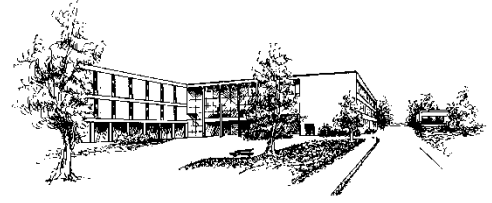


**Helmholtz-Gymnasium Bonn**



**schulinterner Lehrplan  
Sekundarstufe II**

# **Informatik**

**(Fassung vom 24.04.2015)**

## **Inhaltsverzeichnis**

<b>1</b>	<b>DIE FACHGRUPPE INFORMATIK DES HELMHOLTZ-GYMNASIUMS .....</b>	<b>3</b>
<b>2</b>	<b>ENTSCHEIDUNGEN ZUM UNTERRICHT .....</b>	<b>4</b>
<b>2.1</b>	<b>UNTERRICHTSVORHABEN .....</b>	<b>5</b>
<b>2.1.1</b>	<b>ÜBERSICHT ÜBER DIE UNTERRICHTSVORHABEN .....</b>	<b>5</b>
I)	EINFÜHRUNGSPHASE – GRUNDKURS .....	5
II)	QUALIFIKATIONSPHASE – GRUNDKURS/LEISTUNGSKURS .....	7
<b>2.1.2</b>	<b>KONKRETISIERTE UNTERRICHTSVORHABEN .....</b>	<b>10</b>
I)	EINFÜHRUNGSPHASE – GRUNDKURS .....	10
II)/III)	QUALIFIKATIONSPHASE – GRUNDKURS/LEISTUNGSKURS .....	18
<b>2.2</b>	<b>GRUNDSÄTZE DER LEISTUNGSBEWERTUNG UND LEISTUNGSRÜCKMELDUNG .....</b>	<b>42</b>

# 1 Die Fachgruppe Informatik des Helmholtz-Gymnasiums

Das Fach Informatik wird am Helmholtz-Gymnasium Bonn ab der Jahrgangsstufe 5 unterrichtet. In den Jahrgangsstufen 9 und 10 ist es darüber hinaus möglich, das Fach Informatik im Rahmen des Wahlpflichtbereichs zu wählen. In diesem zweijährigen Kurs werden in altersstufengerechter Weise unter anderem die Grundlagen der Algorithmik am Beispiel verschiedener didaktischer Lernumgebungen vertieft. Außerdem erhalten die Schülerinnen und Schüler dieses Differenzierungskurses einen tiefergehenden Einblick in Tabellenkalkulationen sowie die Gestaltung und Programmierung von Webseiten. In der Sekundarstufe II wird das Fach Informatik am Helmholtz-Gymnasium als Wahlfach angeboten. In der Einführungsphase können die Schülerinnen und Schüler das Fach als dreistündigen Grundkurs belegen. In der Qualifikationsphase bietet das Gymnasium nun schon mehrere Jahre nacheinander einen Grund- und einen Leistungskurs im Fach Informatik an.

Um insbesondere Schülerinnen und Schülern, die in der Sekundarstufe I nicht das Wahlpflichtfach Informatik besucht haben, gerecht zu werden, wird in den Kursen der Einführungsphase besonderer Wert darauf gelegt, dass keine Vorkenntnisse aus dem Wahlpflichtbereich zum erfolgreichen Bestehen des Kurses erforderlich sind. Insofern verfügen alle Schülerinnen und Schüler in der Oberstufe unabhängig von ihrer Fächerwahl über ein grundlegendes Wissen der Informations- und Kommunikationstechnologie, auf die im Unterricht der Sekundarstufe II aufgebaut werden kann.

Durch projektartiges Vorgehen, offene Aufgaben und Möglichkeiten, Problemlösungen zu verfeinern und zu optimieren, entspricht der Informatikunterricht in besonderem Maße den Erziehungszielen, Leistungsbereitschaft zu fördern ohne zu überfordern. Darüber hinaus trägt er zu einer breitgefächerten Allgemeinbildung bei, bietet gleichzeitig Raum für individuelle Spezialisierungen und ermöglicht verantwortungsvolles Handeln in einer sich schnell wandelnden und von technischen Fortschritten geprägten Welt.

Die Leistungsbewertung umfasst im Fach Informatik neben der mündlichen Mitarbeit im Unterricht und den schriftlichen Leistungsüberprüfungen auch die Abgabe von Aufgaben in digitaler Form sowie die Präsentation von Referaten.

Die gemeinsame Entwicklung von Materialien und Unterrichtsvorhaben, die Evaluation von Lehr- und Lernprozessen sowie die stetige Überprüfung und eventuelle Modifikation des schulinternen Curriculums durch die Fachkonferenz Informatik stellt einen wichtigen Beitrag zur Qualitätssicherung und -entwicklung des Unterrichts dar. Weiterhin kann durch die Schulausstattung im Normalfall eine Einzelplatzarbeit gewährleistet werden.

## **2 Entscheidungen zum Unterricht**

## **2.1 Unterrichtsvorhaben**

Die Darstellung der Unterrichtsvorhaben im schulinternen Lehrplan besitzt den Anspruch, sämtliche im Kernlehrplan angeführten Kompetenzen abzudecken. Dies entspricht der Verpflichtung jeder Lehrkraft, Schülerinnen und Schülern Lerngelegenheiten zu ermöglichen, so dass alle Kompetenzerwartungen des Kernlehrplans von ihnen erfüllt werden können.

In der nachfolgenden Übersicht über die Unterrichtsvorhaben wird die für alle Lehrerinnen und Lehrer gemäß Fachkonferenzbeschluss verbindliche Verteilung der Unterrichtsvorhaben dargestellt. Das Übersichtsraster dient dazu, den Kolleginnen und Kollegen einen schnellen Überblick über die Zuordnungen der Unterrichtsvorhaben zu den einzelnen Jahrgangsstufen sowie den im Kernlehrplan genannten Kompetenzen, Inhaltsfeldern und inhaltlichen Schwerpunkten zu verschaffen. Der ausgewiesene Zeitbedarf versteht sich als grobe Orientierungsgröße, die nach Bedarf über- oder unterschritten werden kann. Um Spielraum für Vertiefungen, besondere Schülerinteressen, aktuelle Themen bzw. die Erfordernisse anderer besonderer Ereignisse (z. B. Praktika, Kursfahrten o. ä.) zu erhalten, wurde im Rahmen dieses schulinternen Lehrplans nicht die volle Bruttounterrichtszeit verplant. Sicherzustellen bleibt allerdings auch hier, dass im Rahmen der Umsetzung der Unterrichtsvorhaben insgesamt alle Kompetenzerwartungen des Kernlehrplans Berücksichtigung finden.

Da in den folgenden Unterrichtsvorhaben Inhalte in der Regel anhand von Problemstellungen in Anwendungskontexten bearbeitet werden, werden in einigen Unterrichtsvorhaben jeweils mehrere Inhaltsfelder angesprochen.

### **2.1.1 Übersicht über die Unterrichtsvorhaben**

#### **I) Einführungsphase – Grundkurs**

Im Folgenden sollen Unterrichtsvorhaben für das Fach Informatik dargestellt werden. Alle hier aufgeführten Vorhaben beziehen sich auf Grundkurse in der Einführungsphase.

Zu jedem Unterrichtsvorhaben ist eine Anknüpfung an den Kernlehrplan Informatik in Form von Kompetenzbezügen gegeben. Die aufgeführten Kompetenzen sind dabei so zu verstehen, dass das entsprechende Unterrichtsvorhaben zum Erwerb derselben beiträgt. Kompetenzerwerb ist ein kontinuierlicher und kumulativer Prozess, der sich über längere Zeiträume hinzieht und die wiederholte Beschäftigung mit entsprechenden fachlichen Gegenständen und Themen in variierenden Anwendungssituationen oder auf zunehmenden Anforderungsniveaus voraussetzt. Es kann daher nicht der Anspruch erhoben werden, dass die aufgeführten Kompetenzen nach Abschluss lediglich eines Unterrichtsvorhabens vollständig erworben wurden.

<b>Einführungsphase</b>	
<p>Unterrichtsvorhaben E-I Thema: Überblick über das Fach Informatik, Grundlagen von Informatiksystem und Geschichte der Informatik</p> <p>Zentrale Kompetenzen: Argumentieren Darstellen und Interpretieren Kommunizieren und Kooperieren</p> <p>Inhaltsfelder: Informatiksysteme Informatik, Mensch und Gesellschaft</p> <p>Inhaltliche Schwerpunkte: Digitalisierung Einzelrechner Dateisysteme Internet Einsatz von Informatiksystemen Wirkungen der Automatisierung Geschichte der automatischen Datenverarbeitung</p> <p>Zeitbedarf: 6 Stunden</p>	<p>Unterrichtsvorhaben E-II Thema: Grundlagen der Programmierung mit Java und einfache Algorithmik</p> <p>Zentrale Kompetenzen: Argumentieren Modellieren Implementieren Darstellen und Interpretieren Kommunizieren und Kooperieren</p> <p>Inhaltsfelder: Daten und ihre Strukturierung Algorithmen Formale Sprachen und Automaten Informatiksysteme</p> <p>Inhaltliche Schwerpunkte: Analyse, Entwurf und Implementierung einfacher Algorithmen Syntax und Semantik einer Programmiersprache</p> <p>Zeitbedarf: 33 Stunden</p>
<p>Unterrichtsvorhaben E-III Thema: Entwurf und Implementierung einfacher Sortierverfahren und eines klassischen kryptographischen Verfahrens</p> <p>Zentrale Kompetenzen: Argumentieren Modellieren Implementieren Darstellen und Interpretieren Kommunizieren und Kooperieren</p> <p>Inhaltsfelder: Daten und ihre Strukturierung Algorithmen Formale Sprachen und Automaten</p> <p>Inhaltliche Schwerpunkte: Analyse, Entwurf und Implementierung einfacher Algorithmen Syntax und Semantik einer Programmiersprache Algorithmen zum Suchen und Sortieren</p> <p>Zeitbedarf: 12 Stunden</p>	<p>Unterrichtsvorhaben E-IV Thema: Objektorientierte Modellierung und Implementierung von Klassen- und Objektbeziehungen anhand von grafischen Spielen, Simulationen und grafischen Oberflächen</p> <p>Zentrale Kompetenzen: Argumentieren Modellieren Implementieren Darstellen und Interpretieren Kommunizieren und Kooperieren</p> <p>Inhaltsfelder: Daten und ihre Strukturierung Algorithmen Formale Sprachen und Automaten</p> <p>Inhaltliche Schwerpunkte: Objekte und Klassen Analyse, Entwurf und Implementierung einfacher Algorithmen Syntax und Semantik einer Programmiersprache</p> <p>Zeitbedarf: 36 Stunden</p>

## II) Qualifikationsphase – Grundkurs/**Leistungskurs**

<b>Qualifikationsphase I</b>	
<p>Unterrichtsvorhaben Q1-I Thema: Rekursive Algorithmen und Backtracking in Anwendungskontexten</p> <p>Zentrale Kompetenzen: Argumentieren Modellieren Implementieren Darstellen und Interpretieren Kommunizieren und Kooperieren</p> <p>Inhaltsfelder: Algorithmen Formale Sprachen und Automaten Informatiksysteme Informatik, Mensch und Gesellschaft</p> <p>Inhaltliche Schwerpunkte: Analyse, Entwurf und Implementierung von Algorithmen Algorithmen in ausgewählten informatischen Kontexten Syntax und Semantik einer Programmiersprache Nutzung von Informatiksystemen Grenzen der Automatisierung</p> <p>Zeitbedarf: 20/25 Stunden</p>	<p>Unterrichtsvorhaben Q1-II Thema: Modellierung und Implementierung dynamischer Listenstrukturen und deren Anwendungen</p> <p>Zentrale Kompetenzen: Argumentieren Modellieren Implementieren Darstellen und Interpretieren Kommunizieren und Kooperieren</p> <p>Inhaltsfelder: Daten und ihre Strukturierung Algorithmen Formale Sprachen und Automaten Informatiksysteme</p> <p>Inhaltliche Schwerpunkte: Objekte und Klassen Analyse, Entwurf und Implementierung von Algorithmen Algorithmen in ausgewählten informatischen Kontexten Syntax und Semantik einer Programmiersprache Nutzung von Informatiksystemen</p> <p>Zeitbedarf: 25/30 Stunden</p>
<p>Unterrichtsvorhaben Q1-III Thema: Modellierung und Implementierung dynamische nichtlineare Datenstrukturen am Beispiel der Binärbäume</p> <p>Zentrale Kompetenzen: Argumentieren Modellieren Implementieren Darstellen und Interpretieren Kommunizieren und Kooperieren</p> <p>Inhaltsfelder: Daten und ihre Strukturierung Algorithmen Formale Sprachen und Automaten Informatiksysteme</p> <p>Inhaltliche Schwerpunkte: Objekte und Klassen Analyse, Entwurf und Implementierung von Algorithmen</p>	<p>Unterrichtsvorhaben Q1-IV Thema: Sicherheit und Datenschutz in Netzstrukturen <b>und Modellierung und Implementierung von Client-Server-Anwendungen</b></p> <p>Zentrale Kompetenzen: Argumentieren <b>Modellieren</b> <b>Implementieren</b> Darstellen und Interpretieren Kommunizieren und Kooperieren</p> <p>Inhaltsfelder: <b>Daten und ihre Strukturierung</b> <b>Algorithmen</b> <b>Formale Sprachen und Automaten</b> Informatiksysteme Informatik, Mensch und Gesellschaft</p> <p>Inhaltliche Schwerpunkte: <b>Objekte und Klassen</b></p>

<p>Algorithmen in ausgewählten informatischen Kontexten          Syntax und Semantik einer Programmiersprache          Nutzung von Informatiksystemen</p> <p>Zeitbedarf: 20/25 Stunden</p>	<p>Analyse, Entwurf und Implementierung von Algorithmen          Algorithmen in ausgewählten informatischen Kontexten          Syntax und Semantik einer Programmiersprache          Einzelrechner und Rechnernetzwerke          Sicherheit          Nutzung von Informatiksystemen, Wirkungen der Automatisierung</p> <p>Zeitbedarf: 15/25 Stunden</p>
<b>Summe Qualifikationsphase I: 80 (105)</b>	

<b>Qualifikationsphase II</b>	
<p>Unterrichtsvorhaben Q2-I / Q1-V          Thema: Modellierung und Nutzung relationaler Datenbanken in Anwendungskontexten</p> <p>Zentrale Kompetenzen:          Argumentieren          Modellieren          Implementieren          Darstellen und Interpretieren          Kommunizieren und Kooperieren</p> <p>Inhaltsfelder:          Daten und ihre Strukturierung          Algorithmen          Formale Sprache und Automaten          Informatiksysteme          Informatik, Mensch und Gesellschaft</p> <p>Inhaltliche Schwerpunkte:          Datenbanken          Algorithmen in ausgewählten informatischen Kontexten          Syntax und Semantik einer Programmiersprache          Nutzung von Informatiksystemen          Sicherheit          Wirkung der Automatisierung</p> <p>Zeitbedarf: 25/30 Stunden</p>	<p>Unterrichtsvorhaben Q2-II          Thema: Endliche Automaten und Formale Sprachen</p> <p>Zentrale Kompetenzen:          Argumentieren          Modellieren          Implementieren          Darstellen und Interpretieren          Kommunizieren und Kooperieren</p> <p>Inhaltsfelder:          Daten und ihre Strukturierung          Algorithmen          Formale Sprachen und Automaten          Informatiksysteme</p> <p>Inhaltliche Schwerpunkte:          Endliche Automaten (und Kellerautomaten)          Grammatiken regulärer (und kontextfreier) Sprachen          Scanner, Parser und Interpreter für eine reguläre Sprache          Möglichkeiten und Grenzen von Automaten und formalen Sprachen          Nutzung von Informatiksystemen</p> <p>Zeitbedarf: 25/40 Stunden</p>
<p>Unterrichtsvorhaben Q2-III          Thema: Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit</p> <p>Zentrale Kompetenzen:          Argumentieren          Kommunizieren und Kooperieren</p> <p>Inhaltsfelder:          Informatiksysteme</p>	<p>Q2-IV          Thema: Modellierung und Implementierung dynamischer nichtlinearer Datenstrukturen am Beispiel der Graphen</p> <p>Zentrale Kompetenzen:          Argumentieren          Modellieren          Implementieren          Darstellen und Interpretieren</p>



<p>Informatik, Mensch und Gesellschaft</p> <p>Inhaltliche Schwerpunkte:  Einzelrechner und Rechnernetzwerke  Grenzen der Automatisierung</p> <p>Zeitbedarf: 10 Stunden</p>	<p>Kommunizieren und Kooperieren</p> <p>Inhaltsfelder:  Daten und ihre Strukturierung  Algorithmen  Formale Sprachen und Automaten  Informatiksysteme</p> <p>Inhaltliche Schwerpunkte:  Objekte und Klassen  Analyse, Entwurf und Implementierung von  Algorithmen  Algorithmen in ausgewählten informatischen  Kontexten</p> <p>Zeitbedarf: 15 Stunden</p>
<b>Summe Qualifikationsphase II: 60 (95)</b>	

## 2.1.2 Konkretisierte Unterrichtsvorhaben

Hinweis: Thema, Inhaltsfelder, inhaltliche Schwerpunkte, Kompetenzen und Absprachen zur vorhabenbezogenen Konkretisierung hat die Fachkonferenz verbindlich vereinbart. In allen anderen Bereichen (Unterrichtssequenzen und verwendeten Beispiele, Medien und Materialien) sind Abweichungen von den vorgeschlagenen Vorgehensweisen möglich. Darüber hinaus enthält dieser schulinterne Lehrplan in den Kapiteln 2.2 bis 2.3 übergreifende sowie z. T. auch jahrgangsbezogene Absprachen zur fachmethodischen und fachdidaktischen Arbeit, zur Leistungsbewertung und zur Leistungsrückmeldung. Je nach internem Steuerungsbedarf können solche Absprachen auch vorhabenbezogen vorgenommen werden.

### I) Einführungsphase – Grundkurs

Im Folgenden sollen die in Abschnitt 1 aufgeführten Unterrichtsvorhaben konkretisiert werden. Diese Konkretisierung hat vorschlagenden Charakter, ohne die pädagogische Freiheit des Lehrenden einschränken zu wollen.

Die übergeordneten Kompetenzen des Kompetenzbereichs "Kommunizieren und Kooperieren" werden in jedem Unterrichtsvorhaben erworben bzw. vertieft und sind daher nicht jedes Mal erneut aufgeführt.

#### Kommunizieren und Kooperieren (K)

Schülerinnen und Schüler

- verwenden Fachausdrücke bei der Kommunikation über informatische Sachverhalte,
- kommunizieren und kooperieren in Gruppen und in Partnerarbeit,
- präsentieren Arbeitsabläufe und Arbeitsergebnisse.

## **Unterrichtsvorhaben Nr. 1**

Thema: Überblick über das Fach Informatik, Grundlagen von Informatiksystem und Geschichte der Informatik

Leitfragen: Mit welchen Themen befasst sich das Fach Informatik in der Schule? Wie funktioniert ein moderner Computer? Welche Entwicklung durchlief die moderne Datenverarbeitung?

Zeitbedarf: 6 Stunden

Berufsorientierung:

- Umgang mit IT-Systemen

Abspraken zur vorhabenbezogene Konkretisierung:

Das erste Unterrichtsvorhaben stellt eine allgemeine Einführung in das Fach Informatik dar. Da einige Schülerinnen und Schüler das Fach zum ersten Mal in der Einführungsphase belegen, wird zu Beginn ein Überblick über die Themen des Schulfachs Informatik gegeben. Unter anderem wird auf den zentralen Begriff der Information eingegangen und die Möglichkeit der Codierung von Daten, insbesondere wird die Binärdarstellung von Zahlen thematisiert. Stationen der geschichtlichen Entwicklung werden angesprochen wie z.B. Speichermedien, von-Neumann-Architektur und das EVA-Prinzip. Außerdem werden die SuS in die konkrete Nutzung der Informatiksysteme an der Schule eingewiesen.

<b>Unterrichtssequenzen</b>	<b>zu entwickelnde Kompetenzen</b>	<b>Beispiele, Medien, Materialien</b>
1. Allgemeine Einführung a) Übersicht über das Fach b) Einführung in die Informatiksysteme der Schule	Die Schülerinnen und Schüler <ul style="list-style-type: none"><li>• nutzen die im Unterricht eingesetzten Informatiksysteme selbstständig, sicher, zielführend und verantwortungsbewusst (D),</li><li>• nutzen das verfügbare Informatiksystem zu strukturierten Verwaltung und gemeinsamen Verwendung von Daten unter Berücksichtigung der Rechteverwaltung (K),</li><li>• nutzen das Internet zur Recherche, zum Datenaustausch und zur Kommunikation (K).</li></ul>	Computerräume der Schule
2. Grundlagen von Informatiksystemen	Die Schülerinnen und Schüler	

<p>Überblick über die Geschichte der Informatik</p> <p>a) Darstellung von Zahlen im Binärsystem</p> <p>b) Von-Neumann-Architektur</p> <p>c) Geschichte der Datenverarbeitung</p>	<ul style="list-style-type: none"> <li>• stellen ganze Zahlen und Zeichen in Binärcodes dar (D),</li> <li>• interpretieren Binärcodes als Zahlen und Zeichen (D),</li> <li>• beschreiben und erläutern den strukturellen Aufbau und die Arbeitsweise singulärer Rechner am Beispiel der „Von-Neumann-Architektur“ (A).</li> <li>• erläutern wesentliche Grundlagen der Geschichte der digitalen Datenverarbeitung (A).</li> </ul>	
--	---	--

## **Unterrichtsvorhaben Nr. 2**

Thema: Grundlagen der Programmierung mit Java und einfache Algorithmen

Leitfragen: Aus welchen Bausteinen besteht ein Java-Programm? Welche Datentypen und Kontrollstrukturen stehen in Java zur Verfügung und wie nutzt man diese? Wie verläuft der Entwicklungsprozess eines Java-Programms?

Zeitbedarf: 33 Stunden

Berufsorientierung:

- Softwareentwickler\*in

Absprachen zur vorhabenbezogenen Konkretisierung:

Die Grundlagen zur Programmieretechnik mit Java werden durch die SuS selbstständig und individuell erarbeitet. Dazu zählen: einfache Datentypen, Variablenkonzept, Wertzuweisungen, Kontrollstrukturen, Methodenkonzept, Arrays. Die Schülerinnen und Schüler wenden ihr Wissen kontextbezogen auf kleinere Problemstellungen an. Dabei durchlaufen sie den kontinuierlichen Prozess von Implementieren, Compilieren und Testen.

Die Schülerinnen und Schüler dokumentieren kontinuierlich und selbstständig ihren Lernfortschritt in einem Lerntagebuch.

<b>Unterrichtssequenzen</b>	<b>zu entwickelnde Kompetenzen</b>	<b>Beispiele, Medien, Materialien</b>
1. Einführung in die Programmierung a) Kap. 1: Erste Schritte b) Kap. 2: Programmaufbau c) Kap. 3: Datentypen und Operationen d) Kap. 4: Variablen e) Kap. 5: Verzweigungen f) Kap. 6: Schleifen g) Kap. 7: Arrays	Die Schülerinnen und Schüler <ul style="list-style-type: none"><li>• analysieren und erläutern einfache Algorithmen und Programme (A),</li><li>• modifizieren einfache Algorithmen und Programme (I),</li><li>• entwerfen einfache Algorithmen und stellen sie umgangssprachlich und grafisch dar (M),</li><li>• implementieren Algorithmen unter Verwendung von Variablen und Wertzuweisungen, Kontrollstrukturen sowie Methodenaufrufen (I),</li><li>• testen Programme schrittweise anhand von Beispielen (I).</li><li>• implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I).</li></ul>	Leitprogramm „Einführung in die Programmierung“ JavaEditor BlueJ Nassi-Shneiderman-Diagramm Struktred (Editor für Nassi-Shneiderman-Diagramme) Lerntagebuch

### **Unterrichtsvorhaben Nr. 3**

Thema: Entwurf und Implementierung einfacher Sortierverfahren und eines klassischen kryptologischen Verfahrens

Leitfragen: Wie kann man die bisherigen Erkenntnisse im Umgang mit Arrays zum Sortieren von großen Datenmengen und zur einfachen „Verschlüsselung“ von Daten benutzen? Wie effizient ist ein Algorithmus hinsichtlich seines Laufzeitverhaltens?

Zeitbedarf: 12 Stunden

Berufsorientierung:

- Softwareentwickler\*in
- Cybersecurity

Absprachen zur vorhabenbezogene Konkretisierung:

Aufbauend auf den programmiertechnischen Grundlagen des letzten Unterrichtsvorhabens können die SuS nun ihr Wissen zu Arrays und Strings kontextbezogen anwenden:

Dazu werden mindestens zwei der einfachen Sortierverfahren behandelt, indem sie simuliert, implementiert und getestet werden. In der anschließenden Analyse reflektieren die SuS das Laufzeitverhalten der Algorithmen im best, worst und normal case.

Darüberhinaus wird ein einfaches kryptologisches Verfahren (z.B. CAESAR) behandelt. Neben der Durchführung der einfachen Verschlüsselungstechnik werden die zur Implementierung nötigen String-Operationen vertieft. Hierzu gehört auch die Thematisierung des ASCII-Codes. Die Schülerinnen und Schüler reflektieren die Sicherheitsaspekte des Verfahrens.

<b>Unterrichtssequenzen</b>	<b>zu entwickelnde Kompetenzen</b>	<b>Beispiele, Medien, Materialien</b>
1. Implementierung und Analyse einfacher Sortier- und Suchverfahren auf Arrays	Die Schülerinnen und Schüler <ul style="list-style-type: none"><li>• analysieren Such- und Sortieralgorithmen und wenden sie auf Beispiele an (D),</li><li>• entwerfen einen weiteren Algorithmus zum Sortieren (M),</li><li>• beurteilen die Effizienz von Algorithmen am Beispiel von Sortierverfahren hinsichtlich Zeitaufwand und Speicherplatzbedarf (A).</li></ul>	MathePrisma der Uni Wuppertal Youtube JavaEditor BlueJ
2. Implementierung und Analyse eines einfachen kryptologischen Verfahrens	Die Schülerinnen und Schüler	MathePrisma Uni Wuppertal JavaEditor BlueJ

	<ul style="list-style-type: none"><li>• analysieren und erläutern einfache Algorithmen und Programme (A),</li><li>• implementieren Algorithmen unter Verwendung von Variablen und Wertzuweisungen, Kontrollstrukturen sowie Methodenaufrufen (I),</li><li>• testen Programme schrittweise anhand von Beispielen (I).</li></ul>	
--	--	--

#### **Unterrichtsvorhaben Nr. 4**

Thema: Objektorientierte Modellierung und Implementierung von Klassen- und Objektbeziehungen anhand von grafischen Spielen, Simulationen und grafischen Oberflächen

Leitfragen: Was sind Objekte, was sind Klassen? Wie programmiert man objektorientiert? Was bedeutet Vererbung? Wie lassen sich Animationen und Simulationen optischer Gegenstandsbereiche realisieren? Wie lassen sich komplexere Datenflüsse und Beziehungen zwischen Objekten und Klassen realisieren?

Zeitbedarf: 36 Stunden

Berufsorientierung:

- Softwareentwickler\*in

Ab sprachen zur vorhabenbezogene Konkretisierung:

Ausgehend von einem vorgegebenen Greenfootprojekt (z. B. crazy kara) wird die Entwicklungsumgebung Greenfoot eingeführt. Diese wird genutzt, um den Schülerinnen und Schüler die Bedeutung von Objekten und Klassen deutlich zu machen. Innerhalb des Projekts modellieren sie eigene Klassen und differenzieren diese in ihren Attributen und Methoden. Neben der Erzeugung von Objekten erlernen sie auch deren Benutzung und Interaktionen untereinander.

Die Klassen des Projektes schrittweise durch Vererbung konkretisiert und deren Funktion implementiert. Dabei werden die bereits erlernten Grunddatentypen und verschiedenen Arten von Methoden in anderem Zusammenhang angewendet und eingeübt. Insbesondere wird das Geheimnisprinzip vertieft.

Erlern tes wird in einer komplett selbst erstellten Animation oder einem komplett selbst erstellten Spiel reorganisiert. Das Projekt wird dabei geeignet dokumentiert.

Nach Abschluss des Projektes wird die objektorientierte Programmierung anhand des Entwurfs und der Implementierung grafischer Oberflächen mit dem JavaEditor kontextbezogen vertieft. Dabei werden die Ereignisbehandlungsroutinen der Oberflächen-Objekte verwendet und das Geheimnisprinzip angewendet.

<b>Unterrichtssequenzen</b>	<b>zu entwickelnde Kompetenzen</b>	<b>Beispiele, Medien, Materialien</b>
1. Grundlagen der Entwicklungsumgebung Greenfoot und Einführung in die OO a) Was ist ein Objekt? Was ist eine Klasse? b) Objekte erzeugen c) Darstellung / Verhalten von Objekten (act-Methode) d) grundlegende Methoden der Klassen Actor, Animal und World e) Methoden und Attribute f) Objektinteraktion g) Vererbung	Die Schülerinnen und Schüler <ul style="list-style-type: none"><li>• ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M),</li><li>• modellieren Klassen mit ihren Attributen, ihren Methoden und Assoziationsbeziehungen (M),</li><li>• modellieren Klassen unter Verwendung von Vererbung (M),</li></ul>	Greenfoot Fortbildungsmaterialien zur OO



<p>h) Geheimnisprinzip</p>	<ul style="list-style-type: none"> <li>• ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen oder lineare Datensammlungen zu (M),</li> <li>• ordnen Klassen, Attributen und Methoden ihren Sichtbarkeitsbereich zu (M),</li> <li>• stellen den Zustand eines Objekts dar (D),</li> <li>• stellen die Kommunikation zwischen Objekten grafisch dar (M),</li> <li>• stellen Klassen, Assoziations- und Vererbungsbeziehungen in Diagrammen grafisch dar (D),</li> <li>• dokumentieren Klassen durch Beschreibung der Funktionalität der Methoden (D),</li> <li>• analysieren und erläutern eine objektorientierte Modellierung (A),</li> <li>• implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I).</li> </ul>	
<p>2. weitere Möglichkeiten von Greenfoot und Erstellung einer eigenen Animation/eines eigenen Spiels</p> <p>a) Entwicklung einer Spiel-/Animationsidee (Anforderungsdefinition)</p> <p>b) Implementierung der Idee unter Verwendung von Hilfestellungen aus dem Internet</p> <p>c) Dokumentation des Projekts</p>	<p>zusätzlich: Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> <li>• nutzen das verfügbare Informatiksystem zur strukturierten Verwaltung und gemeinsamen Verwendung von Daten unter Berücksichtigung der Rechteverwaltung (K),</li> <li>• nutzen das Internet zur Recherche, zum Datenaustausch und zur Kommunikation (K).</li> </ul>	<p>Greenfoot Fortbildungsmaterialien zur OO</p>

## II)/III) Qualifikationsphase – Grundkurs/**Leistungskurs**

Die Aufstellung der Unterrichtsvorhaben erfolgt für den Grundkurs und den **Leistungskurs** kombiniert.

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 bereitgestellte Informatiksysteme und das Internet reflektiert zur Erschließung, Aufbereitung und Präsentation fachlicher Inhalte (D),
- nutzen das verfügbare Informatiksystem zur strukturierten Verwaltung von Dateien unter Berücksichtigung der Rechteverwaltung  
**/Daten, zur Organisation von Arbeitsabläufen sowie zur Verteilung und Zusammenführung von Arbeitsanteilen** (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).

Ebenso bieten fast alle Unterrichtsvorhaben, in denen Programme implementiert werden, die Gelegenheit, die folgenden Kompetenzen zu erwerben bzw. zu vertiefen:

Schülerinnen und Schüler

- 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),
- wenden eine didaktisch orientierte Entwicklungsumgebung (**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).**

## ***Unterrichtsvorhaben Q1-I***

Thema: Rekursive Algorithmen und Backtracking in Anwendungskontexten

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

Zeitbedarf: 20/25 Stunden

Berufsorientierung:

- Softwareentwickler\*in
- Wissenschaftspropädeutik

Abspraken zur vorhabenbezogene Konkretisierung:

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

Verschiedene NP-vollständige Probleme (wie z. B. Rucksack, n-Damen, Springer, Irrgarten, etc.) werden algorithmisch rekursiv formuliert **und als Backtracking-Algorithmus implementiert.**

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

Lernmittel / Materialien:

- Arbeitsblätter in der moodle-Plattform
- eine didaktische Entwicklungsumgebung (z. B. Java-Editor o. a.)

Unterrichtssequenzen	zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Entwicklung der Rekursion als fundamentale Idee der Informatik</p> <ul style="list-style-type: none"> <li>• rekursive Formeln</li> <li>• rekursive Funktionen / Methoden</li> <li>• rekursive Programmierung</li> </ul>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> <li>• analysieren und erläutern Algorithmen und Programme (A),</li> <li>• modifizieren Algorithmen und Programme (I),</li> </ul>	<p>Türme von Hanoi mit Schwerpunkt auf</p> <ul style="list-style-type: none"> <li>• Zahl der Versetzungsoperationen</li> <li>• Protokollierung der Versetzungen</li> </ul>
<p>2. Rekursion in mathematischen und informatischen Kontexten</p> <ul style="list-style-type: none"> <li>• Rekursion in mathematischen Kontexten</li> <li>• Analyse und Darstellung des rekursiven Ablaufs einer Methode</li> <li>• Analyse des Laufzeitverhaltens linearer und verzweigter Rekursion</li> </ul>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> <li>• analysieren und erläutern Algorithmen und Programme (A),</li> <li>• modifizieren Algorithmen und Programme (I),</li> <li>• stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D),</li> <li>• testen Programme systematisch anhand von Beispielen <b>und mit Hilfe von Testanwendungen</b> (I).</li> <li>• untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen (A).</li> </ul>	<ul style="list-style-type: none"> <li>• Fakultätsfunktion (lineare Rekursion)</li> <li>• Fibonacci-Funktion (verzweigte Rekursion)</li> <li>• ggT (verzweigte Rekursion)</li> <li>• evtl. Fraktale (Kochkurve, Sierpinski dreieck, etc.)</li> </ul>
<p>3. NP-vollständige Probleme lösen mit Backtracking</p> <ul style="list-style-type: none"> <li>• Erarbeitung verschiedener NP-vollständiger Probleme</li> <li>• Algorithmische Beschreibung einer Lösungsidee</li> <li>• <b>Implementierung eines Problems als Backtrackingalgorithmus</b></li> </ul>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> <li>• analysieren und erläutern Algorithmen und Programme (A),</li> <li>• modifizieren Algorithmen und Programme (I),</li> <li>• stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D),</li> <li>• entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“ und „Teilen und Herrschen“ <b>und "Backtracking"</b> (M)</li> <li>• testen Programme systematisch anhand von Beispielen <b>und mit Hilfe von Testanwendungen</b> (I).</li> <li>• untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen (A).</li> </ul>	<ul style="list-style-type: none"> <li>• Rucksackproblem</li> <li>• Irrgartenproblem</li> <li>• <b>Projektarbeit zu einer Problemstellung</b></li> </ul>
<p>4. Effiziente Sortierverfahren / Suchverfahren</p> <ul style="list-style-type: none"> <li>• Wiederholung bereits bekannter Sortier- und Suchverfahren als rekursiver Algorithmus</li> <li>• Erarbeitung eines Sortierverfahrens der Laufzeit <math>O(n \cdot \log(n))</math></li> <li>• Erarbeitung eines Suchverfahrens der Laufzeit <math>O(\log(n))</math>.</li> </ul>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> <li>• analysieren und erläutern Algorithmen und Programme (A),</li> <li>• modifizieren Algorithmen und Programme (I),</li> <li>• stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D),</li> <li>• entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“</li> </ul>	<ul style="list-style-type: none"> <li>• Demonstrationsprogramm zur Visualisierung von Sortierverfahren</li> <li>• Quicksortvisualisierung zur Erarbeitung der Idee</li> <li>• Suchspiel zur Erarbeitung der Binären Suche</li> </ul>

	<p>und „Teilen und Herrschen“ und "Backtracking" (M),</p> <ul style="list-style-type: none"><li>• testen Programme systematisch anhand von Beispielen und mit Hilfe von Testanwendungen (I).</li><li>• implementieren und erläutern iterative und rekursive Such- und Sortierverfahren unterschiedlicher Komplexitätsklassen (Speicherbedarf und Laufzeitverhalten) (I),</li><li>• beurteilen die Effizienz von Algorithmen unter Berücksichtigung des Speicherbedarfs und der Zahl der Operationen (A),</li><li>• untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen (A).</li></ul>	
--	---	--

## ***Unterrichtsvorhaben Q1-II***

Thema: Modellierung und Implementierung dynamischer Listenstrukturen und deren Anwendungen

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

Zeitbedarf: 25/30 Stunden

*Berufsorientierung:*

- Softwareentwickler\*in
- Wissenschaftspropädeutik

Abspraken zur vorhabenbezogene Konkretisierung:

Nach Analyse einer Problemstellung in einem geeigneten Anwendungskontext, in dem Daten nach dem Last-In-First-Out-Prinzip verwaltet werden, werden der Aufbau von Stapeln am Beispiel dargestellt und die Operationen der Klasse Stack anhand der Abiturklasse erläutert. Anschließend werden für die Anwendung notwendige Klassen modelliert und implementiert.

Anschließend wird die Anwendung modifiziert, um den Umgang mit der Datenstruktur zu üben. Anhand einer Anwendung, in der Daten nach dem First-In-First-Out-Prinzip verwaltet werden, werden Unterschiede zwischen den Datenstrukturen Schlange und Stapel erarbeitet. Dabei werden die Operationen der Datenstruktur Schlange thematisiert und die entsprechende Abiturklasse Queue verwendet.

Um einfacher an Objekte zu gelangen, die zwischen anderen gespeichert sind, wird die Klasse List gemäß der Abiturklasse eingeführt und in einem Anwendungskontext verwendet.

(Je nach thematisierten Anwendungskontexten ist eine andere Reihenfolge bei der Behandlung der oben genannten Datenstrukturen möglich und zulässig.)

In mindestens einem weiteren Anwendungskontext wird die Verwaltung von Daten in Schlangen, Stapeln oder Listen vertieft. Modellierungen werden dabei in Entwurfs- und Implementationsdiagrammen dargestellt.

Lernmittel / Materialien:

- Arbeitsblätter in der moodle-Plattform
- eine didaktische Entwicklungsumgebung (z. B. Java-Editor o. a.)
- Zeigermodell
- Modellprogramm zur Visualisierung dynamischer Listenstrukturen

Unterrichtssequenzen	zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Die Datenstruktur Stapel im Anwendungskontext unter Nutzung der Klasse Stack</p> <ul style="list-style-type: none"> <li>Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen</li> <li>Erarbeitung der Funktionalität der Klasse Stack</li> <li>Modellierung und Implementierung der Anwendung unter Verwendung der Klasse Stack.</li> <li><b>Implementierung der Klasse Stack</b></li> </ul>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> <li>ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M),</li> <li>stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D),</li> <li>modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M),</li> <li>ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M),</li> <li>ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M),</li> <li>stellen die Kommunikation zwischen Objekten grafisch dar (D),</li> <li>stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D),</li> <li>dokumentieren Klassen (D),</li> <li>analysieren und erläutern objektorientierte Modellierungen (A),</li> <li>implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I).</li> <li>analysieren und erläutern Algorithmen und Programme (A),</li> <li>modifizieren Algorithmen und Programme (I),</li> <li>stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D),</li> <li>implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I),</li> <li>testen Programme systematisch anhand von Beispielen <b>und mit Hilfe von Testanwendungen</b> (I).</li> <li>erläutern Operationen dynamischer (linearer oder/und nicht-linearer) Datenstrukturen (A),</li> <li><b>implementieren Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (I),</b></li> </ul>	<p>Visualisierungsprogramm zu dynamischen Datenstrukturen</p>
<p>2. Die Datenstruktur Schlange im Anwendungskontext unter Nutzung der Klasse Queue</p> <ul style="list-style-type: none"> <li>Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen</li> <li>Erarbeitung der Funktionalität der Klasse Queue</li> <li>Modellierung und Implementierung der Anwendung unter Verwendung der Klasse Queue.</li> <li><b>Implementierung der Klasse Queue</b></li> </ul>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> <li>ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M),</li> <li>stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D),</li> <li>modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M),</li> <li>ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M),</li> <li>ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M),</li> <li>stellen die Kommunikation zwischen Objekten grafisch dar (D),</li> <li>stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D),</li> <li>dokumentieren Klassen (D),</li> <li>analysieren und erläutern objektorientierte Modellierungen (A),</li> <li>implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I).</li> <li>analysieren und erläutern Algorithmen und Programme (A),</li> <li>modifizieren Algorithmen und Programme (I),</li> <li>stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D),</li> <li>implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I),</li> <li>testen Programme systematisch anhand von Beispielen <b>und mit Hilfe von Testanwendungen</b> (I).</li> <li>erläutern Operationen dynamischer (linearer oder/und nicht-linearer) Datenstrukturen (A),</li> <li><b>implementieren Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (I),</b></li> </ul>	<p>Visualisierungsprogramm zu dynamischen Datenstrukturen</p>
<p>3. Die Datenstruktur lineare Liste im Anwendungskontext unter Nutzung der Klasse List</p> <ul style="list-style-type: none"> <li>Erarbeitung der Vorteile der Klasse List im Gegensatz zu den bereits bekannten linearen Strukturen</li> <li>Modellierung und Implementierung einer kontextbezogenen Anwendung unter Verwendung der Klasse List.</li> <li><b>Implementierung der Klasse List</b></li> </ul>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> <li>ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M),</li> <li>stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D),</li> <li>modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M),</li> <li>ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M),</li> <li>ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M),</li> <li>stellen die Kommunikation zwischen Objekten grafisch dar (D),</li> <li>stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D),</li> <li>dokumentieren Klassen (D),</li> <li>analysieren und erläutern objektorientierte Modellierungen (A),</li> <li>implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I).</li> <li>analysieren und erläutern Algorithmen und Programme (A),</li> <li>modifizieren Algorithmen und Programme (I),</li> <li>stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D),</li> <li>implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I),</li> <li>testen Programme systematisch anhand von Beispielen <b>und mit Hilfe von Testanwendungen</b> (I).</li> <li>erläutern Operationen dynamischer (linearer oder/und nicht-linearer) Datenstrukturen (A),</li> <li><b>implementieren Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (I),</b></li> </ul>	<p>Visualisierungsprogramm zu dynamischen Datenstrukturen Einfache Oberfläche zur Visualisierung der linearen Liste, z. B. die Verwaltung einer Namensliste</p>

4. Vertiefung / Anwendung einer linearen Datenstruktur im Anwendungskontext.	zusätzlich: Die Schülerinnen und Schüler <ul style="list-style-type: none"><li>• modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M),</li><li>• verwenden bei der Modellierung geeigneter Problemstellungen Möglichkeiten der Polymorphie (M),</li></ul>	Umsetzung in einem größeren Projekt
--	---	-------------------------------------



## ***Unterrichtsvorhaben Q1-III***

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

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

Zeitbedarf: 20 Stunden

*Berufsorientierung:*

- Softwareentwickler\*in
- Wissenschaftspropädeutik

Absprachen zur vorhabenbezogene Konkretisierung:

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

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

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

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

Zu einer Problemstellung in einem entsprechenden Anwendungskontext werden die Operationen der Datenstruktur Suchbaum thematisiert und unter der Verwendung der Klasse BinarySearchTree der Vorgaben für das Zentralabitur weitere Klassen oder Methoden in diesem Kontext modelliert und implementiert. Die Suchbäume werden wie zuvor auch grafisch dargestellt.

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

Lernmittel / Materialien:

- eine didaktische Entwicklungsumgebung (z. B. Java-Editor o. a.)
- Arbeitsblätter und Demonstrationsprogramme in der moodle-Plattform
- Modellprogramm zur Visualisierung dynamischer Listenstrukturen

Unterrichtssequenzen	zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Analyse von Baumstrukturen in verschiedenen Kontexten</p> <ul style="list-style-type: none"> <li>• Grundlegende Begriffe (Grad, Tiefe, Höhe, Blatt, Inhalt, Teilbaum, Ebene, Vollständigkeit)</li> <li>• Aufbau und Darstellung von binären Bäumen anhand von Baumstrukturen in verschiedenen Kontexten</li> </ul>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> <li>• ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M),</li> <li>• stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D),</li> <li>• modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M),</li> </ul>	<ul style="list-style-type: none"> <li>• Demoprogramm und Arbeitsblätter zum Projekt Akinator</li> </ul>
<p>2. Die Datenstruktur Binärbaum im Anwendungskontext unter Nutzung der Klasse BinaryTree</p> <ul style="list-style-type: none"> <li>• Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen im Anwendungskontext</li> <li>• Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramms</li> <li>• Erarbeitung der Klasse BinaryTree und beispielhafte Anwendung der Operationen</li> <li>• Implementierung der Anwendung oder von Teilen der Anwendung</li> <li>• Traversierung eines Binärbaums im Pre-, In- und Postorderdurchlauf</li> <li>• <b>Implementierung der Standardoperationen der Klasse BinaryTree</b></li> </ul>	<ul style="list-style-type: none"> <li>• modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M),</li> <li>• ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen <b>sowie/oder</b> lineare und nichtlineare Datensammlungen zu (M),</li> <li>• verwenden bei der Modellierung geeigneter Problemstellungen Möglichkeiten der Polymorphie (M),</li> <li>• ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M),</li> <li>• stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D),</li> </ul>	<ul style="list-style-type: none"> <li>• Arbeitsblätter zur Projektarbeit Akinator</li> <li>• Traversierungsverfahren</li> <li>• Projektarbeit Morsebaum</li> </ul>
<p>3. Die Datenstruktur binärer Suchbaum im Anwendungskontext unter Verwendung der Klasse BinarySearchTree</p> <ul style="list-style-type: none"> <li>• Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen</li> <li>• Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramm,</li> <li>• grafische Darstellung eines binären Suchbaums und Erarbeitung der Struktureigenschaften</li> <li>• Erarbeitung der Klasse BinarySearchTree und Einführung des Interface Item zur Realisierung einer geeigneten Ordnungsrelation</li> <li>• Implementierung der Anwendung oder von Teilen der Anwendung inklusive einer sortierten Ausgabe des Baums</li> </ul>	<ul style="list-style-type: none"> <li>• dokumentieren Klassen (D),</li> <li>• analysieren und erläutern objektorientierte Modellierungen (A),</li> <li>• implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I).</li> <li>• analysieren und erläutern Algorithmen und Programme (A),</li> <li>• modifizieren Algorithmen und Programme (I),</li> <li>• stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D),</li> <li>• entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“</li> </ul>	<ul style="list-style-type: none"> <li>• Arbeitsblätter zum binären Suchbaum</li> <li>• Arbeitsblätter zum Baumsortieren und zu Traversierungsverfahren</li> </ul>

<ul style="list-style-type: none"> <li>Implementierung ausgewählter Standardoperationen der Klasse <code>BinarySearchTree</code></li> </ul>	<p>und „Teilen und Herrschen“ und „Backtracking“ (M),</p>	
<p>4. Übung und Vertiefungen der Verwendung von Binärbäumen oder binären Suchbäumen anhand weiterer Problemstellungen</p>	<ul style="list-style-type: none"> <li>implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I),</li> <li>testen Programme systematisch anhand von Beispielen und mit Hilfe von Testanwendungen (I).</li> <li>erläutern Operationen dynamischer (linearer oder/und nicht-linearer) Datenstrukturen (A),</li> <li>implementieren Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (I),</li> </ul>	<ul style="list-style-type: none"> <li>Projektarbeiten zu einem der Themen Termbäume, Morsebäume, Stichwortbaum, Ahnenbaum</li> </ul>

## **Unterrichtsvorhaben Q1-IV**

Thema: Sicherheit und Datenschutz in Netzstrukturen **und Modellierung und Implementierung von Client-Server-Anwendungen**

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

Zeitbedarf: 15/25 Stunden

Berufsorientierung:

- Softwareentwickler\*in
- Cybersecurity
- Wissenschaftspropädeutik
- Systemadministrator

Abspraken zur vorhabenbezogene Konkretisierung:

Ausgehend von einer Kommunikation zwischen zwei Kommunikationspartnern über eine einfache Leitung wird die Notwendigkeiten einer Datenübertragung erarbeitet. Die Schichten des TCP/IP-Schichtenmodells werden beispielgebunden erarbeitet (Basisbandübertragungsverfahren, Prüfverfahren, Vermittlungsschicht, Anwendungsprotokoll) und an einer Simulationssoftware getestet. Verschiedene Netzwerk-Topologien werden entwickelt und in Client-Server-Anwendungen simuliert.

**Im Leistungskurs werden ausgehend von einer einfachen Echo-Anwendung die beteiligten Komponenten (Echo-Server und Echo-Client) entwickelt und unter Verwendung der ZA-Klassen implementiert.**

**Die Echo-Anwendung wird zu einer Chat-Anwendung erweitert, notwendige Protokolle werden entwickelt und systematisch dargestellt.**

**Die Schülerinnen und Schüler entwickeln eine individuelle Client-Serveranwendung, definieren notwendige Protokolle und erweitern die Chat-Anwendung entsprechend der Vorgaben.**

Über die Sicherheit von Netzwerkanwendungen wird das Augenmerk auf verschiedene symmetrische und asymmetrische kryptografische Verfahren gelenkt, welche analysiert und erläutert werden. Fallbeispiele zur Datenschutzproblematik und zum Urheberrecht runden das Unterrichtsvorhaben ab.

Lernmittel / Materialien:

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

Unterrichtssequenzen	zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Schichten des TCP/IP-Protokolls</p> <ul style="list-style-type: none"> <li>• Erarbeitung der Notwendigkeiten einer Netzwerkkommunikation</li> <li>• Erarbeitung der Schichten des TCP/IP-Protokolls: Ethernet-, Internet-, Transport- und Anwendungsschicht</li> </ul>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> <li>• beschreiben und erläutern Netzwerk-Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken (A),</li> </ul>	<ul style="list-style-type: none"> <li>• Arbeitsblätter zur Einführung in Netzwerke</li> </ul>
<p>2. Simulation von Netzwerken / Netzwerk-Topologien</p> <ul style="list-style-type: none"> <li>• Erarbeitung der Topologien: Peer-to-Peer, Sterntopologie, Baumtopologie, Vermaschtes Netz</li> <li>• Simulation von Client-Server-Anwendungen</li> <li>• Simulation von Protokollen der Anwendungsschicht (POP3, SMTP, etc.)</li> </ul>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> <li>• beschreiben und erläutern Netzwerk-Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken (A),</li> <li>• analysieren und erläutern Protokolle zur Kommunikation in einem Client-Server-Netzwerk (A),</li> </ul>	<ul style="list-style-type: none"> <li>• Simulationssoftware FILIUS</li> <li>• Arbeitsblätter und Skript zu FILIUS</li> </ul>
<p>3. Entwicklung verschiedener Client-Server-Anwendungen</p> <ul style="list-style-type: none"> <li>• Grundlegende Begriffe</li> <li>• ZA-Klassen</li> <li>• Echo-Anwendung</li> <li>• Chat Anwendung (Nutzung der List)</li> <li>• Beliebige Client-Server-Anwendung</li> </ul>	<ul style="list-style-type: none"> <li>• erläutern das Prinzip der Nebenläufigkeit (A),</li> <li>• analysieren und erläutern Algorithmen und Methoden zur Client-Server-Kommunikation (A),</li> <li>• entwickeln und implementieren Algorithmen und Methoden zur Client-Server-Kommunikation (I).</li> <li>• analysieren und erläutern Protokolle zur Kommunikation in einem Client-Server-Netzwerk (A),</li> <li>• entwickeln und erweitern Protokolle zur Kommunikation in einem Client-Server-Netzwerk (M).</li> </ul>	<p>Umsetzung eines größeren Projektes</p> <ul style="list-style-type: none"> <li>• Chat</li> <li>• Anwendungsserver (wie z.B. Buchungsserver für Kinokarten)</li> </ul>
<p>4. Analyse und Erläuterung kryptografischer Verfahren</p> <ul style="list-style-type: none"> <li>• Erläuterung symmetrischer Verfahren: monoalphabetisch: Cäsar, polyalphabetisch: Vigenère</li> <li>• Erläuterung asymmetrischer Verfahren: RSA, Diffie-Hellman</li> <li>• Analyse der Sicherheit verschiedener Verfahren und Auswirkungen auf den Datenschutz/Urheberrecht</li> </ul>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> <li>• analysieren und erläutern Algorithmen und Programme (A),</li> <li>• stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D),</li> <li>• analysieren und erläutern Eigenschaften, Funktionsweisen und Einsatzbereiche symmetrischer und asymmetrischer Verschlüsselungsverfahren (A).</li> <li>• untersuchen und bewerten anhand von Fallbeispielen Auswirkungen des Einsatzes von Informatiksystemen sowie Aspekte der Sicherheit von Informatiksystemen, des Datenschutzes und des Urheberrechts (A),</li> </ul>	<ul style="list-style-type: none"> <li>• Arbeitsblätter zu kryptografischen Verfahren</li> <li>• CrypTool</li> <li>• Materialien von klicksafe (Zusatzmodule „Nicht alles was geht, ist auch erlaubt“, „Ich bin öffentlich ganz privat“, „Datenschutz im Internet“)</li> </ul>

	<ul style="list-style-type: none"><li>• 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).</li></ul>	
--	---	--

## **Unterrichtsvorhaben Q2-I**

*Hinweis: Im Leistungskurs sollte dieses Thema in die Q1 vorgezogen werden.*

Thema: Modellierung und Nutzung relationaler Datenbanken in Anwendungskontexten

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

Zeitbedarf: 25/30 Stunden

Berufsorientierung:

- Softwareentwickler\*in
- Wissenschaftspropädeutik
- Datenbankentwickler\*in

Absprachen zur vorhabenbezogene Konkretisierung:

Ausgehend von einer konkreten Anwendungssituation entwickeln die Schülerinnen und Schüler Ideen zur Modellierung von Daten und erkennen die Vorzüge von Datenbanksystemen.

In weiteren Anwendungskontexten müssen Datenbanken entwickelt werden, um Daten zu speichern und Informationen für die Beantwortung von möglicherweise auftretenden Fragen zur Verfügung zu stellen. Dafür ermitteln Schülerinnen und Schüler in den Anwendungssituationen Entitäten, zugehörige Attribute, Relationen und Kardinalitäten und stellen diese in Entity-Relationship-Modellen dar. Entity-Relationship-Modelle werden interpretiert und erläutert, modifiziert und in das Relationale Modell überführt.

An einem Beispiel wird verdeutlicht, dass in Datenbanken Redundanzen unerwünscht sind und Konsistenz gewährleistet sein sollte. Die 1. bis 3. Normalform wird als Gütekriterium für Datenbankentwürfe eingeführt. Datenbankschemata werden hinsichtlich der 1. bis 3. Normalform untersucht und (soweit nötig) normalisiert.

Ausgehend von einer vorhandenen Datenbasis, **für die eine Datenbank angelegt und mit den Daten gefüllt wird**, entwickeln Schülerinnen und Schüler für sie relevante Fragestellungen, die mit dem vorhandenen Datenbestand beantwortet werden sollen. Zur Beantwortung dieser Fragestellungen wird die vorgegebene Datenbank von den Schülerinnen und Schülern analysiert und die notwendigen Grundbegriffe für Datenbanksysteme sowie die erforderlichen SQL-Abfragen werden erarbeitet.

Mit Hilfe von SQL-Anweisungen können anschließend im Kontext relevante Informationen aus der Datenbank extrahiert werden. Die Operationen der Relationenalgebra werden mit SQL-Abfragen simuliert.

Anhand von Fallbeispielen werden Probleme bei der Nutzung von Datenbanksystemen aufgezeigt und im Hinblick auf gesellschaftliche Auswirkungen diskutiert.

Lernmittel / Materialien:

- Arbeitsblätter und Demonstrationsprogramme in der moodle-Plattform
- Datenbankserver (z. B. XAMPP)
- SQL-Tutorial (z. B. die Seite SQL-Zoo, VideoCenter-Datenbank)

Unterrichtssequenzen	zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Modellierung von relationalen Datenbanken</p> <ul style="list-style-type: none"> <li>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</li> <li>Entwicklung einer Datenbank aus einem Datenbankentwurf: Modellierung eines relationalen Datenbankschematas zu einem Entity-Relationship-Diagramm inklusive der Bestimmung von Primär- und Sekundärschlüsseln</li> <li>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)</li> </ul>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> <li>ermitteln für anwendungsbezogene Problemstellungen Entitäten, zugehörige Attribute, Relationen und Kardinalitäten (M),</li> <li>stellen Entitäten mit ihren Attributen und die Beziehungen zwischen Entitäten mit Kardinalitäten in einem Entity-Relationship-Diagramm grafisch dar (D),</li> <li>modifizieren eine Datenbankmodellierung (M),</li> <li>modellieren zu einem Entity-Relationship-Diagramm ein relationales Datenbankschema (M),</li> <li>bestimmen Primär- und Sekundärschlüssel (M),</li> <li>analysieren und erläutern eine Datenbankmodellierung (A),</li> <li>erläutern die Eigenschaften normalisierter Datenbankschemata (A),</li> <li>überprüfen Datenbankschemata auf vorgegebene Normalisierungseigenschaften (D),</li> <li>überführen Datenbankschemata in die 1. bis 3. Normalform (M).</li> </ul>	<ul style="list-style-type: none"> <li>Arbeitsblätter zur Einführung in Datenbanken in der moodle-Plattform</li> </ul>
<p>2. Nutzung von relationalen Datenbanken</p> <ul style="list-style-type: none"> <li>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</li> <li>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</li> </ul>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> <li>modifizieren eine Datenbankmodellierung (M),</li> <li>bestimmen Primär- und Sekundärschlüssel (M),</li> <li><b>implementieren ein relationales Datenbankschema als Datenbank (I),</b></li> <li>analysieren und erläutern eine Datenbankmodellierung (A),</li> <li>ermitteln Ergebnisse von Datenbankabfragen über mehrere verknüpfte Tabellen (D),</li> <li>analysieren und erläutern die Syntax und Semantik einer Datenbankabfrage (A),</li> <li>verwenden die Syntax und Semantik einer Datenbankabfragesprache, um Informationen aus einem Datenbanksystem zu extrahieren (I).</li> </ul>	<ul style="list-style-type: none"> <li>Arbeitsblätter zur Einführung in SQL in der moodle-Plattform</li> <li>Tutorial der Seite SQL-Zoo (<a href="http://sqlzoo.net/wiki/Main_Page">http://sqlzoo.net/wiki/Main_Page</a>)</li> <li>VideoCenter-Datenbank</li> </ul>



<p>der Fragestellungen (JOIN, UNION, AS, GROUP BY, ORDER BY, ASC, DESC, COUNT, MAX, MIN, SUM, Arithmetische Operatoren: +, -, *, /, (...), Vergleichsoperatoren: =, &lt;&gt;, &gt;, &lt;, &gt;=, &lt;=, LIKE, BETWEEN, IN, IS NULL)</p> <ul style="list-style-type: none"> <li>• Vertiefung an einem weiteren Datenbankbeispiel: Vertiefungen am Beispiel der Relationenalgebra</li> </ul>	<ul style="list-style-type: none"> <li>• erläutern Eigenschaften und Aufbau von Datenbanksystemen unter dem Aspekt der sicheren Nutzung (A),</li> </ul>	
<p>3. Gesellschaftliche Auswirkungen der Nutzung von Datenbanksystemen</p>	<ul style="list-style-type: none"> <li>• untersuchen und bewerten anhand von Fallbeispielen Auswirkungen des Einsatzes von Informatiksystemen sowie Aspekte der Sicherheit von Informatiksystemen, des Datenschutzes und des Urheberrechts (A),</li> <li>• 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).</li> </ul>	<ul style="list-style-type: none"> <li>• Fallbeispiele zur Nutzung von Datenbanksystemen</li> <li>• Spiel zum Missbrauch von Daten: DataDealer (<a href="http://datadealer.com/de">http://datadealer.com/de</a>)</li> </ul>

## **Unterrichtsvorhaben Q2-II**

Thema: Endliche Automaten und Formale Sprachen

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

Zeitbedarf: 25/40 Stunden

Berufsorientierung:

- Softwareentwickler\*in
- Wissenschaftspropädeutik

Absprachen zur vorhabenbezogene Konkretisierung:

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

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

Das Grammatikmodell der regulären Grammatiken wird auf das Modell der kontextfreien Grammatiken erweitert und die Auswirkungen auf das entsprechende Automatenmodell der Kellerautomaten veranschaulicht. Die Unzulänglichkeit der Kellerautomaten und kontextfreien Grammatiken wird an Beispielen verdeutlicht.

Ausgehend von einer einfachen formalen Sprache (z. B. reduziertes Java) werden die Bestandteile eines Compilers dargestellt:

Der Scanner eines Compilers wird in Form eines endlichen Automaten modelliert und implementiert. Die Begriffe der Symboltabelle und der Tokenliste werden inhaltlich gefüllt. Der Sprachumfang der einfachen formalen Sprache wird leicht erweitert und die Auswirkungen auf den Automaten und die Implementierung wird beobachtet.

Der Parser eines Compilers wird in Form einer kontextfreien Grammatik modelliert und implementiert. Der Sprachumfang der einfachen formalen Sprache wird um weitere Regeln ergänzt und der Parser wird angepasst.

Bei Bedarf wird ein Übersetzer-Modul entwickelt, welches die einfache formale Sprache in eine Sprachebene übersetzt (z. B. interpretiert und compiliert).

Lernmittel / Materialien:

- Arbeitsblätter und Demonstrationsprogramme in der moodle-Plattform
- Automaten Simulationsprogramm (z. B. JFlap)

Unterrichtssequenzen	zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Einführung in Automaten/Grammatiken</p> <ul style="list-style-type: none"> <li>Grammatiken: Grammatik einer natürlichen Sprache Grammatik einer künstlichen Sprache Idee des Parsens</li> <li>Automaten: erkennender Automat zu Symbolen einer Sprache Modell des endlichen Automaten Darstellungsformen Sprache eines Automaten als regulärer Ausdruck nichtdeterministische Automaten</li> </ul>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> <li>analysieren und erläutern die Eigenschaften endlicher Automaten <b>und Kellerautomaten</b> einschließlich ihres Verhaltens bei bestimmten Eingaben (A),</li> <li>ermitteln die Sprache, die ein endlicher Automat <b>oder ein Kellerautomat</b> akzeptiert (D),</li> <li>entwickeln und modifizieren zu einer Problemstellung endliche Automaten <b>oder Kellerautomaten</b> (M),</li> <li>stellen endliche Automaten in Tabellen oder Graphen dar und überführen sie in die jeweils andere Darstellungsform (D),</li> <li>entwickeln zur Grammatik einer regulären <b>oder kontextfreien</b> Sprache einen zugehörigen endlichen Automaten <b>oder einen Kellerautomaten</b> (M).</li> </ul>	<p>Mögliche Einstiege:</p> <ul style="list-style-type: none"> <li>Grundidee eines Compilers: Scanner, Parser, Codierer</li> </ul>
<p>2. Zusammenhang zwischen endlichen Automaten und regulären Grammatiken</p> <ul style="list-style-type: none"> <li>reguläre Grammatik: Definition Anwendungen</li> <li>Zusammenhang zu endlichen Automaten</li> <li>Grenzen der endlichen Automaten/regulären Grammatiken</li> </ul>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> <li>analysieren und erläutern Grammatiken regulärer <b>und kontextfreier</b> Sprachen (A),</li> <li>modifizieren Grammatiken regulärer <b>und kontextfreier</b> Sprachen (M),</li> <li>ermitteln die formale Sprache, die durch eine Grammatik erzeugt wird (A),</li> <li>entwickeln zu einer regulären <b>oder kontextfreien</b> Sprache eine Grammatik, die die Sprache erzeugt (M),</li> <li>entwickeln zur akzeptierten Sprache eines Automaten eine zugehörige Grammatik (M),</li> <li>beschreiben an Beispielen den Zusammenhang zwischen Automaten und Grammatiken (D),</li> <li><b>zeigen/erläutern</b> die Grenzen endlicher Automaten und regulärer <b>Grammatiken/Sprachen</b> im Anwendungszusammenhang auf (A).</li> </ul>	<p>Arbeitsblätter in der moodle-Plattform Orientierungsrahmen: Chomsky</p>
<p>3. Zusammenhang zwischen Kellerautomaten und kontextfreien Grammatiken</p> <ul style="list-style-type: none"> <li>kontextfreie Grammatik: Definition Anwendungen</li> </ul>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> <li>analysieren und erläutern die Eigenschaften endlicher Automaten und Kellerautomaten einschließlich ihres Verhaltens bei bestimmten Eingaben (A),</li> </ul>	<p>Arbeitsblätter in der moodle-Plattform Orientierungsrahmen: Chomsky</p>

<ul style="list-style-type: none"> <li>• Modell des Kellerautomaten Definition Darstellungsfomen Anwendungen / Sprache eines Kellerautomaten</li> <li>• Zusammenhang zwischen Kellerautomaten/kontextfreien Grammatiken</li> <li>• Grenzen der Kellerautomaten</li> </ul>	<ul style="list-style-type: none"> <li>• ermitteln die Sprache, die ein endlicher Automat oder ein Kellerautomat akzeptiert (D),</li> <li>• entwickeln und modifizieren zu einer Problemstellung endliche Automaten oder Kellerautomaten (M),</li> <li>• entwickeln zur Grammatik einer regulären oder kontextfreien Sprache einen zugehörigen endlichen Automaten oder einen Kellerautomaten (M),</li> <li>• analysieren und erläutern Grammatiken regulärer und kontextfreier Sprachen (A),</li> <li>• modifizieren Grammatiken regulärer und kontextfreier Sprachen (M),</li> <li>• entwickeln zu einer regulären oder kontextfreien Sprache eine Grammatik, die die Sprache erzeugt (M),</li> <li>• beschreiben an Beispielen den Zusammenhang zwischen Automaten und Grammatiken (D).</li> </ul>	
<p>4. Die Schritte eines Compilers</p> <ul style="list-style-type: none"> <li>• Scanner: endlicher Automat als Grundlage Symboltabelle und Tokenliste zur Verwaltung und Erkennung des Quelltextes Erweiterung des terminalen Alphabets der zu compilierenden formalen Sprache Implementierung als endlicher Automat</li> <li>• Parser: kontextfreie Grammatik als Grundlage Erweiterung des Sprachumfangs Implementierung als kontextfreie Grammatik</li> </ul>	<ul style="list-style-type: none"> <li>• modellieren und implementieren Scanner, Parser und Interpreter zu einer gegebenen regulären Sprache (I).</li> </ul>	<p>Arbeitsblätter und Skript zum Compilerbau in der moodle-Plattform</p>

## **Unterrichtsvorhaben Q2-III**

Thema: Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit

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

Zeitbedarf: 10 Stunden

*Berufsorientierung:*

- Softwareentwickler\*in
- Wissenschaftspropädeutik

Absprachen zur vorhabenbezogene Konkretisierung:

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

Ausgehend von den prinzipiellen Grenzen endlicher Automaten liegt die Frage nach den Grenzen von Computern bzw. nach Grenzen der Automatisierbarkeit nahe. Mit Hilfe einer entsprechenden Java-Methode wird plausibel, dass es unmöglich ist, ein Informatiksystem zu entwickeln, 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.

Lernmittel / Materialien:

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

Unterrichtssequenzen	zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Von-Neumann-Architektur und die Ausführung maschinennaher Programme:</p> <ul style="list-style-type: none"> <li>• prinzipieller Aufbau einer von Neumann-Architektur mit CPU, Rechenwerk, Steuerwerk, Register und Hauptspeicher</li> <li>• einige maschinennahe Befehle und ihre Repräsentation in einem Binär-Code, der in einem Register gespeichert werden kann</li> <li>• Analyse und Erläuterung der Funktionsweise eines einfachen maschinennahen Programms</li> </ul>	<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>geeigneter Modellrechner</p> <ul style="list-style-type: none"> <li>• Johnny</li> <li>• WinAli</li> <li>• Microprocessor Simulator</li> </ul>
<p>2. Grenzen der Automatisierbarkeit:</p> <ul style="list-style-type: none"> <li>• Vorstellung des Halteproblems</li> <li>• Unlösbarkeit des Halteproblems</li> <li>• Beurteilung des Einsatzes von Informatiksystemen hinsichtlich prinzipieller Möglichkeiten und prinzipieller Grenzen</li> </ul>		<p>Arbeitsblätter von der Schulhomepage zum Halteproblem Mögliche Verknüpfung zu Unterrichtsvorhaben Q1-1a (Backtracking)</p>

## **Unterrichtsvorhaben Q2-IV**

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

Leitfragen: Bei welchen Problemstellungen reichen die bekannten Datenstrukturen nicht aus? Welche Möglichkeiten gibt es, flexibel miteinander verknüpfte Informationen zu verwalten? Wie hängen die Datenstrukturen Graph, Baum und Liste zusammen?

Zeitbedarf: 15 Stunden

*Berufsorientierung:*

- Softwareentwickler\*in
- Wissenschaftspropädeutik

Abstrachen zur vorhabenbezogene Konkretisierung:

Nach Analyse einer Problemstellung in einem geeigneten Anwendungskontext (z. B. das Eulerkreisproblem), in dem Daten in Form eines Graphen verwaltet werden, werden der Aufbau und die Darstellungsformen von Graphen am Beispiel dargestellt und ausgewählte Problemstellungen exemplarisch analysiert.

Die Operationen der Klasse Graph werden erläutert und im Anwendungszusammenhang bei der Lösung grundlegender Probleme (wie z.B. der Traversierung eines Graphen) genutzt.

Lernmittel / Materialien:

- Arbeitsblätter in der moodle-Plattform
- eine didaktische Entwicklungsumgebung (z. B. Java-Editor o. a.)

Unterrichtssequenzen	zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
<p>1. Die Datenstruktur des Graphen im Anwendungskontext</p> <ul style="list-style-type: none"> <li>• Grundbegriffe (Knoten, Kanten und anderes)</li> <li>• Darstellungsformen eines Graphen</li> </ul>	<p>Die Schülerinnen und Schüler</p> <ul style="list-style-type: none"> <li>• ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M),</li> <li>• stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D),</li> <li>• modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M),</li> <li>• modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M),</li> <li>• ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M),</li> <li>• verwenden bei der Modellierung geeigneter Problemstellungen Möglichkeiten der Polymorphie (M),</li> <li>• ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M),</li> <li>• stellen die Kommunikation zwischen Objekten grafisch dar (D),</li> <li>• stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D),</li> <li>• dokumentieren Klassen (D),</li> <li>• analysieren und erläutern objektorientierte Modellierungen (A),</li> <li>• implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I).</li> <li>• analysieren und erläutern Algorithmen und Programme (A),</li> <li>• modifizieren Algorithmen und Programme (I),</li> </ul>	<p>Eulerkreisproblem:</p> <ul style="list-style-type: none"> <li>• Arbeitsblätter zum Eulerweg / Eulerkreis</li> <li>• Visualisierungsprogramm für Graphen</li> </ul>



	<ul style="list-style-type: none"> <li>• stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D),</li> <li>• entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“ und „Teilen und Herrschen“ und „Backtracking“ (M),</li> <li>• implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I),</li> <li>• testen Programme systematisch anhand von Beispielen und mit Hilfe von Testanwendungen (I).</li> <li>• erläutern Operationen dynamischer (linearer oder/und nicht-linearer) Datenstrukturen (A),</li> </ul>	
<p>2. Die Datenstruktur des Graphen im Anwendungskontext unter Nutzung der Klasse Graph</p> <ul style="list-style-type: none"> <li>• Modellierung von Anwendungssituationen als Graph</li> <li>• Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen</li> <li>• Erarbeitung der Funktionalität der Klasse Graph</li> <li>• Modellierung und Implementierung verschiedener Problemstellungen unter Verwendung der Klasse Graph.</li> </ul>	<ul style="list-style-type: none"> <li>•</li> </ul>	<p>Projektarbeit in einem größeren Kontext:</p> <ul style="list-style-type: none"> <li>• Navigationssystem</li> <li>• Weiterführung des Eulerkreisproblems</li> </ul> <p>Arbeitsblätter in der moodle-Plattform zu:</p> <ul style="list-style-type: none"> <li>• Tiefensuche / Breitensuche</li> <li>• kürzeste Wege mit Dijkstra</li> </ul> <p>mögliche Vertiefungen:</p> <ul style="list-style-type: none"> <li>• Traveling Salesman Problem</li> <li>• minimale Spannbäume</li> </ul>

## 2.2 Grundsätze der Leistungsbewertung und Leistungsrückmeldung

Unter pädagogischen Gesichtspunkten hat die Leistungsbewertung vornehmlich das Individuum im Blick. Hier soll sie über den Leistungszuwachs rückmelden und dadurch die Motivation für weitere Anstrengungen erhöhen. Sie ermöglicht den Schülerinnen und Schülern ihre noch vorhandenen fachlichen Defizite wie auch ihre Stärken und Fähigkeiten zu erkennen um dadurch ein realistisches Selbstbild aufzubauen. Sie ist Basis für gezielte individuelle Förderung.

Die Fachkonferenz hat auf Grundlage von §48 SchulG sowie Kapitel 3 des Kernlehrplans Informatik im Einklang mit dem entsprechenden schulbezogenen Konzept die nachfolgenden, verbindlichen Grundsätze zur Leistungsbewertung und -rückmeldung beschlossen. Es wird **keine** schriftliche Leistungsbewertung in Form von Klassenarbeiten erhoben. Im Bereich der **sonstigen** Leistungen können ungeachtet dessen schriftliche Überprüfungen (Tests) durchgeführt werden.

### **Grundsätze der Leistungsbewertung**

Die Fachkonferenz Informatik legt die Kriterien für die Leistungsbewertung fest. Die Lehrerinnen und Lehrer machen diese Kriterien den Schülerinnen und Schülern transparent.

Es gelten folgende Grundsätze der Leistungsbewertung:

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

### **I. Beurteilungsbereich „Schriftliche Leistungsüberprüfung“**

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

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

Am Helmholtz-Gymnasium Bonn werden die Kursarbeiten bzw. Klausuren in der Regel nach einem vorab festgelegten Punkteschema bewertet. Dabei ist eine glatt ausreichende Leistung bei 45% der Punktzahl erreicht worden. Die übrigen Notenstufen ergeben sich dann dadurch, dass für jede Notenstufe Intervalle der erreichten Punkte gebildet werden, die in der Regel gleich groß sind:

ab %	Punkte
0,00%	0
20,00%	1
26,67%	2
33,33%	3
40,00%	4
45,00%	5
50,00%	6
55,00%	7
60,00%	8
65,00%	9
70,00%	10
75,00%	11
80,00%	12
85,00%	13
90,00%	14
95,00%	15

In der Sekundarstufe II wird spätestens in der Abiturvorklausur die im Zentralabitur gemäß unten aufgeführter Tabelle vorgegebene Zuordnung der erreichten Punkte (maximale Punktzahl: 100 im GK, 150 im LK) zur Note als Grundlage der Notenfindung genutzt.

Grundkurs (100 Pkt.)		Leistungskurs (150 Pkt.)	
Punkte	Note	Punkte	Note
0-19	6	0-29	6
20-26	5-	30-39	5-
27-32	5	40-48	5
33-38	5+	49-57	5+
39-44	4-	58-67	4-
45-49	4	68-74	4
50-54	4+	75-82	4+
55-59	3-	83-89	3-
60-64	3	90-97	3
65-69	3+	98-104	3+
70-74	2-	105-112	2-
75-79	2	113-119	2
80-84	2+	120-127	2+
85-89	1-	128-134	1-
90-94	1	135-142	1
95-100	1+	143-150	1+

Die Fachkonferenz legt die Dauer der Kursarbeiten und Klausuren fest. Am Helmholtz-Gymnasium bzw. im Rahmen der Kooperation mit dem Ernst-Moritz-Arndt-Gymnasium (Leistungskurs Informatik) gelten für die Sekundarstufe II folgende Regelungen:

Klasse	1. Klausur, 1. HJ	2. Klausur 1. HJ	1. Klausur 2. HJ	2. Klausur, 2. HJ
EF	---	90 min	90 min	90 min
Q1 GK	90 min	90 min	90 min	90 min
Q1 LK	135 min	135 min	135 min	135 min
Q2 GK	135 min	135 min	135 min	---
Q2 LK	135 min	180 min	255 min	---

In der Qualifikationsphase I kann die erste Klausur im 2. Halbjahr durch eine Facharbeit ersetzt werden.

## **II. Beurteilungsbereich „Sonstige Leistungen“:**

Den Schülerinnen und Schülern werden die Kriterien zum Bewertungsbereich sonstige Leistungen zu Beginn des Schuljahres genannt.

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

- Beteiligung am Unterrichtsgespräch
- Zusammenfassungen zur Vor- und Nachbereitung des Unterrichts
- Präsentation von Arbeitsergebnissen
- Mitarbeit in Partner- und Gruppenarbeitsphasen
- Schriftliche Bearbeitung von Aufgaben im Unterricht
- Praktische Leistungen am Computer als Werkzeug im Unterricht
- Kürzere Projektarbeiten
- Lernerfolgsüberprüfungen und schriftliche Übungen

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

## **III. Bewertungskriterien**

Die Bewertungskriterien für eine Leistung müssen auch für Schülerinnen und Schüler **transparent, klar** und **nachvollziehbar** sein. Die folgenden allgemeinen Kriterien gelten sowohl für die schriftlichen als auch für die sonstigen Formen der Leistungsüberprüfung:

- Qualität der Beiträge
- Kontinuität der Beiträge
- Sachliche Richtigkeit
- Angemessene Verwendung der Fachsprache
- Darstellungskompetenz
- Komplexität/Grad der Abstraktion
- Selbstständigkeit im Arbeitsprozess
- Einhaltung gesetzter Fristen
- Präzision
- Differenziertheit der Reflexion
- Bei Gruppenarbeiten
  - Einbringen in die Arbeit der Gruppe
  - Durchführung fachlicher Arbeitsanteile
- Bei Projekten
  - Selbstständige Themenfindung
  - Dokumentation des Arbeitsprozesses
  - Grad der Selbstständigkeit
  - Qualität des Produktes
  - Reflexion des eigenen Handelns
  - Kooperation mit dem Lehrenden / Aufnahme von Beratung

## **IV. Grundsätze der Leistungsrückmeldung und Beratung**

Die Leistungsrückmeldung findet in mündlicher oder schriftlicher Form statt. Sie kann auch an Eltern- und/oder Schülersprechtagen oder in Form von individuellen Lern-/Förderempfehlungen erfolgen.

### ***V. Bildung der Zeugnisnote***

In die Note gehen alle im Unterricht erbrachten Leistungen ein. Bei der Notenfindung ist die individuelle Lernentwicklung der Schülerinnen und Schüler angemessen zu berücksichtigen.