Schulinterner Lehrplan

zum Kernlehrplan für die gymnasiale Oberstufe am Gymnasium Porta Westfalica

Informatik

(Stand: 08.12.15)

Inhalt

1 Die Fachgruppe Informatik am Gymnasium Porta Westfalica

Das Fach Informatik wird am GPW in den Jahrgangsstufen 8 und 9 im Wahlpflichtbereich II zweistündig unterrichtet und in der Regel von etwa der Hälfte der Schüler besucht. In der zweijährigen Laufzeit dieser Kurse wird in altersstufengerechter Weise unter anderem auf Grundlagen der Algorithmik am Beispiel … eingegangen.

In der Sekundarstufe II bietet das GPW für die Schüler in allen Jahrgangsstufen Grundkurse in Informatik an und ermöglicht mündliche wie schriftliche Abiturprüfungen in diesem Fachgebiet.

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

Im Unterricht der Sekundarstufe II wird in der Softwareentwicklung primär mit der Programmiersprache Java gearbeitet, aber auch andere Programmiersprachen finden Anwendung.

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

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

Zurzeit besteht die Fachschaft Informatik am GPW aus drei Lehrkräften, von denen eine ausschließlich die Lehrbefähigung für die Sekundarstufe I besitzt. Die Schule verfügt über zwei Rechnerräume und weitere Arbeitsplätze im Selbstlernzentrum. Über ein Lernplattformsystem haben die Schüler die Möglichkeit, Daten zwischen schulischen und privaten Geräten auszutauschen.

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

2 Entscheidungen zum Unterricht

2.1 Grundlegendes zu Unterrichtsvorhaben

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

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

Im „Übersichtsraster Unterrichtsvorhaben“ (Kapitel 2.2) wird die für alle Lehrerinnen und Lehrer gemäß Fachkonferenzbeschluss verbindliche Verteilung der Unterrichtsvorhaben dargestellt. Das Übersichtsraster dient dazu, den Kolleginnen und Kollegen einen schnellen Überblick über die Zuordnung der Unterrichtsvorhaben zu den einzelnen Jahrgangsstufen sowie den im Kernlehrplan genannten Kompetenzen, Inhaltsfeldern und inhaltlichen Schwerpunkten zu verschaffen. Der ausgewiesene Zeitbedarf versteht sich als grobe Orientierungsgröße, die nach Bedarf über- oder unterschritten werden kann. Um Freiraum für Vertiefungen, besondere Schülerinteressen, aktuelle Themen bzw. die Erfordernisse anderer besonderer Ereignisse (z.B. Praktika, Kursfahrten o.ä.) zu erhalten, wurden im Rahmen dieses schulinternen Lehrplans ca. 75 Prozent der Bruttounterrichtszeit verplant.

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

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

2.2 Übersichtsraster Unterrichtsvorhaben

2.2.1 Einführungsphase

Die Gesamtdauer der geplanten Unterrichtseinheiten beträgt 79 Stunden.

2.2.1.1 Unterrichtsvorhaben EF-1

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

Dauer:

- 6 Stunden

Zentrale Kompetenzen:

- Argumentieren
- Darstellen und Interpretieren
- Kommunizieren und Kooperieren

Inhaltsfelder:

- Informatiksysteme
- Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:

- Einzelrechner (Von-Neumann-Prinzip)
- Dateisystem
- Internet
- Einsatz von Informatiksystemen

2.2.1.2 Unterrichtsvorhaben EF-2

Thema:
Grundlagen der objektorientierten Analyse, Modellierung und Implementierung

Dauer

- 8 Stunden

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

2.2.1.3 Unterrichtsvorhaben EF-3

Thema:
Grundlagen der objektorientierten Programmierung und algorithmischer Grundstrukturen in Java

Dauer

- 18 Stunden

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

2.2.1.4 Unterrichtsvorhaben EF-4

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

Dauer

- 26 Stunden

Zentrale Kompetenzen:

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

Inhaltsfelder:

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

Inhaltliche Schwerpunkte:

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

2.2.1.5 Unterrichtsvorhaben EF-5

Thema:
Such- und Sortieralgorithmen anhand kontextbezogener Beispiele

Dauer

- 13 Stunden

Zentrale Kompetenzen:

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

Inhaltsfelder:

- Algorithmen

Inhaltliche Schwerpunkte:

- Algorithmen zum Suchen und Sortieren
- Analyse, Entwurf und Implementierung einfacher Algorithmen

2.2.1.6 Unterrichtsvorhaben EF-6

Thema:
Geschichte der digitalen Datenverarbeitung und die Grundlagen des Datenschutzes

Dauer

- 8 Stunden

Zentrale Kompetenzen:

- Argumentieren
- Darstellen und Interpretieren
- Kommunizieren und Kooperieren

Inhaltsfelder:

- Informatik, Mensch und Gesellschaft
- Informatiksysteme

Inhaltliche Schwerpunkte:

- Wirkungen der Automatisierung
- Geschichte der automatischen Datenverarbeitung
- Digitalisierung

2.2.2 Qualifikationsphase 1

Die Gesamtdauer der geplanten Unterrichtseinheiten beträgt 74 Stunden für den Grundkursbereich, bzw. 112 Stunden im Leistungskursbereich.

2.2.2.1 Unterrichtsvorhaben Q1-1

Thema:

Wiederholung der objektorientierten Modellierung und Programmierung anhand einer kontextbezogenen Problemstellung

Dauer

- Grundkurs: 8 Stunden
- Leistungskurs: 8 Stunden

Zentrale Kompetenzen:

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

Inhaltsfelder:

- Daten und ihre Strukturierung
- Algorithmen

- Formale Sprachen und Automaten
- Informatiksysteme

Inhaltliche Schwerpunkte:

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

2.2.2.2 Unterrichtsvorhaben Q1-2

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

Dauer

- Grundkurs: 20 Stunden
- Leistungskurs: 28 Stunden

Zentrale Kompetenzen:

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

Inhaltsfelder:

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

Inhaltliche Schwerpunkte:

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

2.2.2.3 Unterrichtsvorhaben Q1-3

Thema:
Suchen und Sortieren auf linearen Datenstrukturen

Dauer

- Grundkurs: 16 Stunden

- Leistungskurs: 26 Stunden

Zentrale Kompetenzen:

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

Inhaltsfelder:

- Algorithmen
- Formale Sprachen und Automaten

Inhaltliche Schwerpunkte:

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

2.2.2.4 Unterrichtsvorhaben Q1-4

Thema:
Modellierung und Nutzung von relationalen Datenbanken in Anwendungskontexten

Dauer

- Grundkurs: 20 Stunden
- Leistungskurs: 26 Stunden

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

2.2.2.5 Unterrichtsvorhaben Q1-5

Thema:
Sicherheit und Datenschutz in Netzstrukturen

Dauer

- Grundkurs: 10 Stunden
- Leistungskurs: 24 Stunden

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

2.2.3 Qualifikationsphase 2

Die Gesamtdauer der geplanten Unterrichtseinheiten beträgt 56 Stunden für den Grundkursbereich, bzw. 84 Stunden im Leistungskursbereich.

2.2.3.1 Unterrichtsvorhaben Q2-1

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

Dauer

- Grundkurs: 24 Stunden
- Leistungskurs: 38 Stunden

Zentrale Kompetenzen:

- Argumentieren
- Modellieren

- Implementieren
- Darstellen und Interpretieren
- Kommunizieren und Kooperieren

Inhaltsfelder:

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

Inhaltliche Schwerpunkte:

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

2.2.3.2 Unterrichtsvorhaben Q2-2

Thema:
Endliche Automaten und formale Sprachen

Dauer

- Grundkurs: 20 Stunden
- Leistungskurs: 28 Stunden

Zentrale Kompetenzen:

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

Inhaltsfelder:

- Endliche Automaten und formale Sprachen

Inhaltliche Schwerpunkte:

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

2.2.3.3 Unterrichtsvorhaben Q2-3 (12 Std)

Thema:
Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit

Dauer

- Grundkurs: 12 Stunden
- Leistungskurs: 14 Stunden

Zentrale Kompetenzen:

- Argumentieren
- Kommunizieren und Kooperieren

Inhaltsfelder:

- Informatiksysteme
- Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:

- Einzelrechner und Rechnernetzwerke
- Grenzen der Automatisierung

2.3 Konkretisierte Unterrichtsvorhaben

Im Folgenden sollen die im Kapitel 2.2 aufgeführten Unterrichtsvorhaben konkretisiert werden. Die Fachkonferenz der Beispielschule hat Themen, Leitfragen und die Ausführungen unter der Überschrift *Vorhabenbezogene Konkretisierung* verbindlich vereinbart, ebenso die Sequenzierung der Unterrichtsvorhaben (erste Tabellenspalte) und die ausgewiesenen Kompetenzen (zweite Tabellenspalte). Alle Mitglieder der Fachkonferenz haben sich darauf verständigt, in ihrem Unterricht Lerngelegenheiten anzubieten, so dass Schüler diese Kompetenzen im Rahmen der festgelegten Unterrichtssequenzen erwerben oder vertiefen können. Die angeführten Beispiele, Medien und Materialien sind dagegen Vorschläge bzw. Hilfen für die Lehrkräfte der Beispielschule. In diesen Bereichen sind Abweichungen von den vorgeschlagenen Vorgehensweisen möglich.

In der Einführungsphase und den Qualifikationsphasen werden unter anderem folgende Werkzeuge verwendet:

- Entwicklungsumgebung Greenfoot: Webseite von greenfoot.org
- Entwicklungsumgebung BlueJ: Webseite von bluej.org

Es kommen Kapitelweise folgende Lehrwerke zum Einsatz:

- Schöningh: Informatik, Band 1, 2
- Klett: Informatik, Band 3, 4 und 5

In der Qualifikationsphase werden die Unterrichtsvorhaben unter Berücksichtigung der Vorgaben für das Zentralabitur Informatik in NRW konkretisiert. Diese sind beim Schulministerium des Landes NRW zu beziehen: Download (abgerufen: 30. 04. 2014)

2.3.1 Einführungsphase

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

Die Schüler

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

2.3.1.1 Unterrichtsvorhaben EF-1

Thema

Einführung in die Nutzung von Informatiksystemen und in grundlegende Begrifflichkeiten

Leitfragen

Womit beschäftigt sich die Wissenschaft der Informatik? Wie kann die in der Schule vorhandene technische Ausstattung genutzt werden?

Vorhabenbezogene Konkretisierung

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

Nach den Einführungsbeispielen aus dem Lehrwerk Informatik 1 (Schöningh, 2. Auflage, 2014) werden die Teilgebiete des Faches Informatik anhand ausgewählter Beispiele vorgestellt. Die Schüler bekommen einen Einblick in typische Aufgabenkontexte aus der praktischen, theoretischen, technischen Informatik (TI) und dem Bereich Informatik, Mensch und Gesellschaft (IMuG).

Zeitbedarf

6 Stunden

Sequenzierung des Unterrichtsvorhabens:

A) Zu entwickelnde Kompetenzen:

Die Schüler

- nutzen die im Unterricht eingesetzten Informatiksysteme selbstständig, sicher, zielführend und verantwortungsbewusst (D),
- nutzen das Internet zur Recherche, zum Datenaustausch und zur Kommunikation (K),
- nutzen das verfügbare Informatiksystem zur strukturierten Verwaltung und gemeinsamen Verwendung von Daten unter Berücksichtigung der Rechteverwaltung (K),

B) Beispiele, Medien, Materialien:

- Unterrichtseinstieg zu den Teilgebieten der Informatik: Am Beispiel einer elektronisch unterstützten Verkehrsführung werden Aspekte aus Teilgebieten der Informatik veranschaulicht und konkretisiert::
  - Strukturieren von Daten zur Darstellung des Verkehrsnetzes in einer durch Rechner verarbeitbaren Form
  - Algorithmen zur Suche des optimalen Weges
  - Formale Sprachen und Automaten zur Beschreibung der Funktionsweise des Navigationsgeräts
  - Informatiksysteme zur Kommunikation im Netzwerk
  - Informatik, Mensch und Gesellschaft zur Veranschaulichung der Auswirkungen des Einsatzes von Informatiksystemen auf die Gesellschaft.
  -

C) Unterrichtssequenzen:

- Information, deren Kodierung und Speicherung
  - Informatik als Wissenschaft der Verarbeitung von Informationen
  - Vereinbarung von Richtlinien zur Datenspeicherung auf den Schulrechnern (z.B. Ordnerstruktur, Dateibezeichner usw.)
  - Darstellung von Informationen in Schrift, Bild und Ton
  - Speichern von Daten mit informatischen Systemen
- Erstellen von Lernplakaten zu den Teilgebieten der Informatik anhand des Beispiels „Navigation“.
-
  - Erarbeitung der Beispiele

- Recherche zu der Frage, welche Inhalte das jeweilige Teilgebiet umfasst
- Präsentation der Ergebnisse

2.3.1.2 Unterrichtsvorhaben EF-2

Thema

Grundlagen der objektorientierten Analyse und Modellierung

Leitfragen

Wie lassen sich Gegenstandsbereiche informatisch abstrahieren und in Form von Klassen und Objekten modellieren?

Vorhabenbezogene Konkretisierung

Ein zentraler Bestandteil des Informatikunterrichts der Einführungsphase ist die Objektorientierte Modellierung. Dieses Unterrichtsvorhaben führt in die Grundlagen der Analyse und Modellierung in diesem Kontext ein.

Dazu werden zunächst konkrete Gegenstandsbereiche analysiert und im Sinne des objektorientierten Paradigmas strukturiert. Es werden grundlegende Begriffe der Objektorientierung und Modellierungswerkzeuge wie Objektkarten und Klassenkarten, sowie Vererbungsbeziehungen und ihre Darstellung im Diagramm eingeführt.

Im Anschluss erfolgt eine Einführung in die Arbeit mit der Programmierumgebung *Greenfoot*. Der grundlegende Aufbau einer Java-Klasse wird thematisiert und vorhandene Klassen werden von Schülern analysiert und entsprechende Objekte anhand einfacher Problemstellungen erzeugt und erkundet. Die Schüler inspizieren Objekte, unterscheiden Objekte anhand ihres Zustands und unterscheiden Aufträge und Anfragen bei Methodenaufrufen, die eventuell Parameterwerte zur Durchführung benötigen. Die Schüler lernen mit der vorhandenen Klassendokumentation zu arbeiten.

Zeitbedarf

10 Stunden

Sequenzierung des Unterrichtsvorhabens:

A) Zu entwickelnde Kompetenzen:

Die Schüler

- ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M),
- modellieren Klassen mit ihren Attributen, ihren Methoden und Assoziationsbeziehungen (M),
- stellen die Kommunikation zwischen Objekten mithilfe von Sequenzdiagrammen grafisch dar (M),
- stellen den Zustand eines Objekts als Objektkarte dar (D).
- fassen Objekte zu Klassen zusammen.
- stellen Klassen in Entwurfsdiagrammen dar.
- analysieren und erläutern eine objektorientierte Modellierung.

B) Beispiele, Medien, Materialien:

- Material zur selbstständigen Erarbeitung der Bedeutung der Begriffe Klasse und Objekt im informatischen Kontext.
- Schöningh-Buch: Einführungsbeispiele
- Greenfoot, Dokumentation zu Greenfoot
- Greenfoot-Projekte: Wombat, Crab, Marsrover, LunarLander, Kamelrennen, etc …

C) Unterrichtssequenzen:

- Identifikation von Objekten
  - Am konkreten Beispiel werden Objekte im Sinne der Objektorientierten Modellierung eingeführt.
  - Objekte werden mit Objektkarten visualisiert und mit sinnvollen Eigenschaften (Attribute, mit Typisierung) und Fähigkeiten (Methoden: Anfragen und Aufträge) versehen.
  - Typgleiche Objekte werden zu einer Objektklasse zusammengefasst.
  - Modellierung weiterer Beispiele ähnlichen Musters
- Klassen und Objekte in Greenfoot
  - Über Recherchen und Experimente mit *Greenfoot* wird die grundlegende Funktionalität des Systems erschlossen.
  - Darstellung von Objekten in Greenfoot
  - Aufrufen von Methoden eines Greenfoot-Objekts; Typen von Methoden: Konstruktor, Anfrage, Auftrag

- Parameter als Information, die einer Methode zur Durchführung zur Laufzeit verfügbar gemacht werden muss.
- Aufbau des Quelltextes einer Klasse
- Erschließen und Verwenden der Fähigkeiten eines Objektes aus der vorhandenen Klassendokumentation

2.3.1.3 Unterrichtsvorhaben EF-3

Thema

Grundlagen der objektorientierten Programmierung und algorithmischer Grundstrukturen in Java

Leitfragen

Wie lassen sich Klassen mit ihren Eigenschaften und Fähigkeiten implementieren? Wie lässt sich das Verhalten von Objekten mithilfe der grundlegenden Kontrollstrukturen Sequenz, Fallunterscheidung und Wiederholung implementieren? Wie lassen sich Informationen mit Hilfe von Variablen speichern und bearbeiten?

Vorhabenbezogene Konkretisierung

Die Schüler modellieren das Verhalten der Objekte in einem Spiel oder spielähnlichen Szenario (Crab, Marsrover, Schatzsuche, Kamelrennen, etc...). Für die Implementation erarbeiten sich die Schüler die Prinzipien von Sequenzen, Fallunterscheidungen und gezählten wie bedingten Schleifen und deren Umsetzung in der Programmiersprache *Java*. Sie lernen, Algorithmen durch Diagramme (Nassi-Shneiderman, Programm-Ablauf-Plan) zu visualisieren.

Zeitbedarf

18 Stunden

Sequenzierung des Unterrichtsvorhabens:

A) Zu entwickelnde Kompetenzen:

Die Schüler

- analysieren und erläutern einfache Algorithmen und Programme (A),
- entwerfen einfache Algorithmen und stellen sie umgangssprachlich und grafisch dar (M),
- ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M),
- modellieren Klassen mit ihren Attributen und ihren Methoden (M),
- ordnen Attributen, Parametern und Rückgaben von Methoden einfache Daten- bzw. Objekttypen zu (M),
- implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I),
- erstellen die Dokumentation zu den selbst erstellten Klassen und Methoden unter Verwendung geeigneter Werkzeuge (z.B. javadoc),
- implementieren Algorithmen unter Verwendung von Variablen und Wertzuweisungen, Kontrollstrukturen (Unterscheidung, Schleife) sowie Methodenaufrufen (I),
- kennen die von der Programmiersprache vorgegebenen Variablentypen und verwenden diese,
- 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),
- nutzen die im Unterricht eingesetzten Informatiksysteme selbstständig, sicher, zielführend und verantwortungsbewusst (D).
- kennen Vererbung als Spezialisierung, stellen diese im UML-Diagramm symbolisch dar und setzen diese in einer Programmiersprache um.
- kennen das Variablenkonzept und weisen den Variablen die passenden Standarddatentypen zu; unterscheiden zwischen Deklaration, Initialisierung und dem lesenden und schreibenden Zugriff auf Variablen; unterscheiden zwischen Lebenszeit und Sichtbarkeit einer Variablen; unterscheiden zwischen Attributen einer Klasse, lokalen Variablen und Parametern als besondere Form lokaler Variable.
- verwenden die Punktnotation zum Aufruf von Methoden aus dem Quelltext heraus und wenden diese auch zum Aufruf von Methoden einer Oberklasse an.

B) Beispiele, Medien, Materialien:

- Schöningh: „Informatik 1“
- Greenfoot, Dokumentation zu Greenfoot
- Greenfoot-Projekte: Crab, Marsrover, Schatzsuche, Kamelrennen, etc...

C) Unterrichtssequenzen:

- Grundlagen von Java
  - Analyse von Klassen; Grundaufbau einer Java-Klasse: Attribute, Methoden, Signaturen von Methoden
  - Vererbung als Instrument der Spezialisierung; abstrakte Klassen
  - Grundlegende Sprachstrukturen und Syntax von Java; Basisdatentypen und Objekte der Klasse String
  - Deklaration und Initialisierung von Objekten und Variablen; Methodenaufrufe ohne und mit Parameterübergabe zur Manipulation von Objekteigenschaften
  - Sichtbarkeit von Attributen und Methoden
  - Variablen: Deklaration, Initialisierung, lesender und schreibender Zugriff; Attribute und Parameter als besondere Variablen; Sichtbarkeit und Lebensdauer; Grenzen des Schubladen-Modells
- Objektorientiertes Arbeiten
  - Objektorientierte Programmierung als modularisiertes Vorgehen: Entwicklung von Problemlösungen auf Grundlage von Klassen
  - Arbeiten mit der Dokumentation; Generieren von Dokumentationen mit javadoc
- Algorithmen und Programmabläufe beschreiben
  - Programm-Ablauf-Pläne (PAP) und Nassi-Shneiderman-Diagramme zur Darstellung von Algorithmenabläufen
  - Implementation einfacher Abläufe, auch unter Verwendung von Variablen, Schleifen und Unterscheidungsanweisungen
- Implementation bzw. Erweiterung eines Spiele-Projekts in Greenfoot
  - Implementierung eigener Methoden mit und ohne Parameterübergabe
  - Realisierung von Zustandsvariablen
  - Thematisierung des Geheimnisprinzips und des Autonomitätsprinzips von Objekten
  - Einfache Interaktion und Kommunikation zwischen Objekten
  - Verwenden eines Debuggers zur Fehlersuche

2.3.1.4 Unterrichtsvorhaben EF-4

Thema

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

Leitfragen

Wie lassen sich komplexere Datenflüsse und Beziehungen zwischen Objekten und Klassen realisieren? Wie kommt man von der Realität über das Modell zur Implementation?

Vorhabenbezogene Konkretisierung

Dieses Unterrichtsvorhaben beschäftigt sich im Schwerpunkt mit dem Aufbau komplexerer Objektbeziehungen. Während in vorangegangenen Unterrichtsvorhaben Objekte nur jeweils solchen Objekten Nachrichten schicken konnten, die sie selbst erstellt haben, soll in diesem Unterrichtsvorhaben diese hierarchische Struktur aufgebrochen werden. Dazu bedarf es zunächst einer präzisen Unterscheidung zwischen Objektreferenzen und Objekten, so dass klar wird, dass Dienste eines Objektes von unterschiedlichen Objekten über unterschiedliche Referenzen in Anspruch genommen werden können. Auch der Aufbau solcher Objektbeziehungen muss thematisiert werden. Durch Überschreiben von Methoden einer Oberklasse wird das Prinzip der späten Bindung veranschaulicht. Zur Konkretisierung der erstellten Modelle werden Entwurfsdiagramme in Implementationsdiagramme überführt.

Zeitbedarf

26 Stunden

Sequenzierung des Unterrichtsvorhabens:

A) Zu entwickelnde Kompetenzen:

Die Schüler

- analysieren und erläutern eine objektorientierte Modellierung (A),
- stellen die Kommunikation zwischen Objekten grafisch dar (M),
- ermitteln bei der Analyse von 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),
- stellen Klassen, Assoziations- und Vererbungsbeziehungen in Diagrammen grafisch dar (D),
- dokumentieren Klassen durch Beschreibung der Funktionalität der Methoden (D).

B) Beispiele, Medien, Materialien:

- BlueJ
- weitere Ergänzungsmaterialien aus dem Lehrplannavigator: Ergänzungsmaterialien EF

C) Unterrichtssequenzen:

-
- Einführung in BlueJ
  - Klassen implementieren, Objekte erzeugen und inspizieren
  - Methoden testen, Debugging
  - Objektreferenzen
- Modellieren
  - Finden von Klassen

  - Klassen- und Objektbeziehungen darstellen
  - CRC-Spiel: Klassen, Objekte und Beziehungen als CRC-Karte
  - Sequenzdiagramme zur Darstellung der Objektkommunikation
  - Aktivitätsdiagramme zur Beschreibung von Methodenabläufen
- Arrays
- Beispiele: Zeichnen mit geometrischen Objekten, Autos, Schiffe versenken, Werkzeugkasten, Hotelbuchung

2.3.1.5 Unterrichtsvorhaben EF-5

Thema

Such- und Sortieralgorithmen anhand kontextbezogener Beispiele

Leitfragen

Wie können Objekte bzw. Daten effizient sortiert werden, so dass eine schnelle Suche möglich wird? Wie lässt sich die Effizienz von Algorithmen messen und vergleichen?

Vorhabenbezogene Konkretisierung

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

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

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

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

Zeitbedarf

12 Stunden

Sequenzierung des Unterrichtsvorhabens:

A) Zu entwickelnde Kompetenzen:

Die Schüler

- beurteilen die Effizienz von Algorithmen am Beispiel von Sortierverfahren hinsichtlich Zeit und Speicherplatzbedarf (A),
- entwerfen einen weiteren Algorithmus zum Sortieren (M),
- analysieren Such- und Sortieralgorithmen und wenden sie auf Beispiele an (D).

B) Beispiele, Medien, Materialien:

- Beispiel „Sortieren mit Waage“ für das Sortieren ohne Computer: Die Schüler bekommen die Aufgabe, kleine, optisch identische Kunststoffbehälter aufsteigend nach ihrem Gewicht zu sortieren. Dazu steht ihnen eine Balkenwaage zur Verfügung, mit deren Hilfe sie das Gewicht zweier Behälter vergleichen können.
- Beispiele: Sortieren durch Auswählen oder Vertauschen, Das Prinzip von Quicksort“: Durchführen von Bubble- und Selectionsort, Implementation. Quicksort ist als Beispiel für einen Algorithmus nach dem Prinzip Teile und Herrsche gut zu behandeln; ohne Implementation.
- Beispiel „Simulationsspiel zur binären Suche nach Tischtennisbällen“: Mehrere Tischtennisbälle sind nummeriert, sortiert und unter Bechern verdeckt. Mit Hilfe der binären Suche kann sehr schnell ein bestimmter Tischtennisball gefunden werden.
- Computer science unplugged – Sorting Algorithms, URL: www.csunplugged.org/sorting-algorithms

C) Unterrichtssequenzen:

- Erarbeitung eines Sortierverfahrens
  - Sortierprobleme im Kontext informatischer Systeme und im Alltag (z.B. Dateisortierung, Tabellenkalkulation, Telefonbuch, Bundesligatabelle, usw.)
  - Vergleich zweier Elemente als Grundlage eines Sortieralgorithmus
  - Erarbeitung eines Sortieralgorithmus
  - Darstellungsalternativen für Algorithmen: Ablaufdiagramme, umgangssprachliche Beschreibung, Pseudo-Code, Quelltext
- Systematisierung von Algorithmen und Effizienzbetrachtungen
  - Formulierung oder Erläuterung von mehreren Algorithmen im Pseudocode (auf jeden Fall: Sortieren durch Vertauschen, Sortieren durch Auswählen)
  - Anwendung von Sortieralgorithmen auf verschiedene Beispiele

- Bewertung von Algorithmen anhand der Anzahl der nötigen Vergleiche
-
- Effizienzbetrachtungen an einem konkreten Beispiel bezüglich der Rechenzeit und des Speicherplatzbedarfs
-
- Implementation eines Verfahrens zum Sortieren von Zahlen in Arrays

- Binäre Suche auf sortierten Daten
  - Suchaufgaben im Alltag und im Kontext informatischer Systeme
  - Evtl. Simulationsspiel zum effizienten Suchen mit binärer Suche
  - Effizienzbetrachtungen zur binären Suche

2.3.1.6 Unterrichtsvorhaben EF-6

Thema

Gesellschaftliche Bedeutung der Informatik in der Vargangenheit, Gegenwart und Zukunft

Leitfragen

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

Vorhabenbezogene Konkretisierung

Schüler informieren sich über die Entwicklung der Informatik und stellen diese in Lernplakaten, Zeitleisten und / oder Präsentationen dar. Sie können bedeutende Erfindungen verschiedener Bereiche der Informatik benennen und zeitlich der gesellschaftlichen Entwicklung zuordnen.

Als ein wesentlicher historischer Aspekt wird der Aufbau eines Rechnersystems im Sinne der Von-Neumann-Architektur thematisiert und mit dem grundlegenden Prinzip der Datenverarbeitung (Eingabe-Verarbeitung-Ausgabe) in Beziehung gesetzt.

Anschließend wird auf den Aspekt des Datenschutzes eingegangen. Dazu wird das Bundesdatenschutzgesetz in Auszügen behandelt und auf schülernahe Beispielsituationen angewandt.

Zeitbedarf

12 Stunden

Sequenzierung des Unterrichtsvorhabens:

A) Zu entwickelnde Kompetenzen:

Die Schüler

- bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen (A),
- erläutern wesentliche Grundlagen der Geschichte der digitalen Datenverarbeitung (A),
- stellen ganze Zahlen und Zeichen in Binärcodes dar (D),
- interpretieren Binärcodes als Zahlen und Zeichen (D),
- nutzen das Internet zur Recherche, zum Datenaustausch und zur Kommunikation. (K).

B) Beispiele, Medien, Materialien:

- Beispiel „Ausstellung zu informatischen Themen“: Die Schüler bereiten eine Ausstellung zu informatischen Themen vor. Dazu werden Stellwände und Plakate vorbereitet, die ggf. auch außerhalb des Informatikunterrichts in der Schule ausgestellt werden können.Materialien: Schüler recherchieren selbstständig im Internet, in der Schulbibliothek, in öffentlichen Bibliotheken, usw.
- Beispiel „Fallbeispiele aus dem aktuellen Tagesgeschehen“: Die Schüler bearbeiten Fallbeispiele aus ihrer eigenen Erfahrungswelt oder der aktuellen Medienberichterstattung.
- Materialien: Materialblatt zum Bundesdatenschutzgesetz (Download EF-VI.1)

C) Unterrichtssequenzen:

- Selbstständige Erarbeitung von Themen durch die Schüler;
  - Mögliche Themen zur Erarbeitung in Kleingruppen: „Eine kleine Geschichte der Digitalisierung: vom Morsen zum modernen Digitalcomputer“; „Eine kleine Geschichte der Kryptographie: von Caesar zur Enigma“; „Von Nullen, Einsen und mehr: Stellenwertsysteme und wie man mit ihnen rechnet“; „Kodieren von Texten und Bildern: ASCII, RGB und mehr“; „Auswirkungen der Digitalisierung: Veränderungen der Arbeitswelt und Datenschutz“
  - Vorstellung und Diskussion durch Schüler
- Vertiefung des Themas Datenschutz
  - Erarbeitung grundlegender Begriffe des Datenschutzen
  - Problematisierung und Anknüpfung an die Lebenswelt der Schüler

- Diskussion und Bewertung von Fallbeispielen aus dem Themenbereich „Datenschutz“

2.3.2 Qualifikationsphase I

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

A) Zu entwickelnde Kompetenzen:

Die Schüler

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

2.3.2.1 Unterrichtsvorhaben Q1-1

Thema

Wiederholung der objektorientierten Modellierung und Programmierung

Leitfragen

Wie modelliert und implementiert man zu einer Problemstellung in einem geeigneten Anwendungskontext Java-Klassen inklusive ihrer Attribute, Methoden und Beziehungen? Wie kann man die Modellierung und die Funktionsweise der Anwendung grafisch darstellen?

Vorhabenbezogenen Konkretisierung

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

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

Zeitbedarf

8 Stunden (GK), 8 Stunden (LK)

Sequenzierung des Unterrichtsvorhabens:

A) Zu entwickelnde Kompetenzen:

Die Schüler

- analysieren und erläutern objektorientierte Modellierungen (A),
- beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A),
- modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M),
- ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M),
- modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M),
- implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I),
- nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I),
- wenden eine didaktisch orientierte Entwicklungsumgebung zur Demonstration, zum Entwurf, zur Implementierung und zum Test von Informatiksystemen an (I),
- interpretieren Fehlermeldungen und korrigieren den Quellcode (I),
- stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D),
- dokumentieren Klassen (D),
- stellen die Kommunikation zwischen Objekten grafisch dar (D).

B) **Beispiele, Medien, Materialien:**

- Beispiel „Wetthuepfen“: Für ein Wetthüpfen zwischen einem Hasen, einem Hund und einem Vogel werden die Tiere gezeichnet. Alle Tiere springen wiederholt nach links. Die Höhe und Weite jedes Hüpfers ist zufällig. Evtl. marschieren sie anschließend hintereinander her.
- Beispiel „Tannenbaum“: Ein Tannenbaum soll mit verschiedenen Arten von Schmuckstücken versehen werden, die durch unterschiedliche geometrische Objekte dargestellt werden. Es gibt Kugeln, Päckchen in der Form von Würfeln und Zuckerringe in Form von Toren.
- Ein Prototyp, der bereits mit Kugeln geschmückt werden kann, kann zur Verfügung gestellt werden. Da alle Schmuckstücke über die Funktion des Auf- und Abschmückens verfügen sollen, liegt es nahe, dass entsprechende Methoden in einer gemeinsamen Oberklasse realisiert werden.

- Materialien: Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.1-Wiederholung (Download Q1-I.1)

C) Unterrichtssequenzen

- Wiederholung und Erweiterung der objektorientierten Modellierung und Programmierung durch Analyse und Erweiterung eines kontextbezogenen Beispiels
  - Analyse der Problemstellung
  - Analyse der Modellierung (Implementationsdiagramm)
  - Erweiterung der Modellierung im Implementationsdiagramm (Vererbung, abstrakte Klasse)
  - Kommunikation zwischen mindestens zwei Objekten (grafische Darstellung)
  - Dokumentation von Klassen
  - Implementierung der Anwendung oder von Teilen der Anwendung
  - *LK: Debugging*
  - *LK: Arbeiten mit professionellen Entwicklungsumgebungen, z.B. Eclipse*

2.3.2.2 Unterrichtsvorhaben Q1-2

Thema

Modellierung und Implementierung von Anwendungen mit dynamischen, linearen Datenstrukturen

Leitfrage

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

Vorhabenbezogene Konkretisierung

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

Zeitbedarf

20 Stunden (GK), 28 Stunden (LK)

Sequenzierung des Unterrichtsvorhabens:

A) Zu entwickelnde Kompetenzen:

Die Schüler

- erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (A),
- analysieren und erläutern Algorithmen und Programme (A),
- beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A),
- ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M),
- ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M),
- modifizieren Algorithmen und Programme (I),
- implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I),
- nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I),
- interpretieren Fehlermeldungen und korrigieren den Quellcode (I),
- testen Programme systematisch anhand von Beispielen (I),
- stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D).

B) Beispiele, Medien, Materialien

- Beispiel „Patientenwarteschlange“: Sobald ein Patient in einer Arztpraxis eintrifft, werden sein Name und seine Krankenkasse erfasst. Die Verwaltung der Patientenwarteschlange geschieht über eine Klasse, die hier als Wartezimmer bezeichnet wird. Wesentliche Operationen sind das „Hinzufügen“ eines Patienten und das „Entfernen“ eines Patienten, wenn er zur Behandlung gerufen wird. Die Simulationsanwendung stellt eine GUI zur Verfügung, legt ein Wartezimmer an und steuert die Abläufe. Wesentlicher Aspekt des Projektes ist die Modellierung des Wartezimmers mit Hilfe der Klasse Queue. Anschließend wird der Funktionsumfang der Anwendung erweitert: Patienten können sich zusätzlich in die Warteschlange zum Blutdruckmessen einreihen. Objekte werden von zwei Schlangen verwaltet. (Materialien: Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.2 – Warteschlange Download Q1-II.1)
- Beispiel „Heftstapel“: In einem Heftstapel soll das Heft einer Schülerin gefunden werden.

- Beispiel „Kisten stapeln“: In einem Stapel nummerierter Kisten soll eine bestimmte Kiste gefunden und an einen Kunden geliefert werden. Dazu müssen Kisten auf verschiedene Stapel gestapelt und wieder zurückgestellt werden.
- Beispiel „Abfahrtslauf“: Bei einem Abfahrtslauf kommen die Skifahrer nacheinander an und werden nach ihrer Zeit in eine Rangliste eingeordnet. Diese Rangliste wird in einer Anzeige ausgegeben. Ankommende Abfahrer müssen an jeder Stelle der Struktur, nicht nur am Ende oder Anfang eingefügt werden können. (Materialien: Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.2 – Listen Download Q1-II.2)
- Beispiel „Skispringen“: Ein Skispringen hat folgenden Ablauf: Nach dem Sprung erhält der Springer eine Punktzahl und wird nach dieser Punktzahl in eine Rangliste eingeordnet. Die besten 30 Springer qualifizieren sich für den zweiten Durchgang. Sie starten in umgekehrter Reihenfolge gegenüber der Platzierung auf der Rangliste. Nach dem Sprung erhält der Springer wiederum eine Punktzahl und wird nach der Gesamtpunktzahl aus beiden Durchgängen in die endgültige Rangliste eingeordnet.
- Beispiel „Terme in Postfix-Notation“: Die sog. UPN (Umgekehrt-Polnische-Notation) bzw. Postfix-Notation eines Terms setzt den Operator hinter die Operanden. Um einen Term aus der gewohnten Infixschreibweise in einen Term in UPN umzuwandeln oder um den Wert des Terms zu berechnen, kann ein Stack verwendet werden.
- Beispiel „Rangierbahnhof“: Auf einem Güterbahnhof gibt es drei Gleise, die nur zu einer Seite offen sind. Wagons können also von einer Seite auf das Gleis fahren und nur rückwärts wieder hinausfahren. Die Wagons tragen Nummern, wobei die Nummer jedoch erst eingesehen werden kann, wenn der Wagon der vorderste an der offenen Gleisseite ist. (Zwischen den Wagons herumzuturnen, um die anderen Wagonnummern zu lesen, wäre zu gefährlich.) Zunächst stehen alle Wagons unsortiert auf einem Gleis. Ziel ist es, alle Wagons in ein anderes Gleis zu fahren, so dass dort die Nummern der Wagons vom Gleisende aus aufsteigend in richtiger Reihenfolge sind. Zusätzlich zu diesen beiden Gleisen gibt es ein Abstellgleis, das zum Rangieren benutzt werden kann.
- Beispiel „Autos an einer Ampel zur Zufahrtsregelung“: Es soll eine Ampel zur Zufahrtsregelung in Java simuliert werden. An einem geradlinigen, senkrecht von unten nach oben verlaufenden Straßenstück, das von Autos nur einspurig in eine Richtung befahren werden kann, ist ein Haltepunkt markiert, an dem die Ampel steht. Bei einem Klick auf eine Schaltfläche mit der Aufschrift „Heranfahren“ soll ein neues Auto an den Haltepunkt heranfahren bzw. bis an das letzte Auto, das vor dem Haltepunkt wartet. Grünphasen der Ampel werden durch einen Klick auf eine Schaltfläche mit der Aufschrift „Weiterfahren“ simuliert. In jeder Grünphase darf jeweils nur ein Auto weiterfahren. Die anderen Autos rücken nach. (Mate-

rialien: Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1-II.3 – Anwendungen für lineare Datenstrukturen Download Q1-II.3)

C) Unterrichtssequenzen

- Die Datenstruktur Schlange im Anwendungskontext unter Nutzung der Klasse Queue
  - Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen
  - Erarbeitung der Funktionalität der Klasse Queue
  - *LK: Implementation von Methoden der Klasse Queue*
  - Modellierung und Implementierung der Anwendung unter Verwendung eines oder mehrerer Objekte der Klasse Queue
- Die Datenstruktur Stapel im Anwendungskontext unter Nutzung der Klasse Stack
  - Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen
  - Erarbeitung der Funktionalität der Klasse Stack
  - *LK: Implementation von Methoden der Klasse Stack*
  - Modellierung und Implementierung der Anwendung unter Verwendung eines oder mehrerer Objekte der Klasse Stack
- Die Datenstruktur lineare Liste im Anwendungskontext unter Nutzung der Klasse List
  - Erarbeitung der Vorteile der Klasse List im Gegensatz zu den bereits bekannten linearen Strukturen
  - Modellierung und Implementierung einer kontextbezogenen Anwendung unter Verwendung der Klasse List.
- Vertiefung - Anwendungen von Listen, Stapeln oder Schlangen in mindestens einem weiteren Kontext

2.3.2.3 Unterrichtsvorhaben Q1-3

Thema

Suchen und Sortieren auf linearen Datenstrukturen

Leitfrage

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

Vorhabenbezogene Konkretisierung

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

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

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

Zeitbedarf

16 Stunden (GK), 26 Stunden (LK)

Sequenzierung des Unterrichtsvorhabens:

A) Zu entwickelnde Kompetenzen:

Die Schüler

- analysieren und erläutern Algorithmen und Programme (A),
- beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A),
- beurteilen die Effizienz von Algorithmen unter Berücksichtigung des Speicherbedarfs und der Zahl der Operationen (A),
- entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“ und „Teilen und Herrschen“ (M),
- modifizieren Algorithmen und Programme (I),
- implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I),
- implementieren und erläutern iterative und rekursive Such- und Sortierverfahren (I),
- nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I),
- interpretieren Fehlermeldungen und korrigieren den Quellcode (I),
- testen Programme systematisch anhand von Beispielen (I),
- stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D).

B) Beispiele, Medien, Materialien:

- Beispiel „Karteiverwaltung“: Für ein Adressverwaltungsprogramm soll eine Methode zum Suchen einer Adresse geschrieben werden.
- Beispiel „Bundesjugendspiele“: Die Teilnehmer an Bundesjugendspielen nehmen an drei Disziplinen teil und erreichen dort Punktzahlen. Diese werden in einer Wettkampfkarte eingetragen und an das Wettkampfbüro gegeben. Zur Vereinfachung sollte sich das Modell auf die drei Disziplinen „Lauf”, „Sprung“ und „Wurf“ beschränken. Im Wettkampfbüro wird das Ergebnis erstellt. Das Programm soll dafür zunächst den Besten einer Disziplin heraussuchen können und später das gesamte Ergebnis nach gewissen Kriterien sortieren können. (Materialien: Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.3 - Suchen und Sortieren Download Q1-III.1).

C) Unterrichtssequenzen:

- Suchen von Daten in Listen und Arrays
  - Lineare Suche in Listen und in Arrays
  - Binäre Suche in Arrays als Beispiel für rekursives Problemlösen

- Untersuchung der beiden Suchverfahren hinsichtlich ihrer Effizienz (Laufzeitverhalten, Speicherbedarf)

- Sortieren in Listen und Arrays - Entwicklung und Implementierung von iterativen und rekursiven Sortierverfahren
  - Entwicklung und Implementierung eines einfachen Sortierverfahrens für eine Liste
  - Implementierung eines einfachen Sortierverfahrens für ein Feld
  - Entwicklung eines rekursiven Sortierverfahren für ein Feld (z.B. Sortieren durch Mischen)
- Untersuchung der Effizienz der Sortierverfahren „Sortieren durch direktes Einfügen“ und „Quicksort“ auf linearen Listen
  - Grafische Veranschaulichung der Sortierverfahren
  - Untersuchung der Anzahl der Vergleichsoperationen und des Speicherbedarf bei beiden Sortierverfahren
  - Beurteilung der Effizienz der beiden Sortierverfahren
  - *LK: Implementation des Quicksort*
  - *LK: O-Kalkül, Bestimmen von Aufwandsklasse für verschiedene Algorithmen*

2.3.2.4 Unterrichtsvorhaben Q1-4

Thema

Modellierung und Nutzung von relationalen Datenbanken in Anwendungskontexten

Leitfragen

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

Vorhabenbezogene Konkretisierung

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

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

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

An einem Beispiel wird verdeutlicht, dass in Datenbanken Redundanzen unerwünscht sind und Konsistenz gewährleistet sein sollte. Die 1. bis 3. Normalform wird als Gütekriterium für Datenbankentwürfe eingeführt. Datenbankschemata werden hinsichtlich der 1. bis 3. Normalform untersucht und (soweit nötig) normalisiert.

Zeitbedarf

20 Stunden (GK), 26 Stunden (LK)

Sequenzierung des Unterrichtsvorhabens

A) Zu entwickelnde Kompetenzen:

Die Schüler

- erläutern die Eigenschaften und den Aufbau von Datenbanksystemen unter dem Aspekt der sicheren Nutzung (A),
- analysieren und erläutern die Syntax und Semantik einer Datenbankabfrage (A),
- analysieren und erläutern eine Datenbankmodellierung (A),
- erläutern die Eigenschaften normalisierter Datenbankschemata (A),
- bestimmen Primär- und Sekundärschlüssel (M),
- ermitteln für anwendungsbezogene Problemstellungen Entitäten, zugehörige Attribute, Relationen und Kardinalitäten (M),
- modifizieren eine Datenbankmodellierung (M),
- modellieren zu einem Entity-Relationship-Diagramm ein relationales Datenbankschema (M),
- bestimmen Primär- und Sekundärschlüssel (M),
- überführen Datenbankschemata in vorgegebene Normalformen (M),
- verwenden die Syntax und Semantik einer Datenbankabfragesprache, um Informationen aus einen Datenbanksystem zu extrahieren (I),
- ermitteln Ergebnisse von Datenbankabfragen über mehrere verknüpfte Tabellen (D),
- stellen Entitäten mit ihren Attributen und die Beziehungen zwischen Entitäten in einem Entity-Relationship-Diagramm grafisch dar (D),
- überprüfen Datenbankschemata auf vorgegebene Normalisierungseigenschaften (D).

B) Beispiele, Medien, Materialien:

- Beispiel „VideoCenter“: VideoCenter ist die Simulation einer Online-Videothek für den Informatik-Unterricht mit Webfrontends zur Verwaltung der Kunden, der Videos und der Ausleihe. Außerdem ist es möglich direkt SQL-Abfragen einzugeben. Es ist auch möglich, die Datenbank herunter zu laden und lokal zu installieren. Unter http://dokumentation.video-center.schule.de/old/video/index.html (abgerufen: 30. 03. 2014) findet man den Link zu dem VideoCenter-System sowie nähere Informationen. Lesenswert ist auch die dort verlinkte „Dokumentation der Fallstudie” mit didaktischem Material, welches alternativ bzw. ergänzend zu der im Folgenden beschriebenen Durchführung verwendet werden kann.

- Beispiel „Schulbuchausleihe“: Unter www.brd.nrw.de/lerntreffs/informatik/structure/material/sek2/datenbanken.php (abgerufen: 30. 03. 2014) wird eine Datenbank zur Verfügung gestellt, die Daten einer Schulbuch-Ausleihe enthält (über 1000 Entleiher, 200 Bücher mit mehreren tausend Exemplaren und viele Ausleihvorgänge). Die Datenbank kann in OpenOffice eingebunden werden.
- Beispiel „Fahrradverleih“: Der Fahrradverleih BTR (BikesToRent) verleiht unterschiedliche Typen von Fahrrädern diverser Firmen an seine Kunden. Die Kunden sind bei BTR registriert (Name, Adresse, Telefon). BTR kennt von den Fahrradfirmen den Namen und die Telefonnummer. Kunden von BTR können CityBikes, Treckingräder und Mountainbikes ausleihen.
- Beispiel „Reederei“: Die Datenverwaltung einer Reederei soll in einem Datenbanksystem umgesetzt werden. Ausgehend von der Modellierung soll mit Hilfe eines ER-Modells und eines Datenbankschemas dieser erste Entwurf normalisiert und in einem Datenbanksystem umgesetzt werden. Es schließen sich diverse SQL-Abfragen an, wobei auf die Relationenalgebra eingegangen wird.
- Beispiel „Buchungssystem“: In dem Online-Buchungssystem einer Schule können die Lehrer Medienräume, Beamer, Laptops, Kameras, usw. für einen bestimmten Zeitpunkt buchen, der durch Datum und die Schulstunde festgelegt ist. Dazu ist die Datenbank zu modellieren, ggf. zu normalisieren und im Datenbanksystem umzusetzen. Weiter sollen sinnvolle Abfragen entwickelt werden. Unter http://mrbs.sourceforge.net (abgerufen: 30.03. 2014) findet man ein freies Online-Buchungssystem inklusive Demo, an Hand derer man erläutern kann, worum es in dem Projekt geht.
- Beispiel „Schulverwaltung“: In einer Software werden die Schulhalbjahre, Jahrgangsstufen, Kurse, Klassen, Schüler, Lehrer und Noten einer Schule verwaltet. Man kann dann ablesen, dass z.B. Schüler X von Lehrer Y im 2. Halbjahr des Schuljahrs 2011 / 2012 in der Jahrgangsstufe 9 im Differenzierungsbereich im Fach Informatik die Note „sehr gut“ erhalten hat. Dazu ist die Datenbank zu modellieren, ggf. zu normalisieren und im Datenbanksystem umzusetzen. Weiter sollen sinnvolle Abfragen entwickelt werden und das Thema Datenschutz besprochen werden.

C) Unterrichtssequenzen:

- Nutzung von relationalen Datenbanken
  - Aufbau von Datenbanken und Grundbegriffe: Entwicklung von Fragestellungen zur vorhandenen Datenbank; Analyse der Struktur der vorgegebenen Datenbank und Erarbeitung der Begriffe Tabelle, Attribut, Datensatz, Datentyp, Primärschlüssel, Fremdschlüssel, Datenbankschema
  - SQL-Abfragen: Analyse vorgegebener SQL-Abfragen und Erarbeitung der Sprachelemente von SQL (SELECT (DISTINCT) …FROM, WHERE, AND, OR, NOT) auf einer Tabelle; Analyse und Erarbeitung von SQL-

Abfragen auf einer und mehrerer Tabelle zur Beantwortung der Fragestellungen (JOIN, UNION, AS, GROUP BY,ORDER BY, ASC, DESC, COUNT, MAX, MIN, SUM, Arithmetische Operatoren: +, -, *, /, (…), Vergleichsoperatoren: =, <>, >, <, >=, <=, LIKE, BETWEEN, IN, IS NULL)

- ◦ Vertiefung an einem weiteren Datenbankbeispiel

- Modellierung von relationalen Datenbanken
  - ◦ Entity-Relationship-Diagramm: Ermittlung von Entitäten, zugehörigen Attributen, Relationen und Kardinalitäten in Anwendungssituationen und Modellierung eines Datenbankentwurfs in Form eines Entity-Relationship-Diagramms; Erläuterung und Modifizierung einer Datenbankmodellierung
  - ◦ Entwicklung einer Datenbank aus einem Datenbankentwurf: Modellierung eines relationalen Datenbankschematas zu einem Entity-Relationship-Diagramm inklusive der Bestimmung von Primär- und Sekundärschlüsseln
  - ◦ Redundanz, Konsistenz und Normalformen: Untersuchung einer Datenbank hinsichtlich Konsistenz und Redundanz in einer Anwendungssituation; Überprüfung von Datenbankschemata hinsichtlich der 1. bis 3. Normalform und Normalisierung (um Redundanzen zu vermeiden und Konsistenz zu gewährleisten)
- *LK: Arbeiten mit einem DBMS (z.B. Access, Base)*

2.3.2.5 Unterrichtsvorhaben Q1-5

Thema

Sicherheit und Datenschutz in Netzstrukturen, *LK: Netzwerke*

Leitfragen

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

Vorhabenbezogene Konkretisierung

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

Zeitbedarf

10 Stunden (GK), 22 Stunden (GK)

Sequenzierung des Unterrichtsvorhabens:

A) Zu entwickelnde Kompetenzen:

Die Schüler

- beschreiben und erläutern Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken (A),
- analysieren und erläutern Eigenschaften und Einsatzbereiche symmetrischer und asymmetrischer Verschlüsselungsverfahren (A),
- untersuchen und bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen, die Sicherheit von Informatiksystemen sowie die Einhaltung der Datenschutzbestimmungen und des Urheberrechts (A),
- untersuchen und bewerten Problemlagen, die sich aus dem Einsatz von Informatiksystemen ergeben, hinsichtlich rechtlicher Vorgaben, ethischer Aspekte und gesellschaftlicher Werte unter Berücksichtigung unterschiedlicher Interessenlagen (A),
- nutzen bereitgestellte Informatiksysteme und das Internet reflektiert zum Erschließen, zur Aufbereitung und Präsentation fachlicher Inhalte (D).
- *LK: modifizieren Algorithmen und Programme (I),*
- *LK: implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I),*
- *LK: nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I),*
- *LK: interpretieren Fehlermeldungen und korrigieren den Quellcode (I),*
- *LK: testen Programme systematisch anhand von Beispielen (I),*
- *LK: stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D).*
- *LK: ermitteln bei der Analyse einfacherer Problemstellungen die Notwendigkeit von nebenläufigen Prozessen und ihrer Steuerung.*

B) Beispiele, Medien, Materialien:

- Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben, Verschlüsselung Q1.5 - Zugriff auf Daten in Netzwerken Download Q1-V.1.
- Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1 5 - Datenschutz beim Videocenter, Materialblatt-Datenschutzgesetz Download Q1-V.2.

C) Unterrichtssequenzen:

- Daten in Netzwerken und Sicherheitsaspekte in Netzen sowie beim Zugriff auf Datenbanken

- ◦ Beschreibung eines Datenbankzugriffs im Netz anhand eines Anwendungskontextes und einer Client-Server-Struktur zur Klärung der Funktionsweise eines Datenbankzugriffs
- ◦ Netztopologien als Grundlage von Client-Server-Strukturen und TCP/IP-Schichtenmodell als Beispiel für eine Paketübermittlung in einem Netz
- ◦ Vertraulichkeit, Integrität, Authentizität in Netzwerken sowie symmetrische und asymmetrische kryptografische Verfahren (Cäsar-, Vigenère-, RSA-Verfahren) als Methoden Daten im Netz verschlüsselt zu übertragen

- Fallbeispiele zur Datenschutzproblematik und zum Urheberrecht
- *LK: Anwendung zur Kommunikation im Netzwerk*
  - ◦ *LK: Implementation von Algorithmen zur Kommunikation in Netzwerken unter Verwendung der Klassen Connection, Client und Server.*
  - ◦ *LK: Entwurf von Kommunikationsprotokollen, z.B. für ein einfaches Netzwerk-Rollenspiel.*
- *LK: Nebenläufige Prozesse*
  - ◦ *Entwicklung und Implementation von Lösungen zur Steuerung von Prozessen in anwendungsbezogenen Projekten, z.B. Server.*

2.3.3 Qualifikationsphase II

2.3.3.1 Unterrichtsvorhaben Q2-1

Thema

Modellierung und Implementierung von Anwendungen mit dynamischen, nichtlinearen Datenstrukturen

Leitfragen

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

Vorhabenbezogene Konkretisierung

Anhand von Beispielen für Baumstrukturen werden grundlegende Begriffe eingeführt und der rekursive Aufbau binärer Bäume dargestellt.

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

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

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

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

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

Zeitbedarf

24 Stunden (GK), 38 Stunden (LK)

Sequenzierung des Unterrichtsvorhabens:

A) Zu entwickelnde Kompetenzen:

Die Schüler

- erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (A),
- analysieren und erläutern Algorithmen und Programme (A),
- beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A),
- ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M),
- ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M),
- modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M),
- verwenden bei der Modellierung geeigneter Problemstellungen die Möglichkeiten der Polymorphie (M),
- entwickeln iterative und rekursive Algorithmen unter Nutzung der Konstruktionsstrategien „Modularisierung“ und „Teilen und Herrschen“ (M),
- implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I),
- modifizieren Algorithmen und Programme (I),
- nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I),
- interpretieren Fehlermeldungen und korrigieren den Quellcode (I),
- testen Programme systematisch anhand von Beispielen (I),
- stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D),
- stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D).
- *LK: erläutern spezialisierte Bäume und ihre funktionsweise (AVL-Bäume).*
- *LK: bewerten die Effizienz binärer Suchbäume im Gegensatz zu spezialisierten Baumstrukturen (AVL-Bäume).*
- *LK: erkennen anhand einfacher Problemstellungen die Baumstruktur als eine besondere netzartige Struktur.*
- *LK: benennen grundlegende Komponenten einer Netzartigen Struktur.*

- *LK: untersuchen sowohl gewichtete als auch ungewichtete Netzartige Strukturen aufgrund ihrer Struktur mithilfe unterschiedlicher Algorithmen (Breitensuche, Dijkstra-Algorithmus)*

B) Beispiele, Medien, Materialien:

- Beispiel „Termbaum“: Der Aufbau von Termen wird mit Hilfe von binären Baumstrukturen verdeutlicht.
- Beispiel „Ahnenbaum“: Die binäre Baumstruktur ergibt sich daraus, dass jede Person genau einen Vater und eine Mutter hat.
- Beispiel „Suchbäume“: Alle Inhalte, die nach einer Ordnung vor dem Inhalt im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Inhalt im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.)
- Beispiel „Entscheidungsbäume“: Um eine Entscheidung zu treffen, werden mehrere Fragen mit ja oder nein beantwortet. Die Fragen, die möglich sind, wenn die Antwort auf eine Frage mit „ja“ beantwortet wird, befinden sich im linken Teilbaum, die Fragen, die möglich sind, wenn die Antwort „nein“ lautet, stehen im rechten Teilbaum.
- Beispiel „Codierungsbäume für Codierungen, deren Alphabet aus genau zwei Zeichen besteht“: Morse hat Buchstaben als Folge von Punkten und Strichen codiert. Diese Codierungen können in einem Binärbaum dargestellt werden, so dass ein Übergang zum linken Teilbaum einem Punkt und ein Übergang zum rechten Teilbaum einem Strich entspricht. Wenn man im Gesamtbaum startet und durch Übergänge zu linken oder rechten Teilbäumen einen Pfad zum gewünschten Buchstaben sucht, erhält man die Morsecodierung des Buchstabens.
- Materialien: Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 – Binärbaum (Download Q2-I.1)
- Beispiel „Informatikerbaum als binärer Baum“: In einem binären Baum werden die Namen und die Geburtsdaten von Informatikern lexikographisch geordnet abgespeichert. Alle Namen, die nach dieser Ordnung vor dem Namen im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Namen im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.). Folgende Funktionalitäten werden benötigt: Einfügen der Informatiker-Daten in den Baum; Suchen nach einem Informatiker über den Schlüssel Name; Ausgabe des kompletten Datenbestands in nach Namen sortierter Reihenfolge.
- Beispiel „Buchindex“: Es soll eine Anwendung entwickelt werden, die anhand von Stichworten und zugehörigen Seitenzahlen ein Stichwortregister erstellt. da die Stichwörter bei der Analyse des Buches häufig gesucht werden müssen, werden sie in der Klasse Buchindex als Suchbaum (Objekt der Klasse BinarySearchTree) verwaltet. Alle Inhalte, die nach einer Ordnung vor dem Inhalt im aktuellen Teilbaum stehen, sind in dessen lin-

kem Teilbaum, alle die nach dem Inhalt im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.)

- Materialien: Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 – Binärbaum Download Q2-I.2, Download Q2-I.3, Download Q2-I.4.

C) Unterrichtssequenzen:

- Analyse von Baumstrukturen in verschiedenen Kontexten
  - Grundlegende Begriffe (Grad, Tiefe, Höhe, Blatt, Inhalt, Teilbaum, Ebene, Vollständigkeit)
  - Aufbau und Darstellung von binären Bäumen anhand von Baumstrukturen in verschiedenen Kontexten
- Die Datenstruktur Binärbaum im Anwendungskontext unter Nutzung der Klasse BinaryTree
  - Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen im Anwendungskontext
  - Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramms
  - Erarbeitung der Klasse BinaryTree und beispielhafte Anwendung der Operationen
  - Implementierung der Anwendung oder von Teilen der Anwendung
  - Traversierung eines Binärbaums im Pre-, In- und Postorderdurchlauf
- Die Datenstruktur binärer Suchbaum im Anwendungskontext unter Verwendung der Klasse BinarySearchTree
  - Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen
  - Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramm, grafische Darstellung eines binären Suchbaums und Erarbeitung der Struktureigenschaften
  - Erarbeitung der Klasse BinarySearchTree und Einführung des Interface Item zur Realisierung einer geeigneten Ordnungsrelation
  - *LK: Implementation von ausgewählten Methoden der Klassen für den binären Suchbaum, darunter die Methoden insert und search*
  - Implementierung der Anwendung oder von Teilen der Anwendung inklusive einer sortierten Ausgabe des Baums
- Übung und Vertiefungen der Verwendung von Binärbäumen oder binären Suchbäumen anhand weiterer Problemstellungen
- *LK: Selbstoptimierende Binärbäume*

- *LK: Aufwandsanalyse für die Operationen von Selbstausgleichenden Bäumen.*
- *LK: Implementation von binären AVL-Bäumen.*

- *LK: Analyse von Graphenstrukturen in verschiedenen Kontexten*
  - *LK: Grundlegende Begriffe (Knoten, Kanten, Grade, Kantengewichte, Kantenrichtung, Vollständigkeit, Quelle, Senke)*
  - *LK: Aufbau und Darstellung von Graphen durch Adjazenzmatrizen und Adjazenzlisten*
  - *LK: Traversierung von Graphen mit Breiten- und Tiefensuche*
  - *LK: Pfadsuche mit Dijkstra, Backtracking*
- *LK: Die Datenstruktur Graph im Anwendungskontext*

2.3.3.2 Unterrichtsvorhaben Q2-2

Thema

Endliche Automaten und formale Sprachen

Leitfragen

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

Vorhabenbezogene Konkretisierung

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

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

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

Zeitbedarf

20 Stunden (GK), 32 Stunden (LK)

Sequenzierung des Unterrichtsvorhabens:

A) Zu entwickelnde Kompetenzen:

Die Schüler

- analysieren und erläutern die Eigenschaften endlicher Automaten einschließlich ihres Verhaltens auf bestimmte Eingaben (A),
- analysieren und erläutern Grammatiken regulärer Sprachen (A),
- zeigen die Grenzen endlicher Automaten und regulärer Grammatiken im Anwendungszusammenhang auf (A),
- ermitteln die formale Sprache, die durch eine Grammatik erzeugt wird (A),
- entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M),
- entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M),
- entwickeln zur akzeptierten Sprache eines Automaten die zugehörige Grammatik (M),
- entwickeln zur Grammatik einer regulären Sprache einen zugehörigen endlichen Automaten (M),
- modifizieren Grammatiken regulärer Sprachen (M),
- entwickeln zu einer regulären Sprache eine Grammatik, die die Sprache erzeugt (M),
- stellen endliche Automaten in Tabellen oder Graphen dar und überführen sie in die jeweils andere Darstellungsform (D),
- ermitteln die Sprache, die ein endlicher Automat akzeptiert (D).
- beschreiben an Beispielen den Zusammenhang zwischen Automaten und Grammatiken (D).

B) Beispiele, Medien, Materialien:

- Beispiele „Cola-Automat“, „Geldspielautomat“: Roboter, Zustandsänderung eines Objekts „Auto“, Akzeptor für bestimmte Zahlen, Akzeptor für Teilwörter in längeren Zeichenketten, Akzeptor für Terme
- Beispiele: reguläre Grammatik für Wörter mit ungerader Parität, Grammatik für Wörter, die bestimmte Zahlen repräsentieren, Satzgliederungsgrammatik
- Beispiele: Klammerausdrücke, $a^n b^n$ im Vergleich zu $(ab)^n$
- Materialien: Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.2 – Endliche Automaten, Formale Sprachen Download Q2-II.1.

C) Unterrichtssequenzen:

- Endliche Automaten
  - Vom Automaten in den Schülern bekannten Kontexten zur formalen Beschreibung eines endlichen Automaten
  - Untersuchung, Darstellung und Entwicklung endlicher Automaten
- Untersuchung und Entwicklung von Grammatiken regulärer Sprachen
  - Erarbeitung der formalen Darstellung regulärer Grammatiken
  - Untersuchung, Modifikation und Entwicklung von Grammatiken
  - Entwicklung von endlichen Automaten zum Erkennen regulärer Sprachen die durch Grammatiken gegeben werden
  - Entwicklung regulärer Grammatiken zu endlichen Automaten
- Grenzen endlicher Automaten
- *LK: Implementation eines Parsers für eine einfache Programmiersprache*
- *LK: Kellerautomaten und Turingmaschinen*
  - *kontextsensitive und kontextfreie Grammatiken*
  - *Fragen der Berechenbarkeit von Problemen*
  - *P und NP*

2.3.3.3 Unterrichtsvorhaben Q2-3

Thema

Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit

Leitfragen

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

Vorhabenbezogene Konkretisierung

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

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

Zeitbedarf

10 Stunden (GK), 12 Stunden (LK)

Sequenzierung des Unterrichtsvorhabens:

A) Zu entwickelnde Kompetenzen:

Die Schüler

- erläutern die Ausführung eines einfachen maschinennahen Programms sowie die Datenspeicherung auf einer „Von-Neumann-Architektur“ (A),
- untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen (A).

B) Beispiele, Medien, Materialien:

- Beispiel: Addition von 4 zu einer eingegeben Zahl mit einem Rechnermodell
- Beispiel: Halteproblem
- Materialien: Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.3 –Von-Neumann-Architektur und maschinennahe Programmierung Download Q2-III.1, Download Q2-III.2.

C) Unterrichtssequenzen:

- Von-Neumann-Architektur und die Ausführung maschinennaher Programme
  - prinzipieller Aufbau einer von Neumann-Architektur mit CPU, Rechenwerk, Steuerwerk, Register und Hauptspeicher
  - einige maschinennahe Befehlen und ihre Repräsentation in einem Binär-Code, der in einem Register gespeichert werden kann
  - Analyse und Erläuterung der Funktionsweise eines einfachen maschinennahen Programms
- Grenzen der Automatisierbarkeit
  - Vorstellung des Halteproblems
  - Unlösbarkeit des Halteproblems
  - Beurteilung des Einsatzes von Informatiksystemen hinsichtlich prinzipieller Möglichkeiten und prinzipieller Grenzen

2.3.3.4 Unterrichtsvorhaben Q2-4

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

2.4 Grundsätze der fachmethodischen und -didaktischen Arbeit

In Absprache mit der Lehrerkonferenz sowie unter Berücksichtigung des Schulprogramms hat die Fachkonferenz Informatik des Gymnasiums Porta Westfalica 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.

2.4.1 Überfachliche Grundsätze:

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

2.4.2 Fachliche Grundsätze:

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

2.5 Grundsätze der Leistungsbewertung und -rückmeldung

Auf der Grundlage von §13 - §16 der APO-GOSt sowie Kapitel 3 des Kernlehrplans Informatik für die gymnasiale Oberstufe hat die Fachkonferenz des Gymnasiums Porta Westfalica im Einklang mit dem entsprechenden schulbezogenen Konzept die nachfolgenden Grundsätze zur Leistungsbewertung und Leistungsrückmeldung beschlossen. Die nachfolgenden Absprachen stellen die Minimalanforderungen an das lerngruppenübergreifende gemeinsame Handeln der Fachgruppenmitglieder dar. Bezogen auf die einzelne Lerngruppe kommen ergänzend weitere der in den Folgeabschnitten genannten Instrumente der Leistungsüberprüfung zum Einsatz.

2.5.1 Beurteilungsbereich Klausuren

2.5.1.1 Verbindliche Absprachen:

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

2.5.1.2 Instrumente:

- Einführungsphase: 1 Klausur je Halbjahr Dauer: 2 Unterrichtsstunden
- Grundkurse Q 1: 2 Klausuren je Halbjahr Dauer: 2 Unterrichts-stunden
- Grundkurse Q 2.1: 2 Klausuren Dauer: 3-4 Unterrichts-stunden
- Grundkurse Q 2.2: 1 Klausur unter Abiturbedingungen
- Anstelle einer Klausur kann gemäß dem Beschluss der Lehrerkonferenz in Q 1.2 eine Facharbeit geschrieben werden.

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

2.5.1.3 Kriterien

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

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

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

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

2.5.2 Beurteilungsbereich Sonstige Mitarbeit

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

2.5.2.1 Verbindliche Absprachen der Fachkonferenz

- Alle Schüler führen in der Einführungsphase in Kleingruppen ein Kurzprojekt durch und fertigen dazu eine Arbeitsmappe mit Arbeitstagebuch an. Dies wird in die Note für die Sonstige Mitarbeit einbezogen.
- In der Qualifikationsphase erstellen, dokumentieren und präsentieren die Schüler in Kleingruppen ein anwendungsbezogenes Softwareprodukt. Dies wird in die Note für die Sonstige Mitarbeit einbezogen.

2.5.2.2 Leistungsaspekte

<u>Mündliche Leistungen</u>

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

<u>Praktische Leistungen am Computer</u>

- Implementierung, Test und Anwendung von Informatiksystemen

<u>Sonstige schriftliche Leistungen</u>

- Arbeitsmappe und Arbeitstagebuch zu einem durchgeführten Unterrichtsvorhaben
- Lernerfolgsüberprüfung durch kurze schriftliche Übungen

- In Kursen, in denen höchstens 50% der Kursmitglieder eine Klausur schreiben, finden schriftliche Übungen mindestens einmal pro Kurshalbjahr statt, in anderen Kursen entscheidet über die Durchführung die Lehrkraft.

  Schriftliche Übung dauern ca. 20 Minuten und umfassen den Stoff der letzten ca. 4–6 Stunden.
- Bearbeitung von schriftlichen Aufgaben im Unterricht

2.5.2.3 Kriterien

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

Die Bewertungskriterien stützen sich auf

- die Qualität der Beiträge,
- die Quantität der Beiträge und
- die Kontinuität der Beiträge.

Besonderes Augenmerk ist dabei auf

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

Bei Gruppenarbeiten auch auf

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

Bei Projektarbeit darüber hinaus auf

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

2.5.2.4 Grundsätze der Leistungsrückmeldung und Beratung

Die Grundsätze der Leistungsbewertung werden zu Beginn eines jeden Halbjahres den Schülern transparent gemacht. Leistungsrückmeldungen können erfolgen

- nach einer mündlichen Überprüfung,
- bei Rückgabe von schriftlichen Leistungsüberprüfungen,
- nach Abschluss eines Projektes,
- nach einem Vortrag oder einer Präsentation,
- bei auffälligen Leistungsveränderungen,
- auf Anfrage,
- als Quartalsfeedback und
- zu Eltern- oder Schülersprechtagen.

Die Leistungsrückmeldung kann

- durch ein Gespräch mit der Schülerin oder dem Schüler,
- durch einen Feedbackbogen,
- durch die schriftliche Begründung einer Note oder
- durch eine individuelle Lern-/Förderempfehlung

erfolgen.

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

3 Qualitätssicherung und Evaluation

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

Das schulinterne Curriculum (siehe 2.1) ist zunächst bis 2017 für den ersten Durchgang durch die gymnasiale Oberstufe nach Erlass des Kernlehrplanes verbindlich. Erstmalig nach Ende der Einführungsphase im Sommer 2015, werden in einer Sitzung der Fachkonferenz Erfahrungen ausgetauscht und ggf. Änderungen für den nächsten Durchgang der Einführungsphase beschlossen, um erkannten ungünstigen Entscheidungen schnellstmöglich entgegenwirken zu können.

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

Minden, den 13.01.2015

Tobias Selms *Frederik Föllmer*