

## Schulinterner Lehrplan Q2 Informatik

<b>Qualifikationsphase 2</b>	
<p><u>Unterrichtsvorhaben Q2-I</u></p> <p><b>Thema:</b></p> <p><i>Modellierung und Implementierung von Anwendungen mit dynamischen, nichtlinearen Datenstrukturen</i></p> <p><b>Zentrale Kompetenzen:</b></p> <ul style="list-style-type: none"> <li>• Argumentieren</li> <li>• Modellieren</li> <li>• Implementieren</li> <li>• Darstellen und Interpretieren</li> <li>• Kommunizieren und Kooperieren</li> </ul> <p><b>Inhaltsfelder:</b></p> <ul style="list-style-type: none"> <li>• Daten und ihre Strukturierung</li> <li>• Algorithmen</li> <li>• Formale Sprachen und Automaten</li> </ul> <p><b>Inhaltliche Schwerpunkte:</b></p> <ul style="list-style-type: none"> <li>• Objekte und Klassen</li> <li>• Analyse, Entwurf und Implementierung von Algorithmen</li> <li>• Algorithmen in ausgewählten informatischen Kontexten</li> <li>• Syntax und Semantik einer Programmiersprache</li> </ul> <p><b>Zeitbedarf:</b> 24 Stunden</p>	<p><u>Unterrichtsvorhaben Q2-II</u></p> <p><b>Thema:</b></p> <p><i>Endliche Automaten und formale Sprachen</i></p> <p><b>Zentrale Kompetenzen:</b></p> <ul style="list-style-type: none"> <li>• Argumentieren</li> <li>• Modellieren</li> <li>• Darstellen und Interpretieren</li> <li>• Kommunizieren und Kooperieren</li> </ul> <p><b>Inhaltsfelder:</b></p> <ul style="list-style-type: none"> <li>• Endliche Automaten und formale Sprachen</li> </ul> <p><b>Inhaltliche Schwerpunkte:</b></p> <ul style="list-style-type: none"> <li>• Endliche Automaten</li> <li>• Grammatiken regulärer Sprachen</li> <li>• Möglichkeiten und Grenzen von Automaten und formalen Sprachen</li> </ul> <p><b>Zeitbedarf:</b> 20 Stunden</p>

## **Qualifikationsphase 2**

### Unterrichtsvorhaben Q2-III

#### **Thema:**

*Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit*

#### **Zentrale Kompetenzen:**

- Argumentieren
- Kommunizieren und Kooperieren

#### **Inhaltsfelder:**

- Informatiksysteme
- Informatik, Mensch und Gesellschaft

#### **Inhaltliche Schwerpunkte:**

- Einzelrechner und Rechnernetzwerke
- Grenzen der Automatisierung

**Zeitbedarf:** 12 Stunden

**Summe Qualifikationsphase 2: 56 Stunden**

# Modellierung und Implementierung von Anwendungen mit dynamischen, nichtlinearen Datenstrukturen (Unterrichtsvorhaben Q2-I)

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

## Vorhabenbezogene Konkretisierung:

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

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

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

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

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

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

**Zeitbedarf:** 24 Stunden

## Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<b>1. Analyse von Baumstrukturen in verschiedenen Kontexten</b>	Die Schülerinnen und Schüler <ul style="list-style-type: none"><li>erläutern Operationen dynamischer (linearer</li></ul>	<i>Beispiel:</i> Termbaum Der Aufbau von Termen wird mit Hilfe von binären

<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>oder nicht-linearer) Datenstrukturen (A),</p> <ul style="list-style-type: none"> <li>• analysieren und erläutern Algorithmen und Programme (A),</li> <li>• beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A),</li> <li>• ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M),</li> <li>• ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M),</li> <li>• modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M),</li> <li>• verwenden bei der Modellierung geeigneter Problemstellungen die Möglichkeiten der Polymorphie (M),</li> <li>• entwickeln iterative und rekursive Algorithmen unter Nutzung der Konstruktionsstrategien „Modularisierung“ und „Teilen und Herrschen“ (M),</li> <li>• implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I),</li> <li>• modifizieren Algorithmen und Programme (I),</li> </ul>	<p>Baumstrukturen verdeutlicht.</p> <p>oder</p> <p><i>Beispiel:</i> Ahnenbaum</p> <p>Die binäre Baumstruktur ergibt sich daraus, dass jede Person genau einen Vater und eine Mutter hat.</p> <p><i>Weitere Beispiele für Anwendungskontexte für binäre Bäume:</i></p> <p><i>Beispiel:</i> Suchbäume (zur sortierten Speicherung von Daten)</p> <p>Alle Inhalte, die nach einer Ordnung vor dem Inhalt im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Inhalt im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.)</p> <p>oder</p> <p><i>Beispiel:</i> Entscheidungsbäume</p> <p>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.</p> <p>oder</p>
---	---	---

	<ul style="list-style-type: none"> <li>• nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I),</li> <li>• interpretieren Fehlermeldungen und korrigieren den Quellcode (I),</li> <li>• testen Programme systematisch anhand von Beispielen (I),</li> <li>• stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D),</li> <li>• stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D).</li> </ul>	<p><i>Beispiel:</i> Codierungsbäume für Codierungen, deren Alphabet aus genau zwei Zeichen besteht</p> <p>Morse hat Buchstaben als Folge von Punkten und Strichen codiert. Diese Codierungen können in einem Binärbaum dargestellt werden, so dass ein Übergang zum linken Teilbaum einem Punkt und ein Übergang zum rechten Teilbaum einem Strich entspricht. Wenn man im Gesamtbaum startet und durch Übergänge zu linken oder rechten Teilbäumen einen Pfad zum gewünschten Buchstaben sucht, erhält man die Morsecodierung des Buchstabens.</p> <p><i>Materialien:</i></p> <p>Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 – Binärbaum</p>
<p><b>2. Die Datenstruktur Binärbaum im Anwendungskontext unter Nutzung der Klasse BinaryTree</b></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><i>Beispiel:</i> Informatikerbaum als binärer Baum</p> <p>In einem <i>binären Baum</i> werden die Namen und die Geburtsdaten von Informatikern lexikographisch geordnet abgespeichert. Alle Namen, die nach dieser Ordnung vor dem Namen im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Namen im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.)</p>

<p>(c) Erarbeitung der Klasse BinaryTree 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>Folgende Funktionalitäten werden benötigt:</p> <ul style="list-style-type: none"> <li>• Einfügen der Informatiker-Daten in den Baum</li> <li>• Suchen nach einem Informatiker über den Schlüssel Name</li> <li>• Ausgabe des kompletten Datenbestands in nach Namen sortierter Reihenfolge</li> </ul> <p><i>Materialien:</i></p> <p>Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 – Binärbaum</p>
<p><b>3. Die Datenstruktur binärer Suchbaum im Anwendungskontext unter Verwendung der Klasse BinarySearchTree</b></p> <p>(a) Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen</p> <p>(b) Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramm, grafische Darstellung eines binären Suchbaums und Erarbeitung der Struktureigenschaften</p> <p>(c) Erarbeitung der Klasse BinarySearchTree und Einführung des Interface Item zur Realisierung einer geeigneten Ordnungsrelation</p>		<p><i>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. Alle Namen, die nach dieser Ordnung vor dem Namen im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Namen im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.)</p> <p>Folgende Funktionalitäten werden benötigt:</p> <ul style="list-style-type: none"> <li>• Einfügen der Informatiker-Daten in den Baum</li> </ul> <p>Suchen nach einem Informatiker über den Schlüssel Name</p>

<p>(d) Implementierung der Anwendung oder von Teilen der Anwendung inklusive einer sortierten Ausgabe des Baums</p>		<p>Ausgabe des kompletten Datenbestands in nach Namen sortierter Reihenfolge</p> <p><i>Materialien:</i></p> <p>Ergänzungsmaterialien zum Lehrplannavigator  Unterrichtsvorhaben Q2.1 – Binärer Suchbaum</p>
<p><b>4. Übung und Vertiefungen der Verwendung von Binärbäumen oder binären Suchbäumen anhand weiterer Problemstellungen</b></p>		<p><i>Beispiel:</i> Buchindex</p> <p>Es soll eine Anwendung entwickelt werden, die anhand von Stichworten und zugehörigen Seitenzahlen ein Stichwortregister erstellt.</p> <p>Da die Stichwörter bei der Analyse des Buches häufig gesucht werden müssen, werden sie in der Klasse Buchindex als Suchbaum (Objekt der Klasse BinarySearchTree) verwaltet.</p> <p>Alle Inhalte, die nach einer Ordnung vor dem Inhalt im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Inhalt im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.)</p> <p><i>oder</i></p> <p><i>Beispiel:</i> Entscheidungsbäume (s.o.)</p> <p><i>oder</i></p> <p><i>Beispiel:</i> Termbaum (s.o.)</p> <p><i>oder</i></p> <p><i>Beispiel:</i> Ahnenbaum (s.o.)</p>

		<p><i>Materialien:</i></p> <p>Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.1 – Anwendung Binärbaum</p>
--	--	--

# Endliche Automaten und formale Sprachen (Unterrichtsvorhaben Q2-II)

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

## Vorhabenbezogene Konkretisierung:

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

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

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

**Zeitbedarf:** 20 Stunden

## Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien oder Materialien
<p><b>1. Endliche Automaten</b></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"> <li>analysieren und erläutern die Eigenschaften endlicher Automaten einschließlich ihres Verhaltens auf bestimmte Eingaben (A),</li> <li>analysieren und erläutern Grammatiken regulärer Sprachen (A),</li> <li>zeigen die Grenzen endlicher Automaten und regulärer Grammatiken im Anwendungszusammenhang auf (A),</li> </ul>	<p><i>Beispiele:</i></p> <p>Cola-Automat, Geldspielautomat, Roboter, Zustandsänderung eines Objekts „Auto“, Akzeptor für bestimmte Zahlen, Akzeptor für Teilwörter in längeren Zeichenketten, Akzeptor für Terme</p> <p><i>Materialien:</i></p> <p>Ergänzungsmaterialien zum Lehrplannavigator</p>

	<ul style="list-style-type: none"> <li>• ermitteln die formale Sprache, die durch eine Grammatik erzeugt wird (A),</li> <li>• entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M),</li> <li>• entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M),</li> <li>• entwickeln zur akzeptierten Sprache eines Automaten die zugehörige Grammatik (M),</li> <li>• entwickeln zur Grammatik einer regulären Sprache einen zugehörigen endlichen Automaten (M),</li> <li>• modifizieren Grammatiken regulärer Sprachen (M),</li> <li>• entwickeln zu einer regulären Sprache eine Grammatik, die die Sprache erzeugt (M),</li> <li>• stellen endliche Automaten in Tabellen oder Graphen dar und überführen sie in die jeweils andere Darstellungsform (D),</li> <li>• ermitteln die Sprache, die ein endlicher Automat akzeptiert (D).</li> <li>• beschreiben an Beispielen den Zusammenhang zwischen Automaten und Grammatiken (D).</li> </ul>	Unterrichtsvorhaben Q2.2 – Endliche Automaten, Formale Sprachen (Download Q2-II.1)
<p><b>2. Untersuchung und Entwicklung von Grammatiken regulärer Sprachen</b></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>		<p><i>Beispiele:</i></p> <p>reguläre Grammatik für Wörter mit ungerader Parität, Grammatik für Wörter, die bestimmte Zahlen repräsentieren, Satzgliederungsgrammatik</p>
<p><b>3. Grenzen endlicher Automaten</b></p>		<p><i>Beispiele:</i></p> <p>Klammerausdrücke, anbn im Vergleich zu (ab)<sup>n</sup></p>

# Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit (Unterrichtsvorhaben Q2-III)

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

## Vorhabenbezogene Konkretisierung:

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

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

**Zeitbedarf:** 12 Stunden

## Sequenzierung des Unterrichtsvorhabens:

Unterrichtssequenzen	Zu entwickelnde Kompetenzen	Beispiele, Medien oder Materialien
<p><b>1. Von-Neumann-Architektur und die Ausführung maschinennaher Programme</b></p> <p>a) prinzipieller Aufbau einer von Neumann-Architektur mit CPU, Rechenwerk, Steuerwerk, Register und Hauptspeicher</p> <p>b) einige maschinennahe Befehle und ihre Repräsentation in einem Binär-Code, der in einem Register gespeichert werden kann</p> <p>c) Analyse und Erläuterung der Funktionsweise eines einfachen maschinennahen Programms</p>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> <li>erläutern die Ausführung eines einfachen maschinennahen Programms sowie die Datenspeicherung auf einer „Von-Neumann-Architektur“ (A),</li> <li>untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen (A).</li> </ul>	<p><i>Beispiel:</i></p> <p>Addition von 4 zu einer eingegebenen Zahl mit einem Rechnermodell</p> <p><i>Materialien:</i></p> <p>Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q2.3 – Von-Neumann-Architektur und maschinennahe Programmierung</p>

<p><b>2. Grenzen der Automatisierbarkeit</b></p> <p>a) Vorstellung des Halteproblems</p> <p>b) Unlösbarkeit des Halteproblems</p> <p>c) Beurteilung des Einsatzes von Informatiksystemen hinsichtlich prinzipieller Möglichkeiten und prinzipieller Grenzen</p>		<p><i>Beispiel:</i> Halteproblem</p> <p><i>Materialien:</i></p> <p>Ergänzungsmaterialien zum Lehrplannavigator  Unterrichtsvorhaben Q2.3 - Halteproblem</p>
---	--	---