvorhaben

I) Einführungsphase

Einführungsphase	
<p><u>Unterrichtsvorhaben E-I</u></p> <p>Thema: <i>Einführung in die Nutzung von Informatiksystemen und in grundlegende Begrifflichkeiten</i></p> <p>Zentrale Kompetenzen:</p> <ul style="list-style-type: none">• Argumentieren• Darstellen und Interpretieren• Kommunizieren und Kooperieren <p>Inhaltsfelder:</p> <ul style="list-style-type: none">• Informatiksysteme• Informatik, Mensch und Gesellschaft <p>Inhaltliche Schwerpunkte:</p> <ul style="list-style-type: none">• Einzelrechner• Dateisystem• Internet• Einsatz von Informatiksystemen <p>Zeitbedarf: 6 Stunden</p>	<p><u>Unterrichtsvorhaben E-II</u></p> <p>Thema: <i>Grundlagen der objektorientierten Programmierung und algorithmischer Grundstrukturen in Java unter Benutzung eines Zeichenwerkzeugs.</i></p> <p>Zentrale Kompetenzen:</p> <ul style="list-style-type: none">• Argumentieren• Modellieren• Implementieren• Darstellen und Interpretieren• Kommunizieren und Kooperieren <p>Inhaltsfelder:</p> <ul style="list-style-type: none">• Daten und ihre Strukturierung• Algorithmen• Formale Sprachen und Automaten <p>Inhaltliche Schwerpunkte:</p> <ul style="list-style-type: none">• Objekte und Klassen• Syntax und Semantik einer Programmiersprache• Analyse, Entwurf und Implementierung einfacher Algorithmen <p>Zeitbedarf: 18 Stunden</p>

Einführungsphase

Unterrichtsvorhaben E-III

Thema:

Leben in einer digitalen Welt 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: 15 Stunden

Unterrichtsvorhaben E-IV

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 Implementierung einfacher Algorithmen

Zeitbedarf: 9 Stunden

Einführungsphase

Unterrichtsvorhaben E-V

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

Unterrichtsvorhaben E-VI

Thema:

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

Zentrale Kompetenzen:

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

Inhaltsfelder:

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

Inhaltliche Schwerpunkte:

- Objekte und Klassen
- Syntax und Semantik einer Programmiersprache
- Analyse, Entwurf und Implementierung einfacher Algorithmen

Zeitbedarf: 18 Stunden

Summe Einführungsphase: 74

Qualifikationsphase 1

Unterrichtsvorhaben Q1-I

Thema:

Wiederholung der objektorientierten Modellierung und Programmierung anhand einer kontextbezogenen Problemstellung

Zentrale Kompetenzen:

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

Inhaltsfelder:

- Daten und ihre Strukturierung
- Algorithmen
- Formale Sprachen und Automaten
- Informatiksysteme

Inhaltliche Schwerpunkte:

- Objekte und Klassen
- Analyse, Entwurf und Implementierung von Algorithmen
- Syntax und Semantik einer Programmiersprache
- Nutzung von Informatiksystemen

Zeitbedarf: 8 Stunden

Unterrichtsvorhaben Q1-II

Thema:

Modellierung und Implementierung von Anwendungen mit dynamischen, linearen Datenstrukturen

Zentrale Kompetenzen:

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

Inhaltsfelder:

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

Inhaltliche Schwerpunkte:

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

Zeitbedarf: 20 Stunden

Qualifikationsphase 1

Unterrichtsvorhaben Q1-III

Thema:

Suchen und Sortieren auf linearen Datenstrukturen

Zentrale Kompetenzen:

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

Inhaltsfelder:

- Algorithmen
- Formale Sprachen und Automaten

Inhaltliche Schwerpunkte:

- Analyse, Entwurf und Implementierung von Algorithmen
- Algorithmen in ausgewählten informatischen Kontexten
- Syntax und Semantik einer Programmiersprache

Zeitbedarf: 16 Stunden

Unterrichtsvorhaben Q1-IV

Thema:

Endliche Automaten und formale Sprachen

Zentrale Kompetenzen:

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

Inhaltsfelder:

- Endliche Automaten und formale Sprachen

Inhaltliche Schwerpunkte:

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

Zeitbedarf: 20 Stunden

Qualifikationsphase 1

Unterrichtsvorhaben Q1-V

Thema:

Sicherheit und Datenschutz in Netzstrukturen

Zentrale Kompetenzen:

- Argumentieren
- Darstellen und Interpretieren
- Kommunizieren und Kooperieren

Inhaltsfelder:

- Informatiksysteme
- Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:

- Einzelrechner und Rechnernetzwerke
- Sicherheit
- Nutzung von Informatiksystemen, Wirkungen der Automatisierung

Zeitbedarf: 10 Stunden

Summe Qualifikationsphase 1: 74 Stunden

Qualifikationsphase 2

Unterrichtsvorhaben Q2-I

Thema:

Modellierung und Implementierung von Anwendungen mit dynamischen, nichtlinearen Datenstrukturen

Zentrale Kompetenzen:

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

Inhaltsfelder:

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

Inhaltliche Schwerpunkte:

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

Zeitbedarf: 24 Stunden

Unterrichtsvorhaben Q2-II

Thema:

Modellierung und Nutzung von relationalen Datenbanken in Anwendungskontexten

Zentrale Kompetenzen:

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

Inhaltsfelder:

- Daten und ihre Strukturierung
- Algorithmen
- Formale Sprachen und Automaten
- Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:

- Datenbanken
- Algorithmen in ausgewählten informatischen Kontexten
- Syntax und Semantik einer Programmiersprache
- Sicherheit

Zeitbedarf: 20 Stunden

Qualifikationsphase 2

Unterrichtsvorhaben Q2-III

Thema:

*Prinzipielle Arbeitsweise eines Computers
und Grenzen der Automatisierbarkeit*

Zentrale Kompetenzen:

- Argumentieren
- Kommunizieren und Kooperieren

Inhaltsfelder:

- Informatiksysteme
- Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:

- Einzelrechner und Rechnernetzwerke
- Grenzen der Automatisierung

Zeitbedarf: 12 Stunden

Summe Qualifikationsphase 2: 56 Stunden

2.1.2 Konkretisierte Unterrichtsvorhaben

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

Hinweis:

Verbindliche Festlegungen der Fachkonferenz:

Themen, Leitfragen und die Ausführungen unter der Überschrift *Vorhabenbezogene Konkretisierung* sind verbindlich vereinbart, ebenso die Sequenzierung der Unterrichtsvorhaben (erste Tabellenspalte) und die ausgewiesenen Kompetenzen (dritte Tabellenspalte).

Unterrichtliche Anregungen:

Die angeführten Beispiele, Medien und Materialien sind Vorschläge für die Lehrkräfte. In diesen Bereichen sind Abweichungen von den vorgeschlagenen Vorgehensweisen möglich.

In der Einführungsphase wird die didaktische Bibliothek "Basis" verwendet. Dazu stehen Installationspakete und Dokumentationen zur Verfügung.

I) Einführungsphase

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

Die Schülerinnen und Schüler

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

Unterrichtsvorhaben EF-I

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

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

Vorhabenbezogene Konkretisierung:

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

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

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

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

Zeitbedarf: 6 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Beispiele, Medien, Materialien	Zu entwickelnde Kompetenzen
<p>1. Information, deren Kodierung und Speicherung</p> <p>(a) Informatik als Wissenschaft der Verarbeitung von Informationen</p> <p>(b) Darstellung von Informationen</p> <p>(c) Speichern von Daten mit informatischen Systemen am Beispiel der Schulrechner</p>	<p><i>Beispiel:</i> Textkodierung Kodierung und Dekodierung von Texten mit unbekanntem Zeichensätzen (z.B. Wingdings)</p> <p><i>Beispiel:</i> Bildkodierung Kodierung von Bildinformationen in Raster- und Vektorgrafiken</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • beschreiben und erläutern den Aufbau und die Arbeitsweise von Rechnern 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).
<p>2. Informations- und Daten-übermittlung in Netzen</p>		
<p>(a) Informatische Kommunikation in Rechnernetzen am Beispiel des Schulnetzwerks (z.B. Benutzeranmeldung, Netzwerkordner, Zugriffsrechte, Client-Server)</p> <p>(b) Grundlagen der technischen Umsetzung von Rechnerkommunikation am Beispiel des Internets (z.B. Netzwerkadresse, Paketvermittlung, Routing)</p> <p>(c) Richtlinien zum verantwortungsvollen Umgang mit dem Internet</p>	<p><i>Beispiel:</i> Rollenspiel zur Paketvermittlung im Internet</p> <p>Schülerinnen und Schüler übernehmen die Rollen von Clients und Routern. Sie schicken spielerisch Informationen auf Karten von einem Schüler-Client zum anderen. Jede Schülerin und jeder Schüler hat eine Adresse, jeder Router darüber hinaus eine Routingtabelle. Mit Hilfe der Tabelle und einem Würfel wird entschieden, wie ein Paket weiter vermittelt wird.</p>	

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“

Beispiel

Durch Demontage eines Demonstrationsrechners lernen Schülerinnen und Schüler die verschiedenen Hardwarekomponenten eines Informatiksystems kennen.

Unterrichtsvorhaben EF-II

Thema: Grundlagen der objektorientierten Programmierung und algorithmischer Grundstrukturen in Java unter Benutzung eines Zeichenwerkzeugs.

Leitfragen: *Wie lassen sich graphische Szenen mit Hilfe einer „Turtle“ bzw. eines „Igels“ darstellen? Wie kommt man von statischen Darstellungen zu Animationen?*

Vorhabenbezogene Konkretisierung:

Der Schwerpunkt dieses Unterrichtsvorhabens liegt auf der Konzeption und Durchführung mehrerer Projekte. Zunächst werden Projekte bearbeitet, die die grafische Darstellung einfacher Gegenstände mit Hilfe der Turtle beinhalten. In diesen Projekten werden im Wesentlichen Objekte der Klassen Fenster und Stift benutzt. Zur Darstellung von komplexeren Szenen werden Methoden zur Strukturierung entworfen und implementiert.

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.

Zunächst wird bei der Umsetzung der ersten Projekte konsequent auf die Verwendung von Kontrollstrukturen verzichtet, da der Quellcode aus einer rein linearen Sequenz besteht. So ist eine Fokussierung auf die Grundlagen der Objektorientierung möglich, ohne dass algorithmische Probleme ablenken.

Die Darstellung von Animationen führt in nachfolgenden Projekten zur Einführung von Schleifen und Verzweigungen.

Die algorithmischen Konzepte sollen an weiteren Beispielen eingeübt werden. Dabei muss es sich nicht zwangsläufig um solche handeln, bei denen Kontrollstrukturen lediglich zur Animation verwendet werden.

In einem abschließenden Projekt soll ein Malprogramm mit einem GUI erstellt werden. In diesem Kontext werden einerseits die bekannten Kontrollstrukturen verwendet und es erfolgt andererseits eine Einarbeitung in die Klassen der Basisbibliothek.

Zeitbedarf: 18 Stunden

Hinweis:

Das eingeführte Schulbuch bietet an, dass die algorithmischen Grundstrukturen mit Hilfe der Software-Umgebung *greenfoot* und den sogenannten Robinski-Szenarien erarbeitet werden können. Da dies vor allem Schülerinnen und Schülern ohne Erfahrung mit Informatik entgegenkommt, kann dieses Modul ggf. vorgeschaltet werden, um den Fokus weniger auf Implementierung zu richten. Es ist auch durchaus möglich, die längere Zeit in der EF mit dieser Umgebung zu arbeiten, insbesondere wenn es um die systematische Erarbeitung informatischer Strukturen geht.

Die Umsetzung eigener kleiner Projekte ist aber essenziell und erhält im Laufe des Schuljahres einen zunehmend größeren Anteil.

Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Beispiele, Medien, Materialien	Zu entwickelnde Kompetenzen
<p>1. Darstellungen grafischer Szenen</p> <p>(a) Darstellung eines Gegenstandes mit Hilfe des Stiftes und relativer sowie absoluter Stiftbewegungen</p> <p>(b) Erstellung von Methoden, gegebenenfalls mit Parametern zur Zeichnung eines Gegenstandes</p>	<p><i>Beispiel: Setzkasten</i></p> <p><i>Beispiel: Landschaft</i></p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • analysieren und er-läutern einfache Algorithmen und Programme (A), • entwerfen einfache Algorithmen und stellen sie umgangssprachlich und graphisch dar (M), • ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • ordnen Parametern von Methoden einfache Datentypen (M), • modifizieren einfache Algorithmen und Programme (I), • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I), • implementieren Algorithmen unter Verwendung von Kontrollstrukturen sowie Methodenaufrufen (I), • implementieren einfache Algorithmen unter Beachtung der Syntax und Semantik einer Programmiersprache (I),
<p>2. Animierte Grafiken</p> <p>(a) Animation durch fortgesetztes Zeichnen und Löschen (While-Schleife)</p> <p>(b) Abfrage eines Knopfes zur Realisierung einer Schleifenbedingung für eine Animationsschleife</p> <p>(c) Gegebenenfalls mehrstufige Animationen mit mehreren sequenziellen Schleifen</p>	<p><i>Beispiel: Hampelmann</i></p> <p><i>Beispiel: Trickfilm</i></p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • analysieren und er-läutern einfache Algorithmen und Programme (A), • entwerfen einfache Algorithmen und stellen sie umgangssprachlich und graphisch dar (M), • ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • ordnen Parametern von Methoden einfache Datentypen (M), • modifizieren einfache Algorithmen und Programme (I), • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I), • implementieren Algorithmen unter Verwendung von Kontrollstrukturen sowie Methodenaufrufen (I), • implementieren einfache Algorithmen unter Beachtung der Syntax und Semantik einer Programmiersprache (I),

<p>3. Interaktives Zeichnen</p> <p>(a) Malen durch fortgesetzte Bewegung des Stiftes zur Mausposition</p> <p>(b) Gestaltung von Benutzungsoberflächen</p> <p>(c) Verwendung weiterer Basisklassen</p> <p>(d) Reaktion auf Benutzereingaben</p>	<p><i>Beispiel:</i> Malprogramm</p>	<ul style="list-style-type: none">• testen Programme schrittweise anhand von Beispielen (I),• interpretieren Fehlermeldungen und korrigieren den Quellcode (I).
---	---	--

Unterrichtsvorhaben EF-III

Thema: Grundlagen des Datenschutzes / Grundlagen der Codierung und Geschichte der digitalen Datenverarbeitung

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

Vorhabenbezogene Konkretisierung:

Schülerinnen und Schüler sollen informatische Themenbereiche aus dem Kontext der Geschichte der Datenverarbeitung und insbesondere den daraus sich ergebenden Fragen des Datenschutzes bearbeiten. Diese Themenbereiche können in Kleingruppen bearbeitet und in Form von Plakatpräsentationen vorgestellt werden. Die Cäsarcodierung soll im Unterricht vertieft behandelt werden und in Teilen auch implementiert werden. In diesem Zusammenhang werden Variablen einfacher Datentypen und des Datentyps String eingeführt. Als Kontrollstruktur wird die Zählschleife behandelt.

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: 18 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Beispiele, Medien, Materialien	Zu entwickelnde Kompetenzen
<p>1. Erarbeitung von Themen durch die Schülerinnen und Schüler</p> <p>(a) Mögliche Themen</p> <ul style="list-style-type: none"> • „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“ <p>(b) Vorstellung und Diskussion durch Schülerinnen und Schüler</p>	<p><i>Beispiel:</i> Ausstellung zu informatischen Themen</p> <p>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.</p> <p><i>Materialien:</i> Schülerinnen und Schüler recherchieren selbstständig im Internet, in der Schulbibliothek, in öffentlichen Bibliotheken, usw.</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • 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). • implementieren Algorithmen unter Verwendung von Variablen und Wertzuweisungen, Kontrollstrukturen sowie Methodenaufrufen (I),
<p>2. Cäsarverschlüsselung</p> <p>(a) Erläuterung des ASCII-Codes</p> <p>(b) Variable für einfache Datentypen und String</p> <p>(c) Umwandlungsoperationen für int nach char und umgekehrt</p> <p>(d) Stringoperationen zur Konkatenation und Zeichenzugriff</p> <p>(e) Zählschleife zur sequentiellen Codierung eines Strings</p>	<p><i>Beispiel:</i> Codierung/Decodierung von Texten</p> <p><i>Beispiel:</i> Häufigkeitszählung</p> <p><i>Beispiel:</i> Palindrome</p> <p><i>Beispiel:</i> Drei Chinesen</p>	<ul style="list-style-type: none"> • implementieren Algorithmen unter Verwendung von Variablen und Wertzuweisungen, Kontrollstrukturen sowie Methodenaufrufen (I),

<p>3. Vertiefung des Themas Datenschutz</p> <p>(a) Erarbeitung grundlegender Begriffe des Datenschutzes</p> <p>(b) Problematisierung und Anknüpfung an die Lebenswelt der Schülerinnen und Schüler</p> <p>(c) Diskussion und Bewertung von Fallbeispielen aus dem Themenbereich „Datenschutz“</p>	<p><i>Beispiel:</i> Fallbeispiele aus dem aktuellen Tagesgeschehen</p> <p>Die Schülerinnen und Schüler bearbeiten Fallbeispiele aus ihrer eigenen Erfahrungswelt oder der aktuellen Medienberichterstattung.</p> <p><i>Materialien:</i> Materialblatt zum Bundesdatenschutzgesetz (<u>Download EF-VI.1</u>)</p>	
--	---	--

Unterrichtsvorhaben EF-IV

Thema: Such- und Sortieralgorithmen anhand kontextbezogener Beispiele

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

Vorhabenbezogene Konkretisierung:

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

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

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

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

Zeitbedarf: 9 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Beispiele, Medien, Materialien	Zu entwickelnde Kompetenzen
<p>1 Explorative Erarbeitung eines Sortierverfahrens</p> <p>(a) Sortierprobleme im Kontext informatischer Systeme und im Alltag (z.B. Dateisortierung, Tabellenkalkulation, Telefonbuch, Bundesligatabelle, usw.)</p> <p>(b) Vergleich zweier Elemente als Grundlage eines Sortieralgorithmus</p> <p>(c) Erarbeitung eines Sortieralgorithmus durch die Schülerinnen und Schüler</p>	<p><i>Beispiel:</i> Sortieren mit Waage</p> <p>Die Schülerinnen und Schüler bekommen die Aufgabe, kleine, optisch identische Kunststoffbehälter aufsteigend nach ihrem Gewicht zu sortieren. Dazu steht ihnen eine Balkenwaage zur Verfügung, mit deren Hilfe sie das Gewicht zweier Behälter vergleichen können.</p> <p><i>Materialien:</i> Computerscience unplugged – Sorting Algorithms, URL: https://classic.csunplugged.org/activities/sorting-algorithms/ (aufgerufen 12.06.2022)</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • beurteilen die Effizienz von Algorithmen am Beispiel von Sortierverfahren hinsichtlich Zeit und Speicherplatzbedarf (A), • entwerfen einen weiteren Algorithmus zum Sortieren (M), • analysieren Such- und Sortieralgorithmen und wenden sie auf Beispiele an (D).
<p>2. Systematisierung von Algorithmen und Effizienzbetrachtungen</p> <p>(a) Formulierung (falls selbst gefunden) oder Erläuterung von mehreren Algorithmen im Pseudocode (auf jeden Fall: Sortieren durch Vertauschen, Sortieren durch Auswählen)</p> <p>(b) Anwendung von Sortieralgorithmen auf verschiedene Beispiele</p> <p>(c) Bewertung von Algorithmen anhand der Anzahl der nötigen Vergleiche</p> <p>(d) Variante des Sortierens durch Auswählen (Nutzung eines einzigen oder zweier Felder bzw.</p>	<p><i>Beispiele:</i> Sortieren durch Auswählen, Sortieren durch Vertauschen, Quicksort</p> <p>Quicksort ist als Beispiel für einen Algorithmus nach dem Prinzip <i>Teile und Herrsche</i> gut zu behandeln. Kenntnisse in rekursiver Programmierung sind nicht erforderlich, da eine Implementierung nicht angestrebt wird.</p> <p><i>Materialien:</i> Computerscience unplugged – Sorting Algorithms, URL: https://classic.csunplugged.org/activities/sorting-algorithms/ (aufgerufen 12.06.2022)</p>	

<p>lediglich eines einzigen zusätzlichen Ablageplatzes oder mehrerer neuer Ablageplätze)</p> <p>(e) Effizienzbetrachtungen an einem konkreten Beispiel bezüglich der Rechenzeit und des Speicherplatzbedarfs</p> <p>(f) Analyse des weiteren Sortieralgorithmus (sofern nicht in Sequenz 1 und 2 bereits geschehen)</p>		
<p>3. Binäre Suche auf sortierten Daten</p> <p>(a) Suchaufgaben im Alltag und im Kontext informatischer Systeme</p> <p>(b) Evtl. Simulationsspiel zum effizienten Suchen mit binärer Suche</p> <p>(c) Effizienzbetrachtungen zur binären Suche</p>	<p><i>Beispiel:</i> 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.</p> <p><i>Materialien:</i> Computerscience unplugged – Sorting Algorithms, URL: https://classic.csunplugged.org/activities/sorting-algorithms/ (aufgerufen 12.06.2022)</p>	

Unterrichtsvorhaben EF-V

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 der Objektorientierten Programmierung ist die Modellierung von Klassen. 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 wird verstärkt Gebrauch von Modellierungswerkzeugen wie Objektkarten, Klassenkarten oder Beziehungsdiagramme gemacht.

Zeitbedarf: 9 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Beispiele, Medien, Materialien	Zu entwickelnde Kompetenzen
<p>1. Identifikation von Objekten</p> <p>(a) Am Beispiel eines lebensweltnahen Beispiels werden Objekte im Sinne der Objektorientierten Modellierung eingeführt.</p> <p>(b) Objekte werden mit Objektkarten visualisiert und mit sinnvollen Attributen und „Fähigkeiten“, d.h. Methoden versehen.</p> <p>(c) Manche Objekte sind prinzipiell typgleich und werden so zu einer Objektsorte bzw. Klasse zusammengefasst.</p> <p>(d) Vertiefung: Modellierung weiterer Beispiele ähnlichen Musters</p> <p>(e) Objektorientierte Programmierung als modularisiertes Vorgehen (Entwicklung von Problemlösungen auf Grundlage vorhandener Klassen)</p>	<p><i>Beispiel:</i> Autos Schülerinnen und Schüler betrachten Autos als Menge gleichartiger Objekte, die in einer Klasse mit Attributen und Methoden zusammengefasst werden können.</p> <p><i>Beispiel:</i> <i>Tannenbaumschmuck</i></p> <p><i>Beispiel:</i> <i>Bäume</i></p> <p><i>Materialien:</i> Ergänzungsmaterialien zum Lehrplannavigator - Allgemeine Objektorientierung (Download EF-II.1)</p> <p><i>Materialien:</i> Ergänzungsmaterialien zum Lehrplannavigator - Sequenzielle Programmierung (Download EF-II.3)</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • modellieren Klassen mit ihren Attributen und ihren Methoden (M), • implementieren einfache Algorithmen unter Beachtung der Syntax und Semantik einer Programmiersprache (I),
<p>2. Implementierung von Klassen</p> <p>(a) Grundaufbau einer Java-Klasse</p> <p>(b) Deklaration und Initialisierung von Objekten</p> <p>Methodenaufrufe mit Parameterübergabe zur Manipulation von Objekteigenschaften</p> <p>(z.B. Farbe, Position, Drehung)</p>		<ul style="list-style-type: none"> • stellen den Zustand eines Objekts dar (D).

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

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

Zeitbedarf: 8 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Wiederholung und Erweiterung der objektorientierten Modellierung und Programmierung durch Analyse und Erweiterung eines kontextbezogenen Beispiels</p> <p>(a) Analyse der Problemstellung</p> <p>(b) Analyse der Modellierung (Implementationsdiagramm)</p> <p>(c) Erweiterung der Modellierung im Implementationsdiagramm (Vererbung, abstrakte Klasse)</p> <p>(d) Kommunikation zwischen mindestens zwei Objekten (grafische Darstellung)</p> <p>(e) Dokumentation von Klassen</p> <p>(f) Implementierung der Anwendung oder von Teilen der Anwendung</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • analysieren und erläutern objektorientierte Modellierungen (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M), • ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M), • modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M), • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), 	<p>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.</p> <p>oder</p> <p>Beispiel: Coolwatch Die Firma Coolwatch hat eine Armbanduhrenkollektion Watch Coollection in der Art eines Baukastensystems entwickelt: Eine Armbanduhr besteht aus einer Grundeinheit mit der Uhrelektronik an die verschiedene Anzeigeeinheiten und Bänder angesteckt werden können. Ein Kunde kann auf diese Weise seine Uhr unterschiedlichen Anlässen anpassen, ohne jeweils eine neue Uhr kaufen zu müssen. Die Anzeigeeinheit wird auf die Grundeinheit aufgesteckt und von dort mit Strom versorgt.</p>

- 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).
-

Bereits zur Einführungsphase gehören zur Watch Coollection analoge Anzeigeeinheiten mit Ziffernblatt und Zeigern und solche mit digitaler dezimaler und binärer Anzeige in unterschiedlichen Farben und Designs.

Im Rahmen einer Simulation sollen Uhren aus den genannten Bestandteilen zusammengestellt werden können. Zu diesem Zweck sind geeignete Klassen zu entwickeln und zu implementieren.

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
<p>1. Die Datenstruktur Schlange im Anwendungskontext unter Nutzung der Klasse Queue</p> <p>(a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen</p> <p>(b) Erarbeitung der Funktionalität der Klasse Queue</p> <p>(c) Modellierung und Implementierung der Anwendung unter Verwendung eines oder mehrerer Objekte der Klasse Queue</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (A), • analysieren und erläutern Algorithmen und Programme (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M), • ermitteln bei der Analyse von Problemstellungen Objekte, 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), 	<p><i>Beispiel:</i> Patientenwarteschlange (jeder kennt seinen Nachfolger bzw. alternativ: seinen Vorgänger)</p> <p>Sobald ein Patient in einer Arztpraxis eintrifft, werden sein Name und seine Krankenkasse erfasst. Die Verwaltung der Patientenwarteschlange geschieht über eine Klasse, die hier als Wartezimmer bezeichnet wird. Wesentliche Operationen sind das „Hinzufügen“ eines Patienten und das „Entfernen“ eines Patienten, wenn er zur Behandlung gerufen wird. 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.</p> <p>Anschließend wird der Funktionsumfang der Anwendung erweitert: Patienten können sich zusätzlich in die Warteschlange zum Blutdruckmessen einreihen. Objekte werden von zwei Schlangen verwaltet.</p>
<p>2. Die Datenstruktur Stapel im Anwendungskontext unter Nutzung der Klasse Stack</p> <p>(a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen</p>	<p>(I),</p>	<p><i>Beispiel:</i> Heftstapel</p> <p>In einem Heftstapel soll das Heft einer Schülerin gefunden werden.</p> <p>oder</p>

<p>(b) Erarbeitung der Funktionalität der Klasse Stack</p> <p>(c) Modellierung und Implementierung der Anwendung unter Verwendung eines oder mehrerer Objekte der Klasse Stack</p>	<ul style="list-style-type: none"> • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • testen Programme systematisch anhand von Beispielen (I), stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D). 	<p><i>Beispiel:</i> Kisten stapeln</p> <p>In einem Stapel nummerierter Kisten soll eine bestimmte Kiste gefunden und an einen Kunden geliefert werden. Dazu müssen Kisten auf verschiedene Stapel gestapelt und wieder zurückgestellt werden.</p>
<p>3. Die Datenstruktur lineare Liste im Anwendungskontext unter Nutzung der Klasse List</p> <p>(a) Erarbeitung der Vorteile der Klasse List im Gegensatz zu den bereits bekannten linearen Strukturen</p> <p>(b) Modellierung und Implementierung einer kontextbezogenen Anwendung unter Verwendung der Klasse List.</p>		<p><i>Beispiel:</i> Abfahrtslauf</p> <p>Bei einem Abfahrtslauf kommen die Skifahrer nacheinander an und werden nach ihrer Zeit in eine Rangliste eingeordnet. Diese Rangliste wird in einer Anzeige ausgegeben. Ankommende Abfahrer müssen an jeder Stelle der Struktur, nicht nur am Ende oder Anfang eingefügt werden können.</p> <p><i>Materialien:</i> Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.2 - Listen (Download Q1-II.2)</p>
<p>4. Vertiefung - Anwendungen von Listen, Stapeln oder Schlangen in mindestens einem weiteren Kontext</p>		<p><i>Beispiel:</i> Skispringen</p> <p>Ein Skispringen hat folgenden Ablauf: Nach dem Sprung erhält der Springer eine Punktzahl und wird nach dieser Punktzahl in eine Rangliste eingeordnet. Die besten 30 Springer qualifizieren sich für den zweiten Durchgang. Sie starten in umgekehrter Reihenfolge gegenüber der Platzierung auf der Rangliste. Nach dem Sprung erhält der Springer wiederum eine Punktzahl und wird nach der Gesamtpunktzahl aus beiden Durchgängen in die endgültige Rangliste eingeordnet.</p> <p><i>Beispiel:</i> Terme in Postfix-Notation</p>

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

Unterrichtsvorhaben Q1-III:

Thema: Suchen und Sortieren auf linearen Datenstrukturen

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

Vorhabenbezogene Konkretisierung:

In einem Anwendungskontext werden zunächst Informationen in einer linearen Liste bzw. einem Feld gesucht. Hierzu werden Verfahren entwickelt und implementiert bzw. analysiert und erläutert, wobei neben einem iterativen auch ein rekursives Verfahren thematisiert wird und mindestens ein Verfahren selbst entwickelt und implementiert wird. Die verschiedenen Verfahren werden hinsichtlich Speicherbedarfs und Zahl der Vergleichsoperationen miteinander verglichen.

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

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

Zeitbedarf: 16 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Suchen von Daten in Listen und Arrays</p> <p>(a) Lineare Suche in Listen und in Arrays</p> <p>(b) Binäre Suche in Arrays als Beispiel für rekursives Problemlösen</p> <p>(c) Untersuchung der beiden Suchverfahren hinsichtlich ihrer Effizienz (Laufzeitverhalten, Speicherbedarf)</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • analysieren und erläutern Algorithmen und Programme (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • beurteilen die Effizienz von Algorithmen unter Berücksichtigung des Speicherbedarfs und der Zahl der Operationen (A), • entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“ und „Teilen und Herrschen“ (M), • modifizieren Algorithmen und Programme (I), • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I), 	<p><i>Beispiel:</i> Karteiverwaltung</p> <p>Für ein Adressverwaltungsprogramm soll eine Methode zum Suchen einer Adresse geschrieben werden.</p> <p>oder</p> <p><i>Beispiel:</i> Bundesjugendspiele</p> <p>Die Teilnehmer an Bundesjugendspielen nehmen an drei Disziplinen teil und erreichen dort Punktzahlen. Diese werden in einer Wettkampfkarte eingetragen und an das Wettkampfbüro gegeben. Zur Vereinfachung sollte sich das Modell auf die drei Disziplinen „Lauf“, „Sprung“ und „Wurf“ beschränken.</p> <p>Im Wettkampfbüro wird das Ergebnis erstellt. Das Programm soll dafür zunächst den Besten einer Disziplin herausuchen können und später das gesamte Ergebnis nach gewissen Kriterien sortieren können.</p>

<p>2. Sortieren in Listen und Arrays - Entwicklung und Implementierung von iterativen und rekursiven Sortierverfahren</p> <p>(a) Entwicklung und Implementierung eines einfachen Sortierverfahrens für eine Liste</p> <p>(b) Implementierung eines einfachen Sortierverfahrens für ein Feld Entwicklung eines rekursiven Sortierverfahrens für ein Feld (z.B. Sortieren durch Mischen)</p>	<ul style="list-style-type: none"> • implementieren und erläutern iterative und rekursive Such- und Sortierverfahren (I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • testen Programme systematisch anhand von Beispielen (I), • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D). 	<p><i>Beispiel:</i> Karteiverwaltung (s.o.)</p> <p>oder</p> <p><i>Beispiel:</i> Bundesjugendspiele (s.o.)</p> <p><i>Materialien:</i> (s.o.)</p>
<p>3. Untersuchung der Effizienz der Sortierverfahren „Sortieren durch direktes Einfügen“ und „Quicksort“ auf linearen Listen</p> <p>(a) Grafische Veranschaulichung der Sortierverfahren</p> <p>(b) Untersuchung der Anzahl der Vergleichsoperationen und des Speicherbedarfs bei beiden Sortierverfahren</p> <p>(c) Beurteilung der Effizienz der beiden Sortierverfahren</p>		<p><i>Beispiel:</i> Karteiverwaltung (s.o.)</p> <p>oder</p> <p><i>Beispiel:</i> Bundesjugendspiele (s.o.)</p> <p><i>Materialien:</i> (s.o.)</p>

Unterrichtsvorhaben Q1-IV:

Thema: Endliche Automaten und formale Sprachen

Leitfragen: *Wie kann man (endliche) Automaten genau beschreiben? Wie können endliche Automaten (in alltäglichen Kontexten oder zu informatischen Problemstellungen) modelliert werden? Wie können Sprachen durch Grammatiken beschrieben werden? Welche Zusammenhänge gibt es zwischen formalen Sprachen, endlichen Automaten und regulären Grammatiken?*

Vorhabenbezogene Konkretisierung:

Anhand kontextbezogener Beispiele werden endliche Automaten entwickelt, untersucht und modifiziert. Dabei werden verschiedene Darstellungsformen für endliche Automaten ineinander überführt und die akzeptierten Sprachen endlicher Automaten ermittelt. An einem Beispiel wird ein nichtdeterministischer Akzeptor eingeführt als Alternative gegenüber einem entsprechenden deterministischen Akzeptor.

Anhand kontextbezogener Beispiele werden Grammatiken regulärer Sprachen entwickelt, untersucht und modifiziert. Der Zusammenhang zwischen regulären Grammatiken und endlichen Automaten wird verdeutlicht durch die Entwicklung von allgemeinen Verfahren zur Erstellung einer regulären Grammatik für die Sprache eines gegebenen endlichen Automaten bzw. zur Entwicklung eines endlichen Automaten, der genau die Sprache einer gegebenen regulären Grammatik akzeptiert.

Auch andere Grammatiken werden untersucht, entwickelt oder modifiziert. An einem Beispiel werden die Grenzen endlicher Automaten ausgelotet.

Zeitbedarf: 20 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien oder Materialien
<p>1. Endliche Automaten</p> <p>(a) Vom Automaten in den Schülerinnen und Schülern bekannten Kontexten zur formalen Beschreibung eines endlichen Automaten</p> <p>(b) Untersuchung, Darstellung und Entwicklung endlicher Automaten</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • analysieren und erläutern die Eigenschaften endlicher Automaten einschließlich ihres Verhaltens auf bestimmte Eingaben (A), • analysieren und erläutern Grammatiken regulärer Sprachen (A), • zeigen die Grenzen endlicher Automaten und regulärer Grammatiken im Anwendungszusammenhang auf (A), • ermitteln die formale Sprache, die durch eine Grammatik erzeugt wird (A), • entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M), • entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M), • entwickeln zur akzeptierten Sprache eines Automaten die zugehörige Grammatik (M), • entwickeln zur Grammatik einer regulären Sprache einen zugehörigen endlichen Automaten (M), • modifizieren Grammatiken regulärer Sprachen (M), • entwickeln zu einer regulären Sprache eine Grammatik, die die Sprache erzeugt (M), • stellen endliche Automaten in Tabellen oder Graphen dar und überführen sie in die jeweils andere Darstellungsform (D), <ul style="list-style-type: none"> • ermitteln die Sprache, die ein endlicher Automat akzeptiert (D). • beschreiben an Beispielen den Zusammenhang zwischen Automaten und Grammatiken (D). 	<p><i>Beispiele:</i> Cola-Automat, Geldspielautomat, Roboter, Zustandsänderung eines Objekts „Auto“, Akzeptor für bestimmte Zahlen, Akzeptor für Teilwörter in längeren Zeichenketten, Akzeptor für Terme</p> <p><i>Materialien:</i> Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.2 – Endliche Automaten, Formale Sprachen (Download Q2-II.1)</p>
<p>2. Untersuchung und Entwicklung von Grammatiken regulärer Sprachen</p> <p>(a) Erarbeitung der formalen Darstellung regulärer Grammatiken</p> <p>(b) Untersuchung, Modifikation und Entwicklung von Grammatiken</p> <p>(c) Entwicklung von endlichen Automaten zum Erkennen regulärer Sprachen die durch Grammatiken gegeben werden</p> <p>(d) Entwicklung regulärer Grammatiken zu endlichen Automaten</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • analysieren und erläutern die Eigenschaften endlicher Automaten einschließlich ihres Verhaltens auf bestimmte Eingaben (A), • analysieren und erläutern Grammatiken regulärer Sprachen (A), • zeigen die Grenzen endlicher Automaten und regulärer Grammatiken im Anwendungszusammenhang auf (A), • ermitteln die formale Sprache, die durch eine Grammatik erzeugt wird (A), • entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M), • entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M), • entwickeln zur akzeptierten Sprache eines Automaten die zugehörige Grammatik (M), • entwickeln zur Grammatik einer regulären Sprache einen zugehörigen endlichen Automaten (M), • modifizieren Grammatiken regulärer Sprachen (M), • entwickeln zu einer regulären Sprache eine Grammatik, die die Sprache erzeugt (M), • stellen endliche Automaten in Tabellen oder Graphen dar und überführen sie in die jeweils andere Darstellungsform (D), <ul style="list-style-type: none"> • ermitteln die Sprache, die ein endlicher Automat akzeptiert (D). • beschreiben an Beispielen den Zusammenhang zwischen Automaten und Grammatiken (D). 	<p><i>Beispiele:</i> reguläre Grammatik für Wörter mit ungerader Parität, Grammatik für Wörter, die bestimmte Zahlen repräsentieren, Satzgliedergrammatik</p> <p><i>Materialien: (s.o.)</i></p>
<p>3. Grenzen endlicher Automaten</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • analysieren und erläutern die Eigenschaften endlicher Automaten einschließlich ihres Verhaltens auf bestimmte Eingaben (A), • analysieren und erläutern Grammatiken regulärer Sprachen (A), • zeigen die Grenzen endlicher Automaten und regulärer Grammatiken im Anwendungszusammenhang auf (A), • ermitteln die formale Sprache, die durch eine Grammatik erzeugt wird (A), • entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M), • entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M), • entwickeln zur akzeptierten Sprache eines Automaten die zugehörige Grammatik (M), • entwickeln zur Grammatik einer regulären Sprache einen zugehörigen endlichen Automaten (M), • modifizieren Grammatiken regulärer Sprachen (M), • entwickeln zu einer regulären Sprache eine Grammatik, die die Sprache erzeugt (M), • stellen endliche Automaten in Tabellen oder Graphen dar und überführen sie in die jeweils andere Darstellungsform (D), <ul style="list-style-type: none"> • ermitteln die Sprache, die ein endlicher Automat akzeptiert (D). • beschreiben an Beispielen den Zusammenhang zwischen Automaten und Grammatiken (D). 	<p><i>Beispiele:</i> Klammerausdrücke, $a^n b^n$ im Vergleich zu $(ab)^n$</p>

Unterrichtsvorhaben Q1-V:

Thema: Sicherheit und Datenschutz in Netzstrukturen

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

Vorhabenbezogene Konkretisierung:

Anschließend an das vorhergehende Unterrichtsvorhaben zum Thema Datenbanken werden der Datenbankzugriff aus dem Netz, Topologien von Netzwerken, eine Client-Server-Struktur, das TCP/IP-Schichtenmodell sowie Sicherheitsaspekte beim Zugriff auf Datenbanken und verschiedene symmetrische und asymmetrische kryptografische Verfahren analysiert und erläutert. Fallbeispiele zur Datenschutzproblematik und zum Urheberrecht runden das Unterrichtsvorhaben ab.

Zeitbedarf: 10 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Daten in Netzwerken und Sicherheitsaspekte in Netzen sowie beim Zugriff auf Datenbanken	Die Schülerinnen und Schüler	
<p>(a) Beschreibung eines Datenbankzugriffs im Netz anhand eines Anwendungskontextes und einer Client-Server-Struktur zur Klärung der Funktionsweise eines Datenbankzugriffs</p> <p>(b) Netztopologien als Grundlage von Client-Server-Strukturen und TCP/IP-Schichtenmodell als Beispiel für eine Paketübermittlung in einem Netz</p> <p>(c) Vertraulichkeit, Integrität, Authentizität in Netzwerken sowie symmetrische und asymmetrische kryptografische Verfahren (Cäsar-, Vigenère-, RSA-Verfahren) als Methoden Daten im Netz verschlüsselt zu übertragen</p>	<ul style="list-style-type: none"> • beschreiben und erläutern Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken (A), • analysieren und erläutern Eigenschaften und Einsatzbereiche symmetrischer und asymmetrischer Verschlüsselungsverfahren (A), • untersuchen und bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen, die Sicherheit von Informatiksystemen sowie die Einhaltung der Datenschutzbestimmungen und des Urheberrechts (A), • untersuchen und bewerten Problemlagen, die sich aus dem Einsatz von Informatiksystemen ergeben, hinsichtlich rechtlicher Vorgaben, ethischer Aspekte und gesellschaftlicher Werte unter Berücksichtigung unterschiedlicher Interessenlagen (A), • nutzen bereitgestellte Informatiksysteme und das Internet reflektiert zum Erschließen, zur Aufbereitung und Präsentation fachlicher Inhalte (D). 	<p><i>Materialien:</i> Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben, Verschlüsselung Q1.5 - Zugriff auf Daten in Netzwerken (Download Q1-V.1)</p>
2. Fallbeispiele zur Datenschutzproblematik und zum Urheberrecht	<ul style="list-style-type: none"> • nutzen bereitgestellte Informatiksysteme und das Internet reflektiert zum Erschließen, zur Aufbereitung und Präsentation fachlicher Inhalte (D). 	<p><i>Materialien:</i> Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1 5 - Datenschutz beim Videocenter, Materialblatt-Datenschutzgesetz (Download Q1-V.2)</p>

Unterrichtsvorhaben Q2-I:

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

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

Vorhabenbezogene Konkretisierung:

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

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

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

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

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

Zeitbedarf: 24 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Analyse von Baumstrukturen in verschiedenen Kontexten</p> <p>(a) Grundlegende Begriffe (Grad, Tiefe, Höhe, Blatt, Inhalt, Teilbaum, Ebene, Vollständigkeit)</p> <p>(b) Aufbau und Darstellung von binären Bäumen anhand von Baumstrukturen in verschiedenen Kontexten</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (A), • analysieren und erläutern Algorithmen und Programme (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M), • modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M), • verwenden bei der Modellierung geeigneter Problemstellungen die Möglichkeiten der Polymorphie (M), • entwickeln iterative und rekursive Algorithmen unter Nutzung der Konstruktionsstrategien 	<p><i>Beispiel:</i> Ternbaum Der Aufbau von Termen wird mit Hilfe von binären Baumstrukturen verdeutlicht.</p> <p>oder</p> <p><i>Beispiel:</i> Ahnenbaum Die binäre Baumstruktur ergibt sich daraus, dass jede Person genau einen Vater und eine Mutter hat.</p> <p><i>Weitere Beispiele für Anwendungskontexte für binäre Bäume:</i></p> <p><i>Beispiel:</i> Suchbäume (zur sortierten Speicherung von Daten) Alle Inhalte, die nach einer Ordnung vor dem Inhalt im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Inhalt im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.)</p> <p>oder</p> <p><i>Beispiel:</i> Entscheidungsbäume Um eine Entscheidung zu treffen, werden mehrere Fragen mit ja oder nein beantwortet. Die Fragen, die möglich sind, wenn die Antwort auf eine Frage mit „ja“ beantwortet wird, befinden sich im linken Teilbaum, die Fragen, die möglich</p>

	<p>„Modularisierung“ und „Teilen und Herrschen“ (M),</p> <ul style="list-style-type: none">• 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).	<p>sind, wenn die Antwort „nein“ lautet, stehen im rechten Teilbaum.</p> <p><i>oder</i></p> <p><i>Beispiel:</i> Codierungsbäume für Codierungen, deren Alphabet aus genau zwei Zeichen besteht Morse hat Buchstaben als Folge von Punkten und Strichen codiert. Diese Codierungen können in einem Binärbaum dargestellt werden, so dass ein Übergang zum linken Teilbaum einem Punkt und ein Übergang zum rechten Teilbaum einem Strich entspricht. Wenn man im Gesamtbaum startet und durch Übergänge zu linken oder rechten Teilbäumen einen Pfad zum gewünschten Buchstaben sucht, erhält man die Morsecodierung des Buchstabens.</p> <p><i>Materialien:</i> Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 – Binärbaum (Download Q2-I.1)</p>
--	--	--

2. Die Datenstruktur Binärbaum im Anwendungskontext unter Nutzung der Klasse BinaryTree

- (a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen im Anwendungskontext
- (b) Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramms
- (c) Erarbeitung der Klasse `BinaryTree` und beispielhafte Anwendung der Operationen
- (d) Implementierung der Anwendung oder von Teilen der Anwendung
- (e) Traversierung eines Binärbaums im Pre-, In- und Post-Orderdurchlauf

Beispiel: Informatikerbaum als binärer Baum

In einem *binären Baum* werden die Namen und die Geburtsdaten von Informatikern lexikographisch geordnet abgespeichert. Alle Namen, die nach dieser Ordnung vor dem Namen im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Namen im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.)

Folgende Funktionalitäten werden benötigt:

- Einfügen der Informatiker-Daten in den Baum
- Suchen nach einem Informatiker über den Schlüssel Name
- Ausgabe des kompletten Datenbestands in nach Namen sortierter Reihenfolge

Materialien:

Ergänzungsmaterialien zum Lehrplannavigators
Unterrichtsvorhaben Q2.1 – Binärbaum
([Download Q2-I.2](#))

3. Die Datenstruktur binärer Suchbaum im Anwendungskontext unter Verwendung der Klasse BinarySearchTree

- (a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen
- (b) Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramm, grafische Darstellung eines binären Suchbaums und Erarbeitung der Struktureigenschaften
- (c) Erarbeitung der Klasse BinarySearchTree und Einführung des Interface Item zur Realisierung einer geeigneten Ordnungsrelation
- (d) Implementierung der Anwendung oder von Teilen der Anwendung inklusive einer sortierten Ausgabe des Baums

Beispiel: Informatikerbaum als *Suchbaum*

In einem binären *Suchbaum* werden die Namen und die Geburtsdaten von Informatikern lexikographisch geordnet abgespeichert. Alle Namen, die nach dieser Ordnung vor dem Namen im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Namen im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.)

Folgende Funktionalitäten werden benötigt:
Einfügen der Informatiker-Daten in den Baum
Suchen nach einem Informatiker über den Schlüssel Name
Ausgabe des kompletten Datenbestands in nach Namen sortierter Reihenfolge

Materialien:

Ergänzungsmaterialien zum Lehrplannavigator
Unterrichtsvorhaben Q2.1 – Binärer Suchbaum
([Download Q2-I.3](#))

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: Ternbaum (s.o.)

oder

Beispiel: Ahnenbaum (s.o.)

Materialien:

Ergänzungsmaterialien zum Lehrplannavigator
Unterrichtsvorhaben Q2.1 – Anwendung
Binärbaum
([Download Q2-I.4](#))

Unterrichtsvorhaben Q2-II:

Thema: Modellierung und Nutzung von relationalen Datenbanken in Anwendungskontexten

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

Vorhabenbezogene Konkretisierung:

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

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

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

An einem Beispiel wird verdeutlicht, dass in Datenbanken Redundanzen unerwünscht sind und Konsistenz gewährleistet sein sollte.

Die 1. bis 3. Normalform wird als Gütekriterium für Datenbankentwürfe eingeführt.

Datenbankschemata werden hinsichtlich der 1. bis 3. Normalform untersucht und (soweit nötig) normalisiert.

Zeitbedarf: 20 Stunden

Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Nutzung von relationalen Datenbanken</p> <p>(a) Aufbau von Datenbanken und Grundbegriffe</p> <ul style="list-style-type: none"> • Entwicklung von Fragestellungen zur vorhandenen Datenbank • Analyse der Struktur der vorgegebenen Datenbank und Erarbeitung der Begriffe Tabelle, Attribut, Datensatz, Datentyp, Primärschlüssel, Fremdschlüssel, Datenbankschema <p>(b) SQL-Abfragen</p> <ul style="list-style-type: none"> • Analyse vorgegebener SQL-Abfragen und Erarbeitung der Sprachelemente von SQL (SELECT (DISTINCT) ...FROM, WHERE, AND, OR, NOT) auf einer Tabelle • Analyse und Erarbeitung von SQL-Abfragen auf einer und mehreren Tabellen zur Beantwortung der Fragestellungen (JOIN, UNION, AS, GROUP BY, ORDER BY, ASC, DESC, COUNT, MAX, MIN, SUM, Arithmetische Operatoren: +, -, *, /, (...), Vergleichsoperatoren: =, <>, >, <, >=, <=, LIKE, BETWEEN, IN, IS NULL) <p>(c) Vertiefung an einem weiteren Datenbankbeispiel</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • erläutern die Eigenschaften und den Aufbau von Datenbanksystemen unter dem Aspekt der sicheren Nutzung (A), • analysieren und erläutern die Syntax und Semantik einer Datenbankabfrage (A), • analysieren und erläutern eine Datenbankmodellierung (A), • erläutern die Eigenschaften normalisierter Datenbankschemata (A), • bestimmen Primär- und Sekundärschlüssel (M), • ermitteln für anwendungsbezogene Problemstellungen Entitäten, zugehörige Attribute, Relationen und Kardinalitäten (M), • modifizieren eine Datenbankmodellierung (M), • modellieren zu einem Entity-Relationship-Diagramm ein relationales Datenbankschema (M), • bestimmen Primär- und Sekundärschlüssel (M), • überführen Datenbankschemata in vorgegebene Normalformen (M), • 	<p><i>Beispiel:</i> VideoCenter</p> <p>VideoCenter ist die Simulation einer OnlineVideothek für den Informatik-Unterricht mit Webfrontends zur Verwaltung der Kunden, der Videos und der Ausleihe. Außerdem ist es möglich direkt SQL-Abfragen einzugeben. Es ist auch möglich, die Datenbank herunter zu laden und lokal zu installieren. Unter http://dokumentation.videocenter.schule.de/old/video/index.html (abgerufen: 31.08.2023) findet man den Link zu dem VideoCenter-System sowie nähere Informationen. Lesenswert ist auch die dort verlinkte „Dokumentation der Fallstudie“ mit didaktischem Material, welches alternativ bzw. ergänzend zu der im Folgenden beschriebenen Durchführung verwendet werden kann.</p>

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-RelationshipDiagramms
- Erläuterung und Modifizierung einer Datenbankmodellierung

(b) Entwicklung einer Datenbank aus einem Datenbankentwurf

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

(c) Redundanz, Konsistenz und Normalformen

- Untersuchung einer Datenbank hinsichtlich Konsistenz und Redundanz in einer Anwendungssituation
- Überprüfung von Datenbankschemata hinsichtlich der 1. bis 3. Normalform und Normalisierung (um Redundanzen zu vermeiden und Konsistenz zu gewährleisten)

- verwenden die Syntax und Semantik einer Datenbankabfragesprache, um Informationen aus einem Datenbanksystem zu extrahieren (I),
- ermitteln Ergebnisse von Datenbankabfragen über mehrere verknüpfte Tabellen (D), stellen Entitäten mit ihren Attributen und die Beziehungen zwischen Entitäten in einem Entity-Relationship-Diagramm grafisch dar
- (D), überprüfen Datenbankschemata auf vorgegebene Normalisierungseigenschaften (D).

Beispiel: Fahrradverleih

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

Beispiel: Reederei

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

Beispiel: Buchungssystem

In dem Online-Buchungssystem einer Schule können die Lehrer Medienräume, Beamer, Laptops, Kameras, usw. für einen bestimmten Zeitpunkt buchen, der durch Datum und die Schulstunde festgelegt ist.

Dazu ist die Datenbank zu modellieren, ggf. zu normalisieren und im Datenbanksystem umzusetzen. Weiter sollen sinnvolle Abfragen entwickelt werden.

Unterrichtsvorhaben Q2-III:

Thema: Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit

Leitfragen: *Was sind die strukturellen Hauptbestandteile eines Computers und wie kann man sich die Ausführung eines maschinenahen Programms mit diesen Komponenten vorstellen? Welche Möglichkeiten bieten Informatiksysteme und wo liegen ihre Grenzen?*

Vorhabenbezogene Konkretisierung:

Anhand einer von-Neumann-Architektur und einem maschinennahen Programm wird die prinzipielle Arbeitsweise von Computern verdeutlicht.

Ausgehend von den prinzipiellen Grenzen endlicher Automaten liegt die Frage nach den Grenzen von Computern bzw. nach Grenzen der Automatisierbarkeit nahe. Mit Hilfe einer entsprechenden Java-Methode wird plausibel, dass es unmöglich ist, ein Informatiksystem zu entwickeln, das für jedes beliebige Computerprogramm und jede beliebige Eingabe entscheidet, ob das Programm mit der Eingabe terminiert oder nicht (Halteproblem). Anschließend werden Vor- und Nachteile der Grenzen der Automatisierbarkeit angesprochen und der Einsatz von Informatiksystemen hinsichtlich prinzipieller Möglichkeiten und prinzipieller Grenzen beurteilt.

Zeitbedarf: 12 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien oder Materialien
1. Von-Neumann-Architektur und die Ausführung maschinennaher Programme	Die Schülerinnen und Schüler	
a) prinzipieller Aufbau einer von Neumann-Architektur mit CPU, Rechenwerk, Steuerwerk, Register und Hauptspeicher b) einige maschinennahe Befehlen und ihre Repräsentation in einem BinärCode, der in einem Register gespeichert werden kann c) Analyse und Erläuterung der Funktionsweise eines einfachen maschinennahen Programms	<ul style="list-style-type: none"> • erläutern die Ausführung eines einfachen maschinennahen Programms sowie die Datenspeicherung auf einer „Von-Neumann-Architektur“ (A), • untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen (A). 	<p><i>Beispiel:</i> Addition von 4 zu einer eingegebenen Zahl mit einem Rechnermodell</p> <p><i>Materialien:</i> Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.3 –Von-Neumann-Architektur und maschinennahe Programmierung (Download Q2-III.1)</p>
2. Grenzen der Automatisierbarkeit a) Vorstellung des Halteproblems b) Unlösbarkeit des Halteproblems c) Beurteilung des Einsatzes von Informatiksystemen hinsichtlich prinzipieller Möglichkeiten und prinzipieller Grenzen		<p><i>Beispiel:</i> Halteproblem</p> <p><i>Materialien:</i> Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.3 - Halteproblem (Download Q2-III.2)</p>

Unterrichtsvorhaben Q2-IV:

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

2.2 Grundsätze der fachmethodischen und fachdidaktischen Arbeit

In Absprache mit der Lehrerkonferenz sowie unter Berücksichtigung des Schulprogramms hat die Fachkonferenz Informatik des Konrad-Heresbach-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:

- 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 Exemplarität und soll ermöglichen, informatische Strukturen und Gesetzmäßigkeiten in den ausgewählten Problemen und Projekten zu erkennen.
- 18) Der Unterricht ist anschaulich sowie gegenwarts- und zukunftsorientiert und gewinnt dadurch für die Schülerinnen und Schüler an Bedeutsamkeit.
- 19) Der Unterricht ist handlungsorientiert, d.h. projekt- und produktorientiert angelegt.
- 20) Im Unterricht werden sowohl für die Schule didaktisch reduzierte als auch reale Informatiksysteme aus der Wissenschafts-, Berufs- und Lebenswelt eingesetzt.

21) Der Unterricht beinhaltet reale Begegnung mit Informatiksystemen.

2.3 Grundsätze der Leistungsbewertung und Leistungsrückmeldung

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 Konrad-Heresbach-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.

I) 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 | 2 Unterrichtsstunden |
| • Q1: | 2 Klausuren je Halbjahr | 2 Unterrichtsstunden |
| • Q2: | 2 Klausuren je Halbjahr | 3 Unterrichtsstunden |

Kriterien

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

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

II) Beurteilungsbereich Sonstige Mitarbeit

Die Kriterien zur Beurteilung der sonstigen Mitarbeit unterscheiden sich nicht von denen anderer Fächer und werden am KHG einheitlich gehandhabt. Man findet sie unter [diesem Link \(zuletzt aufgerufen am 31.08.2023\)](#).

3 Qualitätssicherung und Evaluation

Das schulinterne Curriculum stellt keine starre Größe dar, sondern ist als „lebendes Dokument“ zu betrachten. Dementsprechend sind die Inhalte stetig zu überprüfen, um ggf. Modifikationen vornehmen zu können. Die Fachkonferenz (als professionelle Lerngemeinschaft) trägt durch diesen Prozess zur Qualitätsentwicklung und damit zur Qualitätssicherung des Faches bei.

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 2023 für den ersten Durchgang der EF durch die gymnasiale Oberstufe nach Erlass des Kernlehrplanes verbindlich. Jahresweise wird die Fachkonferenz Informatik auf der Grundlage ihrer Unterrichtserfahrungen eine Gesamtsicht des schulinternen Curriculums vornehmen und ggf. eine Beschlussvorlage für die erste Fachkonferenz des folgenden Schuljahres erstellen.

4 Querschnittsthema Verbraucherbildung

Der Bereich Medien und Informationen in der digitalen Welt greift Themenstellungen u.a. Datenschutz, Persönlichkeitsrecht, Urheberrechte, Nutzungsrechte, Werbung und Algorithmen auf und beleuchtet diese aus unterschiedlichen Perspektiven. Bildung in der digitalen Welt hat somit auch die Aufgabe, Medienhandeln und Medieninhalte kritisch zu hinterfragen: Asymmetrien zwischen Anbieter und Nutzer digitaler Medien und Inhalte können thematisiert sowie Möglichkeiten und Grenzen eigenen Handelns erkennbar gemacht werden. Einsatz und Nutzen sowie Gefahren und Unzulänglichkeiten in dem komplexen Bereich der Technologieentwicklung und ihrer Folgen sollen daher multiperspektivisch beleuchtet werden: als Konsumentin oder Konsument, Bürgerin oder Bürger, Unternehmerin oder Unternehmer, als Privatperson oder politisch Aktive(r) etc. In diesem Zusammenhang können auch die rollenvariablen Beeinflussungsmöglichkeiten und Handlungs- sowie Verhaltensoptionen berücksichtigt werden, die angesichts einer hauptsächlich medial gestützten Informationsbeschaffung und -weitergabe zu beachten sind

Kooperationsnetze wie Schule – Wirtschaft oder Schule und Arbeitswelt ermöglichen Schulen Projekte in verbraucherrelevanten Bereichen, die im Sinne einer systematischen und dauerhaften Lernpartnerschaft zwischen einer Schule und einem Unternehmen in fachspezifischen schulinternen Lehrplänen verankert werden.

Quelle: https://www.schulentwicklung.nrw.de/lehrplaene/upload/klp_gs/vb/Rahmenvorgabe_Verbraucherbildung_PS_SI_2017.pdf

Die Integration von Zielen und Inhaltsreichen der Rahmenvorgabe Verbraucherbildung in der Sek II geschieht durch folgende konkretisierte Kompetenzerwartungen

Schülerinnen und Schüler

- *erläutern an ausgewählten Beispielen Auswirkungen des Einsatzes von Informatiksystemen auf ihre Lebens- und Erfahrungswelt (A/KK), (VB C Z5),*
- *beschreiben anhand von ausgewählten Beispielen die Verarbeitung und Nutzung personenbezogener Daten (DI), (VB C Z5),*
- *erläutern anhand von Beispielen aus ihrer Lebenswelt Nutzen und Risiken beim Umgang mit eigenen und fremden Daten auch im Hinblick auf Speicherorte (A), (VB C Z3) und*
- *beschreiben Maßnahmen zum Schutz von Daten mithilfe von Informatiksystemen (A). (VB C Z2)*