Informatik LK

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.** **(a)** Zeichenkettenoperationen **(b)** GUI-Entwicklung mit dem Java-	Die Schülerinnen und Schüler • analysieren und erläutern Algorithmen und Programme (A), • beurteilen die syntaktische Korrekt-	Cäsar-Verfahren Vigenere-Verfahren Häufigkeitsanalyse Matheprisma: Modul RSA

Editor **(c)** ASCII-Codierung **(d)** symmetrische Verschlüsselung **2. Erarbeitung von asymmetrischen Verschlüsselungverfahren.**	heit 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),	

Unterrichtsvorhaben Q1-II:

Thema: Modellierung und Implementierung dynamischer linearer Datenstrukturen und Implementierung von Anwendungen unter Verwendung dynamischer, linearer Datenstrukturen
Zeitbedarf: 30 Stunden
Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Die Datenstruktur Schlange im Anwendungskontext unter Implementierung	Die Schülerinnen und Schüler	*Einstieg:* Obstampel Analogie Korb mit Haken zu Knotenklasse nutzen. Die Obstampelsimulation soll pro-

einer Klasse `Warteschlange` **(a)** Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen **(b)** Erarbeitung der Funktionalität einer Warteschlange **(c)** Modellierung und Implementierung der Datenstruktur Warteschlange und der Anwendung.	• modellieren Klassen mit ihren Attributen, Methoden und Assoziationsbeziehungen unter Angabe von Multiplizitäten (M), • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I), • erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (A), • analysieren und erläutern Algorithmen und Programme (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen oder lineare und nichtlineare Datensammlungen zu (M), • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • modifizieren Algorithmen und Programme (I), • implementieren Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (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	grammiert werden. *Beispiel:* 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.
2. Die Datenstruktur Stapel im Anwendungskontext unter Nutzung der Klasse `Stack` **(a)** Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen **(b)** Erarbeitung der Funktionalität der Klasse `Stack` **(c)** Modellierung und Implementierung der Anwendung unter Verwendung eines oder mehrerer Objekte der Klasse `Stack`		*Beispiel:* PostFixRechner Es soll ein Taschenrechner mit GUI erstellt werden, der nach dem Postfix-Prinzip mit Hilfe eines Stapel arbeitet. *Beispiel:* Integeraddition Ganzzahlen außerhalb des durch den Datentyp int abgedeckten Bereichs sollen mit Hilfe von Stapeln addiert werden.

3. Die Datenstruktur lineare Liste im Anwendungskontext unter Nutzung 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`.	(I), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • testen Programme systematisch anhand von Beispielen und mit Hilfe von Testanwendungen (I), • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D).	*Einstieg:* Erarbeitung der Operationen der Klasse `List` und einfacher Iterationen mit Hilfe der Listendemo. *Beispiel:* ToDoListe Es sollen Aufgaben mit Priorität in einer Liste gespeichert werden. Hier wird auch das Prinzip der Prioritätswarteschlange deutlich.
4. Vertiefung - Anwendungen von Listen, Stapeln oder Schlangen in mindestens einem weiteren Kontext **(a)** Die Vertiefung kann in Form von Aufgaben oder weiteren Projekten geschehen.		*Mögliches Beispiel:* Skispringen Ein Skispringen hat folgenden Ablauf: Nach dem Sprung erhält der Springer eine Punktzahl und wird nach dieser Punktzahl in eine Rangliste eingeordnet. Die besten 30 Springer qualifizieren sich für den zweiten Durchgang. Sie starten in umgekehrter Reihenfolge gegenüber der Platzierung auf der Rangliste. Nach dem Sprung erhält der Springer wiederum eine Punktzahl und wird nach der Gesamtpunktzahl aus beiden Durchgängen in die endgültige Rangliste eingeordnet. *Mögliches Beispiel*: Rangierbahnhof Auf einem Güterbahnhof gibt es drei Gleise, die nur zu einer Seite offen sind. Wagons können also von einer Seite auf das Gleis fahren und nur rückwärts wieder hinausfahren. Die Wagons tragen Nummern, wobei die Nummer jedoch erst eingesehen werden kann, wenn der Wagon der vorderste an der offenen Gleisseite ist. (Zwischen den Wagons herumzuturnen, um die anderen Wagonnummern zu lesen, wäre zu gefähr-

		lich.) 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.

Unterrichtsvorhaben Q1-III:

Thema: Suchen und Sortieren auf linearen Datenstrukturen
Zeitbedarf: 15 Stunden
Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	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) **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	Die Schülerinnen und Schüler • analysieren und erläutern Algorithmen und Programme (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • beurteilen die Effizienz von Algorithmen unter Berücksichtigung des Speicherbedarfs und der Zahl der Operationen (A), • entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“ , „Teilen und Herrschen“ und „Backtracking“(M), • modifizieren Algorithmen und Programme (I), • implementieren iterative und rekursive	*Beispiel*: Bundesjugendspiele Anhand der Läuferliste eines Laufwettbewerbs mit erreichten Zeiten sollen verschiedene Aufgaben gelöst werden: • Finden des schnellsten Läufers. • Finden des schnellsten Mannes. • Berechnen der Durchschnittszeit. • Sortieren nach der Laufzeit oder alphabetisch. *Beispiel:* Simulationsprogramm mit einem Array von int-Werten An diesem Beispiel können große Datenmengen simuliert werden und damit empirische Laufzeitanalysen vorgenommen werden.

(c) Entwicklung eines rekursiven Sortierverfahren für eine Liste (z.B. Quicksort)	Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I), • implementieren und erläutern iterative und rekursive Such- und Sortierverfahren unterschiedlicher Komplexitätsklassen (Speicherbedarf und Laufzeitverhalten) (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 und mit Hilfe von Testanwendungen (I), • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D).	
3. Untersuchung der Effizienz der Sortierverfahren „Sortieren durch direktes Einfügen“ und „Quicksort“ auf linearen Listen **(a)** Grafische Veranschaulichung der Sortierverfahren **(b)** Untersuchung der Anzahl der Vergleichsoperationen und des Speicherbedarf bei beiden Sortierverfahren **(c)** Beurteilung der Effizienz der beiden Sortierverfahren		

Unterrichtsvorhaben Q1-IV:

Thema: Modellierung und Implementierung von Baumstrukturen und Implementierung von Anwendungen unter Verwendung von Baumstrukturen
Zeitbedarf: 20 Stunden
Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	**Zu entwickelnde Kompetenzen**	**Beispiele, Medien, Materialien**
1. Analyse von Baumstrukturen in verschiedenen Kontexten	Die Schülerinnen und Schüler	*Beispiel:* Morsebaum Das Morsealphabet wird anhand des Kriteri-

(a) Grundlegende Begriffe (Grad, Tiefe, Höhe, Blatt, Inhalt, Teilbaum, Ebene, Vollständigkeit) **(b)** Aufbau und Darstellung von binären Bäumen anhand von Baumstrukturen in verschiedenen Kontexten **2. Die Datenstruktur Binärbaum im Anwendungskontext unter Nutzung der Klasse `BinaryTree`** **(a)** Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen im Anwendungskontext **(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 im Pre-, In- und Postorderdurchlauf	• 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), • beurteilen die Effizienz von Algorithmen unter Berücksichtigung des Speicherbedarfs und der Zahl der Operationen (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 oder lineare und nichtlineare Datensammlungen zu (M), • modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M), • verwenden bei der Modellierung geeigneter Problemstellungen die Möglichkeiten der Polymorphie (M), • entwickeln iterative und rekursive Algorithmen unter Nutzung der Konstruktionsstrategien „Modularisierung“ und „Teilen und Herrschen“ und „Backtraing“(M), • entwickeln mit didaktisch orientierten Entwicklungsumgebungen einfache Be-	ums Punkt (links) und Strich (rechts) in einem Binärbaum codiert. Mit Hilfe dieses Morsebaumes sollen Morsenachrichten decodiert und codiert werden. *Mögliches Beispiel:* Termbaum Der Aufbau von Termen wird mit Hilfe von binären Baumstrukturen verdeutlicht. *Mögliches Beispiel*: Ahnenbaum Die binäre Baumstruktur ergibt sich daraus, dass jede Person genau einen Vater und eine Mutter hat. *Mögliches Beispiel*: Entscheidungsbäume Um eine Entscheidung zu treffen, werden mehrere Fragen mit ja oder nein beantwortet. Die Fragen, die möglich sind, wenn die Antwort auf eine Frage mit „ja“ beantwortet wird, befinden sich im linken Teilbaum, die Fragen, die möglich sind, wenn die Antwort „nein“ lautet, stehen im rechten Teilbaum.
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, **(c)** grafische Darstellung eines binären Suchbaums und Erarbeitung der		*Mögliches Beispiel:* Suchbaum mit Ganzzahlen *Mögliches Beispiel*: Informatikerbaum als *Suchbaum* In einem binären *Suchbaum* werden die Namen und die Geburtsdaten von Informatikern lexikographisch geordnet abgespeichert.

Struktureigenschaften **(d)** Erarbeitung der Klasse `BinarySearchTree` und Einführung des Interface `Item` zur Realisierung einer geeigneten Ordnungsrelation **(e)** Implementierung der Anwendung oder von Teilen der Anwendung inklusive einer sortierten Ausgabe des Baums	nutzungsoberflächen zur Kommunikation mit einem Informatiksystem (M), • implementieren Operationen dynamischer (linearer oder nichtlinearer) Datenstrukturen (I), • implementieren und erläutern iterative und rekursive Such- und Sortierverfahren unterschiedlicher Komplexitätsklassen (Speicherbedarf und Laufzeitverhalten) (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 und mithilfe von Testanwendungen(I), • wenden didaktisch orientierte Entwicklungsumgebungen zur Demonstration, zum Entwurf, zur Implementierung und zum Test von Informatiksystemen an (I), • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D), • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D), • nutzen bereitgestellte Informatiksysteme und das Internet reflektiert zur Erschließung, Aufbereitung und Präsentation fachlicher Inhalte (D),	
4. Übung und Vertiefungen der Verwendung von Binärbäumen oder binären Suchbäumen anhand weiterer Problemstellungen		*Mögliche Beispiel:* Codierungsbäume oder Huffman-Codierung *mögliches Beispiel*: Buchindex

Unterrichtsvorhaben Q1-V:

Thema: Modellierung, Implementierung und Nutzung von relationalen Datenbanken in Anwendungskontexten
Zeitbedarf: 30 Stunden
Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Nutzung von relationalen Datenbanken **(a)** 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 **(b)** SQL-Abfragen • Analyse vorgegebener SQL-Abfragen und Erarbeitung der Sprachelemente von SQL (`SELECT (DISTINCT) … FROM, WHERE, AND, OR, NOT`) auf einer Tabelle • Analyse und Erarbeitung von SQL-Abfragen auf einer und mehrerer Tabelle zur Beantwortung der Fragestellungen (`JOIN, UNION, AS, GROUP BY,ORDER BY, ASC, DESC, COUNT, MAX, MIN, SUM`, Arithmetische Operatoren: `+, -, *, /, (…)`, Vergleichsoperatoren: `=, <>, >, <, >=, <=, LIKE, BETWEEN, IN, IS NULL`) **(c)** Vertiefung an einem weiteren Daten-	Die Schülerinnen und Schüler • erläutern die Eigenschaften und den Aufbau von Datenbanksystemen unter dem Aspekt der sicheren Nutzung (A), • analysieren und erläutern die Syntax und Semantik einer Datenbankabfrage (A), • analysieren und erläutern eine Datenbankmodellierung (A), • erläutern die Eigenschaften normalisierter Datenbankschemata (A), • bestimmen Primär- und Sekundärschlüssel (M), • ermitteln für anwendungsbezogene Problemstellungen Entitäten, zugehörige Attribute, Relationen und Kardinalitäten (M), • modifizieren eine Datenbankmodellierung (M), • modellieren zu einem Entity-Relationship-Diagramm ein relationales Datenbankschema (M), • bestimmen Primär- und Sekundärschlüssel (M), • implementieren ein relationales Daten-	*Beispiel*: Fehlstundendatenbank *Beispiel*: Präsidentendatenbank

bankbeispiel	bankschema als Datenbank (I), • überführen Datenbankschemata in vorgegebene Normalformen (M), • verwenden die Syntax und Semantik einer Datenbankabfragesprache, um Informationen aus einen Datenbanksystem zu extrahieren (I), • ermitteln Ergebnisse von Datenbankabfragen über mehrere verknüpfte Tabellen (D), • stellen Entitäten mit ihren Attributen und die Beziehungen zwischen Entitäten in einem Entity-Relationship-Diagramm grafisch dar (D), • überprüfen Datenbankschemata auf vorgegebene Normalisierungseigenschaften (D). • 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),	
2. Modellierung von relationalen Datenbanken **(a)** 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 **(b)** Entwicklung einer Datenbank aus einem Datenbankentwurf • Modellierung eines relationalen Datenbankschematas zu einem Entity-Relationship-Diagramm inklusive der Bestimmung von Primär- und Sekundärschlüsseln **(c)** 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)		*Beispiel: Schulverwaltung* In einer Software werden die Schulhalbjahre, Jahrgangsstufen, Kurse, Klassen, Schüler, Lehrer und Noten einer Schule verwaltet. Man kann dann ablesen, dass z.B. Schüler X von Lehrer Y im 2. Halbjahr des Schuljahrs 2011/2012 in der Jahrgangsstufe 9 im Differenzierungsbereich im Fach Informatik die Note „sehr gut“ erhalten hat. Dazu ist die Datenbank zu modellieren, ggf. zu normalisieren und im Datenbanksystem umzusetzen. Weiter sollen sinnvolle Abfragen entwickelt werden und das Thema Datenschutz besprochen werden.
1. Fallbeispiele zum Datenschutz	• nutzen bereitgestellte Informatiksysteme und das Internet reflektiert zum Erschließen, zur Aufbereitung und Präsentation fachlicher Inhalte (D).	Auszug aus dem Bundesdatenschutzgesetz Volkszählungsurteil passende Fallbeispiele

Unterrichtsvorhaben Q1-VI:

Thema: Informatik, Mensch und Gesellschaft - Sicherheit und Datenschutz in Informatiksystemen, Auswirkungen und Grenzen der Automatisierung
Zeitbedarf: 10 Stunden
Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien oder Materialien
1. Einführung in die Problemfelder Datenschutz, Urheberrecht und moralische Verantwortung **(a)** Vorstellung eines komplexen Fallbeispiels **(b)** Erarbeitung von Interesse verschiedener Interessensgruppen im Hinblick auf die Problemfelder **(c)** Erster Bewertungsversuch auf Grundlage des Vorwissens von Schülerinnen und Schülern	Die Schülerinnen und Schüler • untersuchen und bewerten anhand von Fallbeispielen Auswirkungen des Einsatzes von Informatiksystemen sowie Aspekte der Sicherheit von Informatiksystemen, des Datenschutzes und des Urheberrechts (A). • untersuchen und bewerten Problemlagen, die sich aus dem Einsatz von Informatiksystemen ergeben hinsichtlich rechtlicher Vorgaben, ethischer Aspekte und gesellschaftlicher Werte unter Berücksichtigung unterschiedlicher Interessenlagen (A). • untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen (A). • nutzen bereitgestellte Informatiksysteme und das Internet reflektiert zur Erschließung, Aufbereitung und Präsentation fachlicher Inhalte (D).	*Mögliches Fallbeispiel: autonomes Fahren* Ein namhaftes Softwareunternehmen möchte den Automobilmarkt mit selbstfahrenden Autos revolutionieren. Der Quellcode zu Steuerung der Fahrzeuge ist Firmengeheimnis, zur Verbesserung der Fahrleistung werden Bewegungsprofile erstellt und noch ist nicht klar, wie das Fahrzeug bei drohenden Unfällen mit Personenschaden reagieren soll. Eine erfolgreiche Einführung der Technologie würde den Straßenverkehr sicherer, schneller und ökologischer machen, allerdings auch zu tausenden von Arbeitslosen durch Wegfall ganzer Berufszweige führen.
2. Erarbeitung grundlegender Positionen (ggf. in zieldifferenten Gruppen) und Anwendung auf Fallbeispiele **(a)** Erarbeitung von Grundideen des Datenschutzes (z.B. personenbezogene Daten, informationelle Selbstbestimmung, Datensparsamkeit usw.) **(b)** Erarbeitung von Grundideen des Urheberrechts anhand eines verbreiteten Lizenzsystems **(c)** Erarbeitung eines einfachen moralischen Bewertungsmaßstabes **(d)** Anwendung auf Fallbeispiele		

Unterrichtsvorhaben Q1-VII:

Thema: Projektorientierte Softwareentwicklung am Beispiel einer Anwendung mit Datenbankanbindung
Zeitbedarf: 15 Stunden
Sequenzierung des Unterrichtsvorhabens

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien oder Materialien
1. Analyse einer lebensweltnahen Problemstellung im Hinblick auf die Entwicklung eines Java-Programms mit Datenbankanbindung **(a)** Entwicklung einer Programmidee **(b)** Analyse des Problembereichs **(c)** Entwicklung eines Anforderungskatalogs für das zu entwickelnde Programm	Die Schülerinnen und Schüler • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M), • modellieren Klassen mit ihren Attributen, Methoden und Assoziationsbeziehungen unter Angabe von Multiplizitäten (M), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen oder 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	Hier ist je nach Interessenlage des Kurses ein geeignetes Beispiel abzusprechen. Die folgenden Beispiele sind exemplarisch aufgeführt: *Quizspiel* Verwaltung von Quizfragen, Antworten und Ranglisten *Verwaltung der Schülerbücherei* Verwaltungsprogramm für das Einpflegen und Ausleihe von Büchern der Schülerbibliothek *Fehlstundenverwaltung* Verwaltungsprogramm für die Fehlstunden von Schülerinnen und Schülern *Sportfestverwaltung* Verwaltung von Aufgaben, Sportereignissen und Ergebnissen des Schulsportfestes *Materialverwaltung* Materialien für Vertretungsstunden sollen

	dokumentierter Klassenbibliotheken (I),	verwaltet werden.
2. Modellierung einer datenbankgestützten Problemlösung unter Berücksichtigung des MVC-Prinzips **(a)** Strukturierung nach dem MVC-Prinzip **(b)** Modellierung der Datenbank (ER-Diagramm, Datenbankschema) **(c)** Modellierung der Kontrollklassen und Entwicklung von SQL-Anweisungen entsprechend des Anforderungskatalogs **(d)** Modellierung einer grafischen Benutzungsoberfläche	• 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), • modellieren zu einem Entity-Relationship-Diagramm ein relationales Datenbankschema (M), • bestimmen Primär- und Sekundärschlüssel (M),	
3. Umsetzung des Softwareprojektes (a) Umsetzung der Datenbank in einem Datenbankmanagementsystem (b) Implementierung der Kontrollklassen mit Anbindung an die Datenbank unter Verwendung didaktischer Klassen (c) Integration in den Prototypen der Benutzeroberfläche	• implementieren ein relationales Datenbankschema als Datenbank (I), • 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), • analysieren und erläutern Algorithmen und Programme (A), • modifizieren Algorithmen und Programme (I), • testen Programme systematisch anhand von Beispielen und mit Hilfe von Testanwendungen (I), • ermitteln Ergebnisse von Datenbankabfragen über mehrere verknüpfte Tabellen	

	(D), • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A), • interpretieren Fehlermeldungen und korrigieren den Quellcode (I), • analysieren und erläutern die Syntax und Semantik einer Datenbankabfrage (A), • verwenden die Syntax und Semantik einer Datenbankabfragesprache, um Informationen aus einem Datenbanksystem zu extrahieren (I), • nutzen das verfügbare Informatiksystem zur strukturierten Verwaltung von Daten, zur Organisation von Arbeitsabläufen sowie zur Verteilung und Zusammenführung von Arbeitsanteilen (K), • wenden didaktisch orientierte Entwicklungsumgebungen zur Demonstration, zum Entwurf, zur Implementierung und zum Test von Informatiksystemen an (I), • entwickeln mit didaktisch orientierten Entwicklungsumgebungen einfache Benutzungsoberflächen zur Kommunikation mit einem Informatiksystem (M), • erläutern Eigenschaften und Aufbau von Datenbanksystemen unter dem Aspekt der sicheren Nutzung (A), • untersuchen und bewerten anhand von	

	Fallbeispielen Auswirkungen des Einsatzes von Informatiksystemen sowie Aspekte der Sicherheit von Informatiksystemen, des Datenschutzes und des Urheberrechts (A).	

Unterrichtsvorhaben Q2-I:

Thema: Modellierung und Implementierung von Graphen und Implementierung von Algorithmen auf Graphen in Anwendungssituationen
Zeitbedarf: 25 Stunden
Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien oder Materialien
1. Analyse von Graphen in verschiedenen Kontexten **(a)** Grundlegende Begriffe (Graph, gerichtet – ungerichtet, Knoten, Kanten, Kantengewicht) **(b)** Aufbau und Darstellung von Graphen anhand von Graphenstrukturen in verschiedenen Kontexten (Adjazenzmatrix, Adjazenzliste)	Die Schülerinnen und Schüler • erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (A), • analysieren und erläutern Algorithmen und Programme (A), • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A),	*Beispiel Soziale Netzwerke* Anhand dieses Beispiels werden die Darstellungsformen eines Graphen als Adjazenzmatrix oder mit Adjazenzlisten eingeführt. Das Beispiel wird in beiden Darstellungsformen implementiert und einfache Algorithmen werden umgesetzt.
2. Die Datenstruktur Graph im Anwendungskontext unter Nutzung der Klassen Graph, Vertex und Edge. **(a)** Erarbeitung der Klassen Graph, Vertex und Edge und beispielhafte Anwendung der Operationen **(b)** Bestimmung von Wegen in Graphen im Anwendungskontext (Tiefensuche, Breitensuche) **(c)** Bestimmung von kürzesten Wegen in Graphen im Anwendungskontext (Backtracking, Dijkstra). **(d)** Bestimmung von minimalen Spannbäumen eines Graphen im Anwendungskontext mithilfe des Prim- oder des Kruskal-Algorithmus	• beurteilen die Effizienz von Algorithmen unter Berücksichtigung des Speicherbedarfs und der Zahl der Operationen (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 oder lineare und nichtlineare Datensammlungen zu (M), • modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M), • verwenden bei der Modellierung geeig-	Beispiel: Soziale Netzwerke Ausgehend von dem Problem der Berechnung der Dichte eines sozialen Netzwerkes werden die Funktionalitäten der Methoden der Klassen Graph, Vertex und Edge erarbeitet und erste Beispiele modelliert und implementiert: • Konstruktion eines Beispielgraphen • Anzahl der Knoten • Summe der Kantengewichte • Anzahl der Nachbarn eines Knotens Beispiel: Navigationsgraph Ausgehend von dem Problem der Suche eines beliebigen Weges zwischen zwei Knoten in einem Graphen werden die Traversierungsalgorithmen (Breiten- und Tiefensuche)

	neter Problemstellungen die Möglichkeiten der Polymorphie (M), • entwickeln iterative und rekursive Algorithmen unter Nutzung der Konstruktionsstrategien „Modularisierung“ und „Teilen und Herrschen“ und „Backtraing“(M), • entwickeln mit didaktisch orientierten Entwicklungsumgebungen einfache Benutzungsoberflächen zur Kommunikation mit einem Informatiksystem (M), • implementieren Operationen dynamischer (linearer oder nichtlinearer) Datenstrukturen (I), • implementieren und erläutern iterative und rekursive Such- und Sortierverfahren unterschiedlicher Komplexitätsklassen (Speicherbedarf und Laufzeitverhalten) (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 und mithilfe von Testanwendungen(I), • wenden didaktisch orientierte Entwicklungsumgebungen zur Demonstration, zum Entwurf, zur Implementierung und zum Test von Informatiksystemen an (I), • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D),	erarbeitet. Ausgehend von der Tiefensuche in der Variante des Backtracking wird ein Algorithmus zur Bestimmung eines kürzesten Weges zwischen zwei Knoten implementiert. Anschließend wird der Dijkstra-Algorithmus auf mehrere Beispiele angewendet und implementiert. Der Vergleich der beiden Algorithmen unter Effizienzaspekten ist Bestandteil des Unterrichts. Beispiel: Versorgungsnetz Die Problemstellung, Verbraucher eines Dorfes möglichst kostengünstig an ein Versorgungsnetz (Kabel, Gas, Telefon) anzuschließen, motiviert die Behandlung des minimalen Spannbaumes eines Graphen. Mindestens einer der beiden Algorithmen wird von der Lerngruppe erarbeitet und implementiert. In leistungsstarken Lerngruppen können auch beide Algorithmen behandelt werden. So kann deutlich werden, dass es u.U. zu einem Graphen mehrere minimale Spannbäume gibt.

	• stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D), • nutzen bereitgestellte Informatiksysteme und das Internet reflektiert zur Erschließung, Aufbereitung und Präsentation fachlicher Inhalte (D),	

Unterrichtsvorhaben Q2-II:

Thema: Endliche Automaten und formale Sprachen sowie die Entwicklung eines Parsers zu einer formalen Sprache
Zeitbedarf: 30 Stunden
Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	**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 und Kellerautomaten einschließlich ihres Verhaltens auf bestimmte Eingaben (A), • analysieren und erläutern Grammatiken regulärer und kontextfreier Sprachen (A), • erläutern die Grenzen endlicher Automaten und regulärer Sprachen im Anwendungszusammenhang (A), • ermitteln die formale Sprache, die durch eine Grammatik erzeugt wird (A), • entwickeln und modifizieren zu einer Problemstellung endliche Automaten oder Kellerautomaten (M),	*Mögliche Beispiele:* Cola-Automat, Videorekorder, Akzeptor für bestimmte Zahlen, Akzeptor für Teilwörter in längeren Zeichenketten, Akzeptor für Terme
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 wer-		*Mögliche Beispiele:* reguläre Grammatik für Wörter mit ungerader Parität, Grammatik für Wörter, die bestimmte Zahlen repräsentieren, Satzgliederungsgrammatik

den **(d)** Entwicklung regulärer Grammatiken zu endlichen Automaten. **(e)** Entwicklung eines Parsers für eine einfache reguläre Sprache.	• entwickeln zur akzeptierten Sprache eines Automaten die zugehörige Grammatik (M), • entwickeln zur Grammatik einer regulären oder kontextfreien Sprache einen zugehörigen endlichen Automaten oder einen Kellerautomaten (M), • modifizieren Grammatiken regulärer und kontextfreier Sprachen (M), • entwickeln zu einer regulären oder kontextfreien Sprache eine Grammatik, die die Sprache erzeugt (M), • stellen endliche Automaten in Tabellen oder Graphen dar und überführen sie in die jeweils andere Darstellungsform (D), • ermitteln die Sprache, die ein endlicher Automat oder ein Kellerautomat akzeptiert (D), • beschreiben an Beispielen den Zusammenhang zwischen Automaten und Grammatiken (D), • entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“, „Teilen und Herrschen“ und „Backtracking“ (M).	
3. Grenzen endlicher Automaten und Grenzen der Automatisierbarkeit **a)** Vorstellung des Halteproblems **b)** Unlösbarkeit des Halteproblems		*Beispiele:* Klammerausdrücke, a^nb^n im Vergleich zu $(ab)^n$ Halteproblem
4. Entwicklung eines Kellerautomaten als Antwort auf die Grenzen endlicher Automaten (a) Erweiterung eines DEA um eine einzelne Speichervariable zum Zählen von Eingabezeichen (z.B. Klammern) und Problematisierung dieses Ansatzes (b) Entwicklung eines Automaten mit Kellerspeicher (c) Anwendung eines Kellerautomaten zur Syntaxüberprüfung auf Grundlage von nicht-regulären Grammatiken (d) Implementierung eines Kellerautomaten zur Syntaxüberprüfung (Backtracking)		

Unterrichtsvorhaben Q2-III:

Thema: von-Neumann-Architektur und Ausführung maschinennaher Programme.
Zeitbedarf: 10 Stunden
Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	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 Befehle und ihre Repräsentation in einem Binär-Code, der in einem Register gespeichert werden kann c) Analyse und Erläuterung der Funktionsweise eines einfachen maschinennahen Programms d) Übersetzung der bekannten Kontrollstrukturen Verzweigung und Schleife in die Maschinennahe Programmierung	Die Schülerinnen und Schüler • erläutern die Ausführung eines einfachen maschinennahen Programms sowie die Datenspeicherung auf einer „Von-Neumann-Architektur“ (A),	*Material*: Simulationsprogramm Johnny *Beispiel*: Addition zweier Zahlen, Multiplikation durch Addition abbilden

Unterrichtsvorhaben Q2-IV:

Thema: Grundlagen von Netzwerken und Implementierung von protokollbasierten Client-Server-Systemen in Anwendungskontexten
Zeitbedarf: 20 Stunden
Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien oder Materialien
1. Grundlagen den Datenübertragung in Netzwerken	Die Schülerinnen und Schüler	Alle Inhalte werden an der Simualtionssoftware FILIUS erarbeitet

(a) Schichtenmodell: Erarbeitung eines standardisierten Schichtenmodells für Netzwerkkommunikation **(b)** Topologien: Erarbeitung und Vergleich ausgewählter Netzwerktopologien **(c)** Subnetze und Routing: Analyse von Grundlagen der Adressierung in IP-Netzwerken **(d)** Protokolle: Erarbeitung des beispielhaften Aufbaus eines Protokoll auf der Anwendungsschicht	• beschreiben und erläutern Netzwerk-Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken (A), • analysieren und erläutern Algorithmen und Programme (A), • analysieren und erläutern Protokolle zur Kommunikation in einem Client-Server-Netzwerk (A), • entwickeln und erweitern Protokolle zur Kommunikation in einem Client-Server-Netzwerk (M),	
2. Analyse, Modellierung und Implementierung von Netzwerkanwendungen in Client-Server-Struktur **(a)** (a) Nutzung einfacher Server-Dienste mittels der Klassen Connection und Client • Modellierung und Implementierung von Clients für einfach Serverdienste **(b)** Anbieten von Diensten mittels der Klasse Server • Analyse vorgegebener Implementationen einfacher Server • Modellierung und Implementierung eigener Server **(c)** Entwicklung eines vollständigen Client-Server-Systems • Protokollentwurf, Dialogorientierung • Bedeutung von Nebenläufigkeit • Implementierung	• modifizieren Algorithmen und Programme (I) • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihr Operationen und Beziehungen (M), • modellieren Klassen mit ihren Attributen, Methoden und Assoziationsbeziehungen unter Angabe von Multiplizitäten (M), • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen oder lineare und nichtlineare Datensammlungen zu (M), • analysieren und erläutern objektorientierte Modellierungen (A), • stellen Klassen und ihre Beziehungen grafisch dar (D), • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I).	*Beispiel: Echo- bzw. Time-Clients und –Server sowie eigene Erweiterungen* Anhand der Echo- und Time-Dienste (z.B. lokal im Schulnetz durch den Lehrenden zur Verfügung gestellt) erarbeiten die Schülerinnen und Schüler zunächst den die Funktionsweise bzw. den Aufbau einfacher Clients und verwenden dabei zunächst die Klasse Connection, später die Klasse Client. In einem zweiten Schritt implementieren die Schülerinnen und Schüler Daytime- und Echo-Client bzw. Erweiterungen/Abwandlungen derselben (ggf. mit Steigerung des Interaktionsgrades) selbst. *Beispiel: Messenger-Dienst* Abschließend entwickeln die Schülerinnen und Schüler ein Client-Server-System zum Versenden von Nachrichten zwischen einzelnen Rechnern (einfacher Messenger), basierend auf selbst gewählten „Nicknames“.

Unterrichtsvorhaben Q2-V:

Thema: Entwicklung eines Netzwerkspiels unter Verwendung eines Vorgehensmodells aus der Softwareentwicklung
Zeitbedarf: 20 Stunden
Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien oder Materialien
1. Projekteinstieg / -planung (Spielauswahl, Zeitmanagement) **2. Projektumsetzung (informatische Analyse, Modellierung, Implementierung, Test (ggf. in Zyklen))** **(a)** Analyse des Spiels mit dem Ziel einer späteren informatischen Umsetzung als Netzwerkspiel **(b)** Modellierung und Implementierung des Spiels als Netzwerkanwendung ◦ Entwurf eines Protokolls zur Kommunikation zwischen Spielserver und -client basierend auf den erarbeiteten Spielregeln ◦ (b) Entwicklung von Entwurfs- und Implementationsdiagrammen für den Spielserver ◦ (c) Implementation und Test der Spielserver-Klasse und von dieser benötigter Fachklassen ◦ (d) Modellierung, Implementierung und Test des Spielclients ◦ (e) Test der Zusammenarbeit von Spielserver und Spielclient **(c)** Reflexion des Softwareprodukts	Die Schülerinnen und Schüler • 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), • 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), • ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M), • stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D), • analysieren und erläutern objektorientierte Modellierungen (A), • implementieren Klassen in einer Programmiersprache auch unter Nutzung	Das umzusetzende Spiel wird mit dem Kurs abgesprochen.

3. Projektreflexion (Reflexion der Projektplanung, Präsentation)	dokumentierter Klassenbibliotheken (I), • analysieren und erläutern Algorithmen und Programme (A), • modifizieren Algorithmen und Programme (I), • entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“ und „Teilen und Herrschen“ und „Backtracking“ (M), • implementieren iterative und rekursive Al gorithmen auch unter Verwendung von dynamischen Datenstrukturen (I), • testen Programme systematisch anhand von Beispielen und mit Hilfe von Testanwendungen (I). • analysieren und erläutern Algorithmen und Methoden zur Client-Server-Kommunikation (A), • entwickeln und implementieren Algorithmen und Methoden zur Client-Server-Kommunikation (I). • beschreiben und erläutern Netzwerk-Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken (A), • analysieren und erläutern Protokolle zur Kommunikation in einem Client-Server-Netzwerk (A), • entwickeln und erweitern Protokolle zur Kommunikation in einem Client-Server-Netzwerk (M).	

Unterrichtsvorhaben Q2-VI:

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