Schulinterner Lehrplan zum Kernlehrplan für die gymnasiale Oberstufe am Tannenbusch-Gymnasium

Fach Informatik

(Stand: 21.02.2017)

TABU GYMNASIUM

Inhaltsverzeichnis

1 Die Fachgruppe Informatik am Tannenbusch-Gymnasium

Die Fachgruppe besteht zurzeit aus drei Lehrkräften, denen zwei Computerräume mit 19 bzw. 16 Computerarbeitsplätzen zur Verfügung stehen. Des Weiteren besteht für die Schülerinnen und Schüler im Selbstlernzentrum (SLZ Sek II) noch Zugang über circa 8 Rechner. Alle Arbeitsplätze sind an das schulinterne Rechnernetz angeschlossen, so dass Schülerinnen und Schüler über einen individuell gestaltbaren Zugang zum zentralen Server der Schule in allen drei Räumen Zugriff auf ihre eigenen Daten, zur Recherche im Internet oder zur Bearbeitung schulischer Aufgaben haben.

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

Aufgrund der geringen Anzahl an Lehrkräften wird der Informatikunterricht nur in der Oberstufe in allen Jahrgangsstufen als Grundkurs angeboten. Durch eine Kooperation mit dem Alexander-von-Humboldt-Gymnasium Bornheim ist es für die Schülerinnen und Schüler möglich Informatik als Leistungskurs zu wählen, der Unterricht wird dort verantwortlich durchgeführt.

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

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

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.

2 Entscheidungen zum Unterricht

2.1 Unterrichtsvorhaben

Die Darstellung der Unterrichtsvorhaben im schulinternen Lehrplan besitzt den Anspruch, sämtliche im Kernlehrplan angeführten Kompetenzen abzudecken. Dies entspricht der Verpflichtung jeder Lehrkraft, alle Kompetenzerwartungen des Kernlehrplans bei den Lernenden auszubilden und zu entwickeln.

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

Im „Übersichtsraster Unterrichtsvorhaben" (Kapitel 2.1.1) wird die für alle Lehrerinnen und Lehrer gemäß Fachkonferenzbeschluss verbindliche Verteilung der Unterrichtsvorhaben dargestellt. Das Übersichtsraster dient dazu, den Kolleginnen und Kollegen einen schnellen Überblick über die Zuordnung der Unterrichtsvorhaben zu den einzelnen Jahrgangsstufen sowie den im Kernlehrplan genannten Kompetenzen, Inhaltsfeldern und inhaltlichen Schwerpunkten zu verschaffen. Um Klarheit für die Lehrkräfte herzustellen und die Übersichtlichkeit zu gewährleisten, werden in der Kategorie „Kompetenzen" an dieser Stelle nur die übergeordneten Kompetenzerwartungen ausgewiesen, während die konkretisierten Kompetenzerwartungen erst auf der Ebene konkretisierter Unterrichtsvorhaben Berücksichtigung finden. Der ausgewiesene Zeitbedarf versteht sich als grobe Orientierungsgröße, die nach Bedarf über- oder unterschritten werden kann. Um Spielraum 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 nur 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, besitzt die exemplarische Ausweisung „konkretisierter Unterrichtsvorhaben" (Kapitel 2.1.2) empfehlenden Charakter. 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 und 2.3 zu entnehmen sind. Abweichungen von den

vorgeschlagenen Vorgehensweisen bezüglich der konkretisierten Unterrichtsvorhaben sind im Rahmen der pädagogischen Freiheit der Lehrkräfte jederzeit möglich. Sicherzustellen bleibt allerdings auch hier, dass im Rahmen der Umsetzung der Unterrichtsvorhaben insgesamt alle Sach- und Urteilskompetenzen des Kernlehrplans Berücksichtigung finden.

2.1.1 Übersichtsraster Unterrichtsvorhaben

Einführungsphase

Einführungsphase	
Unterrichtsvorhaben E-I **Thema:** ***Einführung in die Nutzung von Informatiksystemen und in grundlegende Begrifflichkeiten*** **Zentrale Kompetenzen:** Argumentieren Darstellen und Interpretieren Kommunizieren und Kooperieren **Inhaltsfelder:** Informatiksysteme Informatik, Mensch und Gesellschaft **Inhaltliche Schwerpunkte:** Einzelrechner Dateisystem Internet Einsatz von Informatiksystemen Binäre Codierung von Daten **Zeitbedarf**: **6 Stunden**	**Unterrichtsvorhaben E-II** **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**

Einführungsphase	
Unterrichtsvorhaben E-III **Thema**: ***Grundlagen der objektorientierten Programmierung und algorithmischer Grundstrukturen in Java anhand von einfachen Animationen*** **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	**Unterrichtsvorhaben E-IV** **Thema**: ***Modellierung und Implementierung von Klassen- und Objektbeziehungen anhand von grafischen Spielen und Simulationen*** **Zentrale Kompetenzen:** Argumentieren Modellieren Implementieren Darstellen und Interpretieren Kommunizieren und Kooperieren **Inhaltsfelder:** Daten und ihre Strukturierung Algorithmen Formale Sprachen und Automaten **Inhaltliche Schwerpunkte:** Objekte und Klassen Syntax und Semantik einer Programmiersprache Analyse, Entwurf und Implementierung einfacher Algorithmen Binäre Codierung und Verarbeitung **Zeitbedarf**: 18 Stunden

Qualifikationsphase (Q1) – GRUNDKURS

Qualifikationsphase 1	
Unterrichtsvorhaben Q1-I **Thema:** ***Rekursive Algorithmen in Anwendungskontexten*** **Zentrale Kompetenzen:** Argumentieren Modellieren Implementieren Darstellen und Interpretieren Kommunizieren und Kooperieren **Inhaltsfelder:** Algorithmen Formale Sprachen und Automaten Informatiksysteme Informatik, Mensch und Gesellschaft **Inhaltliche Schwerpunkte:** Analyse, Entwurf und Implementierung von Algorithmen Algorithmen in ausgewählten informatischen Kontexten Syntax und Semantik einer Programmiersprache Nutzung von Informatiksystemen Grenzen der Automatisierung **Zeitbedarf: 20 Stunden**	**Unterrichtsvorhaben Q1-II** **Thema:** ***Modellierung und Implementierung dynamischer Listenstrukturen und deren Anwendungen*** **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 Algorithmen in ausgewählten informatischen Kontexten Syntax und Semantik einer Programmiersprache Nutzung von Informatiksystemen **Zeitbedarf: 25 Stunden**

Unterrichtsvorhaben Q1-III

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

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
Algorithmen in ausgewählten informatischen Kontexten
Syntax und Semantik einer Programmiersprache
Nutzung von Informatiksystemen

Zeitbedarf: 20 Stunden

Unterrichtsvorhaben Q1-IV

Thema:
Sicherheit und Datenschutz in Netzstrukturen

Zentrale Kompetenzen:
Argumentieren
Darstellen und Interpretieren
Kommunizieren und Kooperieren

Inhaltsfelder:
Informatiksysteme
Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:
Einzelrechner und Rechnernetzwerke
Sicherheit
Nutzung von Informatiksystemen, Wirkungen der Automatisierung

Zeitbedarf: 15 Stunden

Summe Qualifikationsphase 1: 80 Unterrichtsstunden

Qualifikationsphase (Q2) – GRUNDKURS

Qualifikationsphase 2	
Unterrichtsvorhaben Q2-I **Thema:** ***Modellierung und Nutzung relationaler Datenbanken in Anwendungskontexten*** **Zentrale Kompetenzen:** Argumentieren Modellieren Implementieren Darstellen und Interpretieren Kommunizieren und Kooperieren **Inhaltsfelder:** Daten und ihre Strukturierung Algorithmen Formale Sprache und Automaten Informatiksysteme Informatik, Mensch und Gesellschaft **Inhaltliche Schwerpunkte:** Datenbanken Algorithmen in ausgewählten informatischen Kontexten Syntax und Semantik einer Programmiersprache Nutzung von Informatiksystemen Sicherheit Wirkung der Automatisierung **Zeitbedarf: 25 Stunden**	**Unterrichtsvorhaben Q2-II** **Thema:** ***Endliche Automaten und Formale Sprachen*** **Zentrale Kompetenzen:** Argumentieren Modellieren Implementieren Darstellen und Interpretieren Kommunizieren und Kooperieren **Inhaltsfelder:** Formale Sprachen und Automaten Informatiksysteme **Inhaltliche Schwerpunkte:** Endliche Automaten Grammatiken regulärer Sprachen Möglichkeiten und Grenzen von Automaten und formalen Sprachen Nutzung von Informatiksystemen **Zeitbedarf: 25 Stunden**

Unterrichtsvorhaben Q2-III

Thema:
Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit

Zentrale Kompetenzen:
Argumentieren
Kommunizieren und Kooperieren

Inhaltsfelder:
Informatiksysteme
Informatik, Mensch und Gesellschaft

Inhaltliche Schwerpunkte:
Einzelrechner und Rechnernetzwerke
Grenzen der Automatisierung

Zeitbedarf: 10 Stunden

Summe Qualifikationsphase 2: 60 Unterrichtsstunden

2.1.2 Konkretisierte Unterrichtsvorhaben

Einführungsphase

Unterrichtsvorhaben EF-I:

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

Leitfragen: Womit beschäftigt sich die Wissenschaft der Informatik?
Wie kann die in der Schule vorhandene informatische Ausstattung genutzt werden?
Zeitbedarf: 6 Stunden

Vorhabenbezogene Konkretisierung:
Das erste Unterrichtsvorhaben stellt eine allgemeine Einführung in das Fach Informatik dar. Dabei ist zu berücksichtigen, dass für manche Schülerinnen und Schüler in der Einführungsphase der erste Kontakt mit dem Unterrichtsfach Informatik stattfindet, so dass zu Beginn Grundlagen des Fachs behandelt werden müssen.
Zunächst wird auf den Begriff der Information eingegangen und die Möglichkeit der Kodierung in Form von Daten thematisiert. Anschließend wird auf die Übertragung von Daten im Sinne des Sender-Empfänger-Modells eingegangen. Dabei wird eine überblickartige Vorstellung der Kommunikation von Rechnern in Netzwerken erarbeitet.
Des Weiteren soll der grundlegende Aufbau eines Rechnersystems im Sinne der Von-Neumann-Architektur erarbeitet werden und mit dem grundlegenden Prinzip der Datenverarbeitung (Eingabe-Verarbeitung-Ausgabe) in Beziehung gesetzt werden.
Bei der Beschäftigung mit Datenkodierung, Datenübermittlung und Datenverarbeitung ist jeweils ein Bezug zur konkreten Nutzung der informatischen Ausstattung der Schule herzustellen. So wird in die verantwortungsvolle Nutzung dieser Systeme eingeführt.

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Information, deren Kodierung und Speicherung Informatik als Wissenschaft der Verarbeitung von Informationen • Darstellung von Informationen in Schrift, Bild und Ton • Speichern von Daten mit informatischen Systemen am Beispiel der Schulrechner • Vereinbarung von Richtlinien zur Datenspeicherung auf den Schulrechnern (z.B. Ordnerstruktur, Dateibezeichner usw.)	Die Schülerinnen und Schüler • beschreiben und erläutern den Aufbau und die Arbeitsweise singulärer Rechner am Beispiel der „Von-Neumann-Architektur“ (A), • nutzen die im Unterricht eingesetzten Informatiksysteme selbstständig, sicher, zielführend und verantwortungsbewusst (D), • nutzen das Internet zur Recherche, zum Datenaustausch und zur Kommunikation (K).	Beispiel: Textkodierung Kodierung und Dekodierung von Texten mit unbekannten Zeichensätzen (z.B. Wingdings) Beispiel: Bildkodierung Kodierung von Bildinformationen in Raster- und Vektorgrafiken Alternativ: Navigationssystem als Anschauungsmaterial
2. Informations- und Datenübermittlung in Netzen • „Sender-Empfänger-Modell“ und seine Bedeutung für die Eindeutigkeit von Kommunikation • Informatische Kommunikation in Rechnernetzen am Beispiel des Schulnetzwerks (z.B. Benut-zeranmeldung, Netzwerkordner, Zugriffsrechte, Client-Server) • Grundlagen der technischen Umsetzung von Rechnerkommunikation am Beispiel des Internets (z.B. Netzwerkadresse, Paketvermittlung, Protokoll) • Richtlinien zum verantwortungsvollen Umgang mit dem Internet		Beispiel: Rollenspiel zur Paketvermittlung im Internet Schülerinnen und Schüler übernehmen die Rollen von Clients und Routern. Sie schicken spielerisch Informationen auf Karten von einem Schüler-Client zum anderen. Jede Schülerin und jeder Schüler hat eine Adresse, jeder Router darüber hinaus eine Routingtabelle. Mit Hilfe der Tabelle und einem Würfel wird entschieden, wie ein Paket weiter vermittelt wird. Alternativ oder ergänzend kann FILIUS verwendet werden.
3. Aufbau informatischer Systeme • Identifikation typischer Komponenten informatischer Systeme und anschließende Beschränkung auf das Wesentliche, Herleitung der „Von-Neumann-Architektur“ • Identifikation des EVA-Prinzips (Eingabe-Verarbeitung-Ausgabe) als Prinzip der Verarbeitung von Daten und Grundlage der „Von-Neumann-Architektur“		Material: Demonstrationshardware Durch Demontage eines Demonstrationsrechners entdecken Schülerinnen und Schüler die verschiedenen Hardwarekomponenten eines Informatiksystems. Als Demonstrationsrechner bietet sich ein ausrangierter Schulrechner an.

Unterrichtsvorhaben EF-II:

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

Leitfrage: Wie lassen sich Gegenstandsbereiche informatisch modellieren und in einem Greenfoot-Szenario informatisch realisieren?
Zeitbedarf: 8 Stunden

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 ein Greenfoot-Szenario 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 zu thematisieren. So kann der Fokus hier auf Grundlagen wie der Unterscheidung zwischen Klasse und Objekt, Attribute, Methoden, Objektidentität und Objektzustand gelegt werden.

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

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Identifikation von Objekten • An einem lebensweltnahen Beispiel werden Objekte und Klassen im Sinne der objektorientierten Modellierung eingeführt. • Objekte werden durch Objektdiagramme, Klassen durch Klassendiagramme dargestellt. • Die Modellierungen werden einem konkreten Anwendungsfall entsprechend angepasst.	Die Schülerinnen und Schüler • ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • stellen den Zustand eines Objekts dar (D). • modellieren Klassen mit ihren Attributen, 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)	Beispiel: Vogelschwarm Schülerinnen und Schüler betrachten einen Vogelschwarm als Menge gleichartiger Objekte, die in einer Klasse mit Attributen und Methoden zusammengefasst werden können. Greenfoot-Szenario Wombats weitere Materialien: *Buch Schöningh, Informatik 1, Abschnitt 2.1 Objektorientierte Modellierung*
2. Analyse von Objekten und Klassen im Greenfoot-Szenario • Schritte der objektorientierten Analyse, Modellierung und Implementation.Analyse und Erprobung der Objekte im Greenfoot-Szenario.		*Buch Schöningh, Informatik 1, Abschnitt 2.2 Das Greenfoot-Szenario „Planetenerkundung“* *Von der Realität zu Objekten* *Von den Objekten zu Klassen, Klassendokumentation* *Objekte inspizieren* *Methoden aufrufen* *Objektidentität und Objektzustand*
3. Implementierung einfacher Aktionen in Greenfoot • Grundaufbau des Quelltexts einer Java-Klassenimplementation eigener Methoden, Dokumentationen • Deklaration und Initialisierung von Objekten • Programme übersetzen (Aufgabe des Compilers) und testen.		Beispiel: Wombats, oder Igel o.ä. Programmierung in Greenfoot Methoden schreiben Programme übersetzen

Unterrichtsvorhaben EF-III:

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

Leitfragen: Wie lassen sich Aktionen von Objekten flexibel realisieren?
Zeitbedarf: 18 Stunden

Vorhabenbezogene Konkretisierung:
Der Schwerpunkt dieses Unterrichtsvorhabens besteht darin, das Verhalten von Objekten flexibel zu programmieren, um u.a. ein Objekt sich per Zufallssteuerung bewegen zu lassen oder über die Tastatur zu steuern. Zunächst wird ein Greenfoot-Szenario erarbeitet, um ein einfaches Spiel zu realisieren. Für die Umsetzung dieses Projekts werden Kontrollstrukturen in Form von Wiederholungen und Verzweigungen sowie eine Variable zum Zählen benötigt und eingeführt.
Sind an einem solchen Beispiel im Schwerpunkt Wiederholungen und Verzweigungen eingeführt worden, sollen diese Konzepte an weiteren Beispielprojekten eingeübt werden. Dabei muss es sich nicht zwangsläufig um solche handeln, bei denen Kontrollstrukturen lediglich zur Animation verwendet werden. Auch die Erzeugung größerer Mengen grafischer Objekte und deren Verwaltung in einem Feld kann ein Anlass zur Verwendung von Kontrollstrukturen sein.
Ein weiterer Schwerpunkt des Unterrichtsvorhabens liegt auf der Verwendung logischer Verknüpfungen und 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.

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien

1. Bewegung graphischer Objekte in Greenfoot-Szenarien • Kontinuierliche Verschiebung eines Objekts mit Hilfe einer Schleife (While-Schleife) • Tastaturabfrage zur Steuerung von Objekten • Meldungen zur Kollision zweier Objekte mit Hilfe von Verzweigungen (IF-Anweisungen) • Verknüpfung von Bedingungen durch die logischen Verknüpfungen UND, ODER, NICHT	Die Schülerinnen und Schüler • analysieren und erläutern einfache Algorithmen und Programme (A), • entwerfen einfache Algorithmen und stellen sie umgangssprachlich und grafisch dar (M), • modifizieren einfache Algorithmen und Programme (I), • modellieren Klassen mit ihren Attributen, ihren Methoden und Assoziationsbeziehungen (M), • modifizieren einfache Algorithmen und Programme(I), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen zu (M), • implementieren Algorithmen unter Verwendung von Variablen und Wertzuwei-	Beispiel: Crab1 Die Schülerinnen und Schüler realisieren mit Objekten der Greenfoot-Umgebung ein Spiel, bei dem sich eine Krabbe über den Bildschirm bewegt und Würmer frisst. Beispiel: Crab2 (Räuber-Beute-System) Die Schülerinnen und Schüler modellieren und erstellen eine Klasse „Hummer“, mit deren Hilfe die Räuber simuliert werden können. Die Krabbe ist Beute und wird über Tastatur gesteuert Würmer zählen, Counter Beispiel: Kara mehrere Übungsangebote (mit Differenzierung nach Schwierigkeitsgrad) Materialien: Buch Schöningh, Informatik 1, Kapitel 4 und 4

2. Erstellen und Verwalten größerer Mengen einfacher grafischer Objekte (Objekte) • Implementierung eigener Methoden mit lokalen Variablen, auch zur Realisierung einer Zählschleife • Implementierung eigener Methoden mit Parameterübergabe und/oder Rückgabewert • Implementierung von Konstruktoren • Realisierung von Attributen • Erzeugung von Objekten mit Hilfe von Zählschleifen (FOR-Schleife) • Verwaltung von Objekten in eindimensionalen Feldern (Arrays) • Animation von Objekten, die in eindimensionalen Feldern (Arrays) verwaltet werden	sungen, Kontrollstrukturen sowie Methodenaufrufen (I), • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I), • ordnen Klassen, Attributen und Methoden ihren Sichtbarkeitsbereich zu (M), • 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).	Beispiel: Musiker umdrehen Die Schülerinnen und Schüler realisieren eine festgelegte Anzahl von Musikern, die sich in einer Reihe aufstellen. Die Musiker drehen sich zu Beginn zufällig nach rechts oder links und müssen sich z.T. wieder umdrehen. Beispiel: Game Of Life Die Schülerinnen und Schüler realisieren das Spiel Game Of Life.

Unterrichtsvorhaben EF-IV:

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

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

Zeitbedarf: 18 Stunden

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.

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Umsetzung von Anforderungen in Entwurfsdiagramme • Aus Anforderungsbeschreibungen werden Objekte mit ihren Eigenschaften identifiziert • Gleichartige Objekte werden in Klassen (Entwurf) zusammengefasst und um Datentypen und Methoden erweitert	Die Schülerinnen und Schüler • analysieren und erläutern eine objektorientierte Modellierung (A), 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),	Buch Schöningh, Informatik 1, Kapitel 6 Klassenentwurf 6.1. Von der Realität zum Programm 6.2. Objekte identifizieren 6.3. Klassen und Beziehungen entwerfen Beispiel: Luftballons Die Schülerinnen und Schüler entwickeln ein Spiel, bei dem Luftballons über den Bildschirm schweben und durch anklicken mit der Maus zum Zerplatzen gebracht werden können. Materialien: zugehöriges Greenfoot-Szenario
2. Implementationsdiagramme als erster Schritt der Programmierung • Erweiterung des Entwurfsdiagramms um Konstruktoren und get- und set-Methoden • Festlegung von Datentypen in Java sowie von Rückgaben und Parametern • Entwicklung von Klassendokumentationen • Erstellung von Sequenzdiagrammen als Vorbereitung • Vorbereitung für die Programmierung		Buch Schöningh, Informatik 1, Kapitel 6 Klassenentwurf 6.4 Klassen und Beziehungen implementieren 6.5 Vererbung Informationsblatt: Implementationsdiagramme (Download EF-IV.2)

3. Programmierung anhand der Dokumentation und des Implementations- und Sequenzdiagrammes • Klassen werden in Java-Quellcode umgesetzt • Das Geheimnisprinzip wird umgesetzt • Einzelne Klassen und das Gesamtsystem werden anhand der Anforderungen und Dokumentationen auf ihre Korrektheit überprüft.	• 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)	Buch Schöningh, Informatik 1, Kapitel 6 Klassenentwurf 6.4 Klassen und Beziehungen implementieren 6.5 Vererbung
4.Vererbungsbeziehungen • Das Grundprinzip der Vererbung wird erarbeitet • Die Vorteile der Vererbungsbeziehungen • Vererbung wird implementiert		Buch Schöningh, Informatik 1, Kapitel 6 Klassenentwurf 6.5 Vererbung
5. Softwareprojekt • Analyse und Dekonstruktion eines Spiels (Modelle, Quelltexte) • Erweiterung des Spiels um weitere Funktionalitäten • Modellierung eines Spiels aufgrund einer An-		Buch Schöningh, Informatik 1, Kapitel 8 Softwareprojekte 8.1 Softwareentwicklung 8.2 Oberflächen

forderungsbeschreibung, inklusive einer grafischen Benutzeroberfläche • (arbeitsteilige) Implementation des Spiels		Beispiel: Implementation LOGIN

Unterrichtsvorhaben EF-V:

Thema: **Such- und Sortieralgorithmen anhand kontextbezogener Beispiele**

Leitfragen: Wie können Objekte bzw. Daten effizient sortiert werden, so dass eine schnelle Suche möglich wird?
Zeitbedarf: 9 Stunden

Vorhabenbezogene Konkretisierung:
Dieses Unterrichtsvorhaben beschäftigt sich mit der Erarbeitung von Such- und Sortieralgorithmen. Der Schwerpunkt des Vorhabens liegt dabei auf den Algorithmen selbst und nicht auf deren Implementierung in einer Programmiersprache, auf die in diesem Vorhaben vollständig verzichtet werden soll.
Zunächst erarbeiten die Schülerinnen und Schüler mögliche Einsatzszenarien für Such- und Sortieralgorithmen, um sich der Bedeutung einer effizienten Lösung dieser Probleme bewusst zu werden. Anschließend werden Strategien zur Sortierung mit Hilfe eines explorativen Spiels von den Schülerinnen und Schülern selbst erarbeitet und hinsichtlich der Anzahl notwendiger Vergleiche auf ihre Effizienz untersucht.
Daran anschließend werden die erarbeiteten Strategien systematisiert und im Pseudocode notiert. Die Schülerinnen und Schüler sollen auf diese Weise das Sortieren durch Vertauschen, das Sortieren durch Auswählen und mindestens einen weiteren Sortieralgorithmus kennen lernen.
Des Weiteren soll das Prinzip der binären Suche behandelt und nach Effizienzgesichtspunkten untersucht werden.

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien

1. Explorative 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 durch die Schülerinnen und Schüler	Die Schülerinnen und Schüler • beurteilen die Effizienz von Algorithmen am Beispiel von Sortierverfahren hinsichtlich Zeit und Speicherplatzbedarf (A), • entwerfen einen weiteren Algorithmus zum Sortieren (M), • analysieren Such- und Sortieralgorithmen und wenden sie auf Beispiele an (D).	Beispiel: Sortieren mit Waage Die Schülerinnen und Schüler bekommen die Aufgabe, kleine, optisch identische Kunststoffbehälter aufsteigend nach ihrem Gewicht zu sortieren. Dazu steht ihnen eine Balkenwaage zur Verfügung, mit deren Hilfe sie das Gewicht zweier Behälter vergleichen können. Materialien: Computer science unplugged – Sorting Algorithms, URL: www.csunplugged.org/sorting-algorithms abgerufen: 30. 03. 2014

2. Systematisierung von Algorithmen und Effizienzbetrachtungen • Formulierung (falls selbst gefunden) 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 • Variante des Sortierens durch Auswählen (Nutzung eines einzigen oder zweier Felder bzw. lediglich eines einzigen zusätzlichen Ablageplatzes oder mehrerer neuer Ablageplätze) • Effizienzbetrachtungen an einem konkreten Beispiel bezüglich der Rechenzeit und des Speicherplatzbedarfs Analyse des weiteren Sortieralgorithmus (sofern nicht in Sequenz 1 und 2 bereits geschehen)		Beispiele: Sortieren durch Auswählen, Sortieren durch Vertauschen, Quicksort Quicksort ist als Beispiel für einen Algorithmus nach dem Prinzip Teile und Herrsche gut zu behandeln. Kenntnisse in rekursiver Programmierung sind nicht erforderlich, da eine Implementierung nicht angestrebt wird. Materialien: Computer science unplugged – Sorting Algorithms, URL: www.csunplugged.org/sorting-algorithms abgerufen: 30. 03. 2014
3. 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		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.

Unterrichtsvorhaben EF-VI:

Thema: **Geschichte der digitalen Datenverarbeitung und die Grundlagen des Datenschutzes**

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

Zeitbedarf: 15 Stunden

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

Anschließend wird verstärkt auf den Aspekt des Datenschutzes eingegangen. Dazu wird das Bundesdatenschutzgesetz in Auszügen behandelt und auf schülernahe Beispielsituationen zur Anwendung gebracht. Dabei steht keine formale juristische Bewertung der Beispielsituationen im Vordergrund, die im Rahmen eines Informatikunterrichts auch nicht geleistet werden kann, sondern vielmehr eine persönliche Einschätzung von Fällen im Geiste des Datenschutzgesetzes.

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Selbstständige Erarbeitung von Themen durch die Schülerinnen und 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: Stellenwert-systeme 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 Schülerinnen und Schüler	Die Schülerinnen und Schüler • bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen (A), • erläutern wesentliche Grundlagen der Geschichte der digitalen Datenverarbeitung (A), • stellen ganze Zahlen und Zeichen in Binärcodes dar (D), • interpretieren Binärcodes als Zahlen und Zeichen (D), • nutzen das Internet zur Recherche, zum Datenaustausch und zur Kommunikation (K).	Beispiel: Ausstellung zu informatischen Themen Die Schülerinnen und Schüler bereiten eine Ausstellung zu informatischen Themen vor. Dazu werden Stellwände und Plakate vorbereitet, die ggf. auch außerhalb des Informatikunterrichts in der Schule ausgestellt werden können. Materialien: Schülerinnen und Schüler recherchieren selbstständig im Internet, in der Schulbibliothek, in öffentlichen Bibliotheken, usw.
2. Vertiefung des Themas Datenschutz • Erarbeitung grundlegender Begriffe des Datenschutzen • Problematisierung und Anknüpfung an die Lebenswelt der Schülerinnen und Schüler • Diskussion und Bewertung von Fallbeispielen aus dem Themenbereich „Datenschutz“		Beispiel: Fallbeispiele aus dem aktuellen Tagesgeschehen Die Schülerinnen und Schüler bearbeiten Fallbeispiele aus ihrer eigenen Erfahrungswelt oder der aktuellen Medienberichterstattung. Materialien: Materialblatt zum Bundesdatenschutzgesetz (Download EF-VI.1)

Qualifikationsphase 1 (Grundkurs)

Unterrichtsvorhaben Q1-I:

Thema: **Rekursive Algorithmen in Anwendungskontexten**

Leitfragen: Wie können komplexe, rekursiv definierte Probleme informatisch gelöst werden?
Gibt es schnelle (rekursiv definierte) Sortier- und Suchverfahren?

Zeitbedarf: 20 Stunden

Absprachen zur vorhabenbezogenen Konkretisierung:
Ausgehend vom einem Problem wie z. B. "Türme von Hanoi" wird Rekursion als fundamentale Idee der Informatik zunächst im mathematischen, danach aber auch im informatischen Zusammenhang angewendet. Dabei wird zwischen linearen und verzweigten Rekursionen unterschieden und das Laufzeitverhalten bei hoher Rekursionstiefe analysiert.

Verschiedene Probleme (wie z. B. Rucksack, n-Damen, Springer, Irrgarten, etc.) werden algorithmisch rekursiv formuliert.

Bereits bekannte Such- und Sortierverfahren (z. B. Sortieren durch Einfügen, Sortieren durch Auswahl, Sequentielle Suche) werden rekursiv formuliert und durch leistungsfähigere Verfahren (z. B. Quicksort, Mergesort, Heapsort, Binäre Suche) ergänzt. Die neuen Verfahren werden implementiert.

Lernmittel / Materialien: Arbeitsblätter, eine didaktische Entwicklungsumgebung (z.B. Java-Editor, BlueJ o.a.)

Unterrichtssequenzen	zu **entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Entwicklung der Rekursion als fundamentale Idee der Informatik • rekursive Formeln • rekursive Funktionen / Methoden • rekursive Programmierung	Die Schülerinnen und Schüler • analysieren und erläutern Algorithmen und Programme (A), • modifizieren Algorithmen und Programme (I),	Türme von Hanoi mit Schwerpunkt auf • Zahl der Versetzungsoperationen • Protokollierung der Versetzungen
2. Rekursion in mathematischen und informatischen Kontexten • Rekursion in mathematischen Kontexten • Analyse und Darstellung des rekursiven Ablaufs einer Methode	Die Schülerinnen und Schüler • analysieren und erläutern Algorithmen und Programme (A), • modifizieren Algorithmen und Programme (I), • stellen iterative und rekursive Algorithmen	• Fakultätsfunktion (lineare Rekursion) • Fibonacci-Funktion (verzweigte Rekursion) • ggT (verzweigte Rekursion)

• Analyse des Laufzeitverhaltens linearer und verzweigter Rekursion	umgangssprachlich und grafisch dar (D), • testen Programme systematisch anhand von Beispielen (I). • untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen (A).	• evtl. Fraktale (Kochkurve, Sierpinskidreieck, etc.)
3. Probleme lösen mittels Rekursion • Erarbeitung verschiedener Probleme • Algorithmische Beschreibung einer Lösungsidee	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) • testen Programme systematisch anhand von Beispielen (I). • untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen (A).	• Rucksackproblem • Irrgartenproblem
4. Effiziente Sortierverfahren / Suchverfahren • Wiederholung bereits bekannter Sortier- und Suchverfahren als rekursiver Algorithmus • Erarbeitung eines Sortierverfahrens der Laufzeit O(n*log(n)) • Erarbeitung eines Suchverfahrens der Laufzeit O(log(n)).	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), • 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), • untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen (A).	• Demonstrationsprogramm zur Visualisierung von Sortierverfahren • Quicksortvisualisierung zur Erarbeitung der Idee • Suchspiel zur Erarbeitung der Binären Suche

Unterrichtsvorhaben Q1-II:

Thema: **Modellierung und Implementierung dynamischer Listenstrukturen und deren Anwendungen**

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

Zeitbedarf: 25 Stunden

Absprachen zur vorhabenbezogene Konkretisierung:
Nach Analyse einer Problemstellung in einem geeigneten Anwendungskontext, in dem Daten nach dem Last-In-First-Out-Prinzip verwaltet werden, werden der Aufbau von Stapeln am Beispiel dargestellt und die Operationen der Klasse Stack anhand der Abiturklasse erläutert. Anschließend werden für die Anwendung notwendige Klassen modelliert und implementiert.
Anschließend wird die Anwendung modifiziert, um den Umgang mit der Datenstruktur zu üben. Anhand einer Anwendung, in der Daten nach dem First-In-First-Out-Prinzip verwaltet werden, werden Unterschiede zwischen den Datenstrukturen Schlange und Stapel erarbeitet. Dabei werden die Operationen der Datenstruktur Schlange thematisiert und die entsprechende Abiturklasse Queue verwendet.
Um einfacher an Objekte zu gelangen, die zwischen anderen gespeichert sind, wird die Klasse List gemäß der Abiturklasse eingeführt und in einem Anwendungskontext verwendet.
(Je nach thematisierten Anwendungskontexten ist eine andere Reihenfolge bei der Behandlung der oben genannten Datenstrukturen möglich und zulässig.)
In mindestens einem weiteren Anwendungskontext wird die Verwaltung von Daten in Schlangen, Stapeln oder Listen vertieft. Modellierungen werden dabei in Entwurfs- und Implementationsdiagrammen dargestellt.

Lernmittel / Materialien: Arbeitsblätter, eine didaktische Entwicklungsumgebung (z.B. Java-Editor o.a.), Modellprogramm zur Visualisierung dynamischer Listenstrukturen

Unterrichtssequenzen	**zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. 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 • Modellierung und Implementierung der Anwendung unter Verwendung der Klasse Stack.	Die Schülerinnen und Schüler • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D), • modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen	Visualisierungsprogramm zu dynamischen Datenstrukturen

2. 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 • Modellierung und Implementierung der Anwendung unter Verwendung der Klasse Queue.	unter Angabe von Multiplizitäten (M), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M), • ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M), • stellen die Kommunikation zwischen Objekten grafisch dar (D), • stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D), • dokumentieren Klassen (D), • analysieren und erläutern objektorientierte Modellierungen (A), • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I). • analysieren und erläutern Algorithmen und Programme (A), • modifizieren Algorithmen und Programme (I), • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D), • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I), • testen Programme systematisch anhand von Beispielen (I). • erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (A)	Visualisierungsprogramm zu dynamischen Datenstrukturen
3. 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.		Visualisierungsprogramm zu dynamischen Datenstrukturen Einfache Oberfläche zur Visualisierung der linearen Liste, z. B. die Verwaltung einer Namensliste
4. Vertiefung / Anwendung einer linearen Datenstruktur im Anwendungskontext.	zusätzlich: Die Schülerinnen und Schüler • modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M), • verwenden bei der Modellierung geeigneter Problemstellungen Möglichkeiten der Polymorphie (M),	Umsetzung in einem größeren Projekt

Unterrichtsvorhaben Q1-III:

Thema: **Modellierung und Implementierung von Anwendungen mit dynamischen, nichtlinearen Datenstrukturen am Beispiel der Binärbäume**

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?
Zeitbedarf: 20 Stunden

Absprachen zur 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 Vorgaben für das Zentralabitur NRW verwendet. Klassen und ihre Beziehungen werden in Entwurfs- und Implementationsdiagrammen dargestellt. Die Funktionsweise von Methoden wird anhand grafischer Darstellungen von Binärbäumen erläutert.
Unter anderem sollen die verschiedenen Baumtraversierungen (Pre-, Post- und Inorder) implementiert werden. Unterschiede bezüglich der Möglichkeit, den Baum anhand der Ausgabe der Bauminhalte via Pre-, In- oder Postorder-Traversierung zu rekonstruieren, werden dabei ebenfalls angesprochen, indem die fehlende Umkehrbarkeit der Zuordnung Binärbaum => Inorder-Ausgabe an einem Beispiel verdeutlicht wird.
Eine Tiefensuche wird verwendet, um einen in der Baumstruktur gespeicherten Inhalt zu suchen.
Zu einer Problemstellung in einem entsprechenden Anwendungskontext werden die Operationen der Datenstruktur Suchbaum thematisiert und unter der Verwendung der Klasse BinarySearchTree der Vorgaben für das Zentralabitur weitere Klassen oder Methoden in diesem Kontext modelliert und implementiert. Die Suchbäume werden wie zuvor auch grafisch dargestellt.
Die Verwendung von binären Bäumen und Suchbäumen wird anhand weiterer Problemstellungen oder anderer Kontexten weiter geübt.

Lernmittel / Materialien: eine didaktische Entwicklungsumgebung (z.B. Java-Editor o.a.), Arbeitsblätter und Demonstrationsprogramme (aus dem Internet), Modellprogramm zur Visualisierung dynamischer Listenstrukturen

Unterrichtssequenzen	**zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. 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 Schülerinnen und Schüler • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D), • modellieren Klassen mit ihren Attributen, Methoden	• Demoprogramm und Arbeitsblätter zum Projekt Akinator

2. 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	und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M), • modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen oder lineare und nichtlineare Datensammlungen zu (M), • verwenden bei der Modellierung geeigneter Problemstellungen Möglichkeiten der Polymorphie (M), • ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M), • stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D), • dokumentieren Klassen (D), • analysieren und erläutern objektorientierte Modellierungen (A), • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokum. Klassenbibliotheken (I). • 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). • erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (A)	• Arbeitsblätter zur Projektarbeit Akinator • Traversierungsverfahren • Projektarbeit Morsebaum
3. 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 • Implementierung der Anwendung oder von Teilen der Anwendung inklusive einer sortierten Ausgabe des Baums		• Arbeitsblätter zum binären Suchbaum • Arbeitsblätter zum Baumsortieren und zu Traversierungsverfahren
4. Übung und Vertiefungen der Verwendung von Binärbäumen oder binären Suchbäumen anhand weiterer Problemstellungen		• Projektarbeiten zu einem der Themen Termbäume, Morsebäume, Stichwortbaum, Ahnenbaum

Unterrichtsvorhaben Q1-IV:

Thema: **Sicherheit und Datenschutz in Netzstrukturen**

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

Zeitbedarf: 15 Stunden (Das Thema bietet Vertiefungen darüber hinaus an vielen Stellen an. Eine erste Erprobung des Unterrichtsvorhabens kann über Möglichkeiten hierzu Aufschluss geben und den zeitlichen Bedarf ggf. verändern.)

Absprachen zur vorhabenbezogene Konkretisierung:
Ausgehend von einer Kommunikation zwischen zwei Kommunikationspartnern über eine einfache Leitung wird die Notwendigkeiten einer Datenübertragung erarbeitet. Die Schichten des TCP/IP-Schichtenmodells werden beispielgebunden erarbeitet (Basisbandübertragungsverfahren, Prüfverfahren, Vermittlungsschicht, Anwendungsprotokoll) und an einer Simulationssoftware getestet. Verschiedene Netzwerk-Topologien werden entwickelt und in Client-Server-Anwendungen simuliert.
Über die Sicherheit von Netzwerkanwendungen wird das Augenmerk auf verschiedene symmetrische und asymmetrische kryptografische Verfahren gelenkt, welche analysiert und erläutert werden. Fallbeispiele zur Datenschutzproblematik und zum Urheberrecht runden das Unterrichtsvorhaben ab.

Lernmittel / Materialien: Arbeitsblätter zur Einführung in Netzwerke, Simulationsprogramm "Filius", Arbeitsblätter und Skript zu "Filius", Arbeitsblätter zum Chat-Projekt, Kryptografie-Programm "Cryptool", Arbeitsblätter zu kryptografischen Verfahren

Unterrichtssequenzen	zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Schichten des TCP/IP-Protokolls • Erarbeitung der Notwendigkeiten einer Netzwerkkommunikation • Erarbeitung der Schichten des TCP/IP-Protokolls: Ethernet-, Internet-, Transport- und Anwendungsschicht	Die Schülerinnen und Schüler • beschreiben und erläutern Netzwerk-Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken (A),	• Arbeitsblätter zur Einführung in Netzwerke
2. Simulation von Netzwerken / Netzwerk-Topologien • Erarbeitung der Topologien: Peer-to-Peer, Sterntopologie, Baumtopologie, vermaschtes Netz • Simulation von Client-Server-Anwendungen • Simulation von Protokollen der Anwendungsschicht (POP3, SMTP, etc.)	Die Schülerinnen und Schüler • beschreiben und erläutern Netzwerk-Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken (A),	• Simulationssoftware FILIUS • Arbeitsblätter und Skript zu FILIUS
		Umsetzung eines Projektes • Chat • Anwendungsserver (z.B. Buchungsserver für Kinokarten)
4. Analyse und Erläuterung kryptografischer Verfahren • Erläuterung symmetrischer Verfahren: monoalphabetisch: Cäsar, polyalphabetisch: Vigenère • Erläuterung asymmetrischer Verfahren: RSA, Diffie-Hellman • Analyse der Sicherheit verschiedener Verfahren und Auswirkungen auf den Datenschutz/Urheberrecht	Die Schülerinnen und Schüler • analysieren und erläutern Algorithmen und Programme (A), • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D), • analysieren und erläutern Eigenschaften und Einsatzbereiche symmetrischer und asymmetrischer Verschlüsselungsverfahren (A). • untersuchen und bewerten anhand von Fallbeispielen Auswirkungen des Einsatzes von IF-systemen sowie Aspekte der Sicherheit von IF-systemen, 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).	• Arbeitsblätter zu kryptografischen Verfahren • CrypTool • Materialien von klicksafe (Zusatzmodule „Nicht alles was geht, ist auch erlaubt“, „Ich bin öffentlich ganz privat“, „Datenschutz im Internet“)

Qualifikationsphase 2 (Grundkurs)

Unterrichtsvorhaben Q2-I:

Thema: **Modellierung und Nutzung relationaler Datenbanken in Anwendungskontexten**

Leitfragen: Wie können Fragestellungen mit Hilfe einer Datenbank beantwortet werden?
Wie entwickelt man selbst eine Datenbank für einen Anwendungskontext?
Zeitbedarf: 25 Stunden

Absprachen zur vorhabenbezogene Konkretisierung:
Ausgehend von einer konkreten Anwendungssituation entwickeln die Schülerinnen und Schüler Ideen zur Modellierung von Daten und erkennen die Vorzüge von Datenbanksystemen.
In weiteren Anwendungskontexten müssen Datenbanken entwickelt werden, um Daten zu speichern und Informationen für die Beantwortung von möglicherweise auftretenden Fragen zur Verfügung zu stellen. Dafür ermitteln Schülerinnen und Schüler in den Anwendungssituationen Entitäten, zugehörige Attribute, Relationen und Kardinalitäten und stellen diese in Entity-Relationship-Modellen dar. Entity-Relationship-Modelle werden interpretiert und erläutert, modifiziert und in das Relationale Modell überführt.
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.
Ausgehend von einer vorhandenen Datenbasis, entwickeln Schülerinnen und Schüler für sie relevante Fragestellungen, die mit dem vorhandenen Datenbestand beantwortet werden sollen. Zur Beantwortung dieser Fragestellungen wird die vorgegebene Datenbank von den Schülerinnen und Schülern analysiert und die notwendigen Grundbegriffe für Datenbanksysteme sowie die erforderlichen SQL-Abfragen werden erarbeitet.
Mit Hilfe von SQL-Anweisungen können anschließend im Kontext relevante Informationen aus der Datenbank extrahiert werden. Die Operationen der Relationenalgebra werden mit SQL-Abfragen simuliert.
Anhand von Fallbeispielen werden Probleme bei der Nutzung von Datenbanksystemen aufgezeigt und im Hinblick auf gesellschaftliche Auswirkungen diskutiert.
(Die exakte Abfolge der Teilvorhaben sowie die Verzahnung theoretischer und praktischer Anteile kann hierbei je nach Anwendungskontext flexibel angepasst werden.)

Lernmittel / Materialien: Arbeitsblätter und Demonstrationsprogramme, WebVideo Datenbank (videocenter.schule.de), SQL-Seiten (z. B. die Seite www.tinohempel.de, VideoCenter-Datenbank der Senatsverwaltung für Bildung, Jugend und Wissenschaft, Berlin)

Unterrichtssequenzen	zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. 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)	Die Schülerinnen und Schüler • ermitteln für anwendungsbezogene Problemstellungen Entitäten, zugehörige Attribute, Relationen und Kardinalitäten (M), • stellen Entitäten mit ihren Attributen und die Beziehungen zwischen Entitäten mit Kardinalitäten in einem Entity-Relationship-Diagramm grafisch dar (D), • modifizieren eine Datenbankmodellierung (M), • modellieren zu einem Entity-Relationship-Diagramm ein relationales Datenbankschema (M), • bestimmen Primär- und Sekundärschlüssel (M), • analysieren und erläutern eine Datenbankmodellierung (A), • erläutern die Eigenschaften normalisierter Datenbankschemata (A), • überprüfen Datenbankschemata auf vorgegebene Normalisierungseigenschaften (D), • überführen Datenbankschemata in die 1. bis 3. Normalform (M).	• Arbeitsblätter zur Einführung in Datenbanken (Link: webvideo.schule.de)
2. 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,	Die Schülerinnen und Schüler • modifizieren eine Datenbankmodellierung (M), • bestimmen Primär- und Sekundärschlüssel (M), • analysieren und erläutern eine Datenbankmodellierung (A), • ermitteln Ergebnisse von Datenbankabfragen über mehrere verknüpfte Tabellen (D), • analysieren und erläutern die Syntax und Semantik einer Datenbankabfrage (A), • verwenden die Syntax und Semantik einer Datenbankabfragesprache, um Informationen aus einen Datenbanksystem zu extrahieren (I). • erläutern Eigenschaften und Aufbau von	• ABs zur Einführung in SQL • Informationen der Seite (http://tinoHempel.de) • VideoCenter-Datenbank mit Infos der Seiten (http://oszhdl.be.schule.de)

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: Vertiefungen am Beispiel der Relationenalgebra	Datenbanksystemen unter dem Aspekt der sicheren Nutzung (A),	
3. Gesellschaftliche Auswirkungen der Nutzung von Datenbanksystemen	• 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).	• Fallbeispiele zur Nutzung von Datenbanksystemen • Spiel zum Missbrauch von Daten: DataDealer (http://datadealer.com/de)

Unterrichtsvorhaben Q2-II:

Thema: **Endliche Automaten und Formale Sprachen**

Leitfragen: Wie kann man endliche Automaten genau beschreiben?
Wie können endliche Automaten modelliert werden?
Wie können Sprachen durch Grammatiken beschrieben werden?
Welche Zusammenhänge gibt es zwischen formalen Sprachen, Automaten und Grammatiken?
Zeitbedarf: 25 Stunden

Absprachen zur vorhabenbezogene Konkretisierung:
Ausgehend von einem konkreten Anwendungsbeispiel entwickeln die Schülerinnen und Schüler das Modell der Grammatik einer formalen Sprache und das Modell des endlichen Automaten. Die Schülerinnen und Schüler überführen Automaten in verschiedene Darstellungsformen und ermitteln die akzeptierte Sprache eines Automaten (z. B. in Form von regulären Ausdrücken). An einem Beispiel wird ein nichtdeterministischer Akzeptor als Alternative gegenüber einem entsprechenden deterministischen Akzeptor eingeführt.

Der Zusammenhang zwischen endlichen Automaten und regulären Grammatiken wird durch die Entwicklung allgemeingültiger Verfahren zur Transformation zwischen Automat und Grammatik dargestellt. Die Unzulänglichkeit endlicher Automaten und regulärer Grammatiken wird an Beispielen verdeutlicht.

Lernmittel / Materialien: Arbeitsblätter und Demonstrationsprogramme, Automatensimulationsprogramm (z. B. JFlap)

Unterrichtssequenzen	**zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Einführung in Automaten/Grammatiken • Grammatiken: Grammatik einer natürlichen Sprache Grammatik einer künstlichen Sprache Idee des Parsens • Automaten: erkennender Automat zu Symbolen einer Sprache Modell des endlichen Automaten Darstellungsformen Sprache eines Automaten als regulärer Ausdruck nichtdeterministische Automaten	Die Schülerinnen und Schüler • 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 endliche Automaten (M), • stellen endliche Automaten in Tabellen oder Graphen dar und überführen sie in die jeweils andere Darstellungsform (D), • entwickeln zur Grammatik einer regulären Sprache einen zugehörigen endlichen Automaten (M).	Mögliche Einstiege: • Grundidee eines Compilers: Scanner, Parser, Codierer
2. Zusammenhang zwischen endlichen Automaten und regulären Grammatiken • reguläre Grammatik: Definition Anwendungen • Zusammenhang zu endlichen Automaten • Grenzen der endlichen Automaten/regulären Grammatiken	Die Schülerinnen und Schüler • analysieren und erläutern Grammatiken regulärer Sprachen (A), • modifizieren Grammatiken regulärer Sprachen (M), • ermitteln die formale Sprache, die durch eine Grammatik erzeugt wird (A), • 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 im Anwendungszusammenhang auf (A).	Arbeitsblätter bzw. Leitprogramm Automatentheorie

Unterrichtsvorhaben Q2-III:

Thema: **Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit**

Leitfragen: Was sind die strukturellen Hauptbestandteile eines Computers und wie kann man sich die Ausführung eines maschinenahen Programms mit diesen Komponenten vorstellen?
Welche Möglichkeiten bieten Informatiksysteme und wo liegen ihre Grenzen?
Zeitbedarf: 10 Stunden

Absprachen zur 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.

Lernmittel / Materialien:

- Arbeitsblätter
- geeigneter Modellrechner (z. B. Johnny, WinAli, Microprocessor Simulator, o.a.)

Unterrichtssequenzen	**zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. 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	Die Schülerinnen und Schüler • erläutern die Ausführung eines einfachen maschinennahen Programms sowie die Datenspeicherung auf einer „Von-Neumann-Architektur“ (A), • untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen (A).	geeigneter Modellrechner • Johnny • WinAli • Microprocessor Simulator
2. Grenzen der Automatisierbarkeit: • Vorstellung und Unlösbarkeit des Halteproblems • Beurteilung des Einsatzes von Informatiksystemen hinsichtlich prinzipieller Möglichkeiten und prinzipieller Grenzen		Arbeitsblätter zum Halteproblem, Mögliche Verknüpfung zum Q1-I Unterricht

2.2 Grundsätze der Leistungsbewertung und Leistungsrückmeldung

Transparenz der Leistungsbeurteilung

Schulische Leistungsbewertung steht im Spannungsfeld pädagogischer und gesellschaftlicher Zielsetzung.
Unter pädagogischen Gesichtspunkten hat sie vornehmlich das Individuum im Blick. Hier soll sie über den Leistungszuwachs rückmelden und dadurch die Motivation für weitere Anstrengungen erhöhen. Sie ermöglicht den Schülerinnen und Schülern ihre noch vorhandenen fachlichen Defizite wie auch ihre Stärken und Fähigkeiten zu erkennen um dadurch ein realistisches Selbstbild aufzubauen. Sie ist Basis für gezielte individuelle Förderung.
Für die Erziehungsberechtigten sind Noten eine einfache und zentrale Information zum Leistungsstand ihrer Kinder. Sie bieten den Anlass, über die Ursache von Defiziten und über die Beseitigung von Lernschwierigkeiten verschiedenster Art Rücksprache zu halten. Noten sind zudem Grundlage und Anlass, in den halbjährlich stattfindenden pädagogischen Konferenzen über die Schwierigkeiten und besonderen Probleme einzelner Schüler wie auch Klassen zu beraten und Maßnahmen zur Verbesserung zu beschließen.
Schulische Leistungsbewertung ist eingebettet in die durch das Schulgesetz § 48 (Grundsätze der Leistungsbewertung), APO - GOSt §13 bis §17 sowie Kapitel 3 des Kernlehrplans Informatik für die gymnasiale Oberstufe vorgegebene Grundsätze und Verfahren. Daraus erwächst für die Schulen konkret die Aufgabe, sowohl die individuellen Schwächen und Stärken der Schüler zu diagnostizieren und gegebenenfalls die Defizite durch gezielte Maßnahmen zu beseitigen sowie besondere Begabungen zu fördern.
Die gesellschaftliche Funktion von Noten zu erfüllen ist der Schule aufgegeben. Noten entscheiden mit über Schullaufbahnen, Versetzungen und Abschlüsse. Zeugnisse sind mit entscheidender Parameter bei der Zuteilung von Berufs- und Lebenschancen. Daraus erwachsen für die Beurteilenden eine besondere Verantwortung und die Pflicht einer größtmöglichen Objektivität bei der Notenfindung.
Die Fachkonferenz Informatik legt die Kriterien für die Leistungsbeurteilung fest. Die Lehrerinnen und Lehrer machen diese Kriterien den Schülerinnen und Schülern transparent.

Grundsätze der Leistungsbeurteilung

Es gelten folgende Grundsätze der Leistungsbewertung:

- Lernerfolgsüberprüfungen sind ein kontinuierlicher Prozess. Bewertet werden alle im Zusammenhang mit dem Unterricht erbrachten Leistungen (schriftliche Arbeiten, mündliche Beiträge, praktische Leistungen).
- Leistungsbewertung bezieht sich auf die im Unterricht geförderten Kompetenzen.
- Die Lehrperson gibt den Schülerinnen und Schülern im Unterricht hinreichend Gelegenheit, die entsprechenden Anforderungen der Leistungsbewertung im Unterricht in Umfang und Anspruch kennenzulernen und sich auf sie vorzubereiten.
- Bewertet werden der Umfang, die selbstständige und richtige Anwendung der Kenntnisse, Fähigkeiten und Fertigkeiten sowie die Art der Darstellung.

Formen der Leistungsüberprüfung

Kursarbeiten bzw. Klausuren

Kursarbeiten bzw. Klausuren dienen der schriftlichen Überprüfung der Lernergebnisse einer vorausgegangenen Unterrichtsreihe. Sie sind so anzulegen, dass Sachkenntnisse und methodische Fertigkeiten nachgewiesen werden können. Sie bedürfen einer angemessenen Vorbereitung und verlangen klare Aufgabenstellungen. Im Umfang und Anforderungsniveau sind Kursarbeiten bzw. Klausuren abhängig von den kontinuierlich ansteigenden Anforderungen entsprechend dem Lehrplan.

Es ist darauf zu achten, dass nicht nur die Richtigkeit der Ergebnisse und die inhaltliche Qualität, sondern auch die angemessene Form der Darstellung unabdingbare Kriterien der Bewertung der geforderten Leistung sind.

Mitarbeit im Unterricht

Der Beurteilungsbereich „Mitarbeit im Unterricht“ erfasst die Qualität und Kontinuität der Beiträge, die die Schülerinnen und Schüler im Unterricht erbringen. Diese Beiträge sollen unterschiedliche mündliche und schriftliche Formen in enger Bindung an die Aufgabenstellung, die inhaltliche Reichweite und das Anspruchsniveau der jeweiligen Unterrichtseinheit umfassen.

Bei den mündlichen Leistungen im Unterricht sind zu bewerten:

- Beteiligung am Unterrichtsgespräch
- Zusammenfassungen zur Vor- und Nachbereitung des Unterrichts
- Präsentation von Arbeitsergebnissen
- Mitarbeit in Partner- und Gruppenarbeitsphase
- Umsetzung von Konzepten in Programme am Computer

Neben der Richtigkeit, Vollständigkeit und Komplexität der Gedankengänge sind die der Altersstufe angemessene sprachliche Darstellung und die Verwendung der Fachsprache von Bedeutung.

Bei der Unterrichtsgestaltung sind den Schülerinnen und Schülern hinreichend Möglichkeiten zur Mitarbeit zu eröffnen, z.B. durch

- praktische Leistungen am Computer als Werkzeug im Unterricht,
- Protokolle und Referate,
- Projektarbeit (oft in Form von Gruppenarbeit),
- Lernerfolgsüberprüfungen und schriftliche Übungen.

Individuelle Förderung

Die Lehrerinnen und Lehrer beobachten die individuellen Leistungen in allen Bereichen der Informatik über einen längeren Zeitraum, um auf dieser Grundlage ein Leistungsbild zu erhalten. Neben der Orientierung an den Kompetenzstandards der jeweiligen Jahrgangsstufe kann bei der Leistungsbewertung auch die jeweilige Entwicklung des Schülers bzw. der Schülerin, gemäß der zu beobachtenden Lern– und Denkfortschritte, berücksichtigt werden.

Der Informatikunterricht lebt von der verantwortungsvollen und selbständigen Arbeit der Schülerinnen und Schüler, so dass die Lehrperson die nötige Zeit hat, bei Bedarf gezielt und individuell zu fördern.

Leistungsstärkere Schülerinnen und Schüler können ihr Wissen anhand von vertiefenden Problemstellungen erweitern.

Bildung der Zeugnisnote

In die Note gehen alle im Unterricht erbrachten Leistungen ein. Dabei nehmen die Beurteilung der Kursarbeiten bzw. Klausuren den gleichen Stellenwert wie die Leistungen im Bereich der Mitarbeit im Unterricht ein. Zudem ist bei der Notenfindung die individuelle Lernentwicklung der Schülerinnen und Schüler angemessen zu berücksichtigen.

2.3 Lehr- und Lernmittel

Eingesetzte Lehrbücher und Arbeitsmaterialien:

- Skripte und Arbeitsblätter
- Informatik, Lehrwerk für die gymnasiale Oberstufe – Neubearbeitung, Schöningh-Verlag
- Arbeitsblätter und Programmvorlagen.

Mögliche einzusetzende Software (jeweils in der aktuellen Version):

- Java SDK
- BlueJ
- Greenfoot
- Java-Editor
- SQL (Internet)
- Weitere Demonstrationsprogramme