# Schulinterner Lehrplan INFORMATIK

## zum Kernlehrplan für die gymnasiale Oberstufe am Städtischen Gymnasium Rheinbach

**Gültig für die Einführungsphasen ab dem Schuljahr 2014/2015 und die Qualifikationsphasen, di ab dem Schuljahr 2015/2016 einsetzen nach Beschluss der Fachkonferenz Informatik vom 21.09.2015. Letzte Änderungen durch die Fachkonferenz am 30.09.2019**

# Informatik

## Inhalt

# 1 Die Fachgruppe Informatik des Städtischen Gymnasium

Beim Städtischen Gymnasium Rheinbach handelt es sich um eine vierzügige Schule in Rheinbach mit zurzeit ca. 980 Schülerinnen und Schülern, ca. 85 Lehrerinnen und Lehrern. Das Einzugsgebiet der Schule umfasst die Stadt Rheinbach sowie umliegender Städte und Gemeinden was zum Teil auf das Angebot der Schule im Fach Informatik zurückzuführen ist.

Das Fach Informatik wird am Städtischen Gymnasium ab der Jahrgangsstufe 8 im Wahlpflichtbereich II (WP II) als Fächerkombination mit Mathematik und Physik zweistündig unterrichtet und von etwa der Hälfte der Schülerinnen und Schüler besucht. In der zweijährigen Laufzeit dieser Kurse wird in altersstufengerechter Weise unter anderem auf Grundlagen der Algorithmik am Beispiel einer didaktischen Lernumgebung, auf die technische Informatik am Beispiel von Schaltwerken und Schaltnetzen und auf Kryptologie und Webdesign eingegangen. Der Unterricht erfolgt dabei in enger Verzahnung mit Inhalten der Mathematik und Physik und wird zum Teil in Form von fächerverbindenden Projekten und in Kooperation mit außerschulischen Partnern gestaltet.

Organisatorisch ist das Fach Informatik in der Sekundarstufe I in den MINT-Zweig der Schule eingebunden, den Schülerinnen und Schüler als Alternative zu einem bilingualen Zweig anwählen können.

In der Sekundarstufe II bietet das Städtische Gymnasium Rheinbach für die eigenen Schülerinnen und Schüler in allen Jahrgangsstufen Grundkurse und Leistungskurse an in Informatik an. Im Schuljahr 2014/2015 ist nach langen Jahren erstmals wieder ein Leistungskurs zustande gekommen. Auch im Schuljahr 2015/2016 wurde ein Leistungskurs mit 20 Schüler/innen eingerichtet.

Um insbesondere Schülerinnen und Schülern gerecht zu werden, die in der Sekundarstufe I keinen Informatikunterricht besucht haben, wird in Kursen der Einführungsphase besonderer Wert darauf gelegt, dass keine Vorkenntnisse aus der Sekundarstufe I zum erfolgreichen Durchlaufen des Kurses erforderlich sind.

Der Unterricht der Sekundarstufe II wird mit Hilfe der Programmiersprache Java durchgeführt. In der Einführungsphase kommt dabei zusätzlich eine didaktische Bibliothek zum Einsatz, welche das Erstellen von grafischen Programmen erleichtert.

Durch projektartiges Vorgehen, offene Aufgaben und Möglichkeiten, Problemlösungen zu verfeinern oder zu optimieren, entspricht der Informatikunterricht der Oberstufe in besonderem Maße den Erziehungszielen, Leistungsbereitschaft zu fördern, ohne zu überfordern.

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 stellen einen wichtigen Beitrag zur Qualitätssicherung und -entwicklung des Unterrichts dar.

Zurzeit besteht die Fachschaft Informatik des Städtischen Gymnasium Rheinbach aus einer Lehrkraft, denen zwei Computerräume mit 15 bzw. 17 Computerarbeitsplätzen und ein Wagen mit 30 Laptops zur Verfügung stehen. Alle Arbeitsplätze sind an das schulinterne Rechnernetz angeschlossen, so dass Schülerinnen und Schüler über einen individuell gestaltbaren Zugang zum zentralen Server der Schule alle Arbeitsplätze der drei Räume zum Zugriff auf ihre eigenen Daten, zur Recherche im Internet oder zur Bearbeitung schulischer Aufgaben verwenden können.

Der Unterricht erfolgt im 45-Minuten-Takt. Die Kursblockung sieht grundsätzlich für Grundkurse eine Doppelstunde und eine Einzelstunde und für Leistungskurse zwei

Doppel- und eine Einzelstunde vor. Im Differenzierungsbereich der S1 findet der Unterricht in jeweils einer Doppelstunde statt.

# 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.

Die entsprechende Umsetzung erfolgt auf zwei Ebenen: der Übersichts- und der Konkretisierungsebene.

Im „Übersichtsraster Unterrichtsvorhaben“ (Kapitel 2.1.1) 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 Zuordnung 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 Freiraum für Vertiefungen, besondere Schülerinteressen, aktuelle Themen bzw. die Erfordernisse anderer besonderer Ereignisse (z.B. Praktika, Kursfahrten o.ä.) zu erhalten, wurden im Rahmen dieses schulinternen Lehrplans ca. 75 Prozent der Bruttounterrichtszeit verplant.

Während der Fachkonferenzbeschluss zum „Übersichtsraster Unterrichtsvorhaben“ zur Gewährleistung vergleichbarer Standards sowie zur Absicherung von Lerngruppenübertritten und Lehrkraftwechseln für alle Mitglieder der Fachkonferenz Bindekraft entfalten soll, beinhaltet die Ausweisung „konkretisierter Unterrichtsvorhaben“ (Kapitel 2.1.2) Beispiele und Materialien, die empfehlenden Charakter haben. Referendarinnen und Referendaren sowie neuen Kolleginnen und Kollegen dienen diese vor allem zur standardbezogenen Orientierung in der neuen Schule, aber auch zur Verdeutlichung von unterrichtsbezogenen fachgruppeninternen Absprachen zu didaktisch-methodischen Zugängen, fächerübergreifenden Kooperationen, Lernmitteln und -orten sowie vorgesehenen Leistungsüberprüfungen, die im Einzelnen auch den Kapiteln 2.2 bis 2.3 zu entnehmen sind.

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

**Es liegt grundsätzlich im Ermessen der unterrichtenden Lehrkraft, die Reihenfolge der Unterrichtsvorhaben zu ändern. Ebenfalls liegt es im Ermessen der jeweils unterrichtenden Lehrkraft andere Problemsituationen zu thematisieren und andere Schwerpunkte zu setzen, welche gleichwertig die zu entwickelnden Kompetenzen abdecken.**

## 2.1.1 Übersichtsraster Unterrichtsvorhaben

### 2.1.1.1 Einführungsphase

<table>
<tr><th colspan="2">Einführungsphase</th></tr>
<tr>
<td><u>Unterrichtsvorhaben E-I</u><br><br>Thema:<br>Die digitalisierte Gesellschaft, Bedeutung, Verbreitung und Gefahren der Nutzung von Informatiksystemen<br><br>Zentrale Kompetenzen:<br>• Argumentieren<br>• Darstellen und Interpretieren<br>• Kommunizieren und Kooperieren<br><br>Inhaltsfelder:<br>• Informatiksysteme<br>• Informatik, Mensch und Gesellschaft<br><br>Inhaltliche Schwerpunkte:<br>• Einzelrechner<br>• Digitalisierung<br>• Dateisystem<br>• Internet<br>• Einsatz von Informatiksystemen<br><br>Zeitbedarf: 9 Stunden</td>
<td><u>Unterrichtsvorhaben E-II</u><br><br>Thema:<br>Grundlagen der objektorientierten Analyse, Modellierung und Implementierung anhand von statischen Grafikszenen<br><br>Zentrale Kompetenzen:<br>• Modellieren<br>• Implementieren<br>• Darstellen und Interpretieren<br>• Kommunizieren und Kooperieren<br><br>Inhaltsfelder:<br>• Daten und ihre Strukturierung<br>• Formale Sprachen und Automaten<br><br>Inhaltliche Schwerpunkte:<br>• Objekte und Klassen<br>• Syntax und Semantik einer Programmiersprache<br><br>Zeitbedarf: 8 Stunden</td>
</tr>
</table>

| Einführungsphase | |
|---|---|
| *Unterrichtsvorhaben E-III*<br><br>**Thema**:<br>*Grundlagen der objektorientierten Programmierung und algorithmischer Grundstrukturen in Java anhand von einfachen Animationen*<br><br>**Zentrale Kompetenzen:**<br>• Argumentieren<br>• Modellieren<br>• Implementieren<br>• Kommunizieren und Kooperieren<br><br>**Inhaltsfelder:**<br>• Daten und ihre Strukturierung<br>• Algorithmen<br>• Formale Sprachen und Automaten<br><br>**Inhaltliche Schwerpunkte:**<br>• Objekte und Klassen<br>• Syntax und Semantik einer Programmiersprache<br>• Analyse, Entwurf und Implementierung einfacher Algorithmen<br><br>**Zeitbedarf**: 18 Stunden | *Unterrichtsvorhaben E-IV*<br><br>**Thema**:<br>*Modellierung und Implementierung von Klassen- und Objektbeziehungen anhand von Spielen und Simulationen*<br><br>**Zentrale Kompetenzen:**<br>• Argumentieren<br>• Modellieren<br>• Implementieren<br>• Darstellen und Interpretieren<br>• Kommunizieren und Kooperieren<br><br>**Inhaltsfelder:**<br>• Daten und ihre Strukturierung<br>• Algorithmen<br>• Formale Sprachen und Automaten<br><br>**Inhaltliche Schwerpunkte:**<br>• Objekte und Klassen<br>• Syntax und Semantik einer Programmiersprache<br>• Analyse, Entwurf und Implementierung einfacher Algorithmen<br><br>**Zeitbedarf**: 18 Stunden |

<table>
<tr><th colspan="2">Einführungsphase</th></tr>
<tr><td><u>Unterrichtsvorhaben E-V</u><br><br>Thema:<br>Umgang mit Textdateien, Suchen in Texten, Entwurf einfacher GUIs<br><br>Zentrale Kompetenzen:<br>• Argumentieren<br>• Modellieren<br>• Implementieren<br>• Darstellen und Interpretieren<br>• Kommunizieren und Kooperieren<br><br>Inhaltsfelder:<br>• Daten und ihre Strukturierung<br>• Algorithmen<br>• Formale Sprachen und Automaten<br>• Informatik, Mensch und Gesellschaft<br><br>Inhaltliche Schwerpunkte:<br>• Objekte und Klassen<br>• Algorithmen in ausgewählten informatischen Kontexten<br>• Einsatz von Informatiksystemen<br>• Sicherheit<br><br>Zeitbedarf: 16 Stunden</td><td><u>Unterrichtsvorhaben E-VI</u><br><br>Thema:<br>Analyse-, Such- und Sortieralgorithmen mit quadratischer Laufzeit anhand einer kontextbezogenen Problemstellung (z.B. Lottosimulation) und deren Modellierung und Implementierung<br><br>Zentrale Kompetenzen:<br>• Argumentieren<br>• Modellieren<br>• Implementieren<br>• Darstellen und Interpretieren<br>• Kommunizieren und Kooperieren<br><br>Inhaltsfelder:<br>• Daten und ihre Strukturierung<br>• Algorithmen<br><br>Inhaltliche Schwerpunkte:<br>• Objekte und Klassen<br>• Algorithmen zum Suchen und Sortieren<br>• Analyse, Entwurf und Implementierung einfacher Algorithmen<br><br>Zeitbedarf: 12 Stunden</td></tr>
<tr><td><u>Unterrichtsvorhaben EF-VII</u><br><br>Thema: Informatiksysteme, Datencodierung, Datenspeicherung im Computersystem (im Einzelrechner).<br><br>Zentrale Kompetenzen:<br>• Präsentieren<br>• Implementieren<br>• Darstellen und Interpretieren<br>• Kommunizieren und Kooperieren<br>• Analysieren<br><br>Inhaltsfelder:<br>• Funktionsweise eines Rechners</td><td></td></tr>
</table>

| | |
|---|---|
| • Codierung und Darstellung von Daten<br><br>**Inhaltliche Schwerpunkte:**<br>• Von Neumann-Rechner (Johnny)<br>• Binärcodierung<br>• Verarbeitung von Daten in einem Rechenwerk<br><br>**Zeitbedarf**: 8 Stunden | |
| **Summe Einführungsphase: 89** | |

**Fakultatives Unterrichtsvorhaben: Exkursion ins HNF nach Paderborn (s.u.)**

### 2.1.1.2 Qualifikationsphase

In ROT werden die zusätzlichen Unterrichtsinhalte für den Leistungskurs dargestellt.

<table>
<tr><th colspan="2">Qualifikationsphase 1</th></tr>
<tr>
<td>

*Unterrichtsvorhaben Q1-I*

**Thema:**
*Wiederholung der objektorientierten Modellierung und Programmierung anhand einer kontextbezogenen Problemstellung und Vertiefung der OOM mit Implementations- und Entwurfsdiagrammen*

**Zentrale Kompetenzen:**
- Argumentieren
- Modellieren
- Implementieren
- Darstellen und Interpretieren
- Kommunizieren und Kooperieren

**Inhaltsfelder:**
- Daten und ihre Strukturierung
- Algorithmen
- Formale Sprachen und Automaten
- Informatiksysteme

**Inhaltliche Schwerpunkte:**
- Objekte und Klassen
- Analyse, Entwurf und Implementierung von Algorithmen
- Syntax und Semantik einer Programmiersprache
- Nutzung von Informatiksystemen

**Zeitbedarf**: 10 Stunden

</td>
<td>

*Unterrichtsvorhaben Q1-II*

**Thema:**
*Rekursive Algorithmen und Backtracking in Anwendungskontexten*

**Zentrale Kompetenzen:**
- Argumentieren
- Modellieren
- Implementieren
- Darstellen und Interpretieren
- Kommunizieren und Kooperieren

**Inhaltsfelder:**
- Algorithmen
- Formale Sprachen und Automaten
- Informatiksysteme
- Informatik, Mensch und Gesellschaft

**Inhaltliche Schwerpunkte:**
- Analyse, Entwurf und Implementierung von Algorithmen
- Algorithmen in ausgewählten informatischen Kontexten
- Syntax und Semantik einer Programmiersprache
- Nutzung von Informatiksystemen
- Grenzen der Automatisierung

**Zeitbedarf:** 20/25 Stunden

</td>
</tr>
</table>

| Qualifikationsphase 1 | |
|---|---|
| *Unterrichtsvorhaben Q1-III*<br><br>**Thema:**<br>*Modellierung und Implementierung dynamische Listenstrukturen und deren Anwendungen*<br><br>**Zentrale Kompetenzen:**<br>• Argumentieren<br>• Modellieren<br>• Implementieren<br>• Darstellen und Interpretieren<br>• Kommunizieren und Kooperieren<br><br>**Inhaltsfelder:**<br>• Daten und ihre Strukturierung<br>• Algorithmen<br>• Formale Sprachen und Automaten<br>• Informatiksysteme<br><br>**Inhaltliche Schwerpunkte:**<br>• Objekte und Klassen<br>• Analyse, Entwurf und Implementierung von Algorithmen<br>• Algorithmen in ausgewählten informatischen Kontexten<br>• Syntax und Semantik einer Programmiersprache<br>• Nutzung von Informatiksystemen<br><br>**Zeitbedarf:** 25/30 Stunden | *Unterrichtsvorhaben Q1-IVa*<br><br>**Thema:**<br>*Modellierung und Implementierung dynamische nichtlineare Datenstrukturen am Beispiel der Binärbäume*<br><br>**Zentrale Kompetenzen:**<br>• Argumentieren<br>• Modellieren<br>• Implementieren<br>• Darstellen und Interpretieren<br>• Kommunizieren und Kooperieren<br><br>**Inhaltsfelder:**<br>• Daten und ihre Strukturierung<br>• Algorithmen<br>• Formale Sprachen und Automaten<br>• Informatiksysteme<br><br>**Inhaltliche Schwerpunkte:**<br>• Objekte und Klassen<br>• Analyse, Entwurf und Implementierung von Algorithmen<br>• Algorithmen in ausgewählten informatischen Kontexten<br>• Syntax und Semantik einer Programmiersprache<br>• Nutzung von Informatiksystemen<br><br>**Zeitbedarf:** 20 Stunden |

| Qualifikationsphase 1 | |
|---|---|
| *Unterrichtsvorhaben Q1-IVb*<br><br>**Thema:**<br>*Modellierung und Implementierung dynamische nichtlineare Datenstrukturen am Beispiel der Graphen*<br><br>**Zentrale Kompetenzen:**<br>• Argumentieren<br>• Modellieren<br>• Implementieren<br>• Darstellen und Interpretieren<br>• Kommunizieren und Kooperieren<br><br>**Inhaltsfelder:**<br>• Daten und ihre Strukturierung<br>• Algorithmen<br>• Formale Sprachen und Automaten<br>• Informatiksysteme<br><br>**Inhaltliche Schwerpunkte:**<br>• Objekte und Klassen<br>• Analyse, Entwurf und Implementierung von Algorithmen<br>• Algorithmen in ausgewählten informatischen Kontexten<br><br>**Zeitbedarf:** 15 Stunden | *Unterrichtsvorhaben Q1-Va*<br><br>**Thema:**<br>*Sicherheit und Datenschutz in Netzstrukturen*<br><br>**Zentrale Kompetenzen:**<br>• Argumentieren<br>• Darstellen und Interpretieren<br>• Kommunizieren und Kooperieren<br><br>**Inhaltsfelder:**<br>• Informatiksysteme<br>• Informatik, Mensch und Gesellschaft<br><br>**Inhaltliche Schwerpunkte:**<br>• Einzelrechner und Rechnernetzwerke<br>• Sicherheit<br>• Nutzung von Informatiksystemen, Wirkungen der Automatisierung<br><br>**Zeitbedarf:** 15 Stunden |

| Qualifikationsphase 1 | |
|---|---|
| *Unterrichtsvorhaben Q1-Vb*<br><br>**Thema:**<br>*Modellierung und Implementierung von Client-Server-Anwendungen*<br><br>**Zentrale Kompetenzen:**<br>• Argumentieren<br>• Modellieren<br>• Implementieren<br>• Darstellen und Interpretieren<br>• Kommunizieren und Kooperieren<br><br>**Inhaltsfelder:**<br>• Daten und ihre Strukturierung<br>• Algorithmen<br>• Formale Sprachen und Automaten<br>• Informatiksysteme<br><br>**Inhaltliche Schwerpunkte:**<br>• Objekte und Klassen<br>• Analyse, Entwurf und Implementierung von Algorithmen<br>• Algorithmen in ausgewählten informatischen Kontexten<br>• Syntax und Semantik einer Programmiersprache<br>• Einzelrechner und Rechnernetzwerke<br>• Nutzung von Informatiksystemen<br><br>**Zeitbedarf:** 10 Stunden | |
| **Summe Qualifikationsphase 1: 90/105** | |

| Qualifikationsphase 2 | |
|---|---|
| *Unterrichtsvorhaben Q2-Ia*<br><br>**Thema:**<br>*Modellierung und Nutzung relationaler Datenbanken in Anwendungskontexten*<br><br>**Zentrale Kompetenzen:**<br>• Argumentieren<br>• Modellieren<br>• Implementieren<br>• Darstellen und Interpretieren<br>• Kommunizieren und Kooperieren<br><br>**Inhaltsfelder:**<br>• Daten und ihre Strukturierung<br>• Algorithmen<br>• Formale Sprache und Automaten<br>• Informatiksysteme<br>• Informatik, Mensch und Gesellschaft<br><br>**Inhaltliche Schwerpunkte:**<br>• Datenbanken<br>• Algorithmen in ausgewählten informatischen Kontexten<br>• Syntax und Semantik einer Programmiersprache<br>• Nutzung von Informatiksystemen<br>• Sicherheit<br>• Wirkung der Automatisierung<br><br>**Zeitbedarf:** 25/30 Stunden | *Unterrichtsvorhaben Q2-Ib*<br><br>**Thema:**<br>*Prädikative / Logische Programmiersprachen*<br><br>**Zentrale Kompetenzen:**<br>• Argumentieren<br>• Modellieren<br>• Implementieren<br>• Darstellen und Interpretieren<br>• Kommunizieren und Kooperieren<br><br>**Inhaltsfelder:**<br>• Daten und ihre Strukturierung<br>• Algorithmen<br>• Formale Sprache und Automaten<br><br>**Inhaltliche Schwerpunkte:**<br>• Datenbanken<br>• Analyse, Entwurf und Implementierung von Algorithmen<br>• Algorithmen in Ausgewählten Kontexten<br>• Syntax und Semantik einer Programmiersprache<br><br>**Zeitbedarf:** 15 Stunden |

| Qualifikationsphase 2 | |
|---|---|
| *Unterrichtsvorhaben Q2-IIa*<br><br>**Thema:**<br>*Endliche Automaten und Formale Sprachen*<br><br>**Zentrale Kompetenzen:**<br>• Argumentieren<br>• Modellieren<br>• Implementieren<br>• Darstellen und Interpretieren<br>• Kommunizieren und Kooperieren<br><br>**Inhaltsfelder:**<br>• Formale Sprachen und Automaten<br>• Informatiksysteme<br><br>**Inhaltliche Schwerpunkte:**<br>• Endliche Automaten (und Kellerautomaten)<br>• Grammatiken regulärer (und kontextfreier) Sprachen<br>• Möglichkeiten und Grenzen von Automaten und formalen Sprachen<br>• Nutzung von Informatiksystemen<br><br>**Zeitbedarf:** 25/30 Stunden | *Unterrichtsvorhaben Q2-IIb*<br><br>**Thema:**<br>*Schritte eines Compilers einer formalen Sprache*<br><br>**Zentrale Kompetenzen:**<br>• Argumentieren<br>• Modellieren<br>• Implementieren<br>• Darstellen und Interpretieren<br>• Kommunizieren und Kooperieren<br><br>**Inhaltsfelder:**<br>Daten und ihre Strukturierung<br>Algorithmen<br>Formale Sprachen und Automaten<br>Informatiksysteme<br><br>**Inhaltliche Schwerpunkte:**<br>Endliche Automaten<br>Grammatiken regulärer Sprachen<br>Scanner, Parser und Interpreter für eine reguläre Sprache<br>Nutzung von Informatiksystemen<br><br>**Zeitbedarf:** 15 Stunden |

| Qualifikationsphase 2 | |
|---|---|
| *Unterrichtsvorhaben Q2-III*<br><br>**Thema:**<br>*Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit*<br><br>**Zentrale Kompetenzen:**<br>• Argumentieren<br>• Kommunizieren und Kooperieren<br><br>**Inhaltsfelder:**<br>• Informatiksysteme<br>• Informatik, Mensch und Gesellschaft<br><br>**Inhaltliche Schwerpunkte:**<br>• Einzelrechner und Rechnernetzwerke<br>• Grenzen der Automatisierung<br><br>**Zeitbedarf:** 10 Stunden | |
| **Summe Qualifikationsphase 2: 50/100** | |

### 2.1.2 Konkretisierte Unterrichtsvorhaben

Im Folgenden sollen die im *Unterkapitel 2.1.1* aufgeführten Unterrichtsvorhaben konkretisiert werden.

In der Einführungsphase wird die didaktische Entwicklungsumgebung BlueJ sowie die Entwicklungsumgebung Netbeans verwendet. Die Freeware steht im Internet zum Download zur Verfügung. Die Quellen werden im Unterricht angegeben und sind ebenfalls auf der Informatik-Seite der SGR-Webpräsenz angegeben.

Darüber hinaus werden weitere, frei verfügbare Software-Werkzeuge eingesetzt. Auch hier werden die Quellen wie oben bekannt gegeben.

In der Qualifikationsphase werden die Unterrichtsvorhaben unter Berücksichtigung der Vorgaben für das Zentralabitur Informatik in NRW konkretisiert. Diese sind zu beziehen unter der Adresse

http://www.standardsicherung.schulministerium.nrw.de/abitur-gost/fach.php?fach=15
(abgerufen: 30. 04. 2014)

Darüber hinaus werden seitens des Ministeriums auf www.standardsicherung.nrw.de und den entsprechenden Unterseiten Materialien, Quelltexte, Bibliotheken u.v.a.m. bereitgestellt, die ebenfalls berücksichtigt werden.

Die in den Tabellen aufgeführten Medien, Beispiele und Materialien können durch gelichwertige ersetzt werden, wenn wichtige Gründe dies bedingen. Das können z.B. unterrichtsorganisatorische Gründe sein (Ausstattung der Schule, Zeit), aktuelle Gründe (Neuerungen in der Gesellschaft, aktuelle Nachrichten) oder didaktische Gründe (Lerngruppe, besseres Projekt wurde gefunden).

### 2.1.2.1 Einführungsphase

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

Die Schülerinnen und Schüler

- verwenden Fachausdrücke bei der Kommunikation über informatische Sachverhalte (K),
- präsentieren Arbeitsabläufe und -ergebnisse (K),
- kommunizieren und kooperieren in Gruppen und in Partnerarbeit (K),
- nutzen das verfügbare Informatiksystem zur strukturierten Verwaltung und gemeinsamen Verwendung von Daten unter Berücksichtigung der Rechteverwaltung (K).

**Unterrichtsvorhaben EF-I**

**Thema:** Die digitalisierte Gesellschaft, Bedeutung, Verbreitung und Gefahren der Nutzung von Informatiksystemen, Einzelrechner.

**Leitfragen:** *Womit beschäftigt sich die Wissenschaft der Informatik? Wie kann die in der Schule vorhandene informatische Ausstattung genutzt werden? Welche Bedeutung haben Informatiksysteme für unsere Gesellschaft und welche Gefahren sind damit verbunden? Wie können Daten kodiert werden? Wie verarbeitet ein Computer Daten?*

**Vorhabenbezogene Konkretisierung:**

Das erste Unterrichtsvorhaben stellt eine allgemeine Einführung in das Fach Informatik dar. Dabei ist zu berücksichtigen, dass für manche Schülerinnen und Schüler in der Einführungsphase der erste Kontakt mit dem Unterrichtsfach Informatik stattfindet, so dass zu Beginn Grundlagen des Fachs behandelt werden müssen.

Zunächst wird auf den Begriff der Information eingegangen und die Möglichkeit der Kodierung in Form von Daten thematisiert. Dabei soll der Binärcode von ganzen Zahlen betrachtet und interpretiert werden. Das Modell des Einzelrechners und die interne Datenverarbeitung in der CPU und im Speicher werden an die Binärcodierung gekoppelt. Anschließend wird auf die Übertragung von Daten im Sinne des Sender-Empfänger-Modells eingegangen. Dabei wird eine überblickartige Vorstellung der Kommunikation von Rechnern in Netzwerken erarbeitet.

Bei der Beschäftigung mit Datenkodierung, Datenübermittlung und Datenverarbeitung ist jeweils ein Bezug zur konkreten Nutzung der informatischen Ausstattung der Schule herzustellen. So wird in die verantwortungsvolle Nutzung dieser Systeme eingeführt.

Die mit der verbreiteten Nutzung von Informatiksystemen verbundenen Gefahren werden anhand eines Beispiels (Filmdokumentation) besprochen und daraus werden „Regeln für einen bewussteren und sichereren Umgang mit Informatiksystemen“ abgeleitet.

**Zeitbedarf**: 9 Stunden

| Unterrichtssequenzen | Zu entwickelnde Kompetenzen | Beispiele, Medien, Materialien |
|---|---|---|
| **1. Information, deren Kodierung und Speicherung**<br>(a) Informatik als Wissenschaft der Verarbeitung von Informationen<br>(b) Binärcodierung ganzer Zahlen.<br>(c) Darstellung von Informationen in Schrift, Bild und Ton<br>(d) Speichern von Daten mit informatischen Systemen am Beispiel der Schulrechner bzw des Einzelrechnermodells nach Von-Neumann.<br>(e) Vereinbarung von Richtlinien zur Datenspeicherung auf den Schulrechnern (z.B. Ordnerstruktur, Dateibezeichner usw.) | Die Schülerinnen und Schüler<br>• nutzen die im Unterricht eingesetzten Informatiksysteme selbstständig, sicher, zielführend und verantwortungsbewusst (D),<br>• nutzen das Internet zur Recherche, zum Datenaustausch und zur Kommunikation (K).<br>• entwickeln ein Bewusstsein für die Gefahren bei der Nutzung von Informatiksystemen. | *Beispiel: Binärdarstellung ganzer Zahlen und Zeichenketten. Inerpretaton des Binärcodes als ganze Zahl.*<br><br>*Fakultatives Beispiel*: Textkodierung z.B. nach dem ASCII-Code und anschließend in Binärcode.<br><br>Die von-Neumann-Architektur und die Bestandteile eines Universalrechners werden mit Hilfe des Jonny-Rechner-Software erarbeitet. |
| **2. Informations- und Datenübermittlung in Netzen**<br>(a) „Sender-Empfänger-Modell“ und seine Bedeutung für die Eindeutigkeit von Kommunikation<br>(b) Informatische Kommunikation in Rechnernetzen am Beispiel des Schulnetzwerks (z.B. Benut-zeranmeldung, Netzwerkordner, Zugriffsrechte, Client-Server)<br>(c) Grundlagen der technischen Umsetzung von Rechnerkommunikation am Beispiel des Internets (z.B. Netzwerkadresse, Paketvermittlung, Protokoll)<br>(d) Richtlinien zum verantwortungsvollen Umgang mit dem Internet | | *Beispiel*: Rollenspiel zur Paketvermittlung<br>Schülerinnen und Schüler übernehmen die Rollen von Clients und Routern. Sie schicken spielerisch Informationen auf Karten von einem Schüler-Client zum anderen. Jede Schülerin und jeder Schüler hat eine Adresse, jeder Router darüber hinaus eine Routingtabelle. Mit Hilfe der Tabelle und einem Würfel wird entschieden, wie ein Paket weiter vermittelt wird.<br>*Medium*: Besprechung einer Filmdokumentation<br>Schülerinnen und Schüler diskutieren die Gefahren bei der Nutzung des Internets anhand der Dokumentation „Exclusiv im Ersten Zugriff Wenn das Netz zum Gegner wird“ und erarbeiten da- |

| | | |
|---|---|---|
| | | ran Regeln für einen bewussteren und sicheren Umgang mit Informatiksystemen. |

**Unterrichtsvorhaben EF-II**

**Thema:** Grundlagen der objektorientierten Analyse, Modellierung und Implementierung anhand von statischen Grafikszenen im Projekt FIGUREN (nach Kölling JAVA lernen mit BlueJ)

**Leitfrage:** *Wie lassen sich reale und virtuelle Objekte informatisch modellieren und im Sinne einer Simulation informatisch realisieren?*

**Vorhabenbezogene Konkretisierung:**

Ein zentraler Bestandteil des Informatikunterrichts der Einführungsphase ist die Objektorientierte Programmierung. Dieses Unterrichtsvorhaben führt in die Grundlagen der Analyse, Modellierung und Implementierung in diesem Kontext ein. Dabei werden Klassen- und Objektkarten benutzt (PlugIn Klassenkarte für BlueJ).

Dazu werden zunächst konkrete geometrische Figuren analysiert und im Sinne des Objektorientierten Paradigmas strukturiert. Dabei werden die grundlegenden Begriffe der Objektorientierung und Modellierungswerkzeuge wie Objektkarten, Klassenkarten oder Beziehungsdiagramme eingeführt.

Im Anschluss wird mit der Realisierung erster Projekte mit Hilfe der didaktischen Programmierumgebung BlueJ begonnen. Die vorgegebenen Klassen des Projektes Figuren werden von Schülerinnen und Schülern in Teilen analysiert und erweitert und entsprechende Objekte anhand einfacher Problemstellungen erprobt. Dazu muss der grundlegende Aufbau einer Java-Klasse thematisiert und zwischen Deklaration, Initialisierung und Methodenaufrufen unterschieden werden.

Da bei der Umsetzung dieser ersten Projekte konsequent auf die Verwendung von Kontrollstrukturen verzichtet wird und der Quellcode aus einer rein linearen Sequenz besteht, ist auf diese Weise eine Fokussierung auf die Grundlagen der Objektorientierung möglich, ohne dass algorithmische Probleme ablenken. Natürlich kann die Arbeit an diesen Projekten unmittelbar zum nächsten Unterrichtsvorhaben führen. Dort stehen unter anderem Kontrollstrukturen im Mittelpunkt.

**Zeitbedarf**: 8 Stunden

| Unterrichtssequenzen | Zu entwickelnde Kompetenzen | Beispiele, Medien, Materialien |
|---|---|---|
| **1. Identifikation von Objekten**<br>(a) Am Beispiel eines lebensweltnahen Beispiels werden Objekte im Sinne der Objektorientierten Modellierung eingeführt.<br>(b) Objekte werden mit Objektkarten visualisiert und mit sinnvollen Attributen und „Fähigkeiten“, d.h. Methoden versehen.<br>(c) Manche Objekte sind prinzipiell typgleich und werden so zu einer Objektsorte bzw. Objektklasse zusammengefasst.<br>(d) Vertiefung: Modellierung weiterer Beispiele ähnlichen Musters | Die Schülerinnen und Schüler<br>• ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M),<br>• modellieren Klassen mit ihren Attributen, ihren Methoden und Assoziationsbeziehungen (M),<br>• stellen die Kommunikation zwischen Objekten grafisch dar (M),<br>• implementieren einfache Algorithmen unter Beachtung der Syntax und Semantik einer Programmiersprache (I),<br>• stellen den Zustand eines Objekts dar (D). | *Beispiel:* Vogelschwarm<br>Schülerinnen und Schüler betrachten einen Vogelschwarm als Menge gleichartiger Objekte, die in einer Klasse mit Attributen und Methoden zusammengefasst werden können.<br><br>*Materialien:*<br>Ergänzungsmaterialien zum Projekt Figuren |
| **2. Analyse von Klassen didaktischer Lernumgebungen**<br>(a) Objektorientierte Programmierung als modularisiertes Vorgehen (Entwicklung von Problemlösungen auf Grundlage vorhandener Klassen)<br>(b) Teilanalyse der Klassen des Projekts Figuren | | *Materialien*:<br>Dokumentation und Kommentare des Projekts Figuren |
| **3. Implementierung zweidimensionaler, statischer Szenen**<br>(a) Grundaufbau einer Java-Klasse<br>(b) Deklaration und Initialisierung von Objekten<br>(c) Methodenaufrufe mit Parameterübergabe zur Manipulation von Objekteigenschaften (z.B. Farbe, Position) | | *Beispiel*: An- und Umbauten am Haus<br>Die Schülerinnen und Schüler erweitern und modifizieren das „virtuelle Haus“ mit den Klassen Rechteck und Oval.<br><br>*Materialien:*<br>Ergänzungsmaterialien zum Projekt Figuren (AB Rechteck) |

**Unterrichtsvorhaben EF-III**

**Thema:** Grundlagen der objektorientierten Programmierung und algorithmischer Grundstrukturen in Java anhand von einfachen Grafiken mit der BlueJ-Turtle nach A. Hermes.

**Leitfragen:** *Wie lassen sich Grafiken mit dem Computer durch Programme erzeugen?*

**Vorhabenbezogene Konkretisierung:**

Der Schwerpunkt dieses Unterrichtsvorhabens liegt auf der Entwicklung mehrerer Projekte, die durch Programme erzeugte Grafiken aufweisen. Für die Umsetzung dieses Projekts werden Kontrollstrukturen in Form von Schleifen und Verzweigungen benötigt und eingeführt.

Sind an einem solchen Beispiel im Schwerpunkt Schleifen und Verzweigungen eingeführt worden, sollen diese Konzepte an weiteren Beispielprojekten eingeübt werden. Dabei muss es sich nicht zwangsläufig um solche handeln, bei denen Kontrollstrukturen lediglich zur Grafik verwendet werden. Auch die Realisierung einfacher mathematischer Funktionen soll behandelt werden.

Das Unterrichtsvorhaben schließt mit einem Projekt, das komplexere grafische Elemente und mathematische Algorithmen (Bruchrechnung, Euklid) beinhaltet, so dass die Schülerinnen und Schüler mehr als nur die Klasse erstellen müssen.

Komplexere Assoziationsbeziehungen zwischen Klassen werden in diesem Unterrichtsvorhaben zunächst nicht behandelt. Sie stellen den Schwerpunkt des folgenden Vorhabens dar.

**Zeitbedarf**: 18 Stunden

| Unterrichtssequenzen | Zu entwickelnde Kompetenzen | Beispiele, Medien, Materialien |
|---|---|---|
| **1. 2D-Zeichnungen unter Ausnutzung von Wiederholungen zeichnen lassen**<br>(a) Zeichnung einfacher geometrischer Figuren mit der TURTLE<br>(b) Zeichnungen vereinfachen durch Wiederholungen (Schleifen)<br>(c) Figuren mit Fallunterscheidungen: Farbgebungen | Die Schüler/innen<br>• analysieren und erläutern einfache Algorithmen und Programme (A),<br>• entwerfen einfache Algorithmen und stellen sie umgangssprachlich und grafisch dar (M),<br>• ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M),<br>• modellieren Klassen mit ihren Attributen, ihren Methoden und Assoziationsbeziehungen (M),<br>• ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen oder lineare Datensammlungen zu (M),<br>• ordnen Klassen, Attributen und Methoden ihren Sichtbarkeitsbereich zu (M),<br>• modifizieren einfache Algorithmen und Programme (I),<br>• implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I),<br>• implementieren Algorithmen unter Verwendung von Variablen und Wertzuweisungen, Kontrollstrukturen sowie Methodenaufrufen (I),<br>• implementieren einfache Algorithmen unter Beachtung der Syntax und Semantik einer Programmiersprache (I),<br>• testen Programme schrittweise anhand von Beispielen (I),<br>• interpretieren Fehlermeldungen und korrigieren den Quellcode (I). | *Beispiel*: Quadrat, Dreieck, regelmäßige n-Ecke<br>Die Schülerinnen und Schüler realisieren mit der Hermes-TURTLE die Zeichnung vonm n-Ecken mit geeigneten Methoden in verschiedenen Farben.<br><br>*Materialien:*<br>Hermes-TURTLE für BlueJ (Bibliothek) |
| **2. Erstellen und Verwalten größerer Mengen einfacher grafischer Objekte**<br>(a) Erzeugung von mehreren verknüpften Objekten mit Hilfe von Zählschleifen (FOR-Schleife)<br>(b) Verwaltung von Objekten in eindimensionalen Feldern (Arrays)<br>(c) Animation von Objekten, die in eindimensionalen Feldern (Arrays) verwaltet werden<br>(d) Vertiefung: Verschiedene Feldbeispiele | | *Beispiel:* verschachtelte n-Ecke<br>Die Schülerinnen und Schüler realisieren mit Hilfe mehrerer n-Ecke kunstvolle Grafiken.<br><br>*Beispiel*: Wasserorgel<br>Die Schülerinnen und Schüler erstellen eine Wasserorgelsimulation (Märchenwald-Video).<br><br>*Materialien:*<br>Hermes-Turtle für BlueJ |

**Unterrichtsvorhaben EF-IV**

**Thema:** Modellierung und Implementierung von Klassen- und Objektbeziehungen anhand von endlichen deterministischen Zweipersonen-Nullsummenspielen (nach Rüdiger Baumann).

**Leitfrage:** *Wie lassen sich komplexere Datenflüsse und Beziehungen zwischen Objekten und Klassen realisieren?*

**Vorhabenbezogene Konkretisierung:**

Dieses Unterrichtsvorhaben beschäftigt sich im Schwerpunkt mit dem Aufbau komplexerer Objektbeziehungen. Während in vorangegangenen Unterrichtsvorhaben Objekte nur jeweils solchen Objekten Nachrichten schicken konnten, die sie selbst erstellt haben, soll in diesem Unterrichtsvorhaben diese hierarchische Struktur aufgebrochen werden. Es sollen einfache Klassendiagramme mit den zugehörigen Assoziationen in UML als Werkzeug der Modellierung eingeführt und genutzt werden (ZA-Vorgaben beachten).

Dazu bedarf es zunächst einer präzisen Unterscheidung zwischen Objektreferenzen und Objekten, so dass klar wird, dass Dienste eines Objektes von unterschiedlichen Objekten über unterschiedliche Referenzen in Anspruch genommen werden können. Auch der Aufbau solcher Objektbeziehungen muss thematisiert werden. Des Weiteren wird das Prinzip der Vererbung im objektorientierten Sinne angesprochen. Dazu werden die wichtigsten Varianten der Vererbung anhand von verschiedenen Projekten vorgestellt. Zunächst wird die Vererbung als Spezialisierung im Sinne einer einfachen Erweiterung einer Oberklasse vorgestellt. Darauf folgt ein Projekt, welches das Verständnis von Vererbung um den Aspekt der späten Bindung erweitert, indem Dienste einer Oberklasse überschrieben werden. Modellierungen sollen in Form von Implementationsdiagrammen erstellt werden.

In einem Projekt zur Textanalyse wird die Klasse String ausführlich behandelt. In weiteren kleineren Projekten werden ein- und zweidimensionale Felder (Arrays) eingeführt und verwendet. Es ist sinnvoll an dieser Stelle auf Wrapper-Klassen und Type-Casts einzugehen.

Auch soll hier in das Konzept der grafischen Benutzerschnittstellen eingegangen werden, dabei spielen nur einfache AWT- und Swing-Komponenten eine Rolle und lediglich die ActionEvents werden zur Eingabesteuerung benutzt. Ggf. kann auf das Speichern und Laden von Spielständen mit den Java-IO-Konzepten (ab Java 7 nio-Klassen, vorher io-Klassen) und Exceptions und ihre Behandlung eingegangen werden.

Zum Abschluss kann kurz auf das Prinzip der abstrakten Klasse eingegangen werden. Dieser Inhalt ist aber nicht obligatorisch für die Einführungsphase. Fakultativ kann das Konzept der GUIs erweitert werden.

Abschließend und begleitend wird das Spiel TicTacToe programmiert (incl. ggf. vorgegebener GUI), die Modellierung in UML wird besprochen nach den Vorgaben zum Zentralabitur.

**Zeitbedarf**: 18 Stunden

| Unterrichtssequenzen | Zu entwickelnde Kompetenzen | Beispiele, Medien, Materialien |
|---|---|---|
| **1. Modellierung und Implementation eines bekannten Zweipersonenspiels**<br>(a) Einführung der UML-Modellierung von Klassen und ihren Assoziationen am Beispiel<br>(b) Einführung in grafische Benutzerschnittstellen und die Ereignisbehandlung | Die Schüler/innen<br>• analysieren und erläutern eine objektorientierte Modellierung (A),<br>• stellen die Kommunikation zwischen Objekten grafisch dar (M),<br>• ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M),<br>• modellieren Klassen mit ihren Attributen, ihren Methoden und Assoziationsbeziehungen (M),<br>• ordnen Attribute, Parameter und Rückgaben von Methoden einfache Datentypen, Objekttypen oder lineare Datensammlungen zu (M),<br>• ordnen Klassen, Attributen & Methoden ihrer Sichtbarkeit zu (M),<br>• modellieren Klassen mittels Vererbung (M),<br>• implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I),<br>• testen Programme schrittweise anhand von Beispielen (I),<br>• interpretieren Fehlermeldungen und korrigieren den Quellcode (I),<br>• modifizieren einfache Algorithmen und Programme (I),<br>• stellen Klassen, Assoziations- und Vererbungsbeziehungen in Diagrammen grafisch dar (D),<br>• dokumentieren Klassen durch Beschreibung der Funktionalität der Methoden (D). | *Beispiel*: TicTacToe<br>Die Schülerinnen und Schüler entwickeln das Spiel TicTacToe in JAVA und modellieren es zuvor in einer einfachen UML-Variante.<br><br>*Materialien:*<br>UML-Editor Violet (Horstmann) – Freeware<br>BlueJ-IDE und Netbeans-IDE<br>ZA-Vorgaben (UML) |
| **2. Entwicklung eines Spiels mit der Notwendigkeit von Kollisionskontrollen zwischen zwei oder mehr grafischen Objekten**<br>(a) Modellierung des Spiels ohne Berücksichtigung der Kollision mit Hilfe eines Implementationsdiagramms<br>(b) Dokumentation der Klassen des Projekts<br>(c) Implementierung eines Prototypen<br>(d) Implementierung einer GUI<br>(e) Vertiefung: Laden und Speichern von Spielständen mittels i/o-Konzepten von JAVA | | *Beispiel:* Schiffe versenken<br>Die Schülerinnen und Schüler entwickeln das bekannte Spiel „Schiffe versenken“, dabei modellieren sie in UML und implementieren in JAVA.<br><br>*Materialien:*<br>UML-Editor Violet (Horstmann) – Freeware<br>BlueJ-IDE und Netbeans-IDE |

**Unterrichtsvorhaben E1-V:**

**Thema:** Umgang mit Textdateien, Suchen in Texten, Entwurf einfacher GUIs

**Leitfragen***: Wie können Texte gefiltert werden, wie können sie vom Compiler lexikalisch untersucht werden?*

**Vorhabenbezogene Konkretisierung:**

Ausgehend von einer Kompilierung wird erarbeitet, wie der Lexer des Java-Compilers die Analyse und Zerteilung in Tokens vornimmt. Dabei wird die Struktur von Strings und Texten vertiefend erarbeitet.

In anderen Anwendungskontexten werden Strings und Texte analysiert und manipuliert. Dabei werden Spiele (z.B. Hangman oder Wortsalat) und Kryptoprogramme (z.B. Cäsar-Verschlüsselung) entwickelt und implementiert, welche abschließend mit GUIs versehen werden.

Zur Umsetzung wird nochmals die Behandlung des ASCII- und UTF-Codes vertieft.

**Zeitbedarf**: 16 Stunden

| Unterrichtssequenzen | Zu entwickelnde Kompetenzen | Beispiele, Medien, Materialien |
|---|---|---|
| **1. Analyse des Kompilierungsvorganges bei JAVA – Stufe 1: Der LEXER**<br>(a) Analyse des Compilers, wie werden die einzelnen Bausteine herausgefiltert?<br>(b) Verarbeitung von Texten ohne Automaten, die Klasse STRING und ihre Methoden | Die Schüler/innen<br>• analysieren und erläutern eine objektorientierte Modellierung (A),<br>• stellen die Kommunikation zwischen Objekten grafisch dar (M),<br>• ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M),<br>• modellieren Klassen mit ihren Attributen, ihren Methoden und Assoziationsbeziehungen (M),<br>• implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I),<br>• testen Programme schrittweise anhand von Beispielen (I),<br>• interpretieren Fehlermeldungen und korrigieren den Quellcode (I),<br>• modifizieren einfache Algorithmen und Programme (I),<br>• dokumentieren Klassen durch Beschreibung der Funktionalität der Methoden (D). | *Beispiel*: JAVA-Lexer<br>Die Schülerinnen und Schüler analysieren die Kompilierung eines JAVA-Programmes, sie erarbeiten, dass der LEXER in erster Stufe eine Tokenizung vornimmt.Sie lernen die Klasse STRING kennen<br><br>*Materialien:*<br>BlueJ-IDE und Netbeans-IDE<br>JAVA-API (STRING) |
| **2. Entwicklung eines Spieles mit Texten (z.B. Hangman) und Implementierung selbigens**<br>(a) Modellierung und Implementierung eines Spieles (Hangman) in JAVA<br>(b) Anbindung an eine GUI (fakultativ)<br>(c) Entwicklung einfacher Kryptoprogramme mit Strings, ASCII- oder UTF-Codes (z.B. Cäsar), ggf. GUI dazu | | *Beispiel*: HANGMAN<br>Die Schülerinnen und Schüler entwickeln das Spiel Hangman in JAVA und modellieren es zuvor in einer einfachen UML-Variante.<br><br>*Materialien:*<br>UML-Editor Violet (Horstmann) – Freeware<br>BlueJ-IDE und Netbeans-IDE<br>ZA-Vorgaben (UML)<br>JAVA-API |

**Unterrichtsvorhaben EF-VI**

**Thema:** Analyse-, Such- und Sortieralgorithmen mit quadratischer Laufzeit anhand einer kontextbezogenen und komplexen Problemstellung (z. B. einer Lottosimulation) und deren Realisierung und Implementierung

**Leitfragen:** Wie können gleichartige Daten einfach verwaltet werden? Wie können diese Daten effizient analysiert und sortiert werden?

**Vorhabenbezogene Konkretisierung:**
Die einzelnen Bestandteile einer komplexen Simulation werden Schritt für Schritt analysiert, algorithmisiert und implementiert. Der Datentyp eines eindimensionalen Feldes/Arrays wird zur Verwaltung gleichartiger Daten eingeführt und verwendet.

Im Falle des Beispiels einer Lottosimulation sind die Bausteine im Einzelnen: Prüfung eines Lottoscheins auf Richtigkeit (keine doppelten Zahlen), Sortierung der Tippzahlen, Ziehung der Lottozahlen, Gewinnrangermittlung.

Das zuvor erlernte MVC-Konzept wird hier angewendet, d. h. es wird strikt zwischen Funktionalität (Controller), Daten (Model) und Oberfläche (View) getrennt.

Zuvor erlernte Kontrollstrukturen werden vertieft und situationsabhängig flexibel angewendet.

**Zeitbedarf:** 12 Stunden

| Unterrichtssequenzen | Zu entwickelnde Kompetenzen | Beispiele, Medien oder Materialien |
|---|---|---|
| **1. Problemanalyse Lottosimulation**<br>(a) Modularisierung und Modellierung der Lottosimulation, MVC-Konzept<br>(b) Die Datenstruktur des Feldes/Array: Modell der Lottosimulation<br>(c) Entwicklung eines Gui der Lottosimulation: View der Lottosimulation | Die Schülerinnen und Schüler<br>• modellieren Klassen mit ihren Attributen, ihren Methoden und Assoziationsbeziehungen (M),<br>• ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen oder lineare Datensammlungen zu (M),<br>• ordnen Klassen, Attributen und Methoden ihren Sichtbarkeitsbereich zu (M),<br>• stellen die Kommunikation zwischen Objekten grafisch dar (M),<br>• stellen den Zustand eines Objekts dar (D),<br>• stellen Klassen, Assoziations- und Vererbungsbeziehungen in Diagrammen grafisch dar (D),<br>• dokumentieren Klassen durch Beschreibung der Funktionalität der Methoden (D). | Skript: "Informatik in der Einführungsphase" von der Schulhomepage<br>leeres Projekt "Lottosimulation"<br>Dieses Projekt ist bezüglich des MVC-Konzepts eines der umfangreichsten Projekte, welches die Schülerinnen und Schüler komplett selber erstellen. |
| **2. Algorithmisierung der Lottosimulation: Controller**<br>(a) Prüfung der Korrektheit des Tippscheins<br>(b) Sortierung der Tippzahlen<br>(c) Ziehung der Lottozahlen / Quicktipp<br>(d) Gewinnrangermittlung | zusätzlich: Die Schülerinnen und Schüler<br>• analysieren Such- und Sortieralgorithmen und wenden sie auf Beispiele an (D),<br>• entwerfen einen weiteren Algorithmus zum Sortieren (M),<br>• beurteilen die Effizienz von Algorithmen am Beispiel von Sortierverfahren hinsichtlich Zeitaufwand und Speicherplatzbedarf (A). | Skript: "Informatik in der Einführungsphase" von der Schulhomepage<br>in Unterrichtssequenz 1 verändertes Projekt "Lottosimulation" |
| **3. Abschluss Lottosimulation**<br>(a) Verknüpfung zw. View, Model und Controller<br>(b) Verbesserung der grafischen Oberfläche durch komfortable Gewinnrangausgabe oder Tippscheinabgabe<br>(c) Nutzung anderer Sortierverfahren | | |

## Unterrichtsvorhaben EF-VII

**Thema:** Informatiksysteme, Datencodierung, Datenspeicherung im Computersystem (im Einzelrechner).

**Leitfragen:** *Wie werden Daten in einem Computersystem verarbeitet und gespeichert. Wie werden Daten aufbereitet und codiert, damit sie vom Computer verarbeitet werden können und welche Schritte werden im Computersystem durchlaufen, bevor die Daten im Speicher abgelegt werden können?*

**Vorhabenbezogene Konkretisierung:**

Das siebte Unterrichtsvorhaben beschäftigt sich mit dem Aufbau und der Funktionsweise eines Computersystems. Hier sollen speziell die Bauteile näher betrachtet werde, die nach John von Neumann und der nach ihm benannten von Neumann Architektur einen Universalrechner ausmachen. Hierbei werden insbesondere das Eingabewerk (Maus, Tastatur, etc), das Speicherwerk, das Ausgabewerk, das Rechenwerk und das Steuerwerk betrachtet. Die SuS sollen nach dieser Unterrichtseinheit eine Vorstellung davon bekommen, dass die heutigen Computer seit ihrem Aufkommen eine Entwicklung durchlaufen haben und das die heutige Rechnerstruktur keineswegs starre Konstrukte sind. Auch heute noch werden die Rechnersysteme optimiert, verändert und weiterentwickelt.

Die zweite Säule dieses Unterrichtsvorhabens stellt die Datencodierung und die Datenspeicherung dar. Hier soll speziell der Binärcode behandelt werden und wie es dem Computer gelingt, einfache mathematische Rechenoperationen in binär codierter Form durchzuführen und zu speichern.

Das Unterrichtsvorhaben wird weitestgehend mit der Simulationssoftware “Johnny” durchgeführt werden. Diese Simulationssoftware ist ein kostenloses Programm, welche den Aufbau und die Funktionsweise eines von Neumann-Rechners simuliert. Der Johnny-Simulator stellt ein vereinfachte Form der CPU (Central Processin Unit) und des Arbeitsspeichers dar und erlaubt es den SuS den Ablauf einer mathematischen Prozedur Schritt für Schritt zu modellieren und zu erfassen.

**Zeitbedarf**: 8 Stunden

| Unterrichtssequenzen | Zu entwickelnde Kompetenzen | Beispiele, Medien, Materialien |
|---|---|---|
| **1. Digitalisierung - Anwendung des Binärcodes auf zu codierende Daten**<br>(a) Stellenwertsystem des Binärcodes.<br>(b) Rechenoperationen im Binärcode durchführen. | Die Schülerinnen und Schüler<br>● stellen ganze Zahlen und Zeichen in Binärcodes dar (D).<br>● interpretieren Binärcodes als Zahlen und Zeichen (D).<br>● Präsentieren ihre Ergebnisse im Kurs (D), (K).<br>● stellen sich gegenseitig Aufgaben und diskutieren anschließend die Ergebnisse (K).<br>● beschreiben und erläutern den strukturellen Aufbau und die Arbeitsweise singulärer Rechner am Beispiel der „Von-Neumann-Architektur“ (A).<br>● Präsentieren ihre Ergebnisse (D), (K).<br>● Veranschaulichen die Arbeitsschritte des Rechenwerks im Flussdiagramm (D), (M).<br>● analysieren und erläutern den Aufbau des von Neumann - Rechners sowie des Johnny - Simulators (A).<br>● testen und korrigieren ihre Programme im Simulator (I). | *Beispiel*: Kodierung und Dekodierung von Zahlen und Buchstaben im Binärcode. Rechnen im Binärcode. |
| **2. Universalrechner - Rechnersysteme nach dem Modell der von Neumann Architektur**<br>(a) Aufbau und die Bestandteile eines von Neumann-Rechners.<br>(b) Speichern von Binärcode im Arbeitsspeicher.<br>(c) Verarbeitung der Daten im Rechenwerk (CPU) und die erneute Ablage der Daten im Arbeitsspeicher. | | *Beispiel*: Die Schülerinnen und Schüler bearbeiten im Johnny - Simulator die Funktionsweise der Addition und der Subtraktion im Rechenwerk der CPU und fertigen ein Flussdiagram zur schrittweisen Arbeitsweise des Rechenwerks an. (entweder zur Addition oder zur Subtraktion) |

**Fakultatives Unterrichtsvorhaben: Exkursion ins HNF nach Paderborn**

**Thema:** Geschichte der digitalen Datenverarbeitung und die Grundlagen des Datenschutzes

**Leitfrage:** *Welche Entwicklung durchlief die moderne Datenverarbeitung und welche Auswirkungen ergeben sich insbesondere hinsichtlich neuer Anforderungen an den Datenschutz daraus?*

**Vorhabenbezogene Konkretisierung:**

Das folgende Unterrichtsvorhaben stellt den Abschluss der Einführungsphase dar. Die erste Unterrichtssequenz („Informatiker verändern die Welt – Auswirkungen der Informationstechnologie auf Mensch und Gesellschaft“) können von den Schülerinnen und Schülern als Referate oder in Kleingruppenarbeit präsentiert werden.

Anschließend wird verstärkt auf den Aspekt des Datenschutzes eingegangen und auf schülernahe Beispielsituationen zur Anwendung gebracht. Inhaltliche Schwerpunkte können hierbei sein: Meilensteine der Informationstechnik, Informatik und Ethik, Automatisierung in der Arbeitswelt, Automatisierung aus Sicht der Informatik, Automatisierung im Alltag, Der gläserne Deutsche, Datenschutz als Grundrecht?, Datenschutz konkret – NSA und Social Media

Im Zusammenhang mit dem Unterrichtsvorhaben soll i.d.R. eine Exkursion ins Heinz-Nixdorf-Museum Paderborn stattfinden.

**Zeitbedarf**: ca. 5 Stunden zzgl. Exkursion und Referatsarbeit zuhause

### 2.1.2.2 Qualifikationsphase

**Unterrichtsvorhaben Q1-I (muss angepasst werden nach Evaluation)**

**Thema:** Wiederholung der objektorientierten Modellierung und Programmierung

**Leitfragen:** *Wie modelliert und implementiert man zu einer Problemstellung in einem geeigneten Anwendungskontext Java-Klassen inklusive ihrer Attribute, Methoden und Beziehungen? Wie kann man die Modellierung und die Funktionsweise der Anwendung grafisch darstellen?*

**Vorhabenbezogenen Konkretisierung:**

Zu einer Problemstellung in einem Anwendungskontext soll eine Java-Anwendung entwickelt werden. Die Problemstellung soll so gewählt sein, dass für diese Anwendung die Verwendung einer abstrakten Oberklasse als Generalisierung verschiedener Unterklassen sinnvoll erscheint und eine Klasse durch eine Unterklasse spezialisiert werden kann. Um die Aufgabe einzugrenzen, können (nach der ersten Problemanalyse) einige Teile (Modellierungen oder Teile von Java-Klassen) vorgegeben werden.

Die Schülerinnen und Schülern erläutern und modifizieren den ersten Entwurf und modellieren sowie implementieren weitere Klassen und Methoden für eine entsprechende Anwendung. Klassen und ihre Beziehungen werden in einem Implementationsdiagramm dargestellt. Dabei werden Sichtbarkeitsbereiche zugeordnet. Exemplarisch wird eine Klasse dokumentiert. Der Nachrichtenaustausch zwischen verschiedenen Objekten wird verdeutlicht, indem die Kommunikation zwischen zwei ausgewählten Objekten grafisch dargestellt wird. In diesem Zusammenhang wird das Nachrichtenkonzept der objektorientierten Programmierung wiederholt.

**Zeitbedarf**: 8 Stunden

| Unterrichtssequenzen | Zu entwickelnde Kompetenzen | Beispiele, Medien, Materialien |
|---|---|---|
| **1. Wiederholung und Erweiterung der objektorientierten Modellierung und Programmierung durch Analyse und Erweiterung eines kontextbezogenen Beispiels**<br>(a) Analyse der Problemstellung<br>(b) Analyse der Modellierung (Implementationsdiagramm)<br>(c) Erweiterung der Modellierung im Implementationsdiagramm (Vererbung, abstrakte Klasse)<br>(d) Kommunikation zwischen mindestens zwei Objekten (grafische Darstellung)<br>(e) Dokumentation von Klassen<br>(f) Implementierung der Anwendung oder von Teilen der Anwendung | Die Schülerinnen und Schüler<br>• analysieren und erläutern objektorientierte Modellierungen (A),<br>• beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A),<br>• modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M),<br>• ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M),<br>• modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M),<br>• implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I),<br>• nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I),<br>• wenden eine didaktisch orientierte Entwicklungsumgebung zur Demonstration, zum Entwurf, zur Implementierung und zum Test von Informatiksystemen an (I),<br>• interpretieren Fehlermeldungen und korrigieren den Quellcode (I),<br>• stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D),<br>• dokumentieren Klassen (D),<br>• stellen die Kommunikation zwischen Objekten grafisch dar (D). | *Beispiel:* Wetthuepfen<br>Für ein Wetthüpfen zwischen einem Hasen, einem Hund und einem Vogel werden die Tiere gezeichnet. Alle Tiere springen wiederholt nach links. Die Höhe und Weite jedes Hüpfers ist zufällig. Evtl. marschieren sie anschließend hintereinander her.<br><br>oder<br><br>*Beispiel*: Tannenbaum<br>Ein Tannenbaum soll mit verschiedenen Arten von Schmuckstücken versehen werden, die durch unterschiedliche geometrische Objekte dargestellt werden. Es gibt Kugeln, Päckchen in der Form von Würfeln und Zuckerringe in Form von Toren.<br>Ein Prototyp, der bereits mit Kugeln geschmückt werden kann, kann zur Verfügung gestellt werden. Da alle Schmuckstücke über die Funktion des Auf- und Abschmückens verfügen sollen, liegt es nahe, dass entsprechende Methoden in einer gemeinsamen Oberklasse realisiert werden.<br><br>*Materialien:*<br>Ergänzungsmaterialien zum Lehrplannavigator Unterrichtsvorhaben Q1.1-Wiederholung (Download Q1-I.1) |

**Unterrichtsvorhaben Q1-II**

**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?

**Vorhabenbezogene Konkretisierung:**

Ausgehend vom einem Problem wie z. B. "Türme von Hanoi" wird Rekursion als fundamentale Idee der Informatik zunächst in mathematischem, 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.

**Zeitbedarf:** 20/25 Stunden

| Unterrichtssequenzen | zu entwickelnde Kompetenzen | Beispiele, Medien, Materialien |
|---|---|---|
| 1. Entwicklung der Rekursion als fundamentale Idee der Informatik<br>• Rekursion in mathematischen Kontexten<br>• rekursive Formeln<br>• rekursive Funktionen / Methoden<br>• rekursive Programmierung | Die Schülerinnen und Schüler<br>• analysieren und erläutern Algorithmen und Programme (A),<br>• modifizieren Algorithmen und Programme (I), | Türme von Hanoi mit Schwerpunkt auf<br>• Zahl der Versetzungsoperationen<br>• Protokollierung der Versetzungen |
| 2. Rekursion in mathematischen und informatischen Kontexten<br>• Analyse und Darstellung des rekursiven Ablaufs einer Methode<br>• Analyse des Laufzeitverhaltens linearer und verzweigter Rekursion | Die Schülerinnen und Schüler<br>• analysieren und erläutern Algorithmen und Programme (A),<br>• modifizieren Algorithmen und Programme (I),<br>• stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D),<br>• testen Programme systematisch anhand von Beispielen und mit Hilfe von Testanwendungen (I).<br>• untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen (A). | • Fakultätsfunktion (lineare Rekursion)<br>• Fibonacci-Funktion (verzweigte Rekursion)<br>• ggT (verzweigte Rekursion)<br>• evt. Fraktale (Kochkurve, Sierpinskidreieck, etc.) |
| 3. NP-vollständige Probleme lösen mit Backtracking<br>• Erarbeitung verschiedener NP-vollständiger Probleme<br>• Algorithmische Beschreibung einer Lösungsidee<br>• Implementierung eines Problems als Backtrackingalgorithmus | Die Schülerinnen und Schüler<br>• analysieren und erläutern Algorithmen und Programme (A),<br>• modifizieren Algorithmen und Programme (I),<br>• stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D),<br>• entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“ und „Teilen und Herrschen“ und "Backtracking" (M)<br>• testen Programme systematisch anhand von Beispielen und mit Hilfe von Testanwendungen (I).<br>• untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen (A). | • Arbeitsblätter und Rasterprogramm zum N-Damenproblem<br>• Arbeitsblätter zum Springerproblem<br>• Rucksackproblem<br>• Projektarbeit zur Irrgartenproblematik |
| 4. Effiziente Sortierverfahren / Suchverfahren<br>• Wiederholung bereits bekannter Sortier- und Suchverfahren als rekursiver Algorithmus<br>• Erarbeitung eines Sortierverfahrens der Laufzeit | Die Schülerinnen und Schüler<br>• analysieren und erläutern Algorithmen und Programme (A),<br>• modifizieren Algorithmen und Programme (I), | • Demonstrationsprogramm zur Visualisierung von Sortierverfahren<br>• Quicksortvisualisierung zur |

| | | |
|---|---|---|
| O(n*log(n))<br>• Erarbeitung eines Suchverfahrens der Laufzeit O(log(n)). | • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D),<br>• entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“ und „Teilen und Herrschen“ und "Backtracking" (M),<br>• testen Programme systematisch anhand von Beispielen und mit Hilfe von Testanwendungen (I).<br>• implementieren und erläutern iterative und rekursive Such- und Sortierverfahren unterschiedlicher Komplexitätsklassen (Speicherbedarf und Laufzeitverhalten) (I),<br>• beurteilen die Effizienz von Algorithmen unter Berücksichtigung des Speicherbedarfs und der Zahl der Operationen (A),<br>• untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen (A). | Erarbeitung der Idee<br>• Suchspiel zur Erarbeitung der Binären Suche |

**Unterrichtsvorhaben Q1-III**

**Thema:** Modellierung und Implementierung dynamische Listenstrukturen und deren Anwendungen

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

**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 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.

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

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

**Zeitbedarf:** 25/30 Stunden

<table>
<tr><th>Unterrichtssequenzen</th><th>zu entwickelnde Kompetenzen</th><th>Beispiele, Medien, Materialien</th></tr>
<tr><td>1. Die Datenstruktur Stapel im Anwendungskontext unter Nutzung der Klasse Stack<br>• Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen<br>• Erarbeitung der Funktionalität der Klasse Stack<br>• Modellierung und Implementierung der Anwendung unter Verwendung der Klasse Stack.<br>• Implementierung der Klasse Stack</td><td rowspan="3">Die Schülerinnen und Schüler<br>• ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M),<br>• stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D),<br>• modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M),<br>• ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M),<br>• ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M),<br>• stellen die Kommunikation zwischen Objekten grafisch dar (D),<br>• stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D),<br>• dokumentieren Klassen (D),<br>• analysieren und erläutern objektorientierte Modellierungen (A),<br>• implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I).<br>• analysieren und erläutern Algorithmen und Programme (A),<br>• modifizieren Algorithmen und Programme (I),<br>• stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D),<br>• implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I),<br>• testen Programme systematisch anhand von Beispielen und mit Hilfe von Testanwendungen (I).<br>• erläutern Operationen dynamischer (linearer oder/und nichtlinearer) Datenstrukturen (A),<br>• implementieren Operationen dynamischer (linearer oder nichtlinearer) Datenstrukturen (I),</td><td>Irrgartenproblematik:<br>• Ersatz des rekursiven Algorithmus durch Datenstruktur Stack.<br>• Visualisierungsprogramm zur Tiefensuche im Irrgarten</td></tr>
<tr><td>2. Die Datenstruktur Schlange im Anwendungskontext unter Nutzung der Klasse Queue<br>• Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen<br>• Erarbeitung der Funktionalität der Klasse Queue<br>• Modellierung und Implementierung der Anwendung unter Verwendung der Klasse Queue.<br>• Implementierung der Klasse Queue</td><td>Irrgartenproblematik:<br>• Veränderung der Datenstruktur im Algorithmus zur Tiefensuche<br>• Visualisierungsprogramm zur Breitensuche im Irrgarten</td></tr>
<tr><td>3. Die Datenstruktur lineare Liste im Anwendungskontext unter Nutzung der Klasse List<br>• Erarbeitung der Vorteile der Klasse List im Gegensatz zu den bereits bekannten linearen Strukturen<br>• Modellierung und Implementierung einer kontextbezogenen Anwendung unter Verwendung der Klasse List.</td><td>Einfache Oberfläche zur Visualisierung der linearen Liste, z. B. die Verwaltung einer Namensliste</td></tr>
<tr><td>4. Vertiefung / Anwendung einer linearen Datenstruktur im Anwendungskontext.</td><td>zusätzlich: Die Schülerinnen und Schüler<br>• modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M),<br>• verwenden bei der Modellierung geeigneter Problemstellungen Möglichkeiten der Polymorphie (M),</td><td>Objektorientiertes Malprogramm (Mini-CoralDraw) Beispielprogramm auf Schulhomepage</td></tr>
</table>

**Unterrichtsvorhaben Q1-IVa**

**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?

**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 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:** 20 Stunden

| Unterrichtssequenzen | zu entwickelnde Kompetenzen | BSP, Medien, Materialien |
|---|---|---|
| 1. Analyse von Baumstrukturen in Kontexten<br>• Grundlegende Begriffe (Grad, Tiefe, Höhe, Blatt, Inhalt, Teilbaum, Ebene, Vollständigkeit)<br>• Aufbau und Darstellung von binären Bäumen anhand von Baumstrukturen in verschiedenen Kontexten | Die Schülerinnen und Schüler<br>• ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M),<br>• stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D),<br>• modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M),<br>• modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M),<br>• ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie/oder lineare und nichtlineare Datensammlungen zu (M),<br>• verwenden bei der Modellierung geeigneter Problemstellungen Möglichkeiten der Polymorphie (M),<br>• ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M),<br>• stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D),<br>• dokumentieren Klassen (D),<br>• analysieren und erläutern objektorientierte Modellierungen (A),<br>• implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I).<br>• analysieren und erläutern Algorithmen und Programme (A),<br>• modifizieren Algorithmen und Programme (I),<br>• stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D),<br>• entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“ und „Teilen und Herrschen“ und „Backtracking“ (M),<br>• implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I),<br>• testen Programme systematisch anhand von Beispielen und mit Hilfe von Testanwendungen (I).<br>• erläutern Operationen dynamischer (linearer oder/und nichtlinearer) Datenstrukturen (A),<br>• implementieren Operationen dynamischer (linearer oder nichtlinearer) Datenstrukturen (I), | • Demoprogramm und Arbeitsblätter zum Projekt Tiere-Raten |
| 2. Die Datenstruktur Binärbaum im Anwendungskontext unter Nutzung der Klasse BinaryTree<br>• Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen im Anwendungskontext<br>• Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramms<br>• Erarbeitung der Klasse BinaryTree und beispielhafte Anwendung der Operationen<br>• Implementierung der Anwendung oder von Teilen der Anwendung<br>• Traversierung eines Binärbaums im Pre-, In- und Postorderdurchlauf | | • Arbeitsblätter zur Projektarbeit Tiere-Raten |
| 3. Die Datenstruktur binärer Suchbaum im Anwendungskontext unter Verwendung der Klasse BinarySearchTree<br>• Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen<br>• Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramm,<br>• grafische Darstellung eines binären Suchbaums und Erarbeitung der Struktureigenschaften<br>• Erarbeitung der Klasse BinarySearchTree und Einführung des Interface Item zur Realisierung einer geeigneten Ordnungsrelation<br>• Implementierung der Anwendung oder von Teilen der Anwendung inklusive einer sortierten Ausgabe des Baums | | • Arbeitsblätter zum binären Suchbaum |
| 4. Übung und Vertiefungen der Verwendung von Binärbäumen oder binären Suchbäumen anhand weiterer Problemstellungen | | • Projektarbeiten zu einem der Themen Termbäume, Mobilebäume, Morsebäume<br>• Projektarbeit zu einem der Themen Stichwortbaum, Ahnenbaum |

**Unterrichtsvorhaben Q1-IVb**

**Thema:** Modellierung und Implementierung dynamische nichtlineare Datenstrukturen am Beispiel der Graphen

**Leitfragen:** Bei welchen Problemstellungen reichen die Datenstrukturen des Binärbaums und der Liste nicht aus? Welche Möglichkeiten gibt es, Daten außer in Listen und Bäumen zu verwalten? Wie hängen die Datenstrukturen Graph, Baum und Liste zusammen?

**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 zur Implementation von Lösungen ausgewählter Graphen-Probleme genutzt.

**Zeitbedarf:** 15 Stunden

<table>
<tr><th>Unterrichtssequenzen</th><th>zu entwickelnde Kompetenzen</th><th>Beispiele, Medien, Materialien</th></tr>
<tr><td>1. Die Datenstruktur des Graphen im Anwendungskontext<br>• Darstellungsformen eines Graphen<br>• Modellierung von Anwendungssituationen als Graph.</td><td rowspan="2">Die Schülerinnen und Schüler<br>• ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M),<br>• stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D),<br>• modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M),<br>• modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M),<br>• ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M),<br>• verwenden bei der Modellierung geeigneter Problemstellungen Möglichkeiten der Polymorphie (M),<br>• ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M),<br>• stellen die Kommunikation zwischen Objekten grafisch dar (D),<br>• stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D),<br>• dokumentieren Klassen (D),<br>• analysieren und erläutern objektorientierte Modellierungen (A),<br>• implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I).<br>• analysieren und erläutern Algorithmen und Programme (A),<br>• modifizieren Algorithmen und Programme (I),<br>• stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D),<br>• entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“ und „Teilen und Herrschen“ und „Backtracking“ (M),<br>• implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I),<br>• testen Programme systematisch anhand von Beispielen und mit Hilfe von Testanwendungen (I).<br>• erläutern Operationen dynamischer (linearer oder/und nicht-linearer) Datenstrukturen (A),</td><td>Eulerkreisproblem:<br>• Arbeitsblätter zum Eulerweg / Eulerkreis<br>• Visualisierungsprogramm für Graphen</td></tr>
<tr><td>2. Die Datenstruktur des Graphen im Anwendungskontext unter Nutzung der Klasse Graph<br>• Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen<br>• Erarbeitung der Funktionalität der Klasse Graph<br>• Modellierung und Implementierung verschiedener Problemstellungen unter Verwendung der Klasse Graph.</td><td>Arbeitsblätter auf der Schulhomepage zu:<br>• Tiefensuche / Breitensuche<br>• kürzeste Wege mit Dijkstra<br>• Traveling Salesman Problem<br>• minimale Spannbäume</td></tr>
</table>

**Unterrichtsvorhaben Q1-Va**

**Thema:** Sicherheit und Datenschutz in Netzstrukturen

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

**Vorhabenbezogene Konkretisierung:**

Ausgehend von einer Kommunikation zwischen zwei Kommunikationspartnern über eine einfache Leitung werden 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.

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.

**Zeitbedarf:** 15 Stunden

| Unterrichtssequenzen | zu entwickelnde Kompetenzen | Beispiele, Medien, Materialien |
|---|---|---|
| 1. Schichten des TCP/IP-Protokolls<br>• Erarbeitung der Notwendigkeiten einer Netzwerkkommunikation<br>• Erarbeitung der Schichten des TCP/IP-Protokolls: Basisbandübertragung, Prüfverfahren, Routing/Vermittlungsschicht, Anwendungsprotokolle | Die Schülerinnen und Schüler<br>• beschreiben und erläutern Netzwerk-Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken (A), | • Arbeitsblätter zur Einführung in Netzwerke |
| 2. Simulation von Netzwerken / Netzwerk-Topologien<br>• Erarbeitung der Topologien: Peer-to-Peer, Sterntopologie, Baumtopologie, Vermaschtes Netz<br>• Simulation von Client-Server-Anwendungen<br>• Simulation von Protokollen der Anwendungsschicht (POP3, SMTP, etc.) | Die Schülerinnen und Schüler<br>• beschreiben und erläutern Netzwerk-Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken (A),<br>• analysieren und erläutern Protokolle zur Kommunikation in einem Client-Server-Netzwerk (A), | • Simulationssoftware FILIUS<br>• Arbeitsblätter und Skript zu FILIUS |
| 3. Analyse und Erläuterung kryptografischer Verfahren<br>• Erläuterung symmetrischer Verfahren: monoalphabetisch: Cäsar, polyalphabetisch: Vigenère<br>• Erläuterung asymmetrischer Verfahren: RSA, Diffie-Hellman<br>• Möglichkeiten der Signierung von Dokumenten (digitale Signatur, Steganografie)<br>• Analyse der Sicherheit verschiedener Verfahren und Auswirkungen auf den Datenschutz/Urheberrecht | Die Schülerinnen und Schüler<br>• analysieren und erläutern Algorithmen und Programme (A),<br>• stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D),<br>• analysieren und erläutern Eigenschaften, Funktionsweisen und Einsatzbereiche symmetrischer und asymmetrischer Verschlüsselungsverfahren (A).<br>• untersuchen und bewerten anhand von Fallbeispielen Auswirkungen des Einsatzes von Informatiksystemen sowie Aspekte der Sicherheit von Informatiksystemen, des Datenschutzes und des Urheberrechts (A),<br>• 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). | • Arbeitsblätter zu kryptografischen Verfahren<br><br>**Hier fehlt noch was zum Datenschutz / Urheberrecht** |

**Unterrichtsvorhaben Q1-Vb**

**Thema:** Modellierung und Implementierung von Client-Server-Anwendungen

**Leitfragen:** Wie lassen sich Client- und Server-Anwendungen programmieren? Wie kommunizieren Computer ausgehend von OSI-Layer 5-7?

**Vorhabenbezogene Konkretisierung:**

Ausgehend von einer einfachen Echo-Anwendung werden die beteiligten Komponenten (Echoserver und Echoclient) 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.

**Zeitbedarf:** 10 Stunden

| Unterrichtssequenzen | zu entwickelnde Kompetenzen | BSP, Medien, Materialien |
|---|---|---|
| 1. Entwicklung einer Echo-Anwendung<br>• Grundlegende Begriffe<br>• ZA-Klassen | Die Schülerinnen und Schüler<br>• ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M),<br>• modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M),<br>• modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M),<br>• verwenden bei der Modellierung geeigneter Problemstellungen Möglichkeiten der Polymorphie (M),<br>• ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M),<br>• stellen die Kommunikation zwischen Objekten grafisch dar (D),<br>• stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D),<br>• dokumentieren Klassen (D),<br>• analysieren und erläutern objektorientierte Modellierungen (A),<br>• implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I).<br>• modifizieren Algorithmen und Programme (I),<br>• implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I),<br>• testen Programme systematisch anhand von Beispielen und mit Hilfe von Testanwendungen (I).<br>• erläutern das Prinzip der Nebenläufigkeit (A),<br>• analysieren und erläutern Algorithmen und Methoden zur Client-Server-Kommunikation (A),<br>• entwickeln und implementieren Algorithmen und Methoden zur Client-Server-Kommunikation (I).<br>• beschreiben und erläutern Netzwerk-Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken (A),<br>• analysieren und erläutern Protokolle zur Kommunikation in einem Client-Server-Netzwerk (A),<br>• entwickeln und erweitern Protokolle zur Kommunikation in einem Client-Server-Netzwerk (M). | Echo-Server, Mailserver, CHAT-Programm, … |

**Unterrichtsvorhaben Q2-Ia**

**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?

**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.

**Zeitbedarf:** 25/30 Stunden

| Unterrichtssequenzen | zu entwickelnde Kompetenzen | Beispiele, Medien, Materialien |
|---|---|---|
| 1. Modellierung von relationalen Datenbanken<br>• Entity-Relationship-Diagramm:<br>Ermittlung von Entitäten, zugehörigen Attributen, Relationen und Kardinalitäten in Anwendungssituationen und Modellierung eines Datenbankentwurfs in Form eines Entity-Relationship-Diagramms<br>Erläuterung und Modifizierung einer Datenbankmodellierung<br>• Entwicklung einer Datenbank aus einem Datenbankentwurf:<br>Modellierung eines relationalen Datenbankschematas zu einem Entity-Relationship-Diagramm inklusive der Bestimmung von Primär- und Sekundärschlüsseln<br>• Redundanz, Konsistenz und Normalformen:<br>Untersuchung einer Datenbank hinsichtlich Konsistenz und Redundanz in einer Anwendungssituation.<br>Überprüfung von Datenbankschemata hinsichtlich der 1. bis 3. Normalform und Normalisierung (um Redundanzen zu vermeiden und Konsistenz zu gewährleisten) | Die Schülerinnen und Schüler<br>• ermitteln für anwendungsbezogene Problemstellungen Entitäten, zugehörige Attribute, Relationen und Kardinalitäten (M),<br>• stellen Entitäten mit ihren Attributen und die Beziehungen zwischen Entitäten mit Kardinalitäten in einem Entity-Relationship-Diagramm grafisch dar (D),<br>• modifizieren eine Datenbankmodellierung (M),<br>• modellieren zu einem Entity-Relationship-Diagramm ein relationales Datenbankschema (M),<br>• bestimmen Primär- und Sekundärschlüssel (M),<br>• analysieren und erläutern eine Datenbankmodellierung (A),<br>• erläutern die Eigenschaften normalisierter Datenbankschemata (A),<br>• überprüfen Datenbankschemata auf vorgegebene Normalisierungseigenschaften (D),<br>• überführen Datenbankschemata in die 1. bis 3. Normalform (M). | • Arbeitsblätter zur Einführung in Datenbanken auf der Schulhomepage:<br>○ Modellierung Zeugnis<br>○ Einführung ERM<br>○ ERD Bücherei<br>○ ERM -> RM<br>○ (Relationenalgebra)<br>○ Normalisierung |
| 2. Nutzung von relationalen Datenbanken<br>• Aufbau von Datenbanken und Grundbegriffe:<br>Entwicklung von Fragestellungen zur vorhandenen Datenbank<br>Analyse der Struktur der vorgegebenen Datenbank und Erarbeitung der Begriffe Tabelle, Attribut, Datensatz, Datentyp, Primärschlüssel, Fremdschlüssel, Datenbankschema<br>• SQL-Abfragen:<br>Analyse vorgegebener SQL-Abfragen und Erarbeitung der Sprachelemente von SQL (SELECT | Die Schülerinnen und Schüler<br>• modifizieren eine Datenbankmodellierung (M),<br>• bestimmen Primär- und Sekundärschlüssel (M),<br>• implementieren ein relationales Datenbankschema als Datenbank (I),<br>• analysieren und erläutern eine Datenbankmodellierung (A),<br>• ermitteln Ergebnisse von Datenbankabfragen über mehrere verknüpfte Tabellen (D),<br>• analysieren und erläutern die Syntax und Semantik einer Datenbankabfrage (A), | • Arbeitsblätter zur Einführung in SQL auf der Schulhomepage<br>• Tutorial der Seite SQL-Zoo (http://sqlzoo.net/wiki/Main_Page) |

| | | |
|---|---|---|
| (DISTINCT) …FROM, WHERE, AND, OR, NOT) auf einer Tabelle<br>Analyse und Erarbeitung von SQL-Abfragen auf einer und mehrerer Tabelle zur Beantwortung der Fragestellungen (JOIN, UNION, AS, GROUP BY,ORDER BY, ASC, DESC, COUNT, MAX, MIN, SUM, Arithmetische Operatoren: +, -, *, /, (…), Vergleichsoperatoren: =, <>, >, <, >=, <=, LIKE, BETWEEN, IN, IS NULL)<br>• Vertiefung an einem weiteren Datenbankbeispiel: Vertiefungen am Beispiel der Relationenalgebra | • verwenden die Syntax und Semantik einer Datenbankabfragesprache, um Informationen aus einen Datenbanksystem zu extrahieren (I).<br>• erläutern Eigenschaften und Aufbau von Datenbanksystemen unter dem Aspekt der sicheren Nutzung (A), | |
| 3. Gesellschaftliche Auswirkungen der Nutzung von Datenbanksystemen | • untersuchen und bewerten anhand von Fallbeispielen Auswirkungen des Einsatzes von Informatiksystemen sowie Aspekte der Sicherheit von Informatiksystemen, des Datenschutzes und des Urheberrechts (A),<br>• 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). | • Fallbeispiele zur Nutzung von Datenbanksystemen<br>• Spiel zum Missbrauch von Daten: DataDealer (http://datadealer.com/de) |

**Unterrichtsvorhaben Q2-Ib**

**Thema:** Prädikative / Logische Programmiersprachen

**Leitfragen:** Welche anderen Ansätze für Programmiersprachen gibt es noch? Muss es immer objektorientiert sein? Wie funktioniert ein Expertensystem?

**Vorhabenbezogene Konkretisierung:**

Der Ansatz einer prädikativen Programmiersprache wird dargestellt und gegen den bekannten Ansatz der objektorientierten Programmierung abgegrenzt. Wichtige Begriffe wie Prädikat, Konstante, Variable und Regel werden an Beispielen erlernt und angewendet. Die Verwendung der prädikativen Programmiersprache als Expertensystem steht im Vordergrund

Erweiterung des Regelsystems führt zu rekursiven Berechnungsverfahren, welche auch für arithmetische Probleme angewendet werden. Grundlegende Ideen des Backtrackings werden in anderen Zusammenhängen neu entdeckt.

Die Verwaltung von Listen in einer prädikativen Programmiersprache wird erläutert und in verschiedenen Anwendungssituationen (Schlange/Stapel) bzw. bei der Durchführung von Sortierverfahren geübt. Ebenfalls werden die bekannten Datenstrukturen des Binärbaums und des Graphen prädikativ implementiert.

Die Darstellung von Datenbanken in einer prädikativen Programmiersprache rundet das Unterrichtsvorhaben ab.

**Zeitbedarf:** 15 Stunden

| Unterrichtssequenzen | zu entwickelnde Kompetenzen | Beispiele, Medien, Materialien |
|---|---|---|
| 1. Einstieg in die Programmierung mit Prolog<br>• Begriffsdefinitionen: Regel, Fakt, Variable, Konstante, Prädikat<br>• Rekursive Programmierung:<br>Erweiterung eines Expertensystems<br>bekannte rekursive arithmetische Verfahren<br>Backtracking<br>• Dynamische Datenstrukturen:<br>Liste in Prolog<br>Schlange / Stapel in Prolog<br>Binärbaum in Prolog<br>Graphen in Prolog<br>Sortierverfahren mit Listen und Bäumen<br>• Datenbanken mit Prolog | Die Schülerinnen und Schüler<br>• stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D),<br>• ermitteln für anwendungsbezogene Problemstellungen Entitäten, zugehörige Attribute, Relationen und Kardinalitäten (M),<br>• implementieren ein relationales Datenbankschema als Datenbank (I),<br>• analysieren und erläutern eine Datenbankmodellierung (A),<br>• analysieren und erläutern Algorithmen und Programme (A),<br>• modifizieren Algorithmen und Programme (I),<br>• stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D),<br>• entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“ und „Teilen und Herrschen“ und „Backtracking“ (M),<br>• implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I),<br>• testen Programme systematisch anhand von Beispielen und mit Hilfe von Testanwendungen (I).<br>• implementieren Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (I),<br>• implementieren und erläutern iterative und rekursive Such- und Sortierverfahren unterschiedlicher Komplexitätsklassen (Speicherbedarf und Laufzeitverhalten) (I),<br>• nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I),<br>• beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A). | Arbeitsbuch PROLOG |

**Unterrichtsvorhaben Q2-IIa**

**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 g

**Vorhabenbezogene Konkretisierung:**

Ausgehend von einem konkreten Anwendungsbeispiel (zum Beispiel Compiler einer formalen Sprache) entwickeln die Schülerinnen und Schüler das Modell der Grammatik einer formalen Sprache und das Modell des endlichen Automaten. Die Schülerinnen 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 zu 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.

**Zeitbedarf:** 25/30 Stunden

| Unterrichtssequenzen | zu entwickelnde Kompetenzen | Beispiele, Medien, Materialien |
|---|---|---|
| 1. Einführung in Automaten/Grammatiken über Compilerbau<br>• Grundidee eines Compilers:<br>Scanner/Parser/Codierer<br>• Grammatiken:<br>Grammatik einer natürlichen Sprache<br>Grammatik einer künstlichen Sprache<br>Idee des Parsens<br>• Automaten:<br>erkennender Automat zu Symbolen einer Sprache<br>Modell des endlichen Automaten<br>Darstellungsformen<br>Sprache eines Automaten als regulärer Ausdruck<br>nichtdeterministische Automaten | Die Schülerinnen und Schüler<br>• analysieren und erläutern die Eigenschaften endlicher Automaten und Kellerautomaten einschließlich ihres Verhaltens bei bestimmten Eingaben (A),<br>• ermitteln die Sprache, die ein endlicher Automat oder ein Kellerautomat akzeptiert (D),<br>• entwickeln und modifizieren zu einer Problemstellung endliche Automaten oder Kellerautomaten (M),<br>• stellen endliche Automaten in Tabellen oder Graphen dar und überführen sie in die jeweils andere Darstellungsform (D),<br>• entwickeln zur Grammatik einer regulären oder kontextfreien Sprache einen zugehörigen endlichen Automaten oder einen Kellerautomaten (M). | |
| 2. Zusammenhang zwischen endlichen Automaten und regulären Grammatiken<br>• reguläre Grammatik:<br>Definition<br>Anwendungen<br>• Zusammenhang zu endlichen Automaten<br>• Grenzen der endlichen Automaten/regulären Grammatiken | Die Schülerinnen und Schüler<br>• analysieren und erläutern Grammatiken regulärer und kontextfreier Sprachen (A),<br>• modifizieren Grammatiken regulärer und kontextfreier Sprachen (M),<br>• ermitteln die formale Sprache, die durch eine Grammatik erzeugt wird (A),<br>• entwickeln zu einer regulären oder kontextfreien Sprache eine Grammatik, die die Sprache erzeugt (M),<br>• entwickeln zur akzeptierten Sprache eines Automaten eine zugehörige Grammatik (M),<br>• beschreiben an Beispielen den Zusammenhang zwischen Automaten und Grammatiken (D),<br>• zeigen/erläutern die Grenzen endlicher Automaten und regulärer Grammatiken/Sprachen im Anwendungszusammenhang auf (A). | |
| 3. Zusammenhang zwischen Kellerautomaten und kontextfreien Grammatiken<br>• kontextfreie Grammatik:<br>Definition<br>Anwendungen<br>• Modell des Kellerautomaten | Die Schülerinnen und Schüler<br>• analysieren und erläutern die Eigenschaften endlicher Automaten und Kellerautomaten einschließlich ihres Verhaltens bei bestimmten Eingaben (A),<br>• ermitteln die Sprache, die ein endlicher Automat | |

| | | |
|---|---|---|
| Definition<br>Darstellungsfomen<br>Anwendungen / Sprache eines Kellerautomaten<br>• Zusammenhang zwischen Kellerautomaten/kontextfreien Grammatiken<br>• Grenzen der Kellerautomaten | oder ein Kellerautomat akzeptiert (D),<br>• entwickeln und modifizieren zu einer Problemstellung endliche Automaten oder Kellerautomaten (M),<br>• entwickeln zur Grammatik einer regulären oder kontextfreien Sprache einen zugehörigen endlichen Automaten oder einen Kellerautomaten (M),<br>• analysieren und erläutern Grammatiken regulärer und kontextfreier Sprachen (A),<br>• modifizieren Grammatiken regulärer und kontextfreier Sprachen (M),<br>• entwickeln zu einer regulären oder kontextfreien Sprache eine Grammatik, die die Sprache erzeugt (M),<br>• beschreiben an Beispielen den Zusammenhang zwischen Automaten und Grammatiken (D). | |

**Unterrichtsvorhaben Q2-IIb**

**Thema:** Schritte eines Compilers einer formalen Sprache

**Leitfragen:** Wie funktioniert ein Compiler genau? Wie erkennt ein Compiler die Bestandteile einer Programmiersprache? Wie erkennt er die syntaktische Korrektheit?

**Vorhabenbezogene Konkretisierung:**

Ausgehend von der einfachen formalen Sprache der korrekt geklammerten arithmetischen Terme werden die Bestandteile eines Compilers dargestellt:

Der Scanner eines Compilers wird in Form eines endlichen Automaten, eines Kellerautomaten oder als rekursiver Grammatik-Parser (LL(k)) 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).

Schulspezifisch wird ganz konkret ein Scanner-Parser-Übersetzer für arithmetische Terme entwickelt, welcher aus einem arithmetischen Term (kontextfreie Sprache) einen Termbaum erzeugt und diesen in einen Stackmaschinen-Befehlssatz übersetzt.

Die Stackmaschine wird entwickelt und kann dann den erzeugten Stackmaschinencode ausführen und so den Termwert berechnen. Die Stackmaschine kann somit als Interpreter für Stackmaschinencode verstanden werden.

**Zeitbedarf:** 15 Stunden

| **Unterrichtssequenzen** | **zu entwickelnde Kompetenzen** | **Beispiele, Medien, Materialien** |
|---|---|---|
| 1. Die Schritte eines Compilers<br>• Scanner:<br>endlicher Automat als Grundlage<br>Symboltabelle und Tokenliste zur Verwaltung Erkennung des Quelltextes<br>Erweiterung des terminalen Alphabets der zu compelierenden formalen Sprache<br>Implementierung als endlicher Automat<br>• Parser:<br>kontextfreie Grammatik als Grundlage<br>Erweiterung des Sprachumfangs<br>Implementierung als kontextfreie Grammatik<br>• Übersetzer:<br>Überführung in eine andere Sprachebene | Die Schülerinnen und Schüler<br>• ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M),<br>• stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D),<br>• modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M),<br>• modellieren abstrakte und nicht abstrakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M),<br>• ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie/oder lineare und nichtlineare Datensammlungen zu (M),<br>• verwenden bei der Modellierung geeigneter Problemstellungen Möglichkeiten der Polymorphie (M),<br>• ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M),<br>• stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D),<br>• analysieren und erläutern objektorientierte Modellierungen (A),<br>• implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I),<br>• analysieren und erläutern Algorithmen und Programme (A),<br>• modifizieren Algorithmen und Programme (I),<br>• entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“ und „Teilen | Arbeitsblätter und Skript zum Compilerbau auf der Schulhomepage |

| | | |
|---|---|---|
| | und Herrschen“ und „Backtracking“ (M),<br>• implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I),<br>• testen Programme systematisch anhand von Beispielen und mit Hilfe von Testanwendungen (I).<br>• analysieren und erläutern die Eigenschaften endlicher Automaten und Kellerautomaten einschließlich ihres Verhaltens bei bestimmten Eingaben (A),<br>• ermitteln die Sprache, die ein endlicher Automat oder ein Kellerautomat akzeptiert (D),<br>• entwickeln und modifizieren zu einer Problemstellung endliche Automaten oder Kellerautomaten (M),<br>• stellen endliche Automaten in Tabellen oder Graphen dar und überführen sie in die jeweils andere Darstellungsform (D),<br>• analysieren und erläutern Grammatiken regulärer und kontextfreier Sprachen (A),<br>• modifizieren Grammatiken regulärer und kontextfreier Sprachen (M),<br>• ermitteln die formale Sprache, die durch eine Grammatik erzeugt wird (A),<br>• entwickeln zu einer regulären oder kontextfreien Sprache eine Grammatik, die die Sprache erzeugt (M),<br>• modellieren und implementieren Scanner, Parser und Interpreter zu einer gegebenen regulären Sprache (I). | |

**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?

**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, dass 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:** 10 Stunden

| **Unterrichtssequenzen** | **zu entwickelnde Kompetenzen** | **Beispiele, Medien, Materialien** |
|---|---|---|
| 1. Von-Neumann-Architektur und die Ausführung maschinennaher Programme:<br>• prinzipieller Aufbau einer von Neumann-Architektur mit CPU, Rechenwerk, Steuerwerk, Register und Hauptspeicher<br>• einige maschinennahe Befehlen und ihre Repräsentation in einem Binär-Code, der in einem Register gespeichert werden kann<br>• Analyse und Erläuterung der Funktionsweise eines einfachen maschinennahen Programms | Die Schülerinnen und Schüler<br>• erläutern die Ausführung eines einfachen maschinennahen Programms sowie die Datenspeicherung auf einer „Von-Neumann-Architektur“ (A),<br>• untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen (A). | geeigneter Modellrechner |
| 2. Grenzen der Automatisierbarkeit:<br>• Vorstellung des Halteproblems<br>• Unlösbarkeit des Halteproblems<br>• Beurteilung des Einsatzes von Informatiksystemen hinsichtlich prinzipieller Möglichkeiten und prinzipieller Grenzen | | Arbeitsblätter von der Schulhomepage zum Halteproblem |

## 2.2 Grundsätze der fachmethodischen und fachdidaktischen Arbeit

In Absprache mit der Lehrerkonferenz sowie unter Berücksichtigung des Schulprogramms hat die Fachkonferenz Informatik des Städtischen Gymnasiums Rheinbach die folgenden fachmethodischen und fachdidaktischen Grundsätze beschlossen. In diesem Zusammenhang beziehen sich die Grundsätze 1 bis 14 auf fächerübergreifende Aspekte, die auch Gegenstand der Qualitätsanalyse sind, die Grundsätze 15 bis 21 sind fachspezifisch angelegt.

*Überfachliche Grundsätze:*

1) Geeignete Problemstellungen zeichnen die Ziele des Unterrichts vor und bestimmen die Struktur der Lernprozesse.
2) Inhalt und Anforderungsniveau des Unterrichts entsprechen dem Leistungsvermögen der Schüler/innen.
3) Die Unterrichtsgestaltung ist auf die Ziele und Inhalte abgestimmt.
4) Medien und Arbeitsmittel sind schülernah gewählt.
5) Die Schüler/innen erreichen einen Lernzuwachs.
6) Der Unterricht fördert eine aktive Teilnahme der Schüler/innen.
7) Der Unterricht fördert die Zusammenarbeit zwischen den Schülern/innen und bietet ihnen Möglichkeiten zu eigenen Lösungen.
8) Der Unterricht berücksichtigt die individuellen Lernwege der einzelnen Schüler/innen.
9) Die Schüler/innen erhalten Gelegenheit zu selbstständiger Arbeit und werden dabei unterstützt.
10) Der Unterricht fördert strukturierte und funktionale Partner- bzw. Gruppenarbeit.
11) Der Unterricht fördert strukturierte und funktionale Arbeit im Plenum.
12) Die Lernumgebung ist vorbereitet; der Ordnungsrahmen wird eingehalten.
13) Die Lehr- und Lernzeit wird intensiv für Unterrichtszwecke genutzt.
14) Es herrscht ein positives pädagogisches Klima im Unterricht.

*Fachliche Grundsätze:*

15) Der Unterricht unterliegt der Wissenschaftsorientierung und ist dementsprechend eng verzahnt mit seiner Bezugswissenschaft.
16) Der Unterricht ist problemorientiert und soll von realen Problemen ausgehen und sich auf solche rückbeziehen.
17) Der Unterricht folgt dem Prinzip der Exemplarizität und soll ermöglichen, informatische Strukturen und Gesetzmäßigkeiten in den ausgewählten Problemen und Projekten zu erkennen.
18) Der Unterricht ist anschaulich sowie gegenwarts- und zukunftsorientiert und gewinnt dadurch für die Schülerinnen und Schüler an Bedeutsamkeit.
19) Der Unterricht ist handlungsorientiert, d.h. projekt- und produktorientiert angelegt.
20) Im Unterricht werden sowohl für die Schule didaktisch reduzierte als auch reale Informatiksysteme aus der Wissenschafts-, Berufs- und Lebenswelt eingesetzt.
21) Der Unterricht beinhaltet reale Begegnung mit Informatiksystemen.

## 2.3 Grundsätze der Leistungsbewertung und Leistungsrückmeldung

Auf der Grundlage von §13 - §16 der APO-GOSt sowie Kapitel 3 des Kernlehrplans Informatik für die gymnasiale Oberstufe hat die Fachkonferenz des Städtischen Gymnasium Rheinbach im Einklang mit dem entsprechenden schulbezogenen Konzept die nachfolgenden Grundsätze zur Leistungsbewertung und Leistungsrückmeldung beschlossen. Die nachfolgenden Absprachen stellen die Minimalanforderungen an das lerngruppenübergreifende gemeinsame Handeln der Fachgruppenmitglieder dar. Bezogen auf die einzelne Lerngruppe kommen ergänzend weitere der in den Folgeabschnitten genannten Instrumente der Leistungsüberprüfung zum Einsatz.

### 2.3.1 Beurteilungsbereich Klausuren & Facharbeiten

**Verbindliche Absprachen:**

Bei der Formulierung von Aufgaben werden die für die Abiturprüfungen geltenden Operatoren des Faches Informatik schrittweise eingeführt, erläutert und dann im Rahmen der Aufgabenstellungen für die Klausuren benutzt.

**Instrumente:**

- Einführungsphase: 1 Klausur im 1. Halbjahr, 2 Klausuren im 2. Halbjahr

  Dauer der Klausur: 90 Minuten

  Der Einsatz eines Computers ist nicht vorgesehen, es sei denn die gültigen Vorgaben zum Zentralabitur für den jeweiligen Jahrgang geben dies explizit an.
- Q 1: 2 Klausuren je Halbjahr

  Dauer der Klausuren: 90 Minuten (GK), 135 Minuten (LK)
- Q 2.1: 2 Klausuren

  Dauer der Klausuren: 135 Minuten (GK), 225 Minuten (LK)
- Q 2.2: 1 Klausur unter Abiturbedingungen
- Der Einsatz eines Computers ist in der gesamten Qualifikationsphase nicht vorgesehen, es sei denn die gültigen Vorgaben zum Zentralabitur für den jeweiligen Jahrgang geben dies explizit an.
- Anstelle einer Klausur kann gemäß dem Beschluss der Lehrerkonferenz in Q 1.2 eine Facharbeit geschrieben werden.

Die Aufgabentypen, sowie die Anforderungsbereiche I-III sind entsprechend den Vorgaben in Kapitel 3 des Kernlehrplans zu beachten.

**Kriterien**

Die Bewertung der schriftlichen Leistungen in Klausuren erfolgt über ein Raster mit Hilfspunkten, die im Erwartungshorizont den einzelnen Kriterien zugeordnet sind.

Spätestens ab der Qualifikationsphase orientiert sich die Zuordnung der Hilfspunktsumme zu den Notenstufen an dem Zuordnungsschema des Zentralabiturs.

Von diesem kann aber im Einzelfall begründet abgewichen werden, wenn sich z.B. besonders originelle Teillösungen nicht durch Hilfspunkte gemäß den Kriterien des Erwartungshorizontes abbilden lassen oder eine Abwertung wegen besonders schwacher Darstellung (APO-GOSt §13 (2)) angemessen erscheint.

Die Note ausreichend minus (4 Punkte) soll bei Erreichen von 40 % der Hilfspunkte erteilt werden.

**Bewertung von Facharbeiten**

Facharbeiten können die erste Klausur im 2. Halbjahr der Q1 ersetzen. Dabei wird das Facharbeitsfach vom Schüler gewählt und durch die Oberstufenkoordination festgelegt.

Die Bewertung der Facharbeiten erfolgt nach folgenden Kriterien:

| | |
|---|---|
| Gesamteindruck und Layout | Titelblatt, Seitenlayout, Schriftsatz, Schriftart, … |
| Verzeichnisse | Automatische Erzeugung? Sinnvolle Abstufung der Überschriften, Bezeichnung der Abbildungen, Tabellen und Quellenangaben (z.B. Citavi verwendet) |
| Rechtschreibung und Grammatik | Häufige Fehler führen zu Abzügen, Rechtschreib- und Grammatikprüfung genutzt? |
| Sinnvoller Einsatz der Textverarbeitung | Möglichkeiten von WORD, WRITER, LaTeX o.ä. sowie von Citavi, BibTeX o.ä. ausgenutzt? |
| Vollständigkeit | Selbstständigkeitserklärung, alle vereinbarten Teilaspekte enthalten? |
| Einhaltung der formalen Vorgaben | vergleiche die schulischen, formalen Vorgaben, vollständig eingehalten? |
| Sinnvoller Anhang auf CD/DVD/USB-Stick | Quelltexte, Datensätze, fertige Programme, zusätzliche Bilder/Fotos, Quellen, pdf-Dokumente, Webseiten, Datenbanken usw. in geeigneter Form und gut gegliedert (Ordnerstruktur, Autostart, html-Übersicht)? |
| Eigenanteil an der Themenfindung | Wurde das Thema selbstständig gewählt, geeignet eingegrenzt und weitgehend frei formuliert (eine leichte Hilfe durch den Lehrer ist normal)? |
| Angemessene Gliederung | Entspricht die Gliederung einem fachmethodisch geeignetem Zugang zum Thema? |
| Sichere Anwendung geeigneter Fachmethoden, Algorithmen und Programmierung | Wurden geeignete Fachmethoden gewinnbringend eingesetzt, z.B. die Erstellung von Struktogrammen oder Programmablaufplänen für Algorithmen, von ER-Diagrammen für Datenbanken, von Sequenzdiagrammen für Objektinteraktionen?<br><br>Ist die Programmierung gelungen, der Bau eines elektrotechnisch-digitalen Gerätes erfolgreich umgesetzt, … worden?<br><br>Der praktische Teil (Implementation, Elektrotechnik, …) soll an dieser Stelle besonderes Gewicht erhalten und entsprechend stark gewichtet in die Note eingehen! |
| Geeignete Auswahl an Quellen und Materialien | Sind die genutzten Quellen und Materialien geeignet gewählt, ihre Validität geprüft und auf angemessenem Niveau? |
| Korrekte und sicher angewendete | Wird konsistent auf angemessenem und fachsprach- |

| | |
|---|---|
| Fachsprache | lich gutem Niveau geschrieben? Werden die Fachbegriffe treffsicher verwendet? |
| Gelungene Fokussierung | Ist der Inhalt auf das Wesentliche fokussiert und Wichtiges von Unwichtigem getrennt, sind dadurch keine Redundanzen enthalten und wird das Wichtige hinreichend ausführlich dargelegt? |
| Sinnvolle Grafiken, Diagramme, … | Werden Diagramme, Grafiken usw. gewinnbringend eingebunden oder stellen sie nur verzichtbare „Platzfüller dar“, haben die eingebundenen Diagramme, Grafiken usw. direkten Bezug zum umgebenden Text? |
| Sinnvolle Einbindung von Zitaten und Abbildungen im Text | Werden Zitate und Abbildungen gewinnbringend eingebunden oder stellen sie nur verzichtbare „Platzfüller dar“, haben die eingebundenen Zitate und Abbildungen direkten Bezug zum umgebenden Text? |
| Lauffähigkeit des ggf. erstellten Programmes, Gerätes, … | Ist das gebaute Gerät, das erstellte Programm, die programmierte Datenbank usw. voll lauffähig und intuitiv bedienbar? Sind noch Fehler enthalten, ist die Bedienung komplex und erschließt sich nicht intuitiv? Ist eine geeignete (knappe) Dokumentation vorhanden? |
| Ertrag der Arbeit und Durchdingungstiefe der Problemstellung | Ist das Problem sorgfältig herausgearbeitet und mit einer angemessenen Tiefe durchdrungen worden. Wurden zielführend Erkenntnisse gesammelt um darauf aufbauend ein Fazit zu ziehen? Oder ist das Thema nur oberflächlich beschrieben, so dass keine tiefgehenden Erkenntnisse möglich sind? |
| Kritische Reflexion der Arbeit hinsichtlich eigener Ergebnisse und deren Bewertung im Hinblick auf die Aufgabenstellung | Wurde der Ertrag und das Vorgehen bei der Arbeit selbstkritisch analysiert und bezogen auf die Aufgaben- bzw. Problemstellung eine eigene Bewertung formuliert? Ist diese treffsicher und passend zur Arbeit? |
| Kritischer und sinnvoller Ausblick auf vertiefenden oder erweiternde Möglichkeiten hinsichtlich im Rahmen der Facharbeit unbehandelter Aspekte | Sind noch offene Aspekte geblieben oder haben sich derartige im Laufe der Facharbeit ergeben? Sind die möglichen Erweiterungen sinnvoll dargestellt und auf die eigene Arbeit bezogen? |

Ermittlung der Note durch gewichteten, aufgerundeten Mittelwert aus der dreifachen Punktnote im Bereich „Reflexion der eigenen Arbeitsweise und der Facharbeit“ der siebenfachen Punktnote im Bereich „Fachliche, methodische und inhaltliche Aspekte der Facharbeit“ und der zweifachen Punktnote im Bereich „Formale und methodische Aspekte“

Die jeweils greifenden Kriterien und i9hre Gewichtung wird nach Festlegung des Facharbeitsthemas im Gespräch mit dem Schüler / der Schülerin festgelegt und in einem individuell angepassten bewertungsbogen festgehalten.

Die Bewertungsbögen sind nach folgendem Muster zu gestalten:

## Bewertungsbogen Facharbeit Qualifikationphase gymnasiale Oberstufe

**Städtisches Gymnasium Rheinbach Fachbereich Informatik**

**Name Schüler*in:** **Datum:**

**Thema der Facharbeit (Wortlaut gemäß Vereinbarung):**

**Bewertung:**

| **Formale und methodische Aspekte** | | | | |
|---|---|---|---|---|
| Gesamteindruck und Layout | | | | |
| Verzeichnisse | | | | |
| Rechtschreibung und Grammatik | | | | |
| Sinnvoller Einsatz der Textverarbeitung | | | | |
| Vollständigkeit | | | | |
| Einhaltung der formalen Vorgaben | | | | |
| Sinnvoller Anhang auf CD/DVD/USB-Stick | | | | |
| **Note im Bereich formale und methodische Aspekte** | | | | |

| **Fachliche, methodische und inhaltliche Aspekte der Facharbeit** | | | | |
|---|---|---|---|---|
| Eigenanteil an der Themenfindung | | | | |
| Angemessene Gliederung | | | | |
| Sichere Anwendung geeigneter Fachmethoden, Algorithmen und Programmierung | | | | |
| Geeignete Auswahl an Quellen und Materialien | | | | |
| Korrekte und sicher angewendete Fachsprache | | | | |
| Gelungene Fokussierung | | | | |
| Sinnvolle Grafiken, Diagramme, … | | | | |
| Sinnvolle Einbindung von Zitaten und Abbildungen im Text | | | | |
| Lauffähigkeit des ggf. erstellten Programmes, Gerätes, … | | | | |
| **Note im Bereich Fachliche, methodische und inhaltlische Aspekte der Facharbeit** | | | | |

| **Reflexion der eigenen Arbeitsweise und der Facharbeit** | | | | |
|---|---|---|---|---|
| Kritische Reflexion der Arbeit hinsichtlich eigener Ergebnisse und deren Bewertung im Hinblick auf die Aufgabenstellung | | | | |
| Kritischer und sinnvoller Ausblick auf vertiefenden oder erweiternde Möglichkeiten hinsichtlich im Rahmen der Facharbeit unbehandelter Aspekte | | | | |
| **Note im Bereich Reflexion der eigenen Arbeitsweise und der Facharbeit** | | | | |

**Gesamtbewertung:**

| **Note „Reflexion der Arbeit“ (3fach)** | **Note „Fachliche Aspekte der Arbeit“ (7fach)** | **Note „Formale & methodische Aspekte“ (2fach)** | **Note** |
|---|---|---|---|
| | | | |

### 2.3.2 Beurteilungsbereich Sonstige Mitarbeit

Den Schülerinnen und Schülern werden die Kriterien zum Beurteilungsbereich „sonstige Mitarbeit“ zu Beginn des Schuljahres genannt.

**Verbindliche Absprachen der Fachkonferenz**

- Alle Schülerinnen und Schüler führen in der Einführungsphase in Kleingruppen ein Kurzprojekt durch und fertigen dazu eine Arbeitsmappe mit Arbeitstagebuch an. Dies wird in die Note für die Sonstige Mitarbeit einbezogen.
- In der Qualifikationsphase erstellen, dokumentieren und präsentieren die Schülerinnen und Schüler in Kleingruppen ein anwendungsbezogenes Softwareprodukt. Dies wird in die Note für die Sonstige Mitarbeit einbezogen.

**Leistungsaspekte**

Mündliche Leistungen

- Beteiligung am Unterrichtsgespräch
- Zusammenfassungen zur Vor- und Nachbereitung des Unterrichts
- Präsentation von Arbeitsergebnissen
- Referate
- Mitarbeit in Partner-/Gruppenarbeitsphasen

Praktische Leistungen am Computer

- Implementierung, Test und Anwendung von Informatiksystemen

Sonstige schriftliche Leistungen

- Arbeitsmappe und Arbeitstagebuch zu einem durchgeführten Unterrichtsvorhaben
- Lernerfolgsüberprüfung durch kurze schriftliche Übungen

  In Kursen, in denen höchstens 50% der Kursmitglieder eine Klausur schreiben, finden schriftliche Übungen mindestens einmal pro Kurshalbjahr statt, in anderen Kursen entscheidet über die Durchführung die Lehrkraft.

  Schriftliche Übung dauern ca. 20 Minuten und umfassen den Stoff der letzten ca. 4–6 Stunden.
- Bearbeitung von schriftlichen Aufgaben im Unterricht

**Kriterien**

Die folgenden allgemeinen Kriterien gelten sowohl für die mündlichen als auch für die schriftlichen Formen der sonstigen Mitarbeit.

Die Bewertungskriterien stützen sich auf

- die Qualität der Beiträge,
- die Quantität der Beiträge und
- die Kontinuität der Beiträge.

Besonderes Augenmerk ist dabei auf

- die sachliche Richtigkeit,
- die angemessene Verwendung der Fachsprache,

- die Darstellungskompetenz,
- die Komplexität und den Grad der Abstraktion,
- die Selbstständigkeit im Arbeitsprozess,
- die Präzision und
- die Differenziertheit der Reflexion zu legen.

Bei Gruppenarbeiten auch auf

- das Einbringen in die Arbeit der Gruppe,
- die Durchführung fachlicher Arbeitsanteile und
- die Qualität des entwickelten Produktes.

Bei Projektarbeit darüber hinaus auf

- die Dokumentation des Arbeitsprozesses,
- den Grad der Selbstständigkeit,
- die Reflexion des eigenen Handelns und
- die Aufnahme von Beratung durch die Lehrkraft.

**Grundsätze der Leistungsrückmeldung und Beratung**

Die Grundsätze der Leistungsbewertung werden zu Beginn eines jeden Halbjahres den Schülerinnen und Schülern transparent gemacht. Leistungsrückmeldungen können erfolgen

- nach einer mündlichen Überprüfung,
- bei Rückgabe von schriftlichen Leistungsüberprüfungen,
- nach Abschluss eines Projektes,
- nach einem Vortrag oder einer Präsentation,
- bei auffälligen Leistungsveränderungen,
- auf Anfrage,
- als Quartalsfeedback und
- zu Eltern- oder Schülersprechtagen.

Die Leistungsrückmeldung kann

- durch ein Gespräch mit der Schülerin oder dem Schüler,
- durch einen Feedbackbogen,
- durch die schriftliche Begründung einer Note oder
- durch eine individuelle Lern-/Förderempfehlung

erfolgen.

Leistungsrückmeldungen erfolgen auch in der Einführungsphase im Rahmen der kollektiven und individuellen Beratung zur Wahl des Faches Informatik als fortgesetztes Grund- oder Leistungskursfach in der Qualifikationsphase.

Eine Leistungsrückmeldung erfolgt verbindlich zum Ende eines jeden Quartals mittels eines bidirektionalen Leistungsrückmeldebogen nach folgendem Verfahren und mit folgendem Bogen:

**Anleitung zum Umgang mit dem bidirektionalen Schülerleistungsrückmeldebogen Informatik S2**

1. **Kriterien zur Leistungsbeurteilung wiederholen**,
   Vorstellung wurde zu Beginn des Schuljahres vorgenommen.
   Ein Hinweis auf die Homepage – Kriterien sind dort u.a. im Schulcurriculum veröffentlicht – ist sicher hilfreich.
2. **Rückmeldebogen erklären**, v.a. auch „WARUM“
   Transparenz der Leistungsbeurteilung, Kompetenz Selbsteinschätzung, bidirektionale Rückmeldung
3. **Rückmeldebogen austeilen**
   **Hausaufgabe**: Ausfüllen (Kriterien beachten) und in der nächsten Stunde abgeben!
4. **Rückmeldebogen um Lehrereinschätzungen und Noten ergänzen**, Abgleich mit Schülereinschätzung vornehmen
5. **In Folgestunde Bögen wieder austeilen**, bei starken Abweichungen zwischen Schülereinschätzung und Lehrereinschätzung **Beratungsgespräch** vereinbaren und durchführen.

## Rückmeldebogen Sonstige Mitarbeit S2

| **Kriterium** | **Schüler** | **Lehrer** |
|---|---|---|
| Quantitative und qualitative Leistungen im Unterrichtsgespräch | | |
| Leistungen in Partnerarbeiten | | |
| Leistungen in Gruppenarbeiten | | |
| Leistungen im Bereich Algorithmen erklären und darstellen (Struktogramme/PAPs) | | |
| Leistungen beim Entwickeln und Anwenden von Algorithmen | | |
| Leistungen Implementationen in JAVA | | |
| Leistungen beim Präsentieren von erarbeiteten Inhalten und Implementationen | | |
| Leistungen beim Beschreiben und Erklären von Quelltexten in JAVA | | |
| Leistungen beim Implementieren von SQL-Befehlen (Reihe Datenbanken) | | |
| Leistungen im Bereich Analyse ohne Algorithmen (z.B. Netzwerkprotokolle, Automaten und Sprachen) | | |
| Transferleistungen und Leistungen bei vernetzten Kontexten (incl. Implementationen) | | |
| **Gesamtnote (Sonstige Mitarbeit)** | | |
| **Ggf. Klausurnoten** | | |
| **Gesamtnote (nur zum Halbjahres-/Schuljahresende)** | | |

# 3 Entscheidungen zu fach- und unterrichtsübergreifenden Fragen

Die Fachkonferenz Informatik hat sich im Rahmen des Schulprogramms für folgende zentrale Schwerpunkte entschieden:

**Zusammenarbeit mit anderen Fächern**

Im Informatikunterricht werden Kompetenzen anhand informatischer Inhalte in verschiedenen Anwendungskontexten erworben, in denen Schülerinnen und Schülern aus anderen Fächern Kenntnisse mitbringen können. Diese können insbesondere bei der Auswahl und Bearbeitung von Softwareprojekten berücksichtigt werden und in einem hinsichtlich der informatischen Problemstellung angemessenem Maß in den Unterricht Eingang finden. Da im Inhaltsfeld Informatik, Mensch und Gesellschaft auch gesellschaftliche und ethische Fragen im Unterricht angesprochen werden, soll eine mögliche Zusammenarbeit mit den Fächern Sozialwissenschaften und Philosophie in einer gemeinsamen Fachkonferenz ausgelotet werden.

**Projekttage**

Alle zwei Jahre werden am Städtischen Gymnasium Rheinbach Projekttage angeboten. Die Fachkonferenz Informatik bietet in diesem Zusammenhang mindestens ein Projekt für Schülerinnen und Schüler der gymnasialen Oberstufe an.

**Vorbereitung auf die Erstellung der Facharbeit**

Möglichst schon zweiten Halbjahr der Einführungsphase, spätestens jedoch im ersten Halbjahr des ersten Jahres der Qualifikationsphase werden im Unterricht an geeigneten Stellen Hinweise zur Erstellung von Facharbeiten gegeben. Das betrifft u. a. Themenvorschläge, Hinweise zu den Anforderungen und zur Bewertung. Es wird vereinbart, dass nur Facharbeiten vergeben werden, die mit der eigenständigen Entwicklung eines Softwareproduktes verbunden sind.

**Exkursionen**

In der Einführungsphase wird i.d.R. im Rahmen des Unterrichtsvorhabens „Geschichte der digitalen Datenverarbeitung und die Grundlagen des Datenschutzes“ eine Exkursion zum Heinz Nixdorf MuseumsForum durchgeführt. Die außerunterrichtliche Veranstaltung wird im Unterricht vor- und nachbereitet.

Im Rahmen der Differenzierungskurse werden Exkursionen ins zdi-Schülerlabor Infosphere der RWTH Aachen durchgeführt.

**Wettbewerbe**

Das SGR bietet die Teilnahme an verschiedenen Wettbewerben im Bereich Informatik an, u.a.:

- Informatik-Biber
- Invent-a-chip
- Bundeswettbewerb Informatik
- Intel-Leibniz-Challenge

# 4 Qualitätssicherung und Evaluation

Durch Diskussion der Aufgabenstellung von Klausuren in Fachdienstbesprechungen und eine regelmäßige Erörterung der Ergebnisse von Leistungsüberprüfungen wird ein hohes Maß an fachlicher Qualitätssicherung erreicht.

Das schulinterne Curriculum (siehe 2.1) ist zunächst bis 2017 für den ersten Durchgang durch die gymnasiale Oberstufe nach Erlass des Kernlehrplanes verbindlich. Erstmalig nach Ende der Einführungsphase im Sommer 2015, werden in einer Sitzung der Fachkonferenz Erfahrungen ausgetauscht und ggf. Änderungen für den nächsten Durchgang der Einführungsphase beschlossen, um erkannten ungünstigen Entscheidungen schnellstmöglich entgegenwirken zu können.

Nach Abschluss des Abiturs 2017 wir die Fachkonferenz Informatik auf der Grundlage ihrer Unterrichtserfahrungen eine Gesamtsicht des schulinternen Curriculums vornehmen und ggf. eine Beschlussvorlage für die erste Fachkonferenz des folgenden Schuljahres erstellen.