
Informatik GK

Schulinternes Curriculum Qualifikationsphase

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

Die Schülerinnen und Schüler

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

Unterrichtsvorhaben Q1-I:

Thema: Kryptographie

Zeitbedarf: 20 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Erarbeitung und Implementierung von klassischen symmetrischen Verschlüsselungsverfahren.	Die Schülerinnen und Schüler <ul style="list-style-type: none">• analysieren und erläutern Algorithmen	Cäsar-Verfahren Vigenere-Verfahren Häufigkeitsanalyse

<p>(a) Zeichenkettenoperationen (b) GUI-Entwicklung mit dem Java-Editor (c) ASCII-Codierung (d) symmetrische Verschlüsselung</p> <p>2. Erarbeitung von asymmetrischen Verschlüsselungsverfahren.</p>	<p>und Programme (A),</p> <ul style="list-style-type: none"> • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • testen Programme systematisch anhand von Beispielen (I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • analysieren und erläutern Eigenschaften und Einsatzbereiche symmetrischer und asymmetrischer Verschlüsselungsverfahren (A), • wenden eine didaktisch orientierte Entwicklungsumgebung zur Demonstration, zum Entwurf, zur Implementierung und zum Test von Informatiksystemen an (I), 	<p>Matheprisma: Modul RSA</p>
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-------------------------------

Unterrichtsvorhaben Q1-II:

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

Zeitbedarf: 30 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
----------------------	-----------------------------	--------------------------------

<p>1. Die Datenstruktur Schlange im Anwendungskontext unter Implementierung einer Klasse Warteschlange</p> <p>(a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen</p> <p>(b) Erarbeitung der Funktionalität einer Warteschlange</p> <p>(c) Modellierung und Implementierung der Datenstruktur Warteschlange und der Anwendung.</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (A), • analysieren und erläutern Algorithmen und Programme (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • analysieren und erläutern objektorientierte Modellierungen (A), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nicht-lineare Datensammlungen zu (M), • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M), 	<p><i>Einstieg:</i> Obstempel Analogie Korb mit Haken zu Knotenklasse nutzen. Die Obstampelsimulation soll programmiert werden.</p> <p><i>Beispiel:</i> Patientenwarteschlange (jeder kennt seinen Nachfolger bzw. alternativ: seinen Vorgänger) Sobald ein Patient in einer Arztpraxis eintrifft, werden sein Name und seine Krankenkasse erfasst. Die Verwaltung der Patientenwarteschlange geschieht über eine Klasse, die hier als Wartezimmer bezeichnet wird. Wesentliche Operationen sind das „Hinzufügen“ eines Patienten und das „Entfernen“ eines Patienten, wenn er zur Behandlung gerufen wird. Die Simulationsanwendung stellt eine GUI zur Verfügung, legt ein Wartezimmer an und steuert die Abläufe. Wesentlicher Aspekt des Projektes ist die Modellierung des Wartezimmers mit Hilfe der Klasse Queue. Anschließend wird der Funktionsumfang der Anwendung erweitert: Patienten können sich zusätzlich in die Warteschlange zum Blutdruckmessen einreihen. Objekte werden von zwei Schlangen verwaltet.</p>
<p>2. Die Datenstruktur Stapel im Anwendungskontext unter Nutzung der Klasse Stack</p> <p>(a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen</p> <p>(b) Erarbeitung der Funktionalität der Klasse Stack</p> <p>(c) Modellierung und Implementierung der Anwendung unter Verwendung ei-</p>	<ul style="list-style-type: none"> • ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M), • stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D), • dokumentieren Klassen (D), • stellen die Kommunikation zwischen Objekten grafisch dar (D). • modifizieren Algorithmen und Programme 	<p><i>Beispiel:</i> PostFixRechner Es soll ein Taschenrechner mit GUI erstellt werden, der nach dem Postfix-Prinzip mit Hilfe eines Stapel arbeitet.</p> <p><i>Beispiel:</i> Integeraddition Ganzzahlen außerhalb des durch den Datentyp int abgedeckten Bereichs sollen mit Hilfe von Stapeln addiert werden.</p>

<p>nes oder mehrerer Objekte der Klasse Stack</p>	<ul style="list-style-type: none"> • (I), • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • testen Programme systematisch anhand von Beispielen (I), • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I), • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D). 	
<p>3. Die Datenstruktur lineare Liste im Anwendungskontext unter Nutzung der Klasse List</p> <p>(a) Erarbeitung der Vorteile der Klasse List im Gegensatz zu den bereits bekannten linearen Strukturen</p> <p>(b) Modellierung und Implementierung einer kontextbezogenen Anwendung unter Verwendung der Klasse List.</p>		<p><i>Einstieg:</i> Erarbeitung der Operationen der Klasse List und einfacher Iterationen mit Hilfe der Listendemo.</p> <p><i>Beispiel:</i> ToDoListe Es sollen Aufgaben mit Priorität in einer Liste gespeichert werden. Hier wird auch das Prinzip der Prioritätswarteschlange deutlich.</p>
<p>4. Vertiefung - Anwendungen von Listen, Stapeln oder Schlangen in mindestens einem weiteren Kontext</p> <p>(a) Die Vertiefung kann in Form von Aufgaben oder weiteren Projekten geschehen.</p>		<p><i>Mögliches Beispiel:</i> Skispringen Ein Skispringer hat folgenden Ablauf: Nach dem Sprung erhält der Springer eine Punktzahl und wird nach dieser Punktzahl in eine Rangliste eingeordnet. Die besten 30 Springer qualifizieren sich für den zweiten Durchgang. Sie starten in umgekehrter Reihenfolge gegenüber der Platzierung auf der Rangliste. Nach dem Sprung erhält der Springer wiederum eine Punktzahl und wird nach der Gesamtpunktzahl aus beiden Durchgängen in die endgültige Rangliste eingeordnet.</p> <p><i>Mögliches Beispiel:</i> Rangierbahnhof Auf einem Güterbahnhof gibt es drei Gleise, die nur zu einer Seite offen sind. Wagons können also von einer Seite auf das Gleis fahren und nur rückwärts wieder hinausfahren. Die Wagons tragen Nummern, wobei die Nummer jedoch erst eingesehen werden kann, wenn der Wagon der</p>

		<p>vorderste an der offenen Gleisseite ist. (Zwischen den Wagons herumzuturnen, um die anderen Wagonnummern zu lesen, wäre zu gefährlich.) Zunächst stehen alle Wagons unsortiert auf einem Gleis. Ziel ist es, alle Wagons in ein anderes Gleis zu fahren, so dass dort die Nummern der Wagons vom Gleisende aus aufsteigend in richtiger Reihenfolge sind. Zusätzlich zu diesen beiden Gleisen gibt es ein Abstellgleis, das zum Rangieren benutzt werden kann.</p>
--	--	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Unterrichtsvorhaben Q2-III:

Thema: von-Neumann-Architektur und Ausführung maschinennaher Programme.

Zeitbedarf: 9 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien oder Materialien
<p>1. Von-Neumann-Architektur und die Ausführung maschinennaher Programme</p> <p>a) prinzipieller Aufbau einer von Neumann-Architektur mit CPU, Rechenwerk, Steuerwerk, Register und Hauptspeicher</p> <p>b) einige maschinennahe Befehlen und ihre Repräsentation in einem Binär-Code, der in einem Register gespeichert werden kann</p> <p>c) Analyse und Erläuterung der Funktionsweise eines einfachen maschi-</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • erläutern die Ausführung eines einfachen maschinennahen Programms sowie die Datenspeicherung auf einer „Von-Neumann-Architektur“ (A), 	<p><i>Material:</i> Simulationsprogramm Johnny</p> <p><i>Beispiel:</i> Addition zweier Zahlen, Multiplikation durch Addition abbilden</p>

Unterrichtsvorhaben Q1-IV:

Thema: Suchen und Sortieren auf linearen Datenstrukturen

Zeitbedarf: 15 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Suchen von Daten in Listen und Arrays</p> <p>(a) Lineare Suche in Listen und in Arrays (b) Binäre Suche in Arrays als Beispiel für rekursives Problemlösen (c) Untersuchung der beiden Suchverfahren hinsichtlich ihrer Effizienz (Laufzeitverhalten, Speicherbedarf)</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • analysieren und erläutern Algorithmen und Programme (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • beurteilen die Effizienz von Algorithmen unter Berücksichtigung des Speicherbedarfs und der Zahl der Operationen (A), 	<p><i>Beispiel:</i> Bundesjugendspiele Anhand der Läuferliste eines Laufwettbewerbs mit erreichten Zeiten sollen verschiedene Aufgaben gelöst werden:</p> <ul style="list-style-type: none"> • Finden des schnellsten Läufers. • Finden des schnellsten Mannes. • Berechnen der Durchschnittszeit. • Sortieren nach der Laufzeit oder alphabetisch.
<p>2. Sortieren in Listen und Arrays - Entwicklung und Implementierung von iterativen und rekursiven Sortierverfahren</p> <p>(a) Entwicklung und Implementierung eines einfachen Sortierverfahrens für eine Liste (b) Implementierung eines einfachen Sortierverfahrens für ein Feld (c) Entwicklung eines rekursiven Sortier-</p>	<ul style="list-style-type: none"> • entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“ und „Teilen und Herrschen“ (M), • modifizieren Algorithmen und Programme (I), • implementieren iterative und rekursive Algorithmen auch unter Verwendung von 	<p><i>Beispiel:</i> Simulationsprogramm mit einem Array von int-Werten An diesem Beispiel können große Datenmengen simuliert werden und damit empirische Laufzeitanalysen vorgenommen werden.</p>

<p>verfahren für eine Liste (z.B. Quicksort)</p>	<p>dynamischen Datenstrukturen (I),</p> <ul style="list-style-type: none"> • implementieren und erläutern iterative und rekursive Such- und Sortierverfahren (I), 	
<p>3. Untersuchung der Effizienz der Sortierverfahren „Sortieren durch direktes Einfügen“ und „Quicksort“ auf linearen Listen</p> <p>(a) Grafische Veranschaulichung der Sortierverfahren</p> <p>(b) Untersuchung der Anzahl der Vergleichsoperationen und des Speicherbedarf bei beiden Sortierverfahren</p> <p>(c) Beurteilung der Effizienz der beiden Sortierverfahren</p>	<ul style="list-style-type: none"> • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • testen Programme systematisch anhand von Beispielen (I), • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D). 	

Unterrichtsvorhaben Q1-V:

Thema: Modellierung und Nutzung von relationalen Datenbanken in Anwendungskontexten

Zeitbedarf: 30 Stunden

Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Nutzung von relationalen Datenbanken</p> <p>(a) Aufbau von Datenbanken und Grundbegriffe</p> <ul style="list-style-type: none">• Entwicklung von Fragestellungen zur vorhandenen Datenbank• Analyse der Struktur der vorgegebenen Datenbank und Erarbeitung der Begriffe Tabelle, Attribut, Datensatz, Datentyp, Primärschlüssel, Fremdschlüssel, Datenbankschema <p>(b) SQL-Abfragen</p> <ul style="list-style-type: none">• Analyse vorgegebener SQL-Abfragen und Erarbeitung der Sprachelemente von SQL (SELECT (DISTINCT) ... FROM, WHERE, AND, OR, NOT) auf einer Tabelle• Analyse und Erarbeitung von SQL-Abfragen auf einer und mehrerer Tabelle zur Beantwortung der Fragestellungen (JOIN, UNION, AS, GROUP BY, ORDER BY, ASC, DESC, COUNT, MAX, MIN, SUM, Arithmetische Operatoren: +, -, *, /, (...), Vergleichsoperatoren: =, <>, >, <, >=, <=, LIKE,	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none">• erläutern die Eigenschaften und den Aufbau von Datenbanksystemen unter dem Aspekt der sicheren Nutzung (A),• analysieren und erläutern die Syntax und Semantik einer Datenbankabfrage (A),• analysieren und erläutern eine Datenbankmodellierung (A),• erläutern die Eigenschaften normalisierter Datenbankschemata (A),• bestimmen Primär- und Sekundärschlüssel (M),• ermitteln für anwendungsbezogene Problemstellungen Entitäten, zugehörige Attribute, Relationen und Kardinalitäten (M),• modifizieren eine Datenbankmodellierung (M),• modellieren zu einem Entity-Relationship-Diagramm ein relationales Datenbankschema (M),• bestimmen Primär- und Sekundärschlüs-	<p><i>Beispiel:</i> Fehlstundendatenbank</p> <p><i>Beispiel:</i> Präsidentendatenbank</p>

<p>BETWEEN, IN, IS NULL)</p> <p>(c) Vertiefung an einem weiteren Datenbankbeispiel</p>	<p>sel (M),</p> <ul style="list-style-type: none"> • überführen Datenbankschemata in vorgegebene Normalformen (M), • verwenden die Syntax und Semantik einer Datenbankabfragesprache, um Informationen aus einem Datenbanksystem zu extrahieren (I), 	
<p>2. Modellierung von relationalen Datenbanken</p> <p>(a) Entity-Relationship-Diagramm</p> <ul style="list-style-type: none"> • 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 <p>(b) Entwicklung einer Datenbank aus einem Datenbankentwurf</p> <ul style="list-style-type: none"> • Modellierung eines relationalen Datenbankschemas zu einem Entity-Relationship-Diagramm inklusive der Bestimmung von Primär- und Sekundärschlüsseln <p>(c) Redundanz, Konsistenz und Normalformen</p> <ul style="list-style-type: none"> • 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) 	<ul style="list-style-type: none"> • ermitteln Ergebnisse von Datenbankabfragen über mehrere verknüpfte Tabellen (D), • stellen Entitäten mit ihren Attributen und die Beziehungen zwischen Entitäten in einem Entity-Relationship-Diagramm grafisch dar (D), • überprüfen Datenbankschemata auf vorgegebene Normalisierungseigenschaften (D). • untersuchen und bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen, die Sicherheit von Informatiksystemen sowie die Einhaltung der Datenschutzbestimmungen und des Urheberrechts (A), • untersuchen und bewerten Problemlagen, die sich aus dem Einsatz von Informatiksystemen ergeben, hinsichtlich rechtlicher Vorgaben, ethischer Aspekte und gesellschaftlicher Werte unter Berücksichtigung unterschiedlicher Interessenlagen (A), • nutzen bereitgestellte Informatiksysteme und das Internet reflektiert zum Erschließen, zur Aufbereitung und Präsentation fachlicher Inhalte (D). 	<p><i>Beispiel: Schulverwaltung</i></p> <p>In einer Software werden die Schulhalbjahre, Jahrgangsstufen, Kurse, Klassen, Schüler, Lehrer und Noten einer Schule verwaltet. Man kann dann ablesen, dass z.B. Schüler X von Lehrer Y im 2. Halbjahr des Schuljahrs 2011/2012 in der Jahrgangsstufe 9 im Differenzierungsbereich im Fach Informatik die Note „sehr gut“ erhalten hat. Dazu ist die Datenbank zu modellieren, ggf. zu normalisieren und im Datenbanksystem umzusetzen.</p> <p>Weiter sollen sinnvolle Abfragen entwickelt werden und das Thema Datenschutz besprochen werden.</p>
<p>1. Fallbeispiele zum Datenschutz</p>		<p>Auszug aus dem Bundesdatenschutzgesetz Volkszählungsurteil passende Fallbeispiele</p>

Unterrichtsvorhaben Q2-I:

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

Zeitbedarf: 26 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Analyse von Baumstrukturen in verschiedenen Kontexten</p> <p>(a) Grundlegende Begriffe (Grad, Tiefe, Höhe, Blatt, Inhalt, Teilbaum, Ebene, Vollständigkeit)</p> <p>(b) Aufbau und Darstellung von binären Bäumen anhand von Baumstrukturen in verschiedenen Kontexten</p> <p>2. Die Datenstruktur Binärbaum im Anwendungskontext unter Nutzung der Klasse BinaryTree</p> <p>(a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen im Anwendungskontext</p> <p>(b) Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramms</p> <p>(c) Erarbeitung der Klasse BinaryTree</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (A), • analysieren und erläutern Algorithmen und Programme (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M), • modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Verer- 	<p><i>Beispiel:</i> Morsebaum Das Morsealphabet wird anhand des Kriteriums Punkt (links) und Strich (rechts) in einem Binärbaum codiert. Mit Hilfe dieses Morsebaumes sollen Morsenachrichten decodiert und codiert werden.</p> <p><i>Mögliches Beispiel:</i> Termbaum Der Aufbau von Termen wird mit Hilfe von binären Baumstrukturen verdeutlicht.</p> <p><i>Mögliches Beispiel:</i> Ahnenbaum Die binäre Baumstruktur ergibt sich daraus, dass jede Person genau einen Vater und eine Mutter hat.</p> <p><i>Mögliches Beispiel:</i> Entscheidungsbäume Um eine Entscheidung zu treffen, werden mehrere Fragen mit ja oder nein beantwortet. Die Fragen, die möglich sind, wenn die Antwort auf eine Frage mit „ja“ beantwortet wird, befinden sich im linken Teilbaum, die</p>

<p>und beispielhafte Anwendung der Operationen</p> <p>(d) Implementierung der Anwendung oder von Teilen der Anwendung</p> <p>(e) Traversierung eines Binärbaums im Pre-, In- und Postorderdurchlauf</p>	<p>bung durch Spezialisieren und Generalisieren (M),</p> <ul style="list-style-type: none"> • verwenden bei der Modellierung geeigneter Problemstellungen die Möglichkeiten der Polymorphie (M), • entwickeln iterative und rekursive Algorithmen unter Nutzung der Konstruktionsstrategien „Modularisierung“ und „Teilen und Herrschen“ (M), • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I), • modifizieren Algorithmen und Programme (I), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • testen Programme systematisch anhand von Beispielen (I), • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D), • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D). 	<p>Fragen, die möglich sind, wenn die Antwort „nein“ lautet, stehen im rechten Teilbaum.</p>
<p>3. Die Datenstruktur binärer Suchbaum im Anwendungskontext unter Verwendung der Klasse BinarySearchTree</p> <p>(a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen</p> <p>(b) Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramm,</p> <p>(c) grafische Darstellung eines binären Suchbaums und Erarbeitung der Struktureigenschaften</p> <p>(d) Erarbeitung der Klasse BinarySearchTree und Einführung des Interface Item zur Realisierung einer geeigneten Ordnungsrelation</p> <p>(e) Implementierung der Anwendung oder von Teilen der Anwendung inklusive einer sortierten Ausgabe des Baums</p>	<p>• verwenden bei der Modellierung geeigneter Problemstellungen die Möglichkeiten der Polymorphie (M),</p> <p>• entwickeln iterative und rekursive Algorithmen unter Nutzung der Konstruktionsstrategien „Modularisierung“ und „Teilen und Herrschen“ (M),</p> <p>• implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I),</p> <p>• modifizieren Algorithmen und Programme (I),</p> <p>• nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I),</p> <p>• interpretieren Fehlermeldungen und korrigieren den Quellcode (I),</p> <p>• testen Programme systematisch anhand von Beispielen (I),</p> <p>• stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D),</p> <p>• stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D).</p>	<p><i>Mögliches Beispiel:</i> Suchbaum mit Ganzzahlen</p> <p><i>Mögliches Beispiel:</i> Informatikerbaum als <i>Suchbaum</i></p> <p>In einem binären <i>Suchbaum</i> werden die Namen und die Geburtsdaten von Informatikern lexikographisch geordnet abgespeichert.</p>
<p>4. Übung und Vertiefungen der Verwendung von Binärbäumen oder binären Suchbäumen anhand weiterer Problemstellungen</p>	<p>• verwenden bei der Modellierung geeigneter Problemstellungen die Möglichkeiten der Polymorphie (M),</p> <p>• entwickeln iterative und rekursive Algorithmen unter Nutzung der Konstruktionsstrategien „Modularisierung“ und „Teilen und Herrschen“ (M),</p> <p>• implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I),</p> <p>• modifizieren Algorithmen und Programme (I),</p> <p>• nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I),</p> <p>• interpretieren Fehlermeldungen und korrigieren den Quellcode (I),</p> <p>• testen Programme systematisch anhand von Beispielen (I),</p> <p>• stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D),</p> <p>• stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D).</p>	<p><i>Mögliche Beispiel:</i> Codierungsbäume oder Huffman-Codierung</p> <p><i>mögliches Beispiel:</i> Buchindex</p>

Unterrichtsvorhaben Q2-II:

Thema: Endliche Automaten und formale Sprachen

Zeitbedarf: 21 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien oder Materialien
<p>1. Endliche Automaten</p> <p>(a) Vom Automaten in den Schülerinnen und Schülern bekannten Kontexten zur formalen Beschreibung eines endlichen Automaten</p> <p>(b) Untersuchung, Darstellung und Entwicklung endlicher Automaten</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • analysieren und erläutern die Eigenschaften endlicher Automaten einschließlich ihres Verhaltens auf bestimmte Eingaben (A), • analysieren und erläutern Grammatiken regulärer Sprachen (A), 	<p><i>Mögliche Beispiele:</i> Cola-Automat, Videorekorder, Akzeptor für bestimmte Zahlen, Akzeptor für Teilwörter in längeren Zeichenketten, Akzeptor für Terme</p>
<p>2. Untersuchung und Entwicklung von Grammatiken regulärer Sprachen</p> <p>(a) Erarbeitung der formalen Darstellung regulärer Grammatiken</p> <p>(b) Untersuchung, Modifikation und Entwicklung von Grammatiken</p> <p>(c) Entwicklung von endlichen Automaten zum Erkennen regulärer Sprachen die durch Grammatiken gegeben werden</p> <p>(d) Entwicklung regulärer Grammatiken zu endlichen Automaten</p>	<ul style="list-style-type: none"> • zeigen die Grenzen endlicher Automaten und regulärer Grammatiken im Anwendungszusammenhang auf (A), • ermitteln die formale Sprache, die durch eine Grammatik erzeugt wird (A), • entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M), • entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M), • entwickeln zur akzeptierten Sprache eines Automaten die zugehörige Grammatik (M), 	<p><i>Mögliche Beispiele:</i> reguläre Grammatik für Wörter mit ungerader Parität, Grammatik für Wörter, die bestimmte Zahlen repräsentieren, Satzgliederungsgrammatik</p>
<p>3. Grenzen endlicher Automaten und</p> <p>4. Grenzen der Automatisierbarkeit</p> <p>a) Vorstellung des Halteproblems</p> <p>b) Unlösbarkeit des Halteproblems</p> <p>c) Beurteilung des Einsatzes von Informatiksystemen hinsichtlich prinzipieller</p>	<ul style="list-style-type: none"> • entwickeln zur Grammatik einer regulären Sprache einen zugehörigen endlichen Automaten (M), • modifizieren Grammatiken regulärer Sprachen (M), • entwickeln zu einer regulären Sprache 	<p><i>Beispiele:</i> Klammerausdrücke, $a^n b^n$ im Vergleich zu $(ab)^n$ Halteproblem</p>

<p>ler Möglichkeiten und prinzipieller Grenzen</p>	<p>eine Grammatik, die die Sprache erzeugt (M),</p> <ul style="list-style-type: none"> • stellen endliche Automaten in Tabellen oder Graphen dar und überführen sie in die jeweils andere Darstellungsform (D), • ermitteln die Sprache, die ein endlicher Automat akzeptiert (D). • beschreiben an Beispielen den Zusammenhang zwischen Automaten und Grammatiken (D). • untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen (A). 	
----------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--

Unterrichtsvorhaben Q2-III:

Thema: Netzstrukturen

Zeitbedarf: 9 Stunden

Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Daten in Netzwerken und Sicherheitsaspekte in Netzen sowie beim Zugriff auf Datenbanken</p> <p>(a) Beschreibung eines Datenbankzugriffs im Netz anhand eines Anwendungskontextes und einer Client-Server-Struktur zur Klärung der Funktionsweise eines Datenbankzugriffs</p> <p>(b) Netztopologien als Grundlage von Client-Server-Strukturen und TCP/IP-Schichtenmodell als Beispiel für eine</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> • beschreiben und erläutern Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken (A), 	<p><i>Material:</i> FILIUS</p>

<p>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</p>		
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--	--

Unterrichtsvorhaben Q2-IV:

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