Gymnasium der Stadt Geseke
ANTONIANUM
seit 1687

Sekundarstufe II

Schulinterner Lehrplan für das Fach Informatik

Inhalt

1. Rahmenbedingungen der fachlichen Arbeit

Das Antonianum ist mit ca. 1200 Schülerinnen und Schülern das einzige Gymnasium in Geseke. Im Bereich der Sekundarstufe II kooperiert die Schule mit dem Gymnasium in Erwitte und bietet zahlreiche gemeinsame Kurse an, auch Leistungskurse in Informatik. Das Einzugsgebiet der Schule umfasst die Stadt Geseke und u.a. große Teile der Stadt Salzkotten.

Vorbereitend auf das Fach Informatik werden am Antonianum bereits in der Jahrgangsstufe 6 die Schülerinnen und Schüler im Fach ‚Informatische Bildung' (IB) verpflichtend im Umgang mit Computern und den an der Schule eingerichteten informationstechnischen Hilfsmitteln (Lernplattform Moodle, Pädagogisches Netzwerk) vertraut gemacht. Das Angebot im Wahlpflichtbereich II (Informatik) wird in der Regel von etwa der Hälfte aller Schülerinnen und Schüler genutzt. In der Sekundarstufe II werden entsprechend der Schülerwünsche regelmäßig mehrere Grundkurse und ein Leistungskurs eingerichtet.

Für den Unterricht können vier große Computerräume mit jeweils 15+1 Arbeitsplätzen und ein kleiner Gruppenraum mit 8 Arbeitsplätzen genutzt werden. Die Eigenarbeit der Schülerinnen und Schüler kann im Selbstlernzentrum an 8 Rechnern erfolgen. Die Recherche im Internet ist für Schülerinnen und Schüler an fünf Rechnern in der Eingangshalle möglich. Alle diese Rechner sind im pädagogischen Netzwerk an einen zentralen Server (LogoDidact) angeschlossen. Dieser Server ermöglicht über eine individuelle, passwortgeschützte Anmeldung den gefilterten Zugriff auf das Internet und das eigene Homeverzeichnis in der Schule aber auch von Computern zu Hause aus. Die Softwareverwaltung erfolgt zentral für alle Rechner gleich. Wenn möglich wird Freeware Software genutzt, die auch auf privaten Rechnern kostenlos installiert werden darf.

Für die Programmierung in der Sek II wird die Programmiersprache Java genutzt. Der Einstieg in der Einführungsphase erfolgt über die Entwicklungsumgebung Greenfoot, danach wird weitgehend mit BlueJ gearbeitet. Da in den Kursen der Mittelstufe mit Scratch programmiert wurde, ist sichergestellt, dass alle Schülerinnen und Schüler, die das Fach neu wählen, von Grund auf beginnen können.

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. Dadurch ist auch ein hohes Maß an innerer Differenzierung gewährleistet.
Die gemeinsame Entwicklung von Materialien und Unterrichtsvorhaben, die Evaluation von Lehr- und Lernprozessen sowie die stetige Überprüfung und eventuelle Modifikation des schulinternen Curriculums durch die Fachkonferenz Informatik stellt einen wichtigen Beitrag zur Qualitätssicherung und -entwicklung des Unterrichts dar.
Insgesamt werden überwiegend kooperative, die Selbstständigkeit des Lerners fördernde Unterrichtsformen genutzt, sodass ein individualisiertes Lernen in der Sekundarstufe II kontinuierlich unterstützt wird.

Die individuelle Förderung wird durch vielfältige Zusatzangebote unterstützt: Regelmäßig werden Angebote der Universität Paderborn und des Schülerlabors coolMINT genutzt. Die Kooperation mit

dem ZDI und dem Unternehmen Ferber Software, beide in Lippstadt, ermöglicht die vertiefende Beschäftigung mit den Inhalten des Faches nicht zuletzt in anwendungsnahen, betriebsbegleiteten Facharbeits- und Praktikumsangeboten. Die Wartung der technischen Komponenten erfolgt weitgehend durch Lehrer und Schüler der Schule und bietet somit interessierten Schülerinnen und Schülern auch die Möglichkeit praktische Erfahrungen zu sammeln.

2. Entscheidungen zum Unterricht

Dieser schulinterne Lehrplan hat den Anspruch, alle im Kernlehrplan geforderten Kompetenzen auf wenige Unterrichtsvorhaben abzubilden.

Die Umsetzung des Lehrplans folgt pro Unterrichtsvorhaben jeweils auf einer Übersichts- und einer dazugehörigen Konkretisierungsebene. Die Übersichtsebene insbesondere die Reihenfolge und die daraus resultierende Zuordnung zu den Halbjahren ist für die Kolleginnen und Kollegen verpflichtend, die anschließend dargestellten Konkretisierungen sind Vorschläge zur Umsetzung mit dem eingeführten Lehrwerk **Informatik 1** (EF) und **Informatik 2** (Q1/Q2) aus dem Schöningh-Verlag. Sie lassen sich anhand der Lehrtexte, Projekte und Aufgaben des Buches umfassend unterrichten. An einigen Stellen müssen die Inhalte des Buches Informatik 1 mit externen Materialien ergänzt werden, da die eingeführte Auflage des Buches noch nicht den inhaltlichen Anforderungen des Kernlehrplans entspricht.

Didaktische Lernumgebung

Zur Einführung in die objektorientierte Programmierung wird die didaktische Lernumgebung Greenfoot gewählt. Die Lehrtexte und Aufgaben des Lehrwerks „Informatik 1“ beziehen sich speziell auf das Greenfoot-Roboter-Szenario.

Nach der Einarbeitungsphase wird mit dem Wechsel zu BlueJ auf eine erweiterte didaktische Lernumgebung umgestellt. In der Q-Phase kann parallel die Entwicklungsumgebung NetBeans verwendet werden. Im LK wird überwiegend mit NetBeans gearbeitet.

Hausaufgaben

Als Ergänzung zu den Vorgaben des Hausaufgabenkonzeptes am Antonianum wird im Fach Informatik erwartet, dass die Schülerinnen und Schüler entweder zu Hause die in der Schule genutzten Programmierumgebungen auf eigenen Rechnern installieren oder ggf. die technischen Angebote der Schule nutzen. Zum Erwerb der Kompetenzen im Implementieren müssen die Schülerinnen und Schüler im Rahmen der Hausaufgaben eigenständige Programmieraufgaben bearbeiten.

2.1. Unterrichtsvorhaben

2.1.1 Übersichtsraster Unterrichtsvorhaben

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

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

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

Inhaltsfelder:
- Informatiksysteme
- Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:
- Einsatz, Nutzung und Aufbau von Informatik-systemen
- Wirkung der Automatisierung

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

Thema:
Algorithmische Grundstrukturen in Java

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

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

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

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

Thema:
Such- und Sortieralgorithmen

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

Inhaltsfelder:
- Algorithmen
- Daten und ihre Strukturierung

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

Zeitbedarf: 9 Stunden</td><td><u>Unterrichtsvorhaben E-IV</u>

Thema:
Grundlagen der objektorientierten Analyse, Modellierung und Implementierung

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

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

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

Zeitbedarf: 8 Stunden</td></tr>
</table>

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

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

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

Inhaltsfelder:
- Informatiksysteme
- Informatik, Mensch und Gesellschaft

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

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

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

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

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

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

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

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

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

Inhaltsfelder:
- Informatiksysteme
- Informatik, Mensch und Gesellschaft

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

Zeitbedarf: 10 Stunden</td><td>Summe Einführungsphase:
77 Stunden</td></tr>
</table>

Übersichtsraster Unterrichtsvorhaben in der Qualifikationsphase - Q1

<table>
<tr><th colspan="2">Qualifikationsphase – Q1 – GK</th></tr>
<tr><td><u>Unterrichtsvorhaben Q1-I</u>

Thema: Wiederholung und Vertiefung der objektorientierten Modellierung

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

Inhaltsfelder:
- Daten und ihre Strukturierung
- Algorithmen
- Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:
- Objekte und Klassen
- Wirkung der Automatisierung

Zeitbedarf: 14 Stunden</td><td><u>Unterrichtsvorhaben Q1-II</u>

Thema: Organisation und Verarbeitung von Daten I – Modellierung und Implementierung von Anwendungen mit dynamischen und linearen Datenstrukturen

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

Inhaltsfelder:
- Daten und ihre Strukturierung
- Algorithmen

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

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

Thema: Algorithmen zum Suchen und Sortieren auf linearen Datenstrukturen

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

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: 20 Stunden</td><td><u>Unterrichtsvorhaben Q1-IV</u>

Thema: Aufbau von und Kommunikation in Netzwerken

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: 16 Stunden</td></tr>
<tr><td colspan="2">Summe Qualifikationsphase 1: 70 Stunden</td></tr>
</table>

Übersichtsraster Unterrichtsvorhaben in der Qualifikationsphase – Q2

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

Thema: Automaten und formale Sprachen

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

Inhaltsfelder:
- Formale Sprachen und Automaten
- Informatiksysteme

Inhaltliche Schwerpunkte:
- Syntax und Semantik einer Programmiersprache
- Endliche Automaten
- Grammatiken regulärer Sprachen
- Möglichkeiten und Grenzen von Automaten und formalen Sprachen

Zeitbedarf: 20 Stunden</td><td><u>Unterrichtsvorhaben Q2-I</u>

Thema: Organisation und Verarbeitung von Daten II – Modellierung und Implementierung von Anwendungen mit dynamischen nicht-linearen Datenstrukturen

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

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

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

Zeitbedarf: 20 Stunden</td></tr>
<tr><td><u>Unterrichtsvorhaben Q2-III</u>

Thema: Nutzung und Modellierung von relationalen Datenbanken in Anwendungskontexten

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

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

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

Zeitbedarf: 20 Stunden</td><td></td></tr>
<tr><td colspan="2">Summe Qualifikationsphase 2: 60 Stunden</td></tr>
</table>

<table>
<tr><th colspan="2">Qualifikationsphase – Q1 – LK</th></tr>
<tr><td><u>Unterrichtsvorhaben Q1-I</u>

Thema: Wiederholung und Vertiefung der objektorientierten Modellierung

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

Inhaltsfelder:
- Daten und ihre Strukturierung
- Algorithmen
- Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:
- Objekte und Klassen
- Wirkung der Automatisierung

Zeitbedarf: 25 Stunden</td><td><u>Unterrichtsvorhaben Q1-II</u>

Thema: Organisation und Verarbeitung von Daten I – Modellierung und Implementierung von Anwendungen mit dynamischen und linearen Datenstrukturen

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

Inhaltsfelder:
- Daten und ihre Strukturierung
- Algorithmen

Inhaltliche Schwerpunkte:
- Objekte und Klassen
- Syntax und Semantik einer Programmiersprache
- Analyse, Entwurf und Implementierung von Algorithmen
- Algorithmen in ausgewählten informatischen Kontexten
-
Zeitbedarf: 30 Stunden</td></tr>
<tr><td><u>Unterrichtsvorhaben Q1-III</u>

Thema: Algorithmen zum Suchen und Sortieren auf linearen Datenstrukturen

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

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: 25 Stunden</td><td><u>Unterrichtsvorhaben Q1-IV</u>

Thema: Aufbau von und Kommunikation in Netzwerken und Modellierung und Implementierung von Client-Server-Anwendungen

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: 24 Stunden</td></tr>
<tr><td colspan="2">Summe Qualifikationsphase 1: 100 Stunden</td></tr>
</table>

Qualifikationsphase – Q2 – LK

Unterrichtsvorhaben Q2-I

Thema: Organisation und Verarbeitung von Daten II – Modellierung und Implementierung von Anwendungen mit dynamischen nicht-linearen Datenstrukturen

Zentrale Kompetenzen:
- Argumentieren
- Darstellen und Interpretieren
- Modellieren
- Implementieren
- 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: 25 Stunden

Unterrichtsvorhaben Q2-II

Thema: Modellierung und Implementierung dynamischer nichtlinearer Datenstrukturen am Beispiel der Graphen

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

Inhaltsfelder:
- Daten und ihre Strukturierung
- Algorithmen
- Formale Sprachen und Automaten
- Informatiksysteme
- Kommunizieren und Kooperieren

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

Zeitbedarf: 25 Stunden

Unterrichtsvorhaben Q2-III

Thema: Nutzung und Modellierung 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: 25 Stunden

Unterrichtsvorhaben Q2-IV

Thema: Automaten und formale Sprachen

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

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

Inhaltliche Schwerpunkte:
- Syntax und Semantik einer Programmiersprache
- Endliche Automaten
- Grammatiken regulärer Sprachen
- Möglichkeiten und Grenzen von Automaten und formalen Sprachen
- Scanner, Parser und Interpreter für reg. Sprachen
- Kellerautomaten
- Kontextfreie Sprachen, Grammatiken

Zeitbedarf: 35 Stunden

Summe Qualifikationsphase 1: 110 Stunden

2.1.2 Mögliche konkretisierte Unterrichtsvorhaben

2.1.2.1 Unterrichtsvorhaben EF-I

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

Leitfragen: Was macht Informatik? Welche fundamentalen Konzepte müssen Informatikerinnen und Informatiker in ihre Arbeit einbeziehen, damit informatische Systeme effizient und zuverlässig arbeiten können? Wo lassen sich diese Konzepte (in Ansätzen) in dem schuleigenen Netzwerk- und Computersystem wiederfinden?

Vorhabenbezogene Konkretisierung:
Im ersten Unterrichtsvorhaben werden die fünf Inhaltsfelder des Faches Informatik beispielhaft an einem Informatiksystem erarbeitet. Das Unterrichtsvorhaben ist so strukturiert, dass die Schülerinnen und Schüler anhand bekannter Alltagstechnik die Grundideen fundamentaler informatischer Konzepte (Inhaltsfelder) größtenteils selbstständig erarbeiten und nachvollziehen.

Ausgehend von dem bekannten Bedienungs- und Funktionalitätswissen eines Navigationsgerätes können die Strukturierung von Daten, das Prinzip der Algorithmik, die Eigenheit formaler Sprachen, die Kommunikationsfähigkeit von Informatiksystemen und die positiven und negativen Auswirkungen auf Mensch und Gesellschaft thematisiert werden. Das so erworbene Wissen kann dann auf weitere den Schülerinnen und Schülern bekannte Informatiksysteme übertragen werden.

Zeitbedarf: 6 Stunden

Sequenzierung des Unterrichtsvorhabens: nächste Seite

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Kapitel und Materialien
1. Informatiksysteme und ihr genereller Aufbau (a) Daten und ihre Strukturierung (b) Algorithmen (c) Formale Sprachen und Automaten (d) Informatiksysteme (e) Informatik, Mensch und Gesellschaft	Die Schülerinnen und Schüler - bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen (A) - nutzen die im Unterricht eingesetzten Informatiksysteme selbstständig, sicher, zielführend und verantwortungsbewusst (D)	***Kapitel 1 „Was ist Informatik“*** Als Anschauungsmaterial bieten sich Navigationsgeräte an
2. Der kompetente Umgang mit dem Schulnetzwerk (a) Erstellen und Anlegen von Ordnerstrukturen (b) Sortieren von Dateien und Ordnern (c) Eingabe von Befehlen über die Eingabeaufforderung (d) Einzelrechner und Netzwerk (e) Sicherheit und Datenschutz		***Kapitel 1 „Was ist Informatik“*** Interview mit Netzwerkadministratoren, Benutzer- und Datenschutzbestimmungen der Schule

2.1.2.2 Unterrichtsvorhaben EF-II

Thema: Algorithmische Grundstrukturen in Java

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

Vorhabenbezogene Konkretisierung:

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

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

Zeitbedarf: 18 Stunden

Sequenzierung des Unterrichtsvorhabens: nächste Seite

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Kapitel und Materialien
1. Algorithmen (a) Wiederholungen (While-Schleifen) (b) bedingte Anweisungen (c) Verknüpfung von Bedingungen durch die logischen Funktionen UND, ODER und NICHT (d) Systematisierung des Vorgehens zur Entwicklung von Algorithmen zur Lösung komplexerer Probleme	Die Schülerinnen und Schüler - analysieren und erläutern einfache Algorithmen und Programme (A), - entwerfen einfache Algorithmen und stellen sie umgangssprachlich und grafisch dar (M), - ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen zu (M), - modifizieren einfache Algorithmen und Programme (I), - implementieren Algorithmen unter Verwendung von Variablen und Wertzuweisungen, Kontrollstrukturen sowie Methodenaufrufen (I), - implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I), - implementieren einfache Algorithmen unter Beachtung der Syntax und Semantik einer Programmiersprache (I), - testen Programme schrittweise anhand von Beispielen (I), - interpretieren Fehlermeldungen und korrigieren den Quellcode (I).	***Kapitel 2 Algorithmen*** • Wiederholungen • Bedingte Anweisungen • Logische Funktionen • Algorithmen entwickeln
2. Variablen und Methoden (a) Implementierung eigener Methoden mit lokalen Variablen, auch zur Realisierung einer Zählschleife (b) Implementierung eigener Methoden mit Parameterübergabe und/oder Rückgabewert (c) Implementierung von Konstruktoren (d) Realisierung von Attributen		***Kapitel 3 Methoden und Variablen*** • Variablen • Methoden • Attribute • Rückgaben und Parameter

2.1.2.3 Unterrichtsvorhaben EF-III

Thema: Such- und Sortieralgorithmen

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

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

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

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

Schließlich wird die Effizienz unterschiedlicher Sortierverfahren beurteilt.

Zeitbedarf: 9 Stunden

Sequenzierung des Unterrichtsvorhabens: nächste Seite

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Kapitel und Materialien
1. Modellierung und Implementation von Datenansammlungen (a) Modellierung von Attributen als Felder (b) Deklaration, Instanziierung und Zugriffe auf ein Feld	Die Schülerinnen und Schüler - analysieren Such- und Sortieralgorithmen und wenden sie auf Beispiele an (D) - entwerfen einen weiteren Algorithmus zum Sortieren (M) - beurteilen die Effizienz von Algorithmen am Beispiel von Sortierverfahren hinsichtlich Zeit und Speicherplatzbedarf (A) - ordnen Attributen lineare Datenansammlungen zu (M)	***Kapitel 6 (Neuauflage) Sortieren und Suchen auf Feldern*** 6.1 Das Feld – Eine Sammlung von Daten
2. Explorative Erarbeitung von Suchverfahren (a) Erkundung von Strategien für das Suchen auf unsortierten Daten, auf sortierten Daten und mithilfe einer Berechnungsfunktion. (b) Vergleich der drei Verfahren durch intuitive Effizienzbetrachtungen.		***Kapitel 6 (Neuauflage) Sortieren und Suchen auf Feldern*** Projekteinstieg 1: Suchen 6.2 Suchen mit System Lineare Suche Binäre Suche Hashing
3. Systematisierung von Algorithmen und Effizienzbetrachtungen (a) Formulierung (falls selbst gefunden) oder Erläuterung von mehreren Algorithmen im Pseudocode (b) Anwendung von Sortieralgorithmen auf verschiedene Beispiele (c) Bewertung von Algorithmen anhand der Anzahl der nötigen Vergleiche (d) Effizienzbetrachtungen an einem konkreten Beispiel bezüglich der Rechenzeit und des Speicherplatzbedarfs (e) Analyse eines weiteren Sortieralgorithmus (sofern nicht in (a) bereits geschehen)		***Kapitel 6 (Neuauflage) Sortieren und Suchen auf Feldern*** Projekteinstieg 2: Sortieren 6.3 Ordnung ist das halbe Leben!? – Sortieren Sortieren Selection Sort Insertion Sort Bubble Sort

2.1.2.4 Unterrichtsvorhaben EF-IV

Thema: Grundlagen der objektorientierten Analyse, Modellierung und Implementierung

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

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

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

Im Anschluss wird die objektorientierte Analyse für das Greenfoot-Szenario Planetenerkundung durchgeführt. Die vom Szenario vorgegebenen Klassen werden von Schülerinnen und Schülern in Teilen analysiert und entsprechende Objekte anhand einfacher Problemstellungen erprobt. Die Lernenden implementieren und testen einfache Programme. Die Greenfoot-Umgebung ermöglicht es, Beziehungen zwischen Klassen zu einem späteren Zeitpunkt (Kapitel 4) zu thematisieren. So kann der Fokus hier auf Grundlagen wie der Unterscheidung zwischen Klasse und Objekt, Attribute, Methoden, Objektidentität und Objektzustand gelegt werden.

Da in Kapitel 2 zudem 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 (Kapitel 3) führen. Dort stehen Kontrollstrukturen im Mittelpunkt.

Zeitbedarf: 8 Stunden

Sequenzierung des Unterrichtsvorhabens: nächste Seite

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Kapitel und Materialien
1. Identifikation von Objekten und Klassen (a) An einem lebensweltnahen Beispiel werden Objekte und Klassen im Sinne der objektorientierten Modellierung eingeführt. (b) Objekte werden durch Objektdiagramme, Klassen durch Klassendiagramme dargestellt. (c) Die Modellierungen werden einem konkreten Anwendungsfall entsprechend angepasst.	Die Schülerinnen und Schüler - ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften und ihre Operationen (M), - stellen den Zustand eines Objekts dar (D), - modellieren Klassen mit ihren Attributen und ihren Methoden (M), - implementieren einfache Algorithmen unter Beachtung der Syntax und Semantik einer Programmiersprache (I), - implementieren Klassen in einer Programmiersprache, auch unter Nutzung dokumentierter Klassenbibliotheken (I).	***Kapitel 2 (Neuauflage) Einführung in die Objektorientierung*** 2.1 Objektorientierte Modellierung
2. Analyse von Objekten und Klassen im Greenfoot-Szenario (a) Schritte der objektorientierten Analyse, Modellierung und Implementation (b) Analyse und Erprobung der Objekte im Greenfoot-Szenario		***Kapitel 2 (Neuauflage) Einführung in die Objektorientierung*** 2.2 Greenfootszenario „Planetenerkundung“
3. Implementierung einfacher Aktionen in Greenfoot (a) Quelltext einer Java-Klasse (b) Implementation eigener Methoden, Dokumentation mit JavaDoc (c) Programme übersetzen (Aufgabe des Compilers) und testen		***Kapitel 2/5 (Neuauflage) Einführung in die Objektorientierung*** 2 und 5 Programmierung eigener Klassen in Greenfoot

2.1.2.5 Unterrichtsvorhaben EF-V

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

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

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

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

Zum Abschluss soll der grundlegende Aufbau eines Rechnersystems im Sinne der von-Neumann-Architektur erarbeitet werden.

Zeitbedarf: 8 Stunden

Sequenzierung des Unterrichtsvorhabens: nächste Seite

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Kapitel und Materialien
1. Analoge und digitale Aufbereitung und Verarbeitung von Daten (a) Erarbeitung der Unterschiede von analog und digital (b) Zusammenfassung und Bewertung der technischen Möglichkeiten von analog und digital	Die Schülerinnen und Schüler - bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen (A) - stellen ganze Zahlen und Zeichen in Binärcodes dar (D), - interpretieren Binärcodes als Zahlen und Zeichen (D) - beschreiben und erläutern den strukturellen Aufbau und die Arbeitsweise singulärer Rechner am Beispiel der „Von-Neumann-Architektur“ (A) - nutzen das Internet zur Recherche, zum Datenaustausch und zur Kommunikation (K) - implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I)	***Exkurs „Analog und Digital“ (Neuauflage)***
2. Der Umgang mit binärer Codierung von Informationen (a) Das binäre (und hexadezimale) Zahlensystem (b) Binäre Informationsspeicherung (c) Binäre Verschlüsselung (d) Implementation eines Binärumrechners		***Exkurs „Binäre Welt“ (Neuauflage)***
3. Aufbau informatischer Systeme (a) Identifikation des EVA-Prinzips als grundlegende Arbeitsweise informatischer Systemen (b) Nachvollziehen der von-Neumann-Architektur als relevantes Modell der Umsetzung des EVA-Prinzips		***Exkurs „Arbeitsweise eines Computers“ (Neuauflage)***

2.1.2.6 Unterrichtsvorhaben EF-VI

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

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

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

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

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

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

Zeitbedarf: 18 Stunden

Sequenzierung des Unterrichtsvorhabens: nächste Seite

Sequenzierung des Unterrichtsvorhabens:

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

(c) Vererbung wird implementiert		
5. Softwareprojekt (a) Analyse und Dekonstruktion eines Spiels (Modelle, Quelltexte) (b) Erweiterung des Spiels um weitere Funktionalitäten (c) Modellierung eines Spiels aufgrund einer Anforderungsbeschreibung, inklusive einer grafischen Benutzeroberfläche (d) (arbeitsteilige) Implementation des Spiels		***Kapitel 7 Softwareprojekte (Neuauflage)*** • Softwareentwicklung • Oberflächen

2.1.2.7 Unterrichtsvorhaben EF-VII

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

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

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

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

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

Zeitbedarf: 12 Stunden

Sequenzierung des Unterrichtsvorhabens: nächste Seite

Sequenzierung des Unterrichtsvorhabens:

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

2.2 Konkretisierte Unterrichtsvorhaben Grundkurs Q1

Unterrichtsvorhaben Q1-I

Thema: Wiederholung und Vertiefung der objektorientierten Modellierung

Leitfragen: Wie wird aus einem anwendungsbezogenen Sachkontext ein informatisches Klassenmodell entwickelt? Wie werden Attribute, Methoden und Beziehungen identifiziert, den Klassen zugeordnet und dargestellt? Welche Auswirkungen hat die informatisch-technische Entwicklung auf das Leben der Menschen?

Vorhabenbezogene Konkretisierung:
Der bereits bekannte objektorientierte Zugang zu informatischer Modellierung wird von einer allgemeinen Betrachtung dieses informatischen Konzepts auf eine konkrete Problematik übertragen. Anhand dieser wird eine anwendungsbezogene Implementation Schritt für Schritt von der Objektidentifikation über das Entwurfs- und Implementationsdiagramm durchlaufen.
Grundlegende Modellierungskonzepte wie Sichtbarkeiten, Assoziationen, Vererbung sowie deren Darstellung in Entwurfs- und Klassendiagrammen und Dokumentationen werden wiederholt. Ebenso wird erneut die grafische Darstellung von Objektkommunikation thematisiert.
Anhand von Gütekriterien und Eigenschaften von Modellierung entwickeln und bewerten die Schülerinnen und Schüler Klassenentwürfe.
Das Konzept der objektorientierten Modellierung wird um die Idee der abstrakten Klasse sowie um das Subtyping erweitert.

Zeitbedarf: 14 Stunden

Sequenzierung des Unterrichtsvorhabens: s. nächste Seite

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Kapitel und Materialien
1. Wiederholung der grundlegenden Konzepte der objektorientierten Programmierung a) Sichtweise der objektorientierten Informatik auf die Welt b) OOP als informatikspezifische Modellierung der Realität c) Schritte der Softwareentwicklung	Die Schülerinnen und Schüler … - analysieren und erläutern objektorientierte Modellierungen (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), - 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), - stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D), - dokumentieren Klassen (D), - stellen die Kommunikation zwischen Objekten grafisch dar (D), - untersuchen und bewerten anhand von Fallbeispielen Auswirkungen des Einsatzes von Informatiksystemen sowie Aspekte der Sicherheit von Informatiksystemen, des Datenschutzes 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).	2.1 Die Welt ist voller Objekte Projekteinstieg: Klassenentwurf – step by step
2. Erweiterung der objektorientierten Programmierung a) Umsetzung einer Anforderung in Entwurfs- und Klassendiagramm b) Objektkommunikation im Sequenzdiagramm c) Klassendokumentation d) Umsetzung von Teilen der Modellierung		2.2 Gut geplant – Klassenentwurf 2.3 Vererbungshierarchien nutzen
3. Mensch und Technik a) Verantwortung von Informatikern b) Automatisierung des Alltags durch Informatik		Die digitale Welt 001 - Mensch und Technik
4. Übung und Vertiefung der OOM / OOP		Prüfungsvorbereitung

Unterrichtsvorhaben Q1-2

Thema:
Organisation und Verarbeitung von Daten I – Modellierung und Implementierung von Anwendungen mit dynamischen und linearen Datenstrukturen

Leitfragen:
Wie müssen Daten linear strukturiert werden, um in den gestellten Anwendungsszenarien eine beliebige Anzahl von Objekten verwalten zu können?

Vorhabensbezogene Konkretisierung:
Ausgehend von einigen Alltagsbeispielen werden als Erstes die Anforderungen an eine Datenstruktur erschlossen. Anschließend werden die Möglichkeiten des Arrays untersucht, lineare Daten zu verwalten und über deren Grenzen/Probleme die Vorteile einer dynamischen linearen Struktur am Beispiel der Struktur Queue erarbeitet (Anwendungskontext Warteschlange). Die Klasse *Queue* selbst wird vorgegeben, die Operationen erläutert. Zur Vertiefung der Kenntnisse wird ein weiteres Anwendungsszenario eingeführt (Polizeikontrolle), dessen Lösung modelliert und implementiert wird. Darauf folgt die Erarbeitung der Struktur Stack, die mithilfe eines einfachen Anwendungsszenarios eingeführt (Biber/Palindrom) wird. Auch hier wird die Klasse *Stack* selbst vorgegeben und die Operationen erläutert. Weitere Aufgaben dienen der Vertiefung und Sicherung.
Um die Unterschiede der beiden Prinzipien FIFO und LIFO zu verstehen, werden zur Lösung der Aufgaben sowohl der Stack als auch die Queue benötigt.

Als letzte lineare dynamische Datenstruktur wird die Liste eingeführt. In dieser Sequenz liegt der Fokus auf der Möglichkeit, auf jedes Element zugreifen zu können. Nachdem die umfangreicheren Standardoperationen dieser Datenstruktur in einem einführenden Beispiel (z. B. Vokabeltrainer) erarbeitet und in einem weiteren Beispiel vertieft (LED) wurden, werden abschließend in einem Anwendungskontext verschiedene lineare Datenstrukturen angewendet. Die Modellierung erfolgt beim gesamten Vorhaben in Entwurfs- und Implementationsdiagrammen.
Zeitbedarf: 20 Stunden

Sequenzierung des Unterrichtsvorhabens: s. nächste Seite

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Kapitel und Materialien
1. Die Datenstruktur **Feld** a) Erarbeitung der Anforderungen an eine Datenstruktur b) Wiederholung der Datenstruktur Array, Eigenschaften der Datenstruktur, Standardoperationen für ein und zweidimensionale Arrays c) Modellierung und Implementierung von Anwendungen	Die Schülerinnen und Schüler… - erläutern und begründen methodische Vorgehensweisen, Entwurfs- und Implementationsentscheidungen sowie Aussagen über Informatiksysteme (A) - konstruieren zu kontextbezogenen Problemstellungen informatische Modelle (M) - 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), - implementieren auf der Grundlage von Modellen oder Modellausschnitten Computerprogramme (I) - testen und korrigieren Computerprogramme (I) - interpretieren Fehlermeldungen und korrigieren den Quellcode (I), - überführen gegebene textuelle und grafische Darstellungen informatischer Zusammenhänge in die jeweils andere Darstellungsform (D) - stellen informatische Modelle und Abläufe in Texten, Tabellen, Diagrammen und Grafiken dar (D) - stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D) - modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M) - ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M) - dokumentieren Klassen (D) - implementieren Klassen in einer	**Kapitel:** - Anforderungen an eine Datenstruktur - Speichern mit Struktur *neue Wdh. Aufgabe entwickeln, z. B. eine Chart-Top-10, eine Aufgabe mit zweidimensionalem Array (vgl. Anforderungen KLP)*
2. Die Datenstruktur **Schlange** a) Modellierung und Implementierung der Verknüpfung von Objekten b) Generische Typen, Trennung von Verwaltung und Inhalt dyn. DS. c) Erläuterung von Problemstellungen, die nach dem FIFO-Prinzip bearbeitet werden d) Funktionalität der Schlange unter Verwendung der Klasse *Queue*; Erschließen der Standardoperationen e) Modellierung und Implementierung einer Anwendung auf der Basis einer Anforderungsbeschreibung mit Objekten der Klasse *Queue*		**Kapitel:** - Wer zuerst kommt - Objekte miteinander verketten - Verwaltung und Inhalt - Funktionen der Queue **Aufgaben:** - Warteschlange Büro (Standardoperationen/Basiskompetenz) Kunden warten auf einem Flur, um in ein Büro vorgelassen zu werden. Sie können sich am Ende der Warteschlange anstellen, vorgelassen werden oder müssen alle gehen, wenn die Sprechzeit vorüber ist. - Erweiterte Queue Verkehrskontrolle (Vertiefung) Die Polizei kontrolliert die Fahrzeuge im Hinblick auf ihre Verkehrstauglichkeit. Für die Kontrolle werden die Fahrzeuge aus dem Verkehr gewunken. Es werden so lange Fahrzeuge kontrolliert, bis eine gewissen Menge an Verstößen vorliegt oder Autos kontrolliert wurden.
3. Die Datenstruktur **Stapel**		**Kapitel:**

a) Erläuterung von Problemstellungen, die nach dem LIFO-Prinzip bearbeitet werden b) Funktionalität der Klasse Stapel unter Verwendung der Klasse *Stack* c) Modellierung und Implementierung einer Anwendung auf Basis einer Anforderungsbeschreibung mit Objekten der Klasse *Queue* d) Modellierung und Implementierung einer Anwendung unter Verwendung verschiedener Datenstrukturen (Objekte der Klassen *Queue*, *Stack* und *Array (Palindrom)*)	Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I)	- Daten gut abgelegt – Stapel - Funktionen der Datenstruktur Stapel **Aufgaben:** - Standardoperationen/Basiskompetenz (Stapel Münzen/ CDs) zur Umsetzung der gegebenen Funktionen der Klasse *Stack* - Biber und Teller Es gibt große und kleine Biber sowie grüne und braune Teller. Es muss überprüft werden, ob die gestapelten Teller zur Schlange der Biber passen, da die großen Biber nur von den braunen Tellern essen und die kleinen von den grünen. Hierbei müssen sowohl Queue als auch Stack verwendet werden. - Palindrom Es wird überprüft, ob ein beliebiges Wort ein Palindrom ist.
4. Die Datenstruktur **Liste** a) Analyse der Möglichkeiten bisheriger Datenstrukturen zwecks Bestimmung notwendiger Funktionalitäten für komplexere Anwendungen (Abgrenzung zu *Stack*/*Queue*, zusätzliche Fähigkeiten der Klasse *List*) b) Erarbeitung der Funktionalität der Liste unter Verwendung der Klasse *List* c) Modellierung und Implementierung einer Anwendung mit Objekten der Klasse *List* d) Modellierung und Implementierung einer Anwendung unter Verwendung verschiedener Datenstrukturen (Stack, Queue, List)		**Kapitel:** - Flexibel für alle Fälle – (die) lineare Liste - Funktionen der Datenstruktur Liste **Aufgaben:** - LEDs - Textzeilen verarbeiten

5. Übungen und Vertiefungen zur Verwendung linearer und dynamischer Datenstrukturen anhand weiterer Problemstellungen		Prüfungsvorbereitung
		Projekteinstieg Heldenspiel *Mit dem Heldenspiel können alle im Kapitel behandelten Datenstrukturen erarbeitet werden. Das Spiel kann bis zu einem beliebigen Grad realisiert werden, sodass es sowohl als Einstieg als auch als ein umfassendes Projekt für lineare Datenstrukturen genutzt werden kann.*

Unterrichtsvorhaben Q1-III

Thema: Algorithmen zum Suchen und Sortieren auf linearen Datenstrukturen

Leitfragen: Nach welchen Grundprinzipien können Algorithmen strukturiert werden? Welche Qualitätseigenschaften sollten Algorithmen erfüllen? Wie können mithilfe von Such- und Sortieralgorithmen Daten in linearen Strukturen effizient (wieder-)gefunden werden?

Vorhabenbezogene Konkretisierung:
Zunächst werden anhand eines Anwendungsbeispiels übergreifende Algorithmuseigenschaften (wie Korrektheit, Effizienz und Verständlichkeit) erarbeitet und Schritte der Algorithmusentwicklung wiederholt. Dabei kommen Struktogramme zur Darstellung von Algorithmen zum Einsatz.
Als besondere Struktur von Algorithmen wird die Rekursion an Beispielen veranschaulicht und gegenüber der Iteration abgegrenzt. Rekursive Algorithmen werden von den Schülerinnen und Schülern analysiert und selbst entwickelt.
In der zweiten Unterrichtssequenz geht es um die Frage, wie Daten in linearen Strukturen (lineare Liste und Array) (wieder-)gefunden werden können. Die lineare Suche als iteratives und die binäre Suche als rekursives Verfahren werden veranschaulicht und implementiert. Die Bewertung der Algorithmen erfolgt, indem jeweils die Anzahl der Vergleichsoperationen und der Speicherbedarf ermittelt wird.
Möchte man Daten effizient in einer linearen Struktur wiederfinden, so rückt zwangsläufig die Frage nach einer Sortierstrategie in den Fokus. Es wird mindestens ein iteratives und ein rekursives Sortierverfahren erarbeitet und implementiert sowie ihre Effizienz bewertet.

Zeitbedarf: 20 Stunden

Sequenzierung des Unterrichtsvorhabens: s. nächste Seite

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Kapitel und Materialien
1. Eigenschaften von Algorithmen a) Qualitätseigenschaften von Algorithmen b) Strukturierung von Algorithmen mit Hilfe der Strategien „Modularisierung“ und „Teile und Herrsche“ c) Analyse und Entwicklung von rekursiven Algorithmen	Die Schülerinnen und Schüler - analysieren und erläutern Algorithmen und Programme (A), - modifizieren Algorithmen und Programme (I), - stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D), - entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“ und „Teilen und Herrschen“ (M), - implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I), - testen Programme systematisch anhand von Beispielen (I). - implementieren und erläutern iterative und rekursive Such- und Sortierverfahren (I), - beurteilen die Effizienz von Algorithmen unter Berücksichtigung des Speicherbedarfs und der Zahl der Operationen (A), - beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), - 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),	4.1 Ohne Algorithmen läuft nichts 4.2 Teile die Arbeit – rekursive Algorithmen
2. Suchen in Listen und Arrays a) Lineare Suche in Listen und Arrays b) Binäre Suche in einem Array c) Untersuchung der beiden Verfahren bzgl. Laufzeit und Speicherplatzbedarf		4.3 Suchen – iterativ und rekursiv
3. Sortieren auf Listen und Arrays a) Entwicklung und Implementierung eines iterativen Sortierverfahrens für eine Liste b) Entwicklung und Implementierung eines rekursiven Sortierverfahrens für ein Array c) Untersuchung der beiden Verfahren bzgl. Laufzeit und Speicherplatzbedarf		4.4 Sortieren – iterativ und rekursiv

Unterrichtsvorhaben Q-I IV

Thema: Aufbau von und Kommunikation in Netzwerken

Leitfragen: Was macht menschliche Kommunikation aus? Welchen Stellenwert haben technische/ informatische Hilfsmittel für die Kommunikation? Wie werden Daten in einem Netzwerk zwischen den Kommunikationspartnern übertragen? Wie ist die Arbeitsteilung in Netzwerken gestaltet? Wie kann sicher in Netzwerken kommuniziert werden?

Vorhabenbezogene Konkretisierung:
Ausgehend von alltäglicher Face-to-Face-Kommunikation werden die Grundprinzipien sowie die Bewertungskriterien von Kommunikation erläutert. Das Netzwerk wird als vorteilhafte Kommunikationsstruktur dargestellt und anhand von Topologien und Reichweiten kategorisiert. Ausgehend davon wird der Protokollbegriff entwickelt und anhand des TCP/IP-Schichtenmodells analysiert. Anschließend wird das Client-Server-Prinzip vorgestellt und angewandt.
Sichere Kommunikation in Netzen ist nur dank kryptografischer Verfahren möglich. Stellvertretend werden zwei symmetrische und ein asymmetrisches Verfahren erläutert, angewandt und bewertet.

Zeitbedarf: 12 Stunden

Sequenzierung des Unterrichtsvorhabens: s. nächste Seite

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Kapitel und Materialien
1. Technische Kommunikation als Fortführung natürlicher Kommunikation a) Kommunikation im Shannon-Weaver-Modell b) Kriterien von technischen Kommunikationsarten c) Die Geschichte der technischen Kommunikation	Die Schülerinnen und Schüler - beschreiben und erläutern Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken (A), - nutzen bereitgestellte Informatiksysteme und das Internet reflektiert zum Erschließen, zur Aufbereitung und Präsentation fachlicher Inhalte (D), - analysieren und erläutern Eigenschaften und Einsatzbereiche symmetrischer und asymmetrischer Verschlüsselungsverfahren (A).	7.1 Menschen kommunizieren – ohne und mit Technik Projekteinstieg: Kommunikation im Wilden Westen
2. Aufbau von Netzwerken und Kommunikationsregeln a) Das Netzwerk als Organisationsprinzip der Kommunikation und Möglichkeiten der Ausformung b) Geregelte technische Kommunikation durch Protokolle in Schichtenmodellen		7.2 Ohne Protokoll läuft nichts - Netzwerke
3. Aufgabenteilung in Netzwerken durch Server und Client a) Aufbau und Aufgaben der Client-Server-Struktur (b) Protokolle zwischen Client und Server		7.3 Einer für alle – Client-Server-Struktur
4. Kryptologie a) Veranschaulichen und Anwenden von symmetrischen und asymmetrischen kryptographischen Verfahren (Caesar, Vigenère, RSA) b) Bewertung der Verfahren hinsichtlich ihrer Sicherheit und ihrem Aufwand		Die digitale Welt 100 – Kryptologie
5. Übung und Vertiefung des		Prüfungsvorbereitung

Aufbaus von und der Kommunikation in Netzwerken		

Unterrichtsvorhaben Q2-I

Thema: Automaten und formale Sprachen

Leitfragen:
Wie lassen sich reale Automaten durch ein Modell formal beschreiben? Wie kann die Art und Weise, wie ein Computer Zeichen (Eingaben) verarbeitet, durch Automaten dargestellt werden? Welche Eigenschaften besitzen Automaten und was können sie leisten? Wie werden sie dargestellt? Wie werden reguläre Sprachen durch eine Grammatik beschrieben? In welchem Verhältnis stehen endliche Automaten und Grammatiken? Welche Anwendungsfälle können durch endliche Automaten und Grammatiken regulärer Sprachen beschrieben werden und welche nicht?

Vorhabensbezogene Konkretisierung:
Ausgehend von der Beschreibung und Untersuchung realer Automaten wird das formale Modell eines endlichen Automaten entwickelt. Neben dem Mealy-Automaten geht es vor allem um den erkennenden endlichen Automaten. Auf die Erarbeitung der Beschreibung folgt die Modellierung
eigener Automaten und die Untersuchung bestehender, um die Eigenschaften und Grenzen eines endlichen Automaten zu erkennen. Hierbei wird dessen Verhalten auf bestimmte Eingaben analysiert.
An den Themenkomplex *Endliche Automaten* schließt sich die Erarbeitung von Grammatiken regulärer Sprachen an. Die Untersuchung beginnt bei die Erschließung der formalen Beschreibung und wird mit der Entwicklung von Grammatiken zu regulären Sprachen fortgeführt. Hierbei wird auch die Beziehung von Grammatiken regulärer Sprachen zu endlichen Automaten an Beispielen erarbeitet und analysiert. Hierzu gehört auch die Untersuchung, welche Problemstellungen durch endliche Automaten und reguläre Grammatiken beschrieben werden können und welche nicht.

Zeitbedarf: 20 Std.

Sequenzierung des Unterrichtsvorhabens: s. nächste Seite

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Kapitel und Materialien
1. Endliche Automaten a) Erarbeitung der formalen Beschreibung eines Mealy-Automaten und der Darstellungsformen b) Erarbeitung der formalen Beschreibung eines deterministischen endlichen Automaten (DEA) sowie dessen Darstellungsformen; Erschließung der Fachbegriffe Alphabet, Wort, (akzeptierte) Sprache, Determinismus c) Analyse der Eigenschaften von DEA durch die Modellierung eines Automaten zu einer gegebenen Problemstellung, der Modifikation eines Automaten sowie die Überführung der gegebenen Darstellungsform in eine andere	Die Schülerinnen und Schüler - stellen informatische Modelle und Abläufe in Texten, Tabellen, Diagrammen und Grafiken dar (D) - analysieren und erläutern die Eigenschaften endlicher Automaten einschließlich ihres Verhaltens bei bestimmten Eingaben (A) - ermitteln die Sprache, die ein endlicher Automat akzeptiert (D) - entwickeln und modifizieren zu einer Problemstellung endlicher Automaten (M) - stellen endliche Automaten in Tabellen und Graphen dar und überführen sie in die jeweils andere Darstellungsform (D) - entwickeln zur Grammatik einer regulären Sprache einen zugehörigen endlichen Automaten (M) - analysieren und erläutern Grammatiken regulärer Sprachen (A) - modifizieren Grammatiken regulärer Sprachen (M) - entwickeln zu einer regulären Sprache eine Grammatik, die die Sprache erzeugt (M) - entwickeln zur akzeptierten Sprache eines Automaten eine zugehörige Grammatik (M) - beschreiben an Beispielen den Zusammenhang zwischen Automaten und Grammatiken (D) - zeigen die Grenzen endlicher Automaten und regulärer Grammatiken auf	Vom realen Automaten zum Modell Projekteinstieg: Schatzsuche Der Mealy-Automat Der erkennende endliche Automat Wort und Sprache
2. Grammatiken regulärer Sprachen a) Erarbeitung der formalen Beschreibung einer regulären Grammatik (Sprache, Terminal und Nicht-Terminal, Produktionen und Produktionsvorschriften) b) Analyse der Eigenschaften einer regulären Grammatik durch deren Entwicklung und Modellierung zu einer gegebenen Problemstellung.		Grammatiken regulärer Sprachen
3. Übungen und Vertiefungen		Prüfungsvorbereitung

Verwendung endlicher Automaten und Grammatiken regulärer Sprachen		
Projekteinstieg *Erarbeitung der formalen Beschreibung und Überprüfung des Verhaltens eines erkennenden Automaten auf bestimmte Eingaben*		

Unterrichtsvorhaben Q2-II

Thema: Organisation und Verarbeitung von Daten II – Modellierung und Implementierung von Anwendungen mit dynamischen nicht-linearen Datenstrukturen

Leitfragen: Wie können Daten mithilfe von Baumstrukturen verwaltet werden? Wie können mit binären Suchbäumen Inhalte sortiert verwaltet werden und welche Vor- und Nachteile bietet dies?

Vorhabenbezogene Konkretisierung:
Anhand des Anwendungskontextes Spielbäume werden zunächst der generelle Aufbau von Baumstrukturen (auch nicht-binäre) und wichtige Grundbegriffe erarbeitet. Die Darstellung von Bäumen mit Knoten und Kanten wird eingeführt.

Anschließend rückt der Fokus auf die binären Bäume, deren rekursiver Aufbau für die Traversierung der Datenstruktur genutzt wird. Die Preorder-Traversierung wird verwendet, um einen gespeicherten Inhalt in einem Binärbaum zu finden (Tiefensuche).
Der Anwendungskontext Ahnenbaum wird mithilfe der Klasse BinaryTree (der Materialien für das Zentralabitur in NRW) modelliert und (ggf. in Teilen) implementiert. Dabei wird u. a. die Erzeugung eines Binärbaums mithilfe der beiden Konstruktoren der Klasse BinaryTree thematisiert.

Möchte man Daten geordnet speichern, so bietet sich die Struktur des binären Suchbaums an. An Beispielen wird zunächst das Prinzip des binären Suchbaums erarbeitet. Die Operationen des Suchens, Einfügens, Löschens und der sortierten Ausgabe werden thematisiert.
Um Daten in einem Anwendungskontext mithilfe eines binären Suchbaums verwalten zu können, müssen sie in eine Ordnung gebracht werden können, d. h. sie müssen vergleichbar sein. Diese Vorgabe wird mithilfe des Interfaces Item realisiert, das alle Klassen, dessen Objekte in einem Suchbaum verwaltet werden sollen, implementieren müssen. Auf diese Weise wird ein Anwendungskontext (Benutzerverwaltung) mithilfe der Klassen BinarySearchTree und Item modelliert und (ggf. in Teilen) implementiert.

Zeitbedarf: 20 Stunden

Sequenzierung des Unterrichtsvorhabens: s. nächste Seite

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Kapitel und Materialien
1. Aufbau von Baumstrukturen und Grundbegriffe a) Erarbeitung der Begriffe Wurzel, Knoten, Blatt, Kante, Grad eines Knotens und eines Baumes, Pfad, Tiefe, Ebene, Teilbaum b) Aufbau und Darstellung von Baumstrukturen in verschiedenen Anwendungskontexten	Die Schülerinnen und Schüler - stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D), - erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (A), - analysieren und erläutern Algorithmen und Programme (A), - stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D).	6.1 Spielen mit Struktur – Baumstrukturen Projekteinstieg 1: Spielbäume
2. Binäre Bäume a) rekursiver Aufbau eines binären Baums b) Traversierungen (pre-, in-, postorder) c) Modellierung eines Binärbaums in einem Anwendungskontext mit Hilfe der Klasse BinaryTree (als Entwurfs- und Implementationsdiagramm) d) Implementation einer Anwendung der Datenstruktur binärer Baum (ggf. in Teilen)	- 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),	6.2 Zwei Nachfolger sind genug! - Binäre Bäume Implementation des Projekts Ahnenbaum
3. Binäre Suchbäume a) Prinzip des binären Suchbaums, Ordnungsrelation b) Operationen auf dem binären Suchbaum (Suchen, Einfügen, Löschen, sortierte Ausgabe) c) Modellierung eines binären Suchbaums in einem Anwendungskontext mit	- 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	6.3 Wer Ordnung hält, spart Zeit beim Suchen – Binäre Suchbäume Projekteinstieg 2: Binäre Suchbäume Implementation des Projekts Benutzerverwaltung

Hilfe der Klasse BinarySearchTree (als Entwurfs- und Implementationsdiagramm) und dem Interface Item d) Implementation einer Anwendung der Datenstruktur binärer Suchbaum (ggf. in Teilen)	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),	
4. Übung und Vertiefungen der Verwendung von Binärbäumen oder binären Suchbäumen anhand weiterer Problemstellungen		Prüfungsvorbereitung

Unterrichtsvorhaben Q2-3

Thema: Nutzung und Modellierung von relationalen Datenbanken in Anwendungskontexten

Leitfragen: Was sind Datenbanken und wie kann man mit ihnen arbeiten? Wie entwickelt man selbst eine Datenbank für einen Anwendungskontext?

Vorhabenbezogene Konkretisierung:
Am Beispiel eines Online-Buchhandels wird der Aufbau einer Datenbank sowie wichtige Grundbegriffe erarbeitet. Die Schülerinnen und Schüler nehmen dabei zunächst die Sicht der Anwender an, die eine bestehende Datenbank beschreiben und analysieren und mithilfe von SQL-Abfragen Daten gezielt herausfiltern.
Mithilfe des Projekteinstiegs „Tabellen“ können bereits zu einem frühen Zeitpunkt des Unterrichtsvorhabens Redundanzen, Inkonsistenzen und Anomalien problematisiert werden.

Nachdem die Lernenden in der ersten Sequenz mit Datenbanken vertraut gemacht wurden, nehmen sie nun die Rolle der Entwickler an, indem sie selbst Datenbanken von Grund auf modellieren und das Modell in ein Relationenschema überführen. Sie arbeiten mit Entity-Relationship-Diagrammen, um Entitäten, zugehörige Attribute, Relationen und Kardinalitäten in Anwendungskontexten darzustellen. Gegebene ER-Diagramme werden analysiert, erläutert und modifiziert.
Der bereits in der ersten Sequenz problematisierte Begriff der Redundanz wird am Ende des Unterrichtsvorhabens wieder aufgegriffen, um die Normalisierung von Datenbanken zu thematisieren. Bestehende Datenbankschemata werden hinsichtlich der 1. bis 3. Normalform untersucht und (soweit nötig) normalisiert.

Zeitbedarf: 20 Stunden

Sequenzierung des Unterrichtsvorhabens: nächste Seite

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Kapitel und Materialien
1. Nutzung von relationalen Datenbanken a) Aufbau von Datenbanksystemen und Grundbegriffe - Aufgaben und Eigenschaften eines Datenbanksystems - Erarbeitung der Begriffe Tabelle, Attribut, Attributwert, Datensatz, Datentyp, Primärschlüssel, Datenbankschema - Problematisierung von Redundanzen, Anomalien und Inkonsistenzen b) SQL-Abfragen - Erarbeitung der grundlegenden Sprachelemente von SQL (SELECT(DISTINCT), FROM, WHERE, JOIN) - Analyse und Erarbeitung von SQL-Abfragen (AND, OR, NOT, UNION, AS, GROUP BY,ORDER BY, ASC, DESC, COUNT, MAX, MIN, SUM, Arithmetische Operatoren: +, -, *, /, (…), Vergleichsoperatoren: =, <>, >, <, >=, <=, LIKE, BETWEEN, IN, IS NULL, geschachtelte Select-Ausdrücke) c) Vertiefung an einem weiteren Datenbankbeispiel	Die Schülerinnen und Schüler - erläutern die Eigenschaften und den Aufbau von Datenbanksystemen unter dem Aspekt der sicheren Nutzung (A), - analysieren und erläutern die Syntax und Semantik einer Datenbankabfrage (A), - analysieren und erläutern eine Datenbankmodellierung (A), - erläutern die Eigenschaften normalisierter Datenbankschemata (A), - bestimmen Primär- und Sekundärschlüssel (M), - ermitteln für anwendungsbezogene Problemstellungen Entitäten, zugehörige Attribute, Relationen und Kardinalitäten (M), - modifizieren eine Datenbankmodellierung (M), - modellieren zu einem Entity-Relationship-Diagramm ein relationales Datenbankschema (M), - 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),	**Kapitel 8 Datenbanken** 8.1 Wissen speichern und verwalten – Datenbanksysteme 8.2 Daten anordnen mit Tabellen Beispiel: Buchhandlung Redundanzen, Anomalien und Inkonsistenzen Projekteinstieg: Tabellen 8.3 Daten filtern mit SQL 8.4 Komplexe Filter Aufgaben
2. Modellierung von relationalen Datenbanken a) Datenbankentwurf durch ER-Diagramme - Ermittlung von Entitäten,	- überprüfen Datenbankschemata auf vorgegebene Normalisierungseigenschaften (D).	8.5 Datenbankentwurf *Beispiel: Online-Buchhandel* *Datenanalyse und Entwurf*

zugehörigen Attributen, Beziehungen und Kardinalitäten in Anwendungssituationen und Modellierung eines Datenbankentwurfs in Form eines Entity-Relationship-Diagramms - Erläuterung und Erweiterung einer Datenbankmodellierung b) Entwicklung eines relationalen Modells aus einem Datenbankentwurf - Überführung eines Entity-Relationship-Diagramms in ein relationales Datenbankschema inklusive der Bestimmung von Primär- und Fremdschlüsseln c) Normalformen - Überprüfung von Datenbankschemata hinsichtlich der 1. bis 3. Normalform und Normalisierung (um Redundanzen zu vermeiden und Konsistenz zu gewährleisten)		*8.6 Umsetzung des ER-Modells* *Entitätsmengen* *m:n-Beziehungen* *1:n-Beziehungen* *1:1-Beziehungen* *Wiederaufgriff des Projekteinstiegs* *8.7 Datenbanken verbessern durch Normalformen*
3. Übung und Vertiefung der Nutzung und Modellierung von relationalen Datenbanken		Prüfungsvorbereitung

2.3 Konkretisierte Unterrichtsvorhaben Leistungskurs Q1

Unterrichtsvorhaben Q1-I

Thema: Wiederholung und Vertiefung der objektorientierten Modellierung

Leitfragen:
Wie wird aus einem anwendungsbezogenen Sachkontext ein informatisches Klassenmodell entwickelt? Wie werden Attribute, Methoden und Beziehungen identifiziert, den Klassen zugeordnet und dargestellt? Welche Auswirkungen hat die informatisch-technische Entwicklung auf das Leben der Menschen?

Vorhabenbezogene Konkretisierung:
Der bereits bekannte objektorientierte Zugang zu informatischer Modellierung wird von einer allgemeinen Betrachtung dieses informatischen Konzepts auf eine konkrete Problematik übertragen. Anhand dieser wird eine anwendungsbezogene Implementation Schritt für Schritt von der Objektidentifikation über das Entwurfs- und Implementationsdiagramm durchlaufen.
Grundlegende Modellierungskonzepte wie Sichtbarkeiten, Assoziationen, Vererbung sowie deren Darstellung in Entwurfs- und Klassendiagrammen und Dokumentationen werden wiederholt. Ebenso wird erneut die grafische Darstellung von Objektkommunikation thematisiert.
Anhand von Gütekriterien und Eigenschaften von Modellierung entwickeln und bewerten die Schülerinnen und Schüler Klassenentwürfe.
Das Konzept der objektorientierten Modellierung wird um die Idee der abstrakten Klasse sowie um das Subtyping erweitert.

Zeitbedarf: 25 Stunden

Sequenzierung des Unterrichtsvorhabens: s. nächste Seite

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Wiederholung der grundlegenden Konzepte der objektorientierten Programmierung a) Sichtweise der objektorientierten Informatik auf die Welt b) OOP als informatikspezifische Modellierung der Realität c) Schritte der Softwareentwicklung	Die Schülerinnen und Schüler … • analysieren und erläutern objektorientierte Modellierungen (**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**), • 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**), • stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (**D**), • dokumentieren Klassen (**D**), • stellen die Kommunikation zwischen Objekten grafisch dar (**D**), • untersuchen und bewerten anhand von Fallbeispielen Auswirkungen des Einsatzes von Informatiksystemen sowie Aspekte der Sicherheit von Informatiksystemen, des Datenschutzes 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**).	2.1 Die Welt ist voller Objekte Projekteinstieg: Klassenentwurf – step by step
2. Erweiterung der objektorientierten Programmierung a) Umsetzung einer Anforderung in Entwurfs- und Klassendiagramm b) Objektkommunikation im Sequenzdiagramm c) Klassendokumentation d) Umsetzung von Teilen der Modellierung		2.2 Gut geplant – Klassenentwurf 2.3 Vererbungshierarchien nutzen
3. Mensch und Technik a) Verantwortung von Informatikern b) Automatisierung des Alltags durch Informatik		Die digitale Welt 001 - Mensch und Technik
4. Übung und Vertiefung der OOM / OOP		Prüfungsvorbereitung

Unterrichtsvorhaben Q1-2

Thema:
Organisation und Verarbeitung von Daten I – Modellierung und Implementierung von Anwendungen mit dynamischen und linearen Datenstrukturen

Leitfragen:
Wie müssen Daten linear strukturiert werden, um in den gestellten Anwendungsszenarien eine beliebige Anzahl von Objekten verwalten zu können?

Vorhabensbezogene Konkretisierung:
Ausgehend von einigen Alltagsbeispielen werden als erstes die Anforderungen an eine Datenstruktur erschlossen. Anschließend werden die Möglichkeiten des Arrays untersucht, lineare Daten zu verwalten und über deren Grenzen/Probleme die Vorteile einer dynamischen linearen Struktur am Beispiel der Struktur Queue erarbeitet (Anwendungskontext Warteschlange). Die Klasse Queue selbst wird vorgegeben, die Operationen erläutert. Zur Vertiefung der Kenntnisse wird ein weiteres Anwendungsszenario eingeführt (Polizeikontrolle), dessen Lösung modelliert und implementiert wird. Darauf folgt die Erarbeitung der Struktur Stack, die mithilfe eines einfachen Anwendungsszenarios eingeführt (Biber/Palindrom) wird. Auch hier wird die Klasse Stack selbst vorgegeben und die Operationen erläutert. Weitere Aufgaben dienen der Vertiefung und Sicherung.
Um die Unterschiede der beiden Prinzipien FIFO und LIFO zu verstehen, werden zur Lösung der Aufgaben sowohl der Stack als auch die Queue benötigt.
Als letzte lineare dynamische Datenstruktur wird die Liste eingeführt. In dieser Sequenz liegt der Fokus auf der Möglichkeit, auf jedes Element zugreifen zu können. Nachdem die umfangreicheren Standardoperationen dieser Datenstruktur in einem einführenden Beispiel (Vokabeltrainer) erarbeitet und in einem weiteren Beispiel vertieft (LED) wurden, werden abschließend in einem Anwendungskontext verschiedene lineare Datenstrukturen angewendet. Die Modellierung erfolgt beim gesamten Vorhaben in Entwurfs- und Implementationsdiagrammen.
Zeitbedarf: 30 Stunden

Sequenzierung des Unterrichtsvorhabens: s. nächste Seite

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Die Datenstruktur **Feld** a) Erarbeitung der Anforderungen an eine Datenstruktur b) Wiederholung der Datenstruktur Array, Eigenschaften der Datenstruktur, Standardoperationen für ein und zweidimensionale Arrays c) Modellierung und Implementierung von Anwendungen	Die Schülerinnen und Schüler… • erläutern und begründen methodische Vorgehensweisen, Entwurfs- und Implementationsentscheidungen sowie Aussagen über Informatiksysteme (A) • konstruieren zu kontextbezogenen Problemstellungen informatische Modelle (M) • 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),	**Aufgaben** Aufgabe entwickeln, z. B. eine Chart-Top-10, eine Aufgabe mit zweidimensionalem Array z.B. Schiffe versenken/Minesweeper (vgl. Anforderungen KLP)
2. Die Datenstruktur **Schlange** a) Modellierung und Implementierung der Verknüpfung von Objekten b) Generische Typen, Trennung von Verwaltung und Inhalt dyn. DS. c) Erläuterung von Problemstellungen, die nach dem FIFO-Prinzip bearbeitet werden d) Funktionalität der Schlange unter Verwendung der Klasse Queue; Erschließen der Standardoperationen e) Modellierung und Implementierung einer Anwendung auf der Basis einer Anforderungsbeschreibung mit Objekten der Klasse Queue	• implementieren auf der Grundlage von Modellen oder Modellausschnitten Computerprogramme (I) • testen und korrigieren Computerprogramme (I) • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • überführen gegebene textuelle und grafische Darstellungen informatischer Zusammenhänge in die jeweils andere Darstellungsform (D) • stellen informatische Modelle und Abläufe in Texten, Tabellen, Diagrammen und Grafiken dar (D) • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D) • modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M) • ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M) • dokumentieren Klassen (D) • -implementieren Klassen in einer Programmiersprache auch unter	**Aufgaben** • Warteschlange Büro (Standardoperationen/Basiskompetenz) Kunden warten auf einem Flur, um in ein Büro vorgelassen zu werden. Sie können sich am Ende der Warteschlange anstellen, vorgelassen werden oder müssen alle gehen, wenn die Sprechzeit vorüber ist. • Erweiterte Queue Verkehrskontrolle (Vertiefung) Die Polizei kontrolliert die Fahrzeuge im Hinblick auf ihre Verkehrstauglichkeit. Für die Kontrolle werden die Fahrzeuge aus dem Verkehr gewunken. Es werden so lange Fahrzeuge kontrolliert, bis eine gewisse Menge an Verstößen vorliegt oder Autos kontrolliert wurden.

3. Die Datenstruktur **Stapel** a) Erläuterung von Problemstellungen, die nach dem LIFO-Prinzip bearbeitet werden b) Funktionalität der Klasse Stapel unter Verwendung der Klasse Stack c) Modellierung und Implementierung einer Anwendung auf Basis einer Anforderungsbeschreibung mit Objekten der Klasse Queue d) Modellierung und Implementierung einer Anwendung unter Verwendung verschiedener Datenstrukturen (Objekte der Klassen Queue, Stack und Array (Palindrom))	Nutzung dokumentierter Klassenbibliotheken (I)	• Standardoperationen/Basiskompetenz (Stapel Münzen/ CDs) zur Umsetzung der gegebenen Funktionen der Klasse Stack • Biber und Teller Es gibt große und kleine Biber sowie grüne und braune Teller. Es muss überprüft werden, ob die gestapelten Teller zur Schlange der Biber passen, da die großen Biber nur von den braunen Tellern essen und die kleinen von den Grünen. Hierbei müssen sowohl Queue als auch Stack verwendet werden. • Palindrom Es wird überprüft, ob ein beliebiges Wort ein Palindrom ist.
4. Die Datenstruktur **Liste** a) Analyse der Möglichkeiten bisheriger Datenstrukturen zwecks Bestimmung notwendiger Funktionalitäten für komplexere Anwendungen (Abgrenzung zu Stack/Queue, zusätzliche Fähigkeiten der Klasse List) b) Erarbeitung der Funktionalität der Liste unter Verwendung der Klasse List c) Modellierung und Implementierung einer Anwendung mit Objekten der Klasse List d) Modellierung und Implementierung einer Anwendung unter Verwendung verschiedener Datenstrukturen (Stack, Queue, List)		**Aufgaben** • LEDs • Textzeilen verarbeiten

5. Übungen und Vertiefungen zur Verwendung linearer und dynamischer Datenstrukturen anhand weiterer Problemstellungen		Prüfungsvorbereitung Projekteinstieg Heldenspiel Mit dem Heldenspiel können alle im Kapitel behandelten Datenstrukturen erarbeitet werden. Das Spiel kann bis zu einem beliebigen Grad realisiert werden, sodass es sowohl als Einstieg als auch als ein umfassendes Projekt für lineare Datenstrukturen genutzt werden kann.

Unterrichtsvorhaben Q1-III

Thema:
Algorithmen zum Suchen und Sortieren auf linearen Datenstrukturen

Leitfragen:
Nach welchen Grundprinzipien können Algorithmen strukturiert werden? Welche Qualitätseigenschaften sollten Algorithmen erfüllen? Wie können mithilfe von Such- und Sortieralgorithmen Daten in linearen Strukturen effizient (wieder-)gefunden werden?

Vorhabenbezogene Konkretisierung:
Zunächst werden anhand eines Anwendungsbeispiels übergreifende Algorithmuseigenschaften (wie Korrektheit, Effizienz und Verständlichkeit) erarbeitet und Schritte der Algorithmusentwicklung wiederholt. Dabei kommen Struktogramme zur Darstellung von Algorithmen zum Einsatz.
Als besondere Struktur von Algorithmen wird die Rekursion an Beispielen veranschaulicht und gegenüber der Iteration abgegrenzt. Rekursive Algorithmen werden von den Schülerinnen und Schülern analysiert und selbst entwickelt.
In der zweiten Unterrichtssequenz geht es um die Frage, wie Daten in linearen Strukturen (lineare Liste und Array) (wieder-)gefunden werden können. Die lineare Suche als iteratives und die binäre Suche als rekursives Verfahren werden veranschaulicht und implementiert. Die Bewertung der Algorithmen erfolgt, indem jeweils die Anzahl der Vergleichsoperationen und der Speicherbedarf ermittelt wird.
Möchte man Daten effizient in einer linearen Struktur wiederfinden, so rückt zwangsläufig die Frage nach einer Sortierstrategie in den Fokus. Es wird mindestens ein iteratives und ein rekursives Sortierverfahren erarbeitet und implementiert sowie ihre Effizienz bewertet.

Zeitbedarf: 25 Stunden

Sequenzierung des Unterrichtsvorhabens: s. nächste Seite

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Eigenschaften von Algorithmen a) Qualitätseigenschaften von Algorithmen b) Strukturierung von Algorithmen mit Hilfe der Strategien „Modularisierung“ und „Teile und Herrsche“ c) Analyse und Entwicklung von rekursiven Algorithmen	Die Schülerinnen und Schüler • analysieren und erläutern Algorithmen und Programme (**A**), • modifizieren Algorithmen und Programme (**I**), • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (**D**), • entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“ und „Teilen und Herrschen“ (**M**), • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (**I**), • testen Programme systematisch anhand von Beispielen (**I**). • implementieren und erläutern iterative und rekursive Such- und Sortierverfahren (**I**), • beurteilen die Effizienz von Algorithmen unter Berücksichtigung des Speicherbedarfs und der Zahl der Operationen (**A**), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (**A**), • 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**),	
2. **Suchen** in Listen und Arrays a) Lineare Suche in Listen und Arrays b) Binäre Suche in einem Array c) Untersuchung der beiden Verfahren bzgl. Laufzeit und Speicherplatzbedarf		
3. **Sortieren** auf Listen und Arrays a) Entwicklung und Implementierung eines iterativen Sortierverfahrens für eine Liste b) Entwicklung und Implementierung eines rekursiven Sortierverfahrens für ein Array c) Untersuchung der beiden Verfahren bzgl. Laufzeit und Speicherplatzbedarf		

Unterrichtsvorhaben Q1-IV

Thema: Aufbau von und Kommunikation in Netzwerken und Modellierung und Implementierung von Client-Server-Anwendungen

Leitfragen:
Was macht menschliche Kommunikation aus? Welchen Stellenwert haben technische/ informatische Hilfsmittel für die Kommunikation? Wie werden Daten in einem Netzwerk zwischen den Kommunikationspartnern übertragen? Wie ist die Arbeitsteilung in Netzwerken gestaltet? Wie kann sicher in Netzwerken kommuniziert werden?

Vorhabenbezogene Konkretisierung:
Ausgehend von alltäglicher Face-to-Face-Kommunikation werden die Grundprinzipien sowie die Bewertungskriterien von Kommunikation erläutert. Das Netzwerk wird als vorteilhafte Kommunikationsstruktur dargestellt und anhand von Topologien und Reichweiten kategorisiert. Ausgehend davon wird der Protokollbegriff entwickelt und anhand des TCP/IP-Schichtenmodells analysiert. Anschließend wird das Client-Server-Prinzip vorgestellt und angewandt.
Sichere Kommunikation in Netzen ist nur dank kryptografischer Verfahren möglich. Stellvertretend werden zwei symmetrische und ein asymmetrisches Verfahren erläutert, angewandt und bewertet.

Zeitbedarf 24 Stunden

Sequenzierung des Unterrichtsvorhabens: s. nächste Seite

Unterrichtssequenz	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Technische Kommunikation als Fortführung natürlicher Kommunikation a. Kommunikation im Shannon-Weaver-Modell b. Kriterien von technischen Kommunikationsarten c. Die Geschichte der technischen Kommunikation	Die Schülerinnen und Schüler • beschreiben und erläutern Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken (**A**), • nutzen bereitgestellte Informatiksysteme und das Internet reflektiert zum Erschließen, zur Aufbereitung und Präsentation fachlicher Inhalte (**D**), • analysieren und erläutern Eigenschaften und Einsatzbereiche symmetrischer und asymmetrischer Verschlüsselungsverfahren (**A**). • analysieren und erläutern Protokolle zur Kommunikation in einem Client-Server-Netzwerk (A), • entwickeln und implementieren Algorithmen und Methoden zur Client-Server-Kommunikation (I).	
2. Aufbau von Netzwerken und Kommunikationsregeln a. Das Netzwerk als Organisationsprinzip der Kommunikation und Möglichkeiten der Ausformung b. Geregelte technische Kommunikation durch Protokolle in Schichtenmodellen		
3. Aufgabenteilung in Netzwerken durch Server und Client im Anwendungskontext a. Aufbau und Aufgaben der Client-Server-Struktur b. Protokolle zwischen Client und Server		Umsetzung eines größeren Projektes, z.B. • Echo-Anwendung • Chat Anwendung (Nutzung der List) • Beliebige Client-Server-Anwendung
4. Kryptologie a. Veranschaulichen und Anwenden von symmetrischen und asymmetrischen kryptographischen Verfahren (Caesar, Vigenère, RSA, Diffie-Hellman) b. Bewertung der Verfahren hinsichtlich ihrer Sicherheit und ihrem Aufwand		
5. Übung und Vertiefung des Aufbaus von und der Kommunikation in Netzwerken		

Unterrichtsvorhaben Q2-I

Thema:
Organisation und Verarbeitung von Daten II – Modellierung und Implementierung von Anwendungen mit dynamischen nicht-linearen Datenstrukturen

Leitfragen:
Wie können Daten mithilfe von Baumstrukturen verwaltet werden? Wie können mit binären Suchbäumen Inhalte sortiert verwaltet werden und welche Vor- und Nachteile bietet dies?

Vorhabenbezogene Konkretisierung:
Anhand des Anwendungskontextes Spielbäume werden zunächst der generelle Aufbau von Baumstrukturen (auch nicht-binäre) und wichtige Grundbegriffe erarbeitet. Die Darstellung von Bäumen mit Knoten und Kanten wird eingeführt.

Anschließend rückt der Fokus auf die binären Bäume, deren rekursiver Aufbau für die Traversierung der Datenstruktur genutzt wird. Die Preorder-Traversierung wird verwendet, um einen gespeicherten Inhalt in einem Binärbaum zu finden (Tiefensuche).
Der Anwendungskontext Ahnenbaum wird mithilfe der Klasse BinaryTree (der Materialien für das Zentralabitur in NRW) modelliert und (ggf. in Teilen) implementiert. Dabei wird u. a. die Erzeugung eines Binärbaums mithilfe der beiden Konstruktoren der Klasse BinaryTree thematisiert.

Möchte man Daten geordnet speichern, so bietet sich die Struktur des binären Suchbaums an. An Beispielen wird zunächst das Prinzip des binären Suchbaums erarbeitet. Die Operationen des Suchens, Einfügens, Löschens und der sortierten Ausgabe werden thematisiert.
Um Daten in einem Anwendungskontext mithilfe eines binären Suchbaums verwalten zu können, müssen sie in eine Ordnung gebracht werden können, d. h. sie müssen vergleichbar sein. Diese Vorgabe wird mithilfe des Interfaces Item realisiert, das alle Klassen, dessen Objekte in einem Suchbaum verwaltet werden sollen, implementieren müssen. Auf diese Weise wird ein Anwendungskontext (Benutzerverwaltung) mithilfe der Klassen BinarySearchTree und Item modelliert und (ggf. in Teilen) implementiert.

Zeitbedarf25 Stunden

Sequenzierung des Unterrichtsvorhabens: s. nächste Seite

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Aufbau von Baumstrukturen und Grundbegriffe a) Erarbeitung der Begriffe Wurzel, Knoten, Blatt, Kante, Grad eines Knotens und eines Baumes, Pfad, Tiefe, Ebene, Teilbaum b) Aufbau und Darstellung von Baumstrukturen in verschiedenen Anwendungskontexten	Die Schülerinnen und Schüler • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D), • erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (A), • analysieren und erläutern Algorithmen und Programme (A), • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D). • 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), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I),	
2. Binäre Bäume a) rekursiver Aufbau eines binären Baums b) Traversierungen (pre-, in-, postorder) c) Modellierung eines Binärbaums in einem Anwendungskontext mit Hilfe der Klasse BinaryTree (als Entwurfs- und Implementationsdiagramm) d) Implementation einer Anwendung der Datenstruktur binärer Baum (ggf. in Teilen)		
3. Binäre Suchbäume a) Prinzip des binären Suchbaums, Ordnungsrelation b) Operationen auf dem binären Suchbaum (Suchen, Einfügen, Löschen, sortierte Ausgabe) c) Modellierung eines binären Suchbaums in einem Anwendungskontext mit Hilfe der Klasse BinarySearchTree (als Entwurfs- und Implementationsdiagramm) und dem Interface Item d) Implementation einer Anwendung der Datenstruktur binärer Suchbaum (ggf. in Teilen)		

4. Übung und Vertiefungen der Verwendung von Binärbäumen oder binären Suchbäumen anhand weiterer Problemstellungen	• testen Programme systematisch anhand von Beispielen (I),	

Unterrichtsvorhaben Q2-II

Thema:

Modellierung und Implementierung dynamischer nichtlinearer Datenstrukturen am Beispiel der Graphen

Leitfragen:

Bei welchen Problemstellungen reichen die bekannten Datenstrukturen nicht aus? Welche Möglichkeiten gibt es, flexibel miteinander verknüpfte Informationen zu verwalten? Wie hängen die Datenstrukturen Graph, Baum und und lineare Strukturen (Liste, Queue, Stack) zusammen?

Vorhabenbezogene Konkretisierung:

Nach Analyse einer Problemstellung in einem geeigneten Anwendungskontext (z. B. Adventure-Game, Wolf, dem Schaf und dem Kohlkopf-Problem, das Eulerkreisproblem, Nikolausproblem), in dem Daten in Form eines Graphen verwaltet werden, werden der Aufbau (Knoten, Kanten, gerichtete und ungerichtete Graphen, Teilgraphen, Knotengrad, Wege und Kreise) behandelt. Anhand exemplarischer Problemstellungen wird die Repräsentation von Graphen (Adjazenzliste/-matrix) analysiert. Die Operationen der Klasse Graph werden erläutert und im Anwendungszusammenhang bei der Lösung grundlegender Probleme (z.B. Zyklensuche, topologisches Sortieren) genutzt. Dabei werden Standardtraversierungsverfahren (Tiefen- und Breitensuche) exemplarisch herausgearbeitet. In einem geeigneten Anwendungskontextes (z.B. Navigationssystem) wird die kürzeste Wege-Problematik mihilfe geeigneter Algorithmen (Dijkstra, Kruskal, Prim) unter Nutzung der Klasse Graph behandelt.

Zeitbedarf: 30 Stunden
Sequenzierung des Unterrichtsvorhabens: s. nächste Seite

Unterrichtssequenz	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Analyse von Graphenstrukturen in verschiedenen Kontexten a) Erarbeitung der Begriffe Knoten, Kanten, gerichtete und ungerichtete Graphen, Teilgraphen, Knotengrad, Wege und Kreise im Anwendungskontext. b) Modellierung eines Graphen als Entwurfs- und Implementationsdiagramm c) Implementation der Graphklassen	Die Schülerinnen und Schüler • analysieren einen gegebenen Anwendungskontext (**A**) • stellen das hinter dem Anwendungskontext liegende Problem unter Zuhilfenahme der ihnen bekannten nichtlinearen Struktur Baum dar (**D**) • modellieren Klassen (Graph, Node, Edge) mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (**M**) • dokumentieren Klassen durch Beschreibung der Funktonalität der Methoden (**D**) • ermitteln bei der Analyse von Graphen deren Eigenschaften (**M**) • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (**I**).	Beispiele: - Eulerkreisproblem - Das Haus vom Nikolaus - Adventure-Game Medien: Materialien: - ZA Graphklassen
2. Repräsentation von Graphen und Suchstrategien auf Graphen a) Adjazenzliste und –matrix als Repräsentationen eines Graphen b) Anpassung der Graphklassen c) Suchstrategien auf Graphen d) **Exkurs**: Zyklensuche und topologisches Sortieren: Anwendungen von Suchstrategien auf Graphen	Die Schülerinnen und Schüler • ordnen Daten linearen und nichtlinearen Datensammlungen zu (**M**), • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (**D**) • analysieren und erläutern Algorithmen und Programme (**A**), • entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“ und „Teile und Herrsche“ und „Backtracking“ (**M**), • implementieren und erläutern rekursive Suchverfahren auf Graphen unter	Arbeitsblätter zu … - Adjazenzmatrix/-liste - Tiefen-/Breitensuche - Topologisches Sortieren - Zyklensuche Materialien … - JAVA-Quellcode für Zyklensuche - JAVA-Quellcode für das topologische Sortieren

	Berücksichtigung dynamischer Datenstrukturen (**I**), • dokumentieren Klassen (**D**), • beurteilen den Einsatz von Suchstrategien auf Graphen im Anwendungskontext (**A**) • interpretieren Fehlermeldungen und korrigieren den Quellcode (**I**), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (**I**),	
3. Die Datenstruktur des Graphen im Anwendungskontext unter Nutzung der Klasse Graph a) Ermittlung minimaler Verbindungskosten als Standardanwendung für Graphen b) Die Datenstruktur PriorityQueue im Anwendungskontext c) Modellierung und Implementierung verschiedener Problemstellungen unter Verwendung der Graphklassen	Die Schülerinnen und Schüler • modellieren Anwendungssituationen als Graphen (**M**) • modellieren die PriorityQueue unter Verwendung von Vererbung (**M**) • analysieren und erläutern Algorithmen zur Bestimmung kürzester Wege (**A**), • implementieren den Dijsktraalgorithmus zur Bestimmung des kürzesten Weges unter Nutzung der Graphklassen (**I**), • testen Programme systematisch anhand von Beispielen (**I**), • dokumentieren Klassen (**D**), • modellieren Klassen (PriorityQueue, Dijkstratupel) mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M), • strukturieren den Arbeitsprozess, vereinbaren Schnittstellen und führen Ergebnisse zusammen (**K**),	Projektarbeit in einem größeren Kontext: Arbeitsblätter - Der chinesische Kaiser Priori-Tii Kiu und seine Tochter Dijsktra - Das Wanderwegproblem - Länge von Telefonleitungen, Wasserleitungen, … - Minimale Spannbäume Materialien - JAVA-Quellcode für PriorityQueue - JAVA-Quellcode für DIjkstraTupel

Unterrichtsvorhaben Q2-III

Thema: Nutzung und Modellierung von relationalen Datenbanken in Anwendungskontexten

Leitfragen: Was sind Datenbanken und wie kann man mit ihnen arbeiten? Wie entwickelt man selbst eine Datenbank für einen Anwendungskontext?

Vorhabenbezogene Konkretisierung:
Am Beispiel eines Online-Buchhandels werden der Aufbau einer Datenbank sowie wichtige Grundbegriffe erarbeitet. Die Schülerinnen und Schüler nehmen dabei zunächst die Sicht der Anwender an, die eine bestehende Datenbank beschreiben und analysieren und mithilfe von SQL-Abfragen Daten gezielt herausfiltern.
Mithilfe des Projekteinstiegs „Tabellen“ können bereits zu einem frühen Zeitpunkt des Unterrichtsvorhabens Redundanzen, Inkonsistenzen und Anomalien problematisiert werden.

Nachdem die Lernenden in der ersten Sequenz mit Datenbanken vertraut gemacht wurden, nehmen sie nun die Rolle der Entwickler an, indem sie selbst Datenbanken von Grund auf modellieren und das Modell in ein Relationenschema überführen. Sie arbeiten mit Entity-Relationship-Diagrammen, um Entitäten, zugehörige Attribute, Relationen und Kardinalitäten in Anwendungskontexten darzustellen. Gegebene ER-Diagramme werden analysiert, erläutert und modifiziert.
Der bereits in der ersten Sequenz problematisierte Begriff der Redundanz wird am Ende des Unterrichtsvorhabens wieder aufgegriffen, um die Normalisierung von Datenbanken zu thematisieren. Bestehende Datenbankschemata werden hinsichtlich der 1. bis 3. Normalform untersucht und (soweit nötig) normalisiert.

Zeitbedarf: 25 Stunden

Sequenzierung des Unterrichtsvorhabens: s. nächste Seite

Unterrichtssequenz	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Nutzung von relationalen Datenbanken a. Aufbau von Datenbanksystemen und Grundbegriffe • Aufgaben und Eigenschaften eines Datenbanksystems • Erarbeitung der Begriffe Tabelle, Attribut, Attributwert, Datensatz, Datentyp, Primärschlüssel, Datenbankschema • Problematisierung von Redundanzen, Anomalien und Inkonsistenzen b. SQL-Abfragen • Erarbeitung der grundlegenden Sprachelemente von SQL (SELECT(DISTINCT), FROM, WHERE, JOIN) • Analyse und Erarbeitung von SQL-Abfragen (AND, OR, NOT, UNION, AS, GROUP BY,ORDER BY, ASC, DESC, COUNT, MAX, MIN, SUM, Arithmetische Operatoren: +, -, *, /, (…), Vergleichsoperatoren: =, <>, >, <, >=, <=, LIKE, BETWEEN, IN, IS NULL, geschachtelte Select-Ausdrücke) c. Vertiefung an einem weiteren Datenbankbeispiel	Die Schülerinnen und Schüler • erläutern die Eigenschaften und den Aufbau von Datenbanksystemen unter dem Aspekt der sicheren Nutzung (**A**), • analysieren und erläutern die Syntax und Semantik einer Datenbankabfrage (**A**), • analysieren und erläutern eine Datenbankmodellierung (**A**), • erläutern die Eigenschaften normalisierter Datenbankschemata (**A**), • bestimmen Primär- und Sekundärschlüssel (**M**), • ermitteln für anwendungsbezogene Problemstellungen Entitäten, zugehörige Attribute, Relationen und Kardinalitäten (**M**), • modifizieren eine Datenbankmodellierung (**M**), • modellieren zu einem Entity-Relationship-Diagramm ein relationales Datenbankschema (**M**), • überführen Datenbankschemata in vorgegebene Normalformen (**M**), • implementieren ein relationales Datenbankschema als Datenbank (I),	Aufgaben

2. Modellierung von relationalen Datenbanken a. Datenbankentwurf durch ER-Diagramme • Ermittlung von Entitäten, zugehörigen Attributen, Beziehungen und Kardinalitäten in Anwendungssituationen und Modellierung eines Datenbankentwurfs in Form eines Entity-Relationship-Diagramms • Erläuterung und Erweiterung einer Datenbankmodellierung b. Entwicklung eines relationalen Modells aus einem Datenbankentwurf • Überführung eines Entity-Relationship-Diagramms in ein relationales Datenbankschema inklusive der Bestimmung von Primär- und Fremdschlüsseln • Normalformen c. 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 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**).	

3. Gesellschaftliche Auswirkungen der Nutzung von Datenbanksystemen	Die Schülerinnen und Schüler • untersuchen und bewerten anhand von Fallbeispielen • Auswirkungen des Einsatzes von Informatiksystemen • sowie Aspekte der Sicherheit von Informatiksystemen, • des Datenschutzes 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).	
4. Übung und Vertiefung der Nutzung und Modellierung von relationalen Datenbanken		Prüfungsvorbereitung

Unterrichtsvorhaben Q2 IV

Thema:
Automaten und formale Sprachen

Leitfragen:
Wie lassen sich reale Automaten durch ein Modell formal beschreiben? Wie kann die Art und Weise, wie ein Computer Zeichen (Eingaben) verarbeitet, durch Automaten dargestellt werden? Welche Eigenschaften besitzen Automaten und was können sie leisten? Wie werden sie dargestellt? Wie werden reguläre Sprachen/kontextfreie Sprachen durch eine Grammatik beschrieben? In welchem Verhältnis stehen endliche Automaten/Kellerautomaten und Grammatiken? Welche Anwendungsfälle können durch endliche Automaten und Grammatiken regulärer Sprachen bzw. kontextfreier Sprachen beschrieben werden und welche nicht?

Vorhabensbezogene Konkretisierung:
Ausgehend von der Beschreibung und Untersuchung realer Automaten wird das formale Modell eines endlichen Automaten entwickelt. Neben dem Mealy-Automaten geht es vor allem um den erkennenden endlichen Automaten. Auf die Erarbeitung der Beschreibung folgt die Modellierung
eigener Automaten und die Untersuchung bestehender, um die Eigenschaften und Grenzen eines endlichen Automaten zu erkennen. Hierbei wird dessen Verhalten auf bestimmte Eingaben analysiert.
An den Themenkomplex Endliche Automaten schließt sich die Erarbeitung von Grammatiken regulärer Sprachen an. Die Untersuchung beginnt bei der Erschließung der formalen Beschreibung und wird mit der Entwicklung von Grammatiken zu regulären Sprachen fortgeführt. Hierbei wird auch die Beziehung von Grammatiken regulärer Sprachen zu endlichen Automaten an Beispielen erarbeitet und analysiert. Hierzu gehört auch die Untersuchung, welche Problemstellungen durch endliche Automaten und reguläre Grammatiken beschrieben werden können und welche nicht.
Der Scanner wird in Form von endlichen Automaten, der Parser in Form einer kontextfreien Grammatik modelliert, implementiert bzw. analysiert.
Aufbauend auf die Erkenntnis der Begrenztheit des Modells der regulären Sprachen wird die nächste Ebene in der Chomsky-Hierarchie betrachtet: kontextfreie Sprachen bzw. Grammatiken und die mit dieser Sprachebene korrespondierenden Kellerautomaten.

Zeitbedarf: 35 Stunden

Sequenzierung des Unterrichtsvorhabens: s. nächste Seite

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Endliche Automaten a) Erarbeitung der formalen Beschreibung eines Mealy-Automaten und der Darstellungsformen b) Erarbeitung der formalen Beschreibung eines deterministischen endlichen Automaten (DEA) (Moore-Automat) sowie dessen Darstellungsformen; Erschließung der Fachbegriffe Alphabet, Wort, (akzeptierte) Sprache, Determinismus c) Analyse der Eigenschaften von DEA durch die Modellierung eines Automaten zu einer gegebenen Problemstellung, der Modifikation eines Automaten sowie die Überführung der gegebenen Darstellungsform in eine andere	Die Schülerinnen und Schüler • stellen informatische Modelle und Abläufe in Texten, Tabellen, Diagrammen und Grafiken dar (D) • analysieren und erläutern die Eigenschaften endlicher Automaten bzw. Kellerautomaten einschließlich ihres Verhaltens bei bestimmten Eingaben (A) • ermitteln die Sprache, die ein endlicher Automat akzeptiert, bzw. die ein Kellerautomat akzeptiert(D) • entwickeln und modifizieren zu einer Problemstellung endlicher Automaten (M) • stellen endliche Automaten in Tabellen und Graphen dar und überführen sie in die jeweils andere Darstellungsform (D) • entwickeln zur Grammatik einer regulären Sprache einen zugehörigen endlichen Automaten (M)	

2. Grammatiken regulärer Sprachen a) Erarbeitung der formalen Beschreibung einer regulären Grammatik (Sprache, Terminal und Nicht-Terminal, Produktionen und Produktionsvorschriften) b) Analyse der Eigenschaften einer regulären Grammatik durch deren Entwicklung und Modellierung zu einer gegebenen Problemstellung. c) Scanner, Parser, Interpreter für reguläre Sprachen d) Erweiterung des Automatenmodells hin zu Kellerautomaten und des Modells der regulären Grammatik hin zu kontextfreien Grammatik	• entwickeln zur Grammatik einer kontextfreien Sprache einen zugehörigen Kellerautomaten (M) • analysieren und erläutern Grammatiken regulärer Sprachen bzw. Kontextfreier Sprachen (A) • modifizieren Grammatiken regulärer Sprachen bzw. kontextfreier Sprachen (M) • entwickeln zu einer regulären Sprache eine Grammatik, die die Sprache erzeugt (M) • entwickeln zu einer kontextfreien Sprache eine Grammatik, die die Sprache erzeugt (M) • entwickeln zur akzeptierten Sprache eines Automaten eine zugehörige Grammatik (M) • beschreiben an Beispielen den Zusammenhang zwischen Automaten und Grammatiken (D) • zeigen die Grenzen endlicher Automaten und regulärer Grammatiken auf • analysieren und entwickeln Scanner, Parser und Interpreter für Beispielsprachen [regulär, kontextfrei] (A, I) • erkennen, dass sich bestimmte Sprachen nicht mehr mit dem Modell der regulären Sprache erfassen lassen (A) • analysieren und erläutern Grammatiken kontextfreier Sprachen (A)	
3. Übungen und Vertiefungen Verwendung endlicher Automaten und Grammatiken regulärer Sprachen Projekteinstieg Erarbeitung der formalen Beschreibung und Überprüfung des Verhaltens eines erkennenden Automaten auf bestimmte Eingaben		

2.2 Grundsätze der fachmethodischen und fachdidaktischen Arbeit

In Absprache mit der Lehrerkonferenz sowie unter Berücksichtigung des Schulprogramms hat die Fachkonferenz Informatik des Antonianum 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:

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

Fachliche Grundsätze:

- Der Unterricht unterliegt der Wissenschaftsorientierung und ist dementsprechend eng verzahnt mit seiner Bezugswissenschaft.
- Der Unterricht ist problemorientiert und soll von realen Problemen ausgehen und sich auf solche rückbeziehen.
- Der Unterricht folgt dem Prinzip der Exemplarizität und soll ermöglichen, informatische Strukturen und Gesetzmäßigkeiten in den ausgewählten Problemen und Projekten zu erkennen.
- Der Unterricht ist anschaulich sowie gegenwarts- und zukunftsorientiert und gewinnt dadurch für die Schülerinnen und Schüler an Bedeutsamkeit.
- Der Unterricht ist handlungsorientiert, d.h. projekt- und produktorientiert angelegt.
- Im Unterricht werden sowohl für die Schule didaktisch reduzierte als auch reale Informatiksysteme aus der Wissenschafts-, Berufs- und Lebenswelt eingesetzt.
- 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 § 48 SchulG, §13 - §16 der APO-GOSt sowie Kapitel 3 des Kernlehrplans Informatik für die gymnasiale Oberstufe hat die Fachkonferenz des Antonianum 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.

Beurteilungsbereich Klausuren

Verbindliche Absprachen:

- Bei der Formulierung von Aufgaben werden die für die Abiturprüfungen geltenden Operatoren des Faches Informatik schrittweise eingeführt, erläutert und im Rahmen der Aufgabenstellungen für die Klausuren benutzt.
- Die Aufgaben für Klausuren in parallelen Grund- bzw. Leistungskursen werden im Vorfeld abgesprochen und nach Möglichkeit gemeinsam gestellt.

Anzahl und Umfang der Klausuren

Einführungsphase: 1 Klausur pro Halbjahr (zwei Unterrichtsstunden)

Qualifikationsphase I:

Grundkurs: 2 Klausuren pro Halbjahr (zwei Unterrichtsstunden)
Leistungskurs: 2 Klausuren pro Halbjahr (drei Unterrichtsstunden)
Anstelle einer Klausur kann gemäß dem Beschluss der Lehrerkonferenz in Q1.2 eine Facharbeit geschrieben werden.

Qualifikationsphase II:

1. Halbjahr:
   - Grundkurs: 2 Klausuren pro Halbjahr (drei Unterrichtstunden)
   - Leistungskurs: 2 Klausuren pro Halbjahr (vier Unterrichtstunden)
2. Halbjahr:
   - Grundkurs: 1 Klausur unter Abiturbedingungen
   - Leistungskurs: 1 Klausur unter Abiturbedingungen

Die Aufgabentypen sowie die Anforderungsbereiche I-III sind entsprechend den Vorgaben in Kapitel 3 des Kernlehrplans zu beachten. In der Qualifikationsphase orientieren sich die Aufgabentypen zunehmend an den Vorlagen des zentralen Abiturs.

Folgende Korrekturzeichen werden verwendet

Fehlerarten (fachlich)

f	Rechen- oder Umformungsfehler
Γ	fehlender Teil
#	fehlende Aufgabe / fehlender Aufgabenteil
s.o.	Wiederholungsfehler

Bewertung der Fehler

–	leichter Fehler
|	„normaler“ Fehler
+	schwerer Fehler oder Fehler, der den weiteren Lösungsweg entscheidend verändert

Sonstige Bewertungszeichen

✓	richtiger Zwischenschritt, richtiges Ergebnis
(✓)	Richtig aus einem fehlerhaften Zwischenergebnis weiter geschlossen oder Zwischenschritt oder Ergebnis richtig, jedoch unvollständiger oder fehlerhafter Lösungsweg oder richtiges Ergebnis, jedoch nicht zur Aufgabenlösung nötig

Sprachliche Fehler

R	Rechtschreibfehler
Z	Zeichensetzungsfehler
Gr	Grammatikfehler
ugs	umgangssprachlich
Sb	Satzbau
A	Ausdruck
Bz	Beziehung
Fs	Fehler in der Fachsprache

Kriterien

Die Bewertung der schriftlichen Leistungen in den Klausuren erfolgt über ein Raster mit Punkten, die im Erwartungshorizont den einzelnen Kriterien zugeordnet sind.
Die Zuordnung der Punktesumme zu den Notenstufen orientiert sich 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 Notenzuordnung orientiert sich an den Vorgaben für das Zentralabiturs:

Note	Punkte	Erreichte Punktzahl
sehr gut plus	15	100 – 95
sehr gut	14	94 – 90
sehr gut minus	13	89 – 85
gut plus	12	84 – 80
gut	11	79 – 75
gut minus	10	74 – 70
befriedigend plus	9	69 – 65
befriedigend	8	64 – 60
befriedigend minus	7	59 – 55
ausreichend plus	6	54 – 50
ausreichend	5	49 – 45
ausreichend minus	4	44 – 40
mangelhaft plus	3	39 – 34
mangelhaft	2	33 – 27
mangelhaft minus	1	26 – 20
ungenügend	0	19 – 0

Beurteilungsbereich Sonstige Mitarbeit

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

Im Unterricht gibt es vielfältige Möglichkeiten für die Schülerinnen und Schüler zu zeigen, wie weit sie ihrem Alter angemessen über fachspezifische Kompetenzen verfügen.
Die Bewertung der sonstigen Mitarbeit erfolgt im Wesentlichen anhand der folgenden Kriterien:

- mündliche Mitarbeit zum Unterricht, z.B.
  - Anwenden fachspezifischer Methoden und Arbeitsweisen
  - Einbringen kreativer Ideen
  - konstruktives Umgehen mit Fehler
  - Finden von Beispielen oder Gegenbeispielen
  - verständliches und präzises Darstellen, Erläutern von Lösungen
  - Veranschaulichen, Zusammenfassen und Beschreiben informatischer Sachverhalte
  - Verfügbarkeit informatischen Grundwissens (Begriffe, Strukturen, Algorithmen)
  - angemessenes Verwenden informatischer Fachsprache
  - Erläutern von Hausaufgaben, z. B. verständliches Vortragen der Lösungswege; (schriftliches)
  - Belegen von Schwierigkeiten bei ungelösten Hausaufgaben, sachgerechtes Einbringen von Lösungen bei unterrichtsvorbereitenden Aufgaben
  - zielgerichtetes Beschaffen von Informationen (z.B. Internet, Lexika, Schulbuch, Umfragen)
  - fehlerfreies Anwenden geübter Fertigkeiten

- sonstige Beiträge zum Unterricht, z. B.
    - Mitarbeit in Partner-/ Gruppenarbeitsphasen
    - Implementierung, Test und Anwendung von Informatiksystemen
    - Präsentation von Ergebnissen von Partner- oder Gruppenarbeiten und deren Darstellung
    - Unterrichtsdokumentation (z.B. Heftführung, Lerntagebuch)
    - Organisation der Unterrichtsmaterialien und Projekte im Homeverzeichnis
    - kurze schriftliche Überprüfungen, z.B. als Kurzübung im Moodle zur Wiederholung der vorangegangenen 2-4 Stunden.

Notenermittlung

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,
- nach Abschluss eines Projektes,
- bei auffälligen Leistungsveränderungen,
- auf Anfrage,
- als Quartalsfeedback und
- zu Eltern- oder Schülersprechtagen,
- zur Unterstützung und Würdigung individueller Fortschritte.

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-und Förderempfehlung

erfolgen.

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

3. Entscheidungen zu fach- und unterrichtsübergreifenden Fragen

Fach- und aufgabenfeldbezogene sowie übergreifende Absprachen, z.B. zur Arbeitsteilung bei der Entwicklung crosscurricularer Kompetenzen (ggf. Methodentage, Projektwoche, Facharbeitsvorbereitung, Schulprofil usw.)

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

Zusammenarbeit mit anderen Fächern
Im Informatikunterricht werden Kompetenzen anhand informatischer Inhalte in verschiedenen Anwendungskontexten erworben, in denen Schülerinnen und Schülern aus anderen Fächern Kenntnisse mitbringen können. Diese können insbesondere bei der Auswahl und Bearbeitung von Softwareprojekten berücksichtigt werden und in einem hinsichtlich der informatischen Problemstellung angemessenem Maß in den Unterricht Eingang finden. Da im Inhaltsfeld Informatik, Mensch und Gesellschaft auch gesellschaftliche und ethische Fragen im Unterricht angesprochen werden, besteht eine Zusammenarbeit mit den Fächern Sozialwissenschaften und Philosophie, z.B. im Angebot eines Projektkurses Informatik / Philosophie.

Projekttage
In der Regel werden alle zwei Jahre am Antonianum Projekttage angeboten. Die Fachkonferenz Informatik bietet in diesem Zusammenhang Projekte für Schülerinnen und Schüler der gymnasialen Oberstufe an.

Vorbereitung auf die Erstellung der Facharbeit
Möglichst schon im 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. Eine Anbindung an unsere Schulpartner ist angestrebt.
Der Umfang einer Facharbeit sollte 15 Seiten nicht überschreiten. Ausgenommen sind Anhänge mit Quelltexten.
Bewertet wird: Eigenständige Vorbereitung, die Einhaltung der Formalia, die inhaltliche Darstellungsweise, die wissenschaftliche Arbeitsweise und der Ertrag der Arbeit. Die Facharbeit ersetzt die zweite Klausur im zweiten Halbjahr der QI.

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. Die Möglichkeiten des Lernlabors CoolMINT sollten genutzt werden.

4. 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.
Dieses Dokument ist zunächst bis 2017 für den ersten Durchgang durch die gymnasiale Oberstufe nach Erlass des Kernlehrplanes verbindlich. Nach Ende der Einführungsphase im Sommer 2016, 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.

Fachgruppenarbeit

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

Bedingungen und Planungen der Fachgruppenarbeit		**Ist-Zustand Auffälligkeiten**	**Änderungen/ Konsequenzen/ Perspektivplanung**	**Wer (Verantwortlich)**	**Bis wann (Zeitrahmen)**
Funktionen					
Fachvorsitz					
Stellvertretung					
Rechnerbetreuung					
Sonstige Funktionen (im Rahmen der schulprogrammatischen fächerübergreifenden Schwerpunkte)					
Ressourcen					
personell	Fachlehrkräfte				
	fachfremd				
	Lerngruppen				
	Lerngruppengröße				
	…				
	Fachräume				

Bedingungen und Planungen der Fachgruppenarbeit		Ist-Zustand Auffälligkeiten	Änderungen/ Konsequenzen/ Perspektivplanung	Wer (Verantwortlich)	Bis wann (Zeitrahmen)
Ressourcen					
räumlich	Fachräume				
materiell/ sachlich	Lehrwerke				
	Fachzeitschriften				
	Ausstattung mit Anschauungsmaterial				
zeitlich	Abstände Fachteamarbeit				
	Dauer Fachteamarbeit				
Unterrichtsvorhaben					
Leistungsbewertung/ Einzelinstrumente					
Klausuren					
Facharbeiten					
Kurswahlen					
Grundkurse					
Leistungskurse					
Projektkurse					
Leistungsbewertung/Grundsätze					
sonstige Mitarbeit					
Arbeitsschwerpunkt(e) SE					
fachintern					
- kurzfristig (Halbjahr)					
- mittelfristig (Schuljahr)					
- langfristig					
fachübergreifend					
- kurzfristig					
- mittelfristig					
- langfristig					
…					
Fortbildung					
Fachspezifischer Bedarf					
- kurzfristig					
- mittelfristig					
- langfristig					
Fachübergreifender Bedarf					
- kurzfristig					
- mittelfristig					
- langfristig					

Fortbildungsplanung

Die Fachgruppe Informatik stellt jährlich in ihrer Fachkonferenz zu Beginn des Schuljahres den Fortbildungsbedarf fest. Nachfolgend ist es Aufgabe der/des Fachvorsitzenden, ggf. zusammen mit dem/der Fortbildungsbeauftragten, der Aufgabenfeldbeauftragten und der didaktischen Leitung des Antonianums entsprechende Veranstaltungen zu organisieren. Als Rahmen hierfür gilt das allgemeine Fortbildungskonzept des Antonianums.