Schulinterner Lehrplan Informatik

FREIHERR-VOM-STEIN-GYMNASIUM

Informatik in der Oberstufe am Freiherr-vom-Stein Gymnasium Münster

Stand Dez. 2019

Das Fach Informatik wird zum Schuljahr 2020/21 am Freiherr-vom-Stein Gymnasium Münster eingeführt.

In der Einführungsphase werden folgende Themen behandelt:

EF-I	Einführung in die Nutzung von Informatiksystemen und in grundlegende Begrifflichkeiten
EF-II	Grundlagen der objektorientierten Analyse, Modellierung und Implementierung anhand von statischen Grafikszenen
EF-III	Grundlagen der objektorientierten Programmierung und algorithmischer Grundstrukturen in Java anhand von einfachen Animationen
EF-IV	Modellierung und Implementierung von Klassen- und Objektbeziehungen anhand von grafischen Spielen und Simulationen
EF-V	Such- und Sortieralgorithmen anhand kontextbezogener Beispiele
EF-VI	Geschichte der digitalen Datenverarbeitung und die Grundlagen des Datenschutzes

In der Einführungsphase wird unter anderem die didaktische Bibliothek „Greenfoot“ verwendet.

In der Qualifikationsphase werden die Unterrichtsvorhaben unter Berücksichtigung der Vorgaben für das Zentralabitur Informatik in NRW konkretisiert:

Q1-I Wiederholung der objektorientierten Modellierung und Programmierung

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

Q1-III Suchen und Sortieren auf linearen Datenstrukturen

Q1-IV Modellierung und Nutzung von relationalen Datenbanken in Anwendungskontexten

Q1-V Sicherheit und Datenschutz in Netzstrukturen

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

Q2-II Endliche Automaten und formale Sprachen

Q2-III Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit

Als Materialien werden die Lehrbücher **Informatik 1 und Informatik 2 des Schöningh – Verlages** eingesetzt.
Darüber hinaus bietet der Lehrplannavigator des Landes NRW zahlreiches Zusatzmaterial an, das auf die Inhalte des Lehrplanes abgestimmt ist.

Auf den folgenden Seiten sind die Unterrichtssequenzen mit den zu entwickelnden Kompetenzen und den jeweiligen Beispielen, Medien und Materialien zusammengefasst.

I) Einführungsphase

Schulinternes Curriculum	**Informatik**	FREIHERR-VOM-STEIN-GYMNASIUM
Einführung in die Nutzung von Informatiksystemen und in grundlegende Begrifflichkeiten EF-I	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Information, deren Kodierung und Speicherung (a) Informatik als Wissenschaft der Verarbeitung von Informationen (b) Darstellung von Informationen in Schrift, Bild und Ton (c) Speichern von Daten mit informatischen Systemen (d) Vereinbarung von Richtlinien zur Datenspeicherung auf den Schulrechnern	Die Schülerinnen und Schüler • beschreiben und erläutern den Aufbau und die Arbeitsweise von Rechnern am Beispiel der „Von-Neumann-Architektur“, • nutzen die im Unterricht eingesetzten Informatiksysteme selbstständig, sicher, zielführend und verantwortungsbewusst, • nutzen das Internet zur Recherche, zum Datenaustausch und zur Kommunikation.	*Beispiel*: Textkodierung *Beispiel*: Bildkodierung
2. Informations- und Datenübermittlung in Netzen (a) „Sender-Empfänger-Modell“ und seine Bedeutung für die Eindeutigkeit von Kommunikation (b) Informatische Kommunikation in Rechnernetzen am Beispiel des Schulnetzwerks (z.B. Benutzeranmeldung, Netzwerkordner, Zugriffsrechte, Client-Server) (c) Grundlagen der technischen Umsetzung von Rechnerkommunikation am Beispiel des Internets (z.B. Netzwerkadresse, Paketvermittlung, Protokoll) (d) Richtlinien zum verantwortungsvollen Umgang mit dem Internet		*Beispiel*: Rollenspiel zur Paketvermittlung im Internet, o.ä.
3. Aufbau informatischer Systeme (a) Identifikation typischer Komponenten informatischer Systeme und anschließende Beschränkung auf das Wesentliche, Herleitung der „Von-Neumann-Architektur“ (b) Identifikation des EVA-Prinzips (Eingabe-Verarbeitung-Ausgabe) als Prinzip der Verarbeitung von Daten und Grundlage der „Von-Neumann-Architektur“ Ca. 6 Ustd		*Material*: Lehrbücher, Demonstrationshardware Durch Demontage eines Demonstrationsrechners entdecken Schülerinnen und Schüler die verschiedenen Hardwarekomponenten

Grundlagen der objektorientierten Analyse, Modellierung und Implementierung anhand von statischen Grafikszenen EF-II	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Identifikation von Objekten (a) Am lebensweltnahen Beispielen werden Objekte im Sinne der Objektorientierten Modellierung eingeführt. (b) Objekte werden mit Objektkarten visualisiert und mit sinnvollen Attributen und Methoden versehen. (c) Objekte werden zu einer Objektsorte bzw. Objektklasse zusammengefasst. (d) Vertiefung: Modellierung weiterer Beispiele ähnlichen Musters	Die Schülerinnen und Schüler • ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen, • modellieren Klassen mit ihren Attributen, ihren Methoden und Assoziationsbeziehungen, • stellen die Kommunikation zwischen Objekten grafisch dar, • implementieren einfache Algorithmen unter Beachtung der Syntax und Semantik einer Programmiersprache, • stellen den Zustand eines Objekts dar.	*Beispiel:* Vogelschwarm Schülerinnen und Schüler betrachten einen Vogelschwarm als Menge gleichartiger Objekte, die in einer Klasse mit Attributen und Methoden zusammengefasst werden können. *Materialien:* Lehrbücher, Ergänzungsmaterialien zum Lehrplannavigator - Allgemeine Objektorientierung
2. Analyse von Klassen didaktischer Lernumgebungen (a) Objektorientierte Programmierung auf Grundlage vorhandener Klassen (b) Teilanalyse der Klassen der didaktischen Lernumgebungen „Greenfoot“.		*Materialien*: Lehrbücher, Dokumentation der didaktischen Bibliothek „Greenfoot“
3. Implementierung dynamischen Szenen (a) Grundaufbau einer Java-Klasse (b) Konzeption einer Szene mit Umgebung, Rover und weiteren sichtbaren Objekten (c) Deklaration und Initialisierung von Objekten (d) Methodenaufrufe mit Parameterübergabe zur Manipulation von Objekteigenschaften Ca. 8 Ustd		*Beispiel:* Umgebungsgestaltung Schülerinnen und Schüler erstellen ein Programm, das mit Hilfe von Objekten der Greenfoot-Umgebung einen Parcours auf den Bildschirm bringt. *Beispiel*: Roverfahrt Die Schülerinnen und Schüler programmieren mit Hilfe von Greenfoot-Objekten einen Rover, der den Parcours meistern kann.

Grundlagen der objektorientierten Programmierung und algorithmischer Grundstrukturen in Java anhand von einfachen Animationen EF-III	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Bewegungsanimationen einfacher grafischer Objekte mit Greenfoot (a) Kontinuierliche Fahrt eines Rovers (b) Tastaturabfrage zur Realisierung einer Schleifenbedingung (c) Mehrstufige Animationen mit mehreren sequenziellen Schleifen (d) Verwaltung von gesammelten Objekten (e) Meldungen zur Kollision (IF-Anweisungen)	Die Schülerinnen und Schüler • analysieren und erläutern einfache Algorithmen und Programme, • entwerfen einfache Algorithmen und stellen sie umgangssprachlich und grafisch dar, • ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen, • modellieren Klassen mit ihren Attributen, ihren Methoden und Assoziationsbeziehungen, • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen oder lineare Datensammlungen zu, • ordnen Klassen, Attributen und Methoden ihren Sichtbarkeitsbereich zu, • modifizieren einfache Algorithmen und Programme, • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken, • implementieren Algorithmen unter Verwendung von Variablen und Wertzuweisungen, Kontrollstrukturen sowie Methodenaufrufen, • implementieren einfache Algorithmen unter Beachtung der Syntax und Semantik einer Programmiersprache, • testen Programme schrittweise anhand von Beispielen, • interpretieren Fehlermeldungen und korrigieren den Quellcode.	*Beispiel*: Die Schülerinnen und Schüler realisieren mit Objekten der Greenfoot-Umgebung ein Spiel *Materialien:* Lehrbücher, Greenfoot
2. Erstellen und Verwalten größerer Mengen einfacher grafischer Objekte der Greenfoot-Umgebung (a) Erzeugung von Objekten mit Hilfe von Schleifen (b) Verwaltung von Objekten in Arrays (c) Animation von Objekten, die in eindimensionalen Feldern (Arrays) verwaltet werden		*Beispiel:* Automatische Parcourserstellung Die Schülerinnen realisieren einen Parcours der bei Programmaufruf automatisch erstellt wird. *Beispiel*: Sammeln und Verwalten von Wasser
3. Modellierung und Animation komplexerer grafisch repräsentierbarer Objekte (a) Modellierung eines Simulationsprogramms mit eigenen Klassen, die sich mit Hilfe von einfachen Greenfoot-Objekten implementieren lassen. (b) Implementierung eigener Methoden mit und ohne Parameterübergabe (c) Realisierung von Zustandsvariablen (d) Thematisierung des Geheimnisprinzips und des Autonomitätsprinzips von Objekten (e) Animation mit Hilfe des Aufrufs von selbstimplementierten Methoden (f) Vertiefung: Weitere Projekte Ca. 18 Ustd		*Beispiel: Gedächtnis für den Rover* Die Schülerinnen und Schüler modellieren und erstellen eine Klasse, mit deren Hilfe simuliert werden kann, wie der Rover sich Wege merken und wieder abrufen kann. *Beispiel:* Komplexe Problemstellungen Die Schülerinnen und Schüler erstellen eine sich gegenseitig komplexere Problemstellungen, die der Rover zu bewältigen hat. *Materialien:* Lehrbücher, Greenfoot

Modellierung und Implementierung von Klassen- und Objektbeziehungen anhand von grafischen Spielen und Simulationen EF-IV	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Vertiefung des Referenzbegriffs und Einführung des Prinzips der dynamischen Referenzierung (a) Einführung der Greenfoot-Objektselektion mit der Maus (b) Einführung der Oberklasse bei Greenfoot, die Eigenschaften an alle sichtbaren Objekte in Greenfoot vererbt.	Die Schülerinnen und Schüler • analysieren und erläutern eine objektorientierte Modellierung, • stellen die Kommunikation zwischen Objekten grafisch dar, • ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen, • modellieren Klassen mit ihren Attributen, ihren Methoden und Assoziationsbeziehungen, • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen oder lineare Datensammlungen zu, • ordnen Klassen, Attributen und Methoden ihren Sichtbarkeitsbereich zu, • modellieren Klassen unter Verwendung von Vererbung, • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken, • testen Programme schrittweise anhand von Beispielen, • interpretieren Fehlermeldungen und korrigieren den Quellcode,	*Beispiel*: Kreation neuer sichtbarer Objekte Die Schülerinnen und Schüler entwickeln neue sichtbare Objekte, mit weiteren anderen Eigenschaften, die sich in die Greenfoot-Umgebung integrieren lassen.
2. Entwicklung eines Spiels mit der Notwendigkeit von Kollisionskontrollen zwischen zwei oder mehr grafischen Objekten (a) Modellierung des Spiels ohne Berücksichtigung der Kollision mit Hilfe eines Implementationsdiagramms (b) Dokumentation der Klassen des Projekts (c) Implementierung eines Prototypen ohne Kollision (d) Ergänzung einer Kollisionsabfrage durch zusätzliche Assoziationsbeziehungen in Diagramm, Dokumentation und Quellcode (e) Verallgemeinerung der neuen Verwendung von Objektreferenzen (f) Vertiefung: Entwicklung weiterer Spiele und Simulationen mit vergleichbarer Grundmodellierung		*Beispiel:* Ausweichspiel Die Schülerinnen und Schüler entwickeln die Simulation eines Ausweichspieles, das weiteren Rovern ausweichen soll mit denen eine Kollision möglich ist. *Beispiel*: Kollisionsspiel Die Schülerinnen und Schüler entwickeln ein Spiel, bei dem autonome Rover mit einem intelligenten Rover eingefangen werden sollen. *Materialien:* Lehrbücher, Greenfoot

3. Erarbeitung einer Simulation mit grafischen Objekten, die sich durch unterschiedliche Ergänzungen voneinander unterscheiden (Vererbung durch Spezialisierung ohne Überschreiben von Methoden) (a) Analyse und Erläuterung einer Basisversion der grafischen Klasse (b) Realisierung von grafischen Erweiterungen zur Basisklasse mit und ohne Vererbung (Implementationsdiagramm und Quellcode) (c) Verallgemeinerung und Reflexion des Prinzips der Vererbung am Beispiel der Spezialisierung	• modifizieren einfache Algorithmen und Programme, • stellen Klassen, Assoziations- und Vererbungsbeziehungen in Diagrammen grafisch dar, • dokumentieren Klassen durch Beschreibung der Funktionalität der Methoden.	*Beispiel*: Die Schülerinnen und Schüler erstellen eine Simulation von Geländetypen, die unterschiedliche weitere Eigenschaften tragen.
4. Entwicklung einer komplexeren Simulation mit grafischen Elementen, die unterschiedliche Animationen durchführen (Vererbung mit Überschreiben von Methoden) (a) Analyse und Erläuterung einer einfachen grafischen Animationsklasse (b) Spezialisierung der Klasse zu Unterklassen mit verschiedenen Animationen durch Überschreiben der entsprechenden Animationsmethode (c) Reflexion des Prinzips der späten Bindung (d) Vertiefung: Entwicklung eines vergleichbaren Projekts mit einer (abstrakten) Oberklasse Ca 18 Ustd		*Beispiel*: Rovervariationen Die Schülerinnen und Schüler entwickeln eine Simulation von Rovern, bei der unterschiedliche Rover unterschiedliche Bewegungen durchführen. *Materialien:* Lehrbücher, Greenfoot

Such- und Sortieralgorithmen anhand kontextbezogener Beispiele EF-V	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Explorative Erarbeitung eines Sortierverfahrens (a) Sortierprobleme im Kontext informatischer Systeme und im Alltag (z.B. Dateisortierung, Tabellenkalkulation, Telefonbuch, Bundesligatabelle, usw.) (b) Vergleich zweier Elemente als Grundlage eines Sortieralgorithmus (c) Erarbeitung eines Sortieralgorithmus durch die Schülerinnen und Schüler	Die Schülerinnen und Schüler • beurteilen die Effizienz von Algorithmen am Beispiel von Sortierverfahren hinsichtlich Zeit und Speicherplatzbedarf, • entwerfen einen weiteren Algorithmus zum Sortieren, • analysieren Such- und Sortieralgorithmen und wenden sie auf Beispiele an.	*Beispiel:* Sortieren mit Waage Die Schülerinnen und Schüler bekommen die Aufgabe, kleine, optisch identische Kunststoffbehälter aufsteigend nach ihrem Gewicht zu sortieren. *Materialien:* Lehrbücher, Computer science unplugged – Sorting Algorithms
2. Systematisierung von Algorithmen und Effizienzbetrachtungen (a) Formulierung oder Erläuterung von mehreren Algorithmen im Pseudocode (auf jeden Fall: Sortieren durch Vertauschen, Sortieren durch Auswählen) (b) Anwendung von Sortieralgorithmen auf verschiedene Beispiele (c) Bewertung von Algorithmen anhand der Anzahl der nötigen Vergleiche (d) Variante des Sortierens durch Auswählen (e) Effizienzbetrachtungen an einem konkreten Beispiel (f) Analyse des weiteren Sortieralgorithmus		*Beispiele*: Sortieren durch Auswählen, Sortieren durch Vertauschen, Quicksort *Materialien:* Lehrbücher, Computer science unplugged – Sorting Algorithms
3. Binäre Suche auf sortierten Daten (a) Suchaufgaben im Alltag und im Kontext informatischer Systeme (b) Evtl. Simulationsspiel zum effizienten Suchen mit binärer Suche (c) Effizienzbetrachtungen zur binären Suche **Ca. 9 Ustd**		*Beispiel:* Simulationsspiel zur binären Suche nach Tischtennisbällen *Materialien:* Lehrbücher, Computer science unplugged – Searching Algorithms,

<table>
<tr><th>Geschichte der digitalen Datenverarbeitung und die Grundlagen des Datenschutzes EF-VI</th><th>Zu entwickelnde Kompetenzen</th><th>Beispiele, Medien, Materialien</th></tr>
<tr><td>1. Selbstständige Erarbeitung von Themen durch die Schülerinnen und Schüler
(a) Mögliche Themen zur Erarbeitung in Kleingruppen:
• „Eine kleine Geschichte der Digitalisierung: vom Morsen zum modernen Digitalcomputer“
• „Eine kleine Geschichte der Kryptographie: von Caesar zur Enigma“
• „Von Nullen, Einsen und mehr: Stellenwertsysteme und wie man mit ihnen rechnet“
• „Kodieren von Texten und Bildern: ASCII, RGB und mehr“
• „Auswirkungen der Digitalisierung: Veränderungen der Arbeitswelt und Datenschutz“
(b) Vorstellung und Diskussion durch Schülerinnen und Schüler</td><td rowspan="2">Die Schülerinnen und Schüler
• bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen,
• erläutern wesentliche Grundlagen der Geschichte der digitalen Datenverarbeitung,
• stellen ganze Zahlen und Zeichen in Binärcodes dar,
• interpretieren Binärcodes als Zahlen und Zeichen,
• nutzen das Internet zur Recherche, zum Datenaustausch und zur Kommunikation.</td><td>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:
Lehrbücher, Schülerinnen und Schüler recherchieren selbstständig im Internet, in der Schulbibliothek, in öffentlichen Bibliotheken, usw.</td></tr>
<tr><td>2. Vertiefung des Themas Datenschutz
(a) Erarbeitung grundlegender Begriffe des Datenschutzen
(b) Problematisierung und Anknüpfung an die Lebenswelt der Schülerinnen und Schüler
(c) Diskussion und Bewertung von Fallbeispielen aus dem Themenbereich „Datenschutz“
Ca 15 Ustd</td><td>Beispiel: Fallbeispiele aus dem aktuellen Tagesgeschehen

Materialien:
Lehrbücher, Materialblatt zum Bundesdatenschutzgesetz</td></tr>
</table>

II) Qualifikationsphase – GRUNDKURS Q1

Wiederholung der objektorientierten Modellierung und Programmierung Q1-I	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Wiederholung und Erweiterung der objektorientierten Modellierung und Programmierung durch Analyse und Erweiterung eines kontextbezogenen Beispiels (a) Analyse der Problemstellung (b) Analyse der Modellierung (Implementationsdiagramm) (c) Erweiterung der Modellierung im Implementationsdiagramm (Vererbung, abstrakte Klasse) (d) Kommunikation zwischen mindestens zwei Objekten (grafische Darstellung) (e) Dokumentation von Klassen (f) Implementierung der Anwendung oder von Teilen der Anwendung Ca 8 Ustd	Die Schülerinnen und Schüler • analysieren und erläutern objektorientierte Modellierungen, • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen, • modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten, • ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu, • modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren, • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken, • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen, • wenden eine didaktisch orientierte Entwicklungsumgebung zur Demonstration, zum Entwurf, zur Implementierung und zum Test von Informatiksystemen an, • interpretieren Fehlermeldungen und korrigieren den Quellcode, • stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar, • dokumentieren Klassen, • stellen die Kommunikation zwischen Objekten grafisch dar.	*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. oder *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. . *Materialien:* Lehrbücher, Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.1-Wiederholung

Modellierung und Implementierung von Anwendungen mit dynamischen, linearen Datenstrukturen Q1-II	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Die Datenstruktur der Klasse `Queue` (a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen (b) Erarbeitung der Funktionalität der Klasse `Queue` (c) Modellierung und Implementierung der Anwendung unter Verwendung eines oder mehrerer Objekte der Klasse `Queue`	Die Schülerinnen und Schüler • erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen, • analysieren und erläutern Algorithmen und Programme, • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen, • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu, • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen, • modifizieren Algorithmen und Programme, • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen, • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen, • interpretieren Fehlermeldungen und korrigieren den Quellcode, • testen Programme systematisch anhand von Beispielen, • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau.	*Beispiel:* Patientenwarteschlange (jeder kennt seinen Nachfolger bzw. alternativ: seinen Vorgänger) *Materialien:* Lehrbücher, Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.2 – Warteschlange
2. Die Datenstruktur der Klasse `Stack` (a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen (b) Erarbeitung der der Klasse `Stack` (c) Modellierung und Implementierung der Anwendung unter Verwendung eines oder mehrerer Objekte der Klasse `Stack`		*Beispiel:* Heftstapel In einem Heftstapel soll das Heft einer Schülerin gefunden werden. oder *Beispiel*: Kisten stapeln In einem Stapel nummerierter Kisten soll eine bestimmte Kiste gefunden und an einen Kunden geliefert werden.
3. Die Datenstruktur der Klasse `List` (a) Erarbeitung der Vorteile der Klasse `List` im Gegensatz zu den bereits bekannten linearen Strukturen (b) Modellierung und Implementierung einer kontextbezogenen Anwendung unter Verwendung der Klasse `List`. Ca 20 Ustd		*Beispiel*: Abfahrtslauf Bei einem Abfahrtslauf kommen die Skifahrer nacheinander an und werden nach ihrer Zeit in eine Rangliste eingeordnet. *Materialien:* Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.2 - Listen

Suchen und Sortieren auf linearen Datenstrukturen Q1-III	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Suchen von Daten in Listen und Arrays (a) Lineare Suche in Listen und in Arrays (b) Binäre Suche in Arrays als Beispiel für rekursives Problemlösen (c) Untersuchung der beiden Suchverfahren hinsichtlich ihrer Effizienz (Laufzeitverhalten, Speicherbedarf)	Die Schülerinnen und Schüler • analysieren und erläutern Algorithmen und Programme, • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen, • beurteilen die Effizienz von Algorithmen unter Berücksichtigung des Speicherbedarfs und der Zahl der Operationen, • entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“ und „Teilen und Herrschen“, • modifizieren Algorithmen und Programme, • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen, • implementieren und erläutern iterative und rekursive Such- und Sortierverfahren, • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen, • interpretieren Fehlermeldungen und korrigieren den Quellcode, • testen Programme systematisch anhand von Beispielen, • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar.	*Beispiel:* Karteiverwaltung oder *Beispiel*: Bundesjugendspiele *Materialien:* Lehrbücher, Ergänzungsmaterialien zum Lehrplannavigator
2. Sortieren in Listen und Arrays - Entwicklung und Implementierung von iterativen und rekursiven Sortierverfahren (a) Entwicklung und Implementierung eines einfachen Sortierverfahrens für eine Liste (b) Implementierung eines einfachen Sortierverfahrens für ein Feld (c) Entwicklung eines rekursiven Sortierverfahren für ein Feld (z.B. Sortieren durch Mischen)		*Beispiel:* Karteiverwaltung (s.o.) oder *Beispiel*: Bundesjugendspiele (s.o.) *Materialien:* (s.o.)
3. Untersuchung der Effizienz der Sortierverfahren „Sortieren durch direktes Einfügen“ und „Quicksort“ auf linearen Listen (a) Grafische Veranschaulichung (b) Untersuchung der Anzahl der Vergleichsoperationen und des Speicherbedarf Beurteilung der Effizienz der beiden Sortierverfahren Ca 16 Ustd		*Beispiel:* Karteiverwaltung (s.o.) oder *Beispiel*: Bundesjugendspiele (s.o.) *Materialien:* (s.o.)

Modellierung und Nutzung von relationalen Datenbanken in Anwendungskontexten **Q1-IV**	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Nutzung von relationalen Datenbanken (a) Aufbau von Datenbanken • Fragen zur vorhandenen Datenbank • Analyse der Struktur der vorgegebenen Datenbank (b) SQL-Abfragen • Analyse vorgegebener SQL-Abfragen • Analyse und Erarbeitung von SQL-Abfragen (c) Vertiefung an einem weiteren Datenbankbeispiel	Die Schülerinnen und Schüler • erläutern die Eigenschaften und den Aufbau von Datenbanksystemen unter dem Aspekt der sicheren Nutzung, • analysieren und erläutern die Syntax und Semantik einer Datenbankabfrage, • analysieren und erläutern eine Datenbankmodellierung, • erläutern die Eigenschaften normalisierter Datenbankschemata, • bestimmen Primär- und Sekundärschlüssel, • ermitteln für anwendungsbezogene Problemstellungen Entitäten, zugehörige Attribute, Relationen und Kardinalitäten, • modifizieren eine Datenbankmodellierung, • modellieren zu einem Entity-Relationship-Diagramm ein relationales Datenbankschema, • bestimmen Primär- und Sekundärschlüssel, • überführen Datenbankschemata in vorgegebene Normalformen, • verwenden die Syntax und Semantik einer Datenbankabfragesprache, um Informationen aus einen Datenbanksystem zu extrahieren, • ermitteln Ergebnisse von Datenbankabfragen über mehrere verknüpfte Tabellen, • stellen Entitäten mit ihren Attributen und die Beziehungen zwischen Entitäten in einem Entity-Relationship-Diagramm grafisch dar, • überprüfen Datenbankschemata auf vorgegebene Normalisierungseigenschaften.	*Beispiel:* VideoCenter VideoCenter ist die Simulation einer Online-Videothek *Beispiel*: Schulbuchausleihe
2. Modellierung von relationalen Datenbanken (a) Entity-Relationship-Diagramm • Ermittlung von Entitäten und Modellierung eines Datenbankentwurfs • Erläuterung und Modifizierung einer Datenbankmodellierung (b) Entwicklung einer Datenbank aus einem Datenbankentwurf (c) Modellierung eines relationalen Datenbankschematas zu einem Entity-Relationship-Diagramm (d) Redundanz, Konsistenz und Normalformen • Untersuchung einer Datenbank hinsichtlich Konsistenz und Redundanz • Überprüfung von Datenbankschemata hinsichtlich der 1. bis 3. Normalform und Normalisierung Ca. 20 Ustd		*Beispiel:* Fahrradverleih *Beispiel*: Reederei *Beispiel:* Buchungssystem . *Beispiel: Schulverwaltung*

Sicherheit und Datenschutz in Netzstrukturen Q1-V	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Daten in Netzwerken und Sicherheitsaspekte in Netzen sowie beim Zugriff auf Datenbanken (a) Beschreibung eines Datenbankzugriffs im Netz anhand eines Anwendungskontextes und einer Client-Server-Struktur zur Klärung der Funktionsweise eines Datenbankzugriffs (b) Netztopologien als Grundlage von Client-Server-Strukturen und TCP/IP-Schichtenmodell als Beispiel für eine Paketübermittlung in einem Netz (c) Vertraulichkeit, Integrität, Authentizität in Netzwerken sowie symmetrische und asymmetrische kryptografische Verfahren (Cäsar-, Vigenère-, RSA-Verfahren) als Methoden Daten im Netz verschlüsselt zu übertragen	Die Schülerinnen und Schüler • beschreiben und erläutern Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken, • analysieren und erläutern Eigenschaften und Einsatzbereiche symmetrischer und asymmetrischer Verschlüsselungsverfahren, • untersuchen und bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen, die Sicherheit von Informatiksystemen sowie die Einhaltung der Datenschutzbestimmungen und des Urheberrechts, • untersuchen und bewerten Problemlagen, die sich aus dem Einsatz von Informatiksystemen ergeben, hinsichtlich rechtlicher Vorgaben, ethischer Aspekte und gesellschaftlicher Werte unter Berücksichtigung unterschiedlicher Interessenlagen, • nutzen bereitgestellte Informatiksysteme und das Internet reflektiert zum Erschließen, zur Aufbereitung und Präsentation fachlicher Inhalte.	*Materialien:* Lehrbücher, Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben, Verschlüsselung Q1.5 - Zugriff auf Daten in Netzwerken
2. Fallbeispiele zur Datenschutzproblematik und zum Urheberrecht Ca 10 UStd		*Materialien:* Lehrbücher, *Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1 5* - Datenschutz beim Videocenter, Materialblatt-Datenschutzgesetz

IIb) Qualifikationsphase – GRUNDKURS Q2

Modellierung und Implementierung von Anwendungen mit dynamischen, nichtlinearen Datenstrukturen Q2-I	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Analyse von Baumstrukturen in verschiedenen Kontexten (a) Grundlegende Begriffe (Grad, Tiefe, Höhe, Blatt, Inhalt, Teilbaum, Ebene, Vollständigkeit) (b) Aufbau und Darstellung von binären Bäumen anhand von Baumstrukturen in verschiedenen Kontexten	Die Schülerinnen und Schüler • erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen, • analysieren und erläutern Algorithmen und Programme, • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen, • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen, • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu, • modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren, • verwenden bei der Modellierung geeigneter Problemstellungen die Möglichkeiten der Polymorphie, • entwickeln iterative und rekursive Algorithmen unter Nutzung der Konstruktionsstrategien „Modularisierung“ und „Teilen und Herrschen“, • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen, • modifizieren Algorithmen und Programme	*Beispiel:* Termbaum oder *Beispiel*: Ahnenbaum Die binäre Baumstruktur ergibt sich daraus, dass jede Person genau einen Vater und eine Mutter hat. *Weitere Beispiele für Anwendungskontexte für binäre Bäume:* *Beispiel*: Suchbäume (zur sortierten Speicherung von Daten) *oder* *Beispiel*: Entscheidungsbäume Um eine Entscheidung zu treffen, werden mehrere Fragen mit ja oder nein beantwortet. *oder* *Beispiel*: Codierungsbäume für Codierungen, deren Alphabet aus genau zwei Zeichen besteht *Materialien:* Lehrbücher, Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 – Binärbaum

2. Die Datenstruktur Binärbaum im Anwendungskontext unter Nutzung der Klasse `BinaryTree` (a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen (b) Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramms (c) Erarbeitung der Klasse `BinaryTree` und beispielhafte Anwendung der Operationen (d) Implementierung der Anwendung oder von Teilen der Anwendung (e) Traversierung eines Binärbaums	• nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen, • interpretieren Fehlermeldungen und korrigieren den Quellcode, • testen Programme systematisch anhand von Beispielen, • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau, • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar	*Beispiel*: Informatikerbaum *als binärer Baum* *Materialien:* Lehrbücher, Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 – Binärbaum
3. Die Datenstruktur binärer Suchbaum im Anwendungskontext unter Verwendung der Klasse BinarySearchTree (a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen (b) Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramm, grafische Darstellung eines binären Suchbaums und Erarbeitung der Struktureigenschaften (c) Erarbeitung der Klasse BinarySearchTree und Einführung des Interface Item zur Realisierung einer geeigneten Ordnungsrelation (d) Implementierung der Anwendung oder von Teilen der Anwendung inklusive einer sortierten Ausgabe des Baums		*Beispiel*: Informatikerbaum als *Suchbaum* Folgende Funktionalitäten werden benötigt: • Einfügen der Informatiker-Daten in den Baum • Suchen nach einem Informatiker über den Schlüssel Name • Ausgabe des kompletten Datenbestands in nach Namen sortierter Reihenfolge *Materialien:* Lehrbücher, Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 – Binärer Suchbaum
4. Übung und Vertiefungen der Verwendung von Binärbäumen oder binären Suchbäumen anhand weiterer Problemstellungen Ca 24 Ustd		*Beispiel:* Codierungsbäume (s.o.) oder Huffman-Codierung

Endliche Automaten und formale Sprachen Q2-II	Zu entwickelnde Kompetenzen	Beispiele, Medien oder Materialien
1. Endliche Automaten (a) Vom Automaten in den Schülerinnen und Schülern bekannten Kontexten zur formalen Beschreibung eines endlichen Automaten (b) Untersuchung, Darstellung und Entwicklung endlicher Automaten	Die Schülerinnen und Schüler • analysieren und erläutern die Eigenschaften endlicher Automaten einschließlich ihres Verhaltens auf bestimmte Eingaben, • analysieren und erläutern Grammatiken regulärer Sprachen, • zeigen die Grenzen endlicher Automaten und regulärer Grammatiken im Anwendungszusammenhang auf, • ermitteln die formale Sprache, die durch eine Grammatik erzeugt wird, • entwickeln und modifizieren zu einer Problemstellung endliche Automaten, • entwickeln und modifizieren zu einer Problemstellung endliche Automaten, • entwickeln zur akzeptierten Sprache eines Automaten die zugehörige Grammatik, • entwickeln zur Grammatik einer regulären Sprache einen zugehörigen endlichen Automaten, • modifizieren Grammatiken regulärer Sprachen, • entwickeln zu einer regulären Sprache eine Grammatik, die die Sprache erzeugt, • stellen endliche Automaten in Tabellen oder Graphen dar und überführen sie in die jeweils andere Darstellungsform, • ermitteln die Sprache, die ein endlicher Automat akzeptiert. • beschreiben an Beispielen den Zusammenhang zwischen Automaten und Grammatiken.	*Beispiele:* Cola-Automat, Geldspielautomat, Roboter, Zustandsänderung eines Objekts Cola-Automat *Materialien:* Lehrbücher, Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.2 – Endliche Automaten, Formale Sprachen
2. Untersuchung und Entwicklung von Grammatiken regulärer Sprachen (a) Erarbeitung der formalen Darstellung regulärer Grammatiken (b) Untersuchung, Modifikation und Entwicklung von Grammatiken (c) Entwicklung von endlichen Automaten zum Erkennen regulärer Sprachen die durch Grammatiken gegeben werden (d) Entwicklung regulärer Grammatiken zu endlichen Automaten		*Beispiele:* reguläre Grammatik für Wörter mit ungerader Parität, Grammatik für Wörter, die bestimmte Zahlen repräsentieren, Satzgliederungsgrammatik *Materialien: (s.o.)*
3. Grenzen endlicher Automaten Ca. 20 Ustd		*Beispiele:* Klammerausdrücke, a^nb^n im Vergleich zu $(ab)^n$

Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit Q2-III	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien oder Materialien**
1. Von-Neumann-Architektur und die Ausführung maschinennaher Programme a) prinzipieller Aufbau einer von Neumann-Architektur mit CPU, Rechenwerk, Steuerwerk, Register und Hauptspeicher b) einige maschinennahe Befehlen und ihre Repräsentation in einem Binär-Code, der in einem Register gespeichert werden kann c) Analyse und Erläuterung der Funktionsweise eines einfachen maschinennahen Programms	Die Schülerinnen und Schüler • erläutern die Ausführung eines einfachen maschinennahen Programms sowie die Datenspeicherung auf einer „Von-Neumann-Architektur“, • untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen.	*Beispiel:* Addition von 4 zu einer eingegeben Zahl mit einem Rechnermodell *Materialien:* Lehrbücher, Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.3 –Von-Neumann-Architektur und maschinennahe Programmierung
2. Grenzen der Automatisierbarkeit a) Vorstellung des Halteproblems b) Unlösbarkeit des Halteproblems c) Beurteilung des Einsatzes von Informatiksystemen hinsichtlich prinzipieller Möglichkeiten und prinzipieller Grenzen Ca 12 Ustd		*Beispiel:* Halteproblem *Materialien:* Lehrbücher, Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.3 - Halteproblem