Schulinterner Lehrplan des Städt. Gymnasiums Wermelskirchen zum Kernlehrplan für die gymnasiale Oberstufe

Informatik

Inhalt

1 Vorwort zu den schulinternen Lehrplänen des Städtischen Gymnasiums Wermelskirchen

Das Städtische Gymnasium Wermelskirchen, gegründet im Jahr 1867 am heutigen Schulstandort an der Stockhauser Straße, ist eine Schule mit einer dementsprechend langen Tradition und einem starken Engagement für die Ausbildung junger Menschen. Unser Leitbild spiegelt unsere Werte und Ziele wider und dient als Ankerpunkt für alles, was wir tun. Es ist uns wichtig, dass wir gemeinschaftlich handeln, mit Herz und Verstand dabei sind und uns für das Leben bilden.

Die Schulgemeinde des Städtischen Gymnasiums Wermelskirchen besteht aus Schülerinnen und Schülern, Lehrerinnen und Lehrern, Eltern und anderen Mitwirkenden. Gemeinsam gehen wir achtsam mit Mitmenschen und Natur um, fördern Empathie, Toleranz, Rücksichtnahme und Wertschätzung und legen Wert auf Demokratiebewusstsein, Kommunikation und Kooperation.

Unsere Schule bietet ein breites Angebot, in dem jedes Kind sein Potential entfalten kann und Verantwortung für das eigenen Lernen übernimmt. Kreativität, Selbstständigkeit, die Fähigkeit zur Selbstreflexion und Medienkompetenz sind hierfür beispielhafte Gelingensbedingungen.

Als einziges Gymnasium vor Ort mit einem Standort in der der Nähe der Wermelskirchener Innenstadt hat das Städtische Gymnasium Wermelskirchen ein weiträumiges Einzugsgebiet. Bis zu 10% der Schülerinnen und Schüler eines Jahrgangs kommen aus umliegenden Städten. Die Stadt selbst mit knapp 35.000 Einwohnerinnen und Einwohnern umfasst ein sehr weiträumiges und in Teilen ländliches Stadtgebiet mit vielen großen und kleineren Ortschaften, aus denen einige Kinder lange Anfahrtswege zur Schule haben. In der Stadt gibt es neben dem Gymnasium noch bis zum Schuljahr 2027/28 eine auslaufende Sekundarschule und seit dem Schuljahr 2022/23 eine sich im Aufbau befindende Gesamtschule.

Unsere Schule ist in der Sekundarstufe I vier- bis fünfzügig ohne gebundenen Ganztag und die gesamte Schülerschaft umfasst derzeit etwa 950 Schülerinnen und Schüler, die von etwa 80 Lehrerinnen und Lehrern unterrichtet werden.

Die vorliegenden schulinternen Lehrpläne für die Sekundarstufe I wurden im Zuge der Umstellung von G8 auf G9 neu erstellt. Sie sind ein Ausdruck unseres Leitbildes und unserer Werte und zeigen, wie wir uns das Lernen und Lehren an unserer Schule vorstellen. Alle Lehrpläne basieren auf den Vorgaben der Kernlehrpläne NRW der einzelnen Fächer.

2 Die Fachgruppe Informatik des Städt. Gymnasiums Wermelskirchen

Das Städt. Gymnasium Wermelskirchen ist eine vier- bis fünfzügige Schule mit derzeit ca. 1200 Schülerinnen und Schülern im ländlich gelegenen Wermelskirchen bei Köln.

Das Fach Informatik wird am Städt. Gymnasium Wermelskirchen in der Jahrgangsstufe 6 und ab der Jahrgangsstufe 9 im Wahlpflichtbereich II unterrichtet. Regelmäßig kommt hier ein Kurs zustande, so dass ca. 25 Schülerinnen und Schüler diesen zweijährigen Kurs besuchen, in dem in altersstufengerechter Weise unter anderem die Grundlagen der Programmierung sowie die technische Informatik thematisiert werden. Darüber hinaus erhalten die Schülerinnen und Schüler dieses Differenzierungskurses einen tiefergehenden Einblick in eine einfache Programmierumgebung wie z.B. zu Processing. In der Regel wird ebenso ein Einblick in die Robotik am Beispiel der Lego - Roboter gegeben.

Das Fach Informatik wird in der Sekundarstufe II am Städt. Gymnasium Wermelskirchen als Wahlfach angeboten. In der Einführungsphase können die Schülerinnen und Schüler das Fach als dreistündigen Grundkurs belegen.

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

Durch projektartiges Vorgehen, offene Aufgaben und Möglichkeiten, Problemlösungen zu verfeinern und zu optimieren, entspricht der Informatikunterricht in besonderem Maße dem Erziehungsziel, Leistungsbereitschaft zu fördern ohne zu überfordern.

Darüber hinaus trägt er zu einer breitgefächerten Allgemeinbildung bei, bietet gleichzeitig Raum für individuelle Spezialisierungen und ermöglicht verantwortungsvolles Handeln in einer sich schnell wandelnden und von technischen Fortschritten geprägten Welt.

Die gemeinsame Entwicklung von Materialien und Unterrichtsvorhaben, die Evaluation von Lehr- und Lernprozessen sowie die stetige Überprüfung und eventuelle Modifikation des schulinternen Curriculums durch die Fachkonferenz Informatik stellt einen wichtigen Beitrag zur Qualitätssicherung und -entwicklung des Unterrichts dar.

Das Städt. Gymnasium Wermelskirchen verfügt über drei Informatikräume und einer Mediothek. Alle Computerarbeitsplätze sind am schulinternen Rechnernetz angeschlossen, so dass Schülerinnen und Schüler über einen Zugang zum zentralen Server der Schule Zugriff auf ihre eigenen Daten zur Recherche im Internet oder Bearbeitung schulischer Aufgaben haben.

Die Arbeit am Rechner erfolgt im Normalfall in Einzel- oder Partnerarbeit.

3 Entscheidungen zum Unterricht

3.1 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, wobei die Reihenfolge innerhalb eines Schuljahres geändert werden darf. Das Übersichtsraster dient dazu, den Kolleginnen und Kollegen einen schnellen Überblick über die Zuordnungen der Unterrichtsvorhaben zu den einzelnen Jahrgangsstufen sowie den im Kernlehrplan genannten Kompetenzen, Inhaltsfeldern und inhaltlichen Schwerpunkten zu verschaffen. Der ausgewiesene Zeitbedarf versteht sich als grobe Orientierungsgröße, die nach Bedarf über- oder unterschritten werden kann. Um Spielraum für Vertiefungen, besondere Schülerinteressen, aktuelle Themen bzw. die Erfordernisse anderer besonderer Ereignisse (z.B. Praktika, Kursfahrten o.ä.) zu erhalten, wurden im Rahmen dieses schulinternen Lehrplans nur 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, besitzt die exemplarische Ausweisung "konkretisierter Unterrichtsvorhaben" (Kapitel 2.1.2) empfehlenden Charakter. 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, fachübergreifenden Kooperationen, Lernmitteln und Lernorten sowie vorgesehenen Leistungsüberprüfungen, die im Einzelnen auch den Kapiteln 2.2 bis 2.3 zu entnehmen sind. Abweichungen von den vorgeschlagenen Vorgehensweisen bezüglich der konkretisierten Unterrichtsvorhaben sind im Rahmen der pädagogischen Freiheit der Lehrkräfte jederzeit möglich. Sicherzustellen bleibt allerdings auch hier, dass im Rahmen der Umsetzung der Unterrichtsvorhaben insgesamt alle konkretisierten Kompetenzerwartungen des Kernlehrplans Berücksichtigung finden.

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

3.1.1 Übersichtsraster Unterrichtsvorhaben

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

3.1.1.1 Einführungsphase

Thema	Inhaltsfelder	inhaltliche Schwerpunkte
Unterrichtsvorhaben E-I Grundlagen der objektorientierten Analyse und Programmierung anhand von grafischen Beispielen	• Daten und ihre Strukturierung • Algorithmen • Formale Sprachen und Automaten	• Dateisystem • Informatiksysteme • Objekte und Klassen • Analyse, Entwurf und Implementierung einfacher Algorithmen • Syntax und Semantik einer Programmiersprache
Unterrichtsvorhaben E-II Modellierung und Implementierung von Klassen- und Objektbeziehungen anhand von grafischen Beispielen	• Daten und ihre Strukturierung • Algorithmen • Formale Sprachen und Automaten	• Objekte und Klassen • Analyse, Entwurf und Implementierung einfacher Algorithmen • Syntax und Semantik einer Programmiersprache
Unterrichtsvorhaben E-III Vorbereitung von dynamischen Datenstrukturen und Verwendung abstrakter Klassen	• Daten und ihre Strukturierung • Algorithmen • Formale Sprachen und Automaten	• Objekte und Klassen • Analyse, Entwurf und Implementierung einfacher Algorithmen • Syntax und Semantik einer Programmiersprache
Unterrichtsvorhaben E-IV Analyse von Such- und Sortieralgorithmen und Auswirkungen von Informatiksystemen auf die Gesellschaft	• Daten und ihre Strukturierung • Algorithmen • Formale Sprachen und Automaten • Informatiksysteme • Informatik, Mensch und Gesellschaft	• Objekte und Klassen • Analyse, Entwurf und Implementierung einfacher Algorithmen • Algorithmen zum Suchen und Sortieren • Syntax und Semantik einer Programmiersprache • Digitalisierung • Wirkung der Automatisierung • Geschichte der automatischen Datenverarbeitung

3.1.1.2 Qualifikationsphase – Grundkurs

Qualifikationsphase I

Thema	Inhaltsfelder	inhaltliche Schwerpunkte
Unterrichtsvorhaben Q1-I Wiederholung der objektorientierten Modellierung und Programmierung anhand einer kontextbezogenen Problemstellung	• Daten und ihre Strukturierung • Algorithmen • Formale Sprachen und Automaten • Informatiksysteme	• Objekte und Klassen • Analyse, Entwurf und Implementierung von Algorithmen • Syntax und Semantik einer Programmiersprache • Nutzung von Informatiksystemen
Unterrichtsvorhaben Q1-II Modellierung und Implementierung von Anwendungen mit dynamischen, linearen Datenstrukturen	• Daten und ihre Strukturierung • Algorithmen • Formale Sprachen und Automaten	• Objekte und Klassen • Analyse, Entwurf und Implementierung einfacher Algorithmen • Algorithmen in ausgewählten informatischen Kontexten • Syntax und Semantik einer Programmiersprache
Unterrichtsvorhaben Q1-III Suchen und Sortieren auf linearen Datenstrukturen	• Daten und ihre Strukturierung • Algorithmen • Formale Sprachen und Automaten	• Analyse, Entwurf und Implementierung von Algorithmen • Algorithmen in ausgewählten informatischen Kontexten • Syntax und Semantik einer Programmiersprache
Unterrichtsvorhaben Q1-IV Modellierung und Implementierung von Anwendungen mit dynamischen, nichtlinearen Datenstrukturen	• Daten und ihre Strukturierung • Algorithmen • Formale Sprachen und Automaten	• Objekte und Klassen • Analyse, Entwurf und Implementierung einfacher Algorithmen • Algorithmen in ausgewählten informatischen Kontexten • Syntax und Semantik einer Programmiersprache
Unterrichtsvorhaben Q1-V Modellierung und Implementierung von Anwendungen mit dynamischen, nichtlinearen Datenstrukturen	• Informatiksysteme • Informatik, Mensch und Gesellschaft	• Einzelrechner und Rechnernetzwerke • Sicherheit • Nutzung von Informatiksystemen, Wirkungen der Automatisierung

Qualifikationsphase II

Thema	Inhaltsfelder	inhaltliche Schwerpunkte
Unterrichtsvorhaben Q2-I Modellierung und Nutzung von relationalen Datenbanken in Anwendungskontexten	• Daten und ihre Strukturierung • Algorithmen • Formale Sprachen und Automaten • Informatik, Mensch und Gesellschaft	• Datenbanken • Algorithmen in ausgewählten informatischen Kontexten • Syntax und Semantik einer Programmiersprache • Sicherheit
Unterrichtsvorhaben Q2-II Endliche Automaten und formale Sprache	• Endliche Automaten und formale Sprachen • Algorithmen	• Endliche Automaten • Grammatiken regulärer Sprachen • Möglichkeiten und Grenzen von Automaten und formalen Sprachen
Unterrichtsvorhaben Q2-III Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit	• Informatiksysteme • Informatik, Mensch und Gesellschaft	• Einzelrechner und Rechnernetzwerke • Grenzen der Automatisierung

Die Reihenfolge der Unterrichtsvorhaben in der Qualifikationsphase 2 ist freigestellt, um Rahmenbedingungen – etwa der aufgrund von Studienfahrten ggf. sehr frühen ersten Klausur – besser gerecht werden zu können.

3.1.2 Konkretisierte Unterrichtsvorhaben

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

3.1.2.1 Einführungsphase

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

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

Kommunizieren und Kooperieren (K)

Schülerinnen und Schüler

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

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

Schülerinnen und Schüler

- ermitteln bei der Analyse einfacher Problemstellung Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M),
- dokumentieren Klassen durch Beschreibung der Funktionalität der Methoden (D),
- analysieren und erläutern eine objektorientierte Modellierung (A),
- implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I),
- analysieren und erläutern einfache Algorithmen und Programme (A),
- modifizieren einfache Algorithmen und Programme (I),
- entwerfen einfache Algorithmen und stellen sie umgangssprachlich und grafisch dar (M),
- implementieren Algorithmen unter Verwendung von Variablen und Wertzuweisungen, Kontrollstrukturen sowie Methodenaufrufen (I),
- testen Programme schrittweise anhand von Beispielen (I),
- implementieren einfache Algorithmen unter Beachtung der Syntax und Semantik einer Programmiersprache (I),
- interpretieren Fehlermeldungen und korrigieren den Quellcode (I),
- nutzen das verfügbare Informatiksystem zu strukturierten Verwaltung und gemeinsamen Verwendung von Daten unter Berücksichtigung der Rechteverwaltung (K),
- nutzen das Internet zur Recherche, zum Datenaustausch und zur Kommunikation (K),
- nutzen die im Unterricht eingesetzten Informatiksysteme selbstständig, sicher, zielführend und verantwortungsbewusst (D).

Da in der Einführungsphase das Hauptaugenmerk auf die Einführung der objektorientierten Programmiersprache liegt, werden die oben angegeben Kompetenzbezüge nicht mehr explizit bei den einzelnen Themenblöcken genannt.

Unterrichtsvorhaben E-I: Grundlagen der objektorientierten Programmierung und der Algorithmik anhand von grafischen Beispielen

Unterrichtssequenzen	zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Einführung in eine Programmierumgebung und Nutzen von Klassenbibliotheken • Vorstellung einer Programmierumgebung und der Programmiersprache Java • Grundbegriffe der Objektorientierung (Klasse, Objekt, Nachricht, Methode, Auftrag, Anfrage) • Nutzung von Klassenbibliotheken und Verwendung von Dokumentationen • Erarbeitung der Begriffe „sondierende Methode", "verändernde Methode" und Konstruktor	Die Schülerinnen und Schüler • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I). • implementieren einfache Algorithmen unter Beachtung der Syntax und Semantik einer Programmiersprache (I). • interpretieren Fehlermeldungen und korrigieren den Quellcode (I).	**Projekt: Programmgesteuertes Zeichnen** • Kennenlernen von BlueJ • SuM-Klassen Stift und Bildschirm mit Dokumentationen • Nachrichten an Objekte in der Werkbank • Direkteingabe von Java-Befehlen • Erste Programme als Nachrichtenfolge
2. Modellierung von Klassen • Modellierung benötigter Klassen aufgrund einer Aufgabenbeschreibung • Erstellung der Dokumentationen der Klassen unter Berücksichtigung von Parametern und Rückgabewerten	Die Schülerinnen und Schüler • Ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften und ihre Operationen (M). • Modellieren Klassen mit ihren Attributen und ihren Methoden (M). • Ordnen Attributen, Parametern und Rückgaben von Methoden von einfachen Datentypen zu (M). • Dokumentieren Klassen durch Beschreibung der Funktionalität der Methoden (D)	**Projekt: Pfeilwurf** • Klassenwurf (Pfeil, Zielscheibe) als Auftrag an die Lehrkraft
3. Einfache Algorithmen, Verwendung von Kontrollstrukturen • Implementierung einfacher Algorithmen unter Benutzung der selbst modellierten Klassen • Verwendung und Dokumentation von Kontrollstrukturen • Verwendung weiterer Klassenbibliotheken	Die Schülerinnen und Schüler • analysieren und erläutern einfache Algorithmen und Programme (A). • entwerfen einfache Algorithmen und stellen sie umgangssprachlich und grafisch dar (M). • implementieren Algorithmen unter Verwendung von Kontrollstrukturen sowie Methodenaufrufen (I).	**Projekt: Pfeilwurf** • Verwendung der von der Lehrkraft realisierten Klassenbibliothek • Schleife, Verzweigung • Struktogramme • SuM: Tastatur

• Der Tastaturpuffer als Beispiel einer Schlange		
4. Sicherung und Anwendung von einfachen Algorithmen und Kontrollstrukturen • Eigenständige Anwendung des gelernten Wissens auf ein neues Projekt • Verwendung weiterer Klassenbibliotheken • Beobachtung einer Vererbung • UML-Diagramme der Klassenbibliothek ohne Sichtbarkeiten	Die Schülerinnen und Schüler • Stellen Klassen und Vererbungsbeziehungen in Diagrammen grafisch dar (D).	**Projekt: Freihandzeichnen** • SuM: Maus, Buntstift

Unterrichtsvorhaben E-II: Modellierung und Implementierung von Klassen. Und Objektbeziehungen anhand von grafischen Beispielen

Unterrichtssequenzen	zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Entwurf eigener Klassen • Entwicklung eines ersten UML-Diagramms zum gewählten Projekt • Einführung private und öffentlicher Methoden • Hole- und Setze-Methoden für Attribute • Assoziation von Klassen am Beispiel der Hat-Beziehung	Die Schülerinnen und Schüler • ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M). • modellieren Klassen mit ihren Attributen, ihren Methoden und Assoziationsbeziehungen (M). • ordnen Klassen, Attributen und Methoden ihren Sichtbarkeitsbereich zu (M). • stellen Klassen, Assoziations- und Vererbungsbeziehungen in Diagrammen grafisch dar (D).	**Projekt: Billiard oder Flappy Bird** • UML-Editor für BlueJ
2. Implementierung eigener Klassen • Implementation der Klassen aus der ersten Sequenz • Erweiterung um weitere Funktionalitäten • Einführung von Zustandsvariablen für Attribute • Verdeutlichung des Factory-Aspekts, indem zu einem Bauplan viele Exemplare (Objekte) erzeugt werden	Die Schülerinnen und Schüler • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I). • implementieren Algorithmen unter Verwendung von Variablen und Wertzuweisungen, Kontrollstrukturen sowie Methodenaufrufen (I). • implementieren einfache Algorithmen unter Beachtung der Syntax und Semantik einer Programmiersprache (I). • interpretieren Fehlermeldungen und korrigieren den Quellcode (I).	**Projekt: Billiard oder Flappy Bird** • Sammeln von Fehlermeldungen, Ursachen und mögliche Lösungen auf IServ
3. Erweiterung des Modells • Ergänzung um eine weitere Klasse, so dass Kennt- Beziehungen notwendig werden		**Projekt: Billiard oder Flappy Bird** • Tisch bzw. Hindernisse

4. Einführung einer speziellen Unterklasse • Einführung der Vererbung durch Spezialisierung • Entwurf und Implementation von Unterklassen	Die Schülerinnen und Schüler • stellen Klassen, Assoziations- und Vererbungsbeziehungen in Diagrammen grafisch dar (D). • modellieren Klassen unter Verwendung von Vererbung (M). • modifizieren einfache Algorithmen und Programme (I).	**Projekt: Billiard oder Flappy Bird** • verschiedene Kugeltypen bzw. Spielfiguren
5. Arbeitsteiliger Entwurf einer komplexeren Modellierung • Entwicklung einzelner Klassendokumentationen in Kleingruppen • Zusammenstellung aller entworfenen Klassen zu einem Gesamtmodell als Grundlage für die Implementation • Entwurf einer abstrakten Oberklasse für Klassen gemeinsamen Eigenschaften und Methoden (Vererbung durch Generalisierung)	Die Schülerinnen und Schüler • analysieren und erläutern eine objektorientierte Modellierung (A)	**Projekt: Archicat** • Beim Entwurf der Klassen darauf achten, dass Polymorphie und späte Bindung möglich sind
6. Implementation in arbeitsteiligen Gruppen • Entwicklung der Dokumentation der entworfenen Klassen • Implementation der Klassen • Genauere Projektplanung (Arbeitsplan) • Polymorphie bei den Unterklassen der abstrakten Oberklasse	Die Schülerinnen und Schüler • testen Programme schrittweise anhand von Beispielen (I).	
7. Späte Bindung • Erzeugen von beliebigen unterschiedlichen Objekten während der Laufzeit	Die Schülerinnen und Schüler • stellen den Zustand eines Objekts dar (D).	• Debugger

• Einsatz des Debuggers zur Veranschaulichung der Laufzeit-Situation		

Unterrichtsvorhaben E-III: Vorbereitung von dynamischen Datenstrukturen und Verwendung abstrakter Klassen

Unterrichtssequenzen	zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Modellierung der Ausgangssituation • Projekt mit Objekten vieler gemeinsamer Eigenschaften, die zu einer Modellierung einer abstrakten Oberklasse mit mehreren Unterklassen führen. Diese Objekte müssen so gewählt sein, dass eine anschauliche Verkettung möglich ist.	Die Schülerinnen und Schüler • ermitteln bei der Analyse einfacher Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M). • modellieren Klassen mit ihren Attributen, ihren Methoden und Assoziationsbeziehungen (M). • ordnen Klassen, Attributen und Methoden ihren Sichtbarkeitsbereich zu (M). • stellen Klassen, Assoziations- und Vererbungsbeziehungen in Diagrammen grafisch dar (D).	**Projekt: Zug**
2. Implementation der Klassen • Gemeinsame Erstellung der abstrakten Oberklasse • arbeitsteilige Erstellung der Unterklassen • Bereitstellung der class-Dateien und Zusammenstellung zu einem kombinierten Projekt	Die Schülerinnen und Schüler • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I). • implementieren Algorithmen unter Verwendung von Variablen und Wertzuweisungen, Kontrollstrukturen sowie Methodenaufrufen (I). • implementieren einfache Algorithmen unter Beachtung der Syntax und Semantik einer Programmiersprache (I). • interpretieren Fehlermeldungen und korrigieren den Quellcode (I).	**Projekt: Zug** • Waggon • Spezielle Waggons • Zusammenstellung der Waggons zu einem Zug
3. Verkettung von Objekten • Verkettung als Möglichkeit, beliebig viele Objekte hintereinander zu hängen • Gemeinsame Ergänzung der Oberklasse um die Verkettungseigenschaft	Die Schülerinnen und Schüler • ordnen Attributen, Parametern und Ruckgaben von Methoden einfache Datentypen, Objekttypen oder lineare Datensammlungen zu (M).	**Projekt: Zug** • statt alle Waggons einzeln zu bewegen, zieht die Lok den ersten Waggon, diese den zweiten usw.

• Zusammenstellung beliebiger Objekte unter Verwendung anonymer Objekte		

Unterrichtsvorhaben E-IV: Analyse von Such- und Sortieralgorithmen und Auswirkungen von Auswirkungen von Informatiksystemen auf die Gesellschaft

Unterrichtssequenzen	zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Speichern von Zeichen • Umwandlung von ganzen Zahlen in Binärcodes und Zeichen	Die Schülerinnen und Schüler • stellen ganze Zahlen und Zeichen in Binärcodes dar (D). • interpretieren Binärcodes als Zahlen und Zeichen (D).	• Tabelle des ASCII-Codes in Dezimal- und Hexadezimaldarstellung • Arbeitsblatt mit „Geheimcode“ in binärer Darstellung • Projekte mit Komponenten und Programmgenerator: Cäsar-Codierung Umwandlungen Datenübertragung (Parität) • Weitere Beispiele: IBAN, ISBN, Barcodes
2. Such- und Sortieralgorithmen • Analyse eines fertigen Programmtextes zu Such- und Sortieralgorithmen • Das Feld als Datenstruktur mit direktem Zugriff • Umwandlungen von Zeichen in Zahlen und umgekehrt	Die Schülerinnen und Schüler • ordnen Attributen, Parametern und Ruckgaben von Methoden einfache Datentypen, Objekttypen oder lineare Datensammlungen zu (M). • analysieren Such- und Sortieralgorithmen und wenden sie auf Beispiele an (D).	**Projekt: Dekodieren**
3. Weitere Sortieralgorithmen • Entwurf zusätzlicher Sortieralgorithmen • Zählen von Vergleichs- und Vertauschoperationen an realisierten Sortieralgorithmen	Die Schülerinnen und Schüler • entwerfen einen weiteren Algorithmus zum Sortieren (M). • beurteilen die Effizienz von Algorithmen am Beispiel von Sortierverfahren hinsichtlich Zeitaufwandes und Speicherplatzbedarf (A).	• Verdecktes Sortieren von Karte • Ein bestimmte Kartenfolge wird nach den Verfahren sortiert und die Operationen werden ausgezählt • Ein fertiges Programm zählt für größere Datenmengen.
4. Informatiksysteme und ihre Auswirkungen auf die Gesellschaft • Auswirkungen der Automatisierung mit historischen Beispielen	Die Schülerinnen und Schüler • bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen (A). • erläutern wesentliche Grundlagen der Geschichte der digitalen Datenverarbeitung (A).	• Code knacken (Enigma, NSA) • Bibliotheksarchive • Volkszählungen • Buchausleihe

3.1.2.2 Qualifikationsphase – Grundkurs

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

Die Schülerinnen und Schüler

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

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

Schülerinnen und Schüler

- nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I),
- beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A),
- interpretieren Fehlermeldungen und korrigieren den Quellcode (I),
- wenden eine didaktisch orientierte Entwicklungsumgebung zur Demonstration, zum Entwurf, zur Implementierung und zum Test von Informatiksystemen an (I),

Unterrichtsvorhaben Q1-I: Wiederholung der objektorientierten Modellierung und Programmierung anhand einer kontextbezogenen Problemstellung

Unterrichtssequenzen	zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Entwicklung und Testen einer Fachklasse (Modell) • Erstellen eines Implementationsdiagramms für die Klasse • Schreiben einer Dokumentation der Klasse • Wiederholen der Entwicklung von Klassen sowie Get- und Set-Methoden	Die Schülerinnen und Schüler • analysieren und erläutern objektorientierte Modellierungen (A). • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A). • modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M). • ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M). • dokumentieren Klassen (D). • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I). • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I). • wenden eine didaktisch orientierte Entwicklungsumgebung zur Demonstration, zum Entwurf, zur Implementierung und zum Test von Informatiksystemen an (I). • testen Programme systematisch anhand von Beispielen (I). • interpretieren Fehlermeldungen und korrigieren den Quellcode (I). • stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D). • stellen die Kommunikation zwischen Objekten grafisch dar (D).	**Projekt: Bruchrechner** • Klasse Bruch • Testen der Klasse in der Werkbank
2. Entwicklung einer Rechnerklasse • Kommunikation zwischen den Klassen • Entwicklung von Algorithmen		**Projekt: Bruchrechner** • Klasse Bruchrechner
3. Entwicklung einer Oberflächenklasse (Control, View) • Kommunikation zwischen den Klassen		

	• modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M).	
4. MVC-Konzept	• Analysieren und erläutern objektorientierte Modellierungen (A)	• Erläutern das Konzept anhand des Beispiels

Unterrichtsvorhaben Q1-II: Modellierung und Implementierung von Anwendungen mit dynamischen, linearen Datenstrukturen

Unterrichtssequenzen	zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Die Datenstruktur Schlange im Anwendungskontext unter Nutzung der Klasse Queue • Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen • Erarbeitung der Funktionalität der Klasse Queue • Modellierung und Implementierung der Anwendung unter Verwendung eines oder mehrerer Objekte der Klasse Queue	Die Schülerinnen und Schüler • erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (A). • analysieren und erläutern Algorithmen und Programme (A). • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A). • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M). • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M). • modifizieren Algorithmen und Programme (I). • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I). • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I).	**Projekt: Wartezimmer** Patientenwarteschlange (jeder kennt seinen Nachfolger bzw. alternativ: seinen Vorgänger) Sobald ein Patient in einer Arztpraxis eintrifft, werden sein Name und seine Krankenkasse erfasst. Die Verwaltung der Patientenwarteschlange geschieht über eine Klasse, die hier als Wartezimmer bezeichnet wird. Wesentliche Operationen sind das „Hinzufügen“ eines Patienten und das „Entfernen“ eines Patienten, wenn er zur Behandlung gerufen wird. Die Simulationsanwendung stellt eine GUI zur Verfügung, legt ein Wartezimmer an und steuert die Abläufe. Wesentlicher Aspekt des Projektes ist die Modellierung des Wartezimmers mit Hilfe der Klasse Queue. Anschließend wird der Funktionsumfang der Anwendung erweitert: Patienten können sich zusätzlich in die Warteschlange zum Blutdruckmessen einreihen. Objekte werden von zwei Schlangen verwaltet.

2. Die Datenstruktur Stapel im Anwendungskontext unter Nutzung der Klasse Stack • Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen • Erarbeitung der Funktionalität der Klasse Stack • Modellierung und Implementierung der Anwendung unter Verwendung eines oder mehrerer Objekte der Klasse Stack		
3. Die Datenstruktur lineare Liste im Anwendungskontext unter Nutzung der Klasse List • Erarbeitung der Vorteile der Klasse List im Gegensatz zu den bereits bekannten linearen Strukturen • Modellierung und Implementierung einer kontextbezogenen Anwendung unter Verwendung der Klasse List.		**Projekt: Abfahrtslauf** Bei einem Abfahrtslauf kommen die Skifahrer nacheinander an und werden nach ihrer Zeit in eine Rangliste eingeordnet. Diese Rangliste wird in einer Anzeige ausgegeben. Ankommende Abfahrer müssen an jeder Stelle der Struktur, nicht nur am Ende oder Anfang eingefügt werden können.
4. Vertiefung – Anwendung von Listen, Stapeln oder Schlangen in mindestens einem weiteren Kontext		**Projekt: Skispringen** Ein Skispringen hat folgenden Ablauf: Nach dem Sprung erhält der Springer eine Punktzahl und wird nach dieser Punktzahl in eine Rangliste eingeordnet. Die besten 30 Springer qualifizieren sich für den zweiten Durchgang. Sie starten in umgekehrter Reihenfolge gegenüber der Platzierung auf der Rangliste. Nach dem Sprung erhält der Springer wiederum eine Punktzahl und wird nach der Gesamtpunktzahl aus beiden Durchgängen in die endgültige Rangliste eingeordnet.

Unterrichtsvorhaben Q1-III: Suchen und Sortieren auf linearen Datenstrukturen

Unterrichtssequenzen	zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Suchen von Daten in Listen und Arrays • Lineare Suche in Listen und in Arrays (auch zweidimensional) • Binäre Suche in Arrays als Beispiel für rekursives Problemlösen (ggf. aber auch iterativ) • Untersuchung der beiden Suchverfahren hinsichtlich ihrer Effizienz (Laufzeitverhalten, Speicherbedarf) **2. Sortieren in Listen und Arrays – Entwicklung und Implementierung von iterativen und rekursiven Sortierverfahren** • Entwicklung und Implementierung eines einfachen Sortierverfahrens für eine Liste • Entwicklung eines rekursiven Sortierverfahren für eine Liste (Quicksort) **3. Untersuchung der Effizienz der Sortierverfahren „Sortieren durch direktes Einfügen“ und „Quicksort“ auf lineare Listen** • Grafische Veranschaulichung der Sortierverfahren • Untersuchung der Anzahl der Vergleichsoperationen und des Speicherbedarfs bei beiden Sortierverfahren • Beurteilung der Effizienz der beiden Sortierverfahren	Die Schülerinnen und Schüler • analysieren und erläutern Algorithmen und Programme (A). • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A). • beurteilen die Effizienz von Algorithmen unter Berücksichtigung des Speicherbedarfs und der Zahl der Operationen (A). • entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“ und „Teilen und Herrschen“ (M). • modifizieren Algorithmen und Programme (I). • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I). • implementieren und erläutern iterative und rekursive Such- und Sortierverfahren (I). • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I). • interpretieren Fehlermeldungen und korrigieren den Quellcode (I). • testen Programme systematisch anhand von Beispielen (I). • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D).	**Projekt: Bundesjugendspiele** Die Teilnehmer an Bundesjugendspielen nehmen an drei Disziplinen teil und erreichen dort Punktzahlen. Diese werden in einer Wettkampfkarte eingetragen und an das Wettkampfbüro gegeben. Zur Vereinfachung sollte sich das Modell auf die drei Disziplinen „Lauf”, „Sprung“ und „Wurf“ beschränken. Im Wettkampfbüro wird das Ergebnis erstellt. Das Programm soll dafür zunächst den Besten einer Disziplin heraussuchen können und später das gesamte Ergebnis nach gewissen Kriterien sortieren können.

	• implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I).	

Unterrichtsvorhaben Q1-IV: Modellierung und Implementierung von Anwendungen mit dynamischen, nichtlinearen Datenstrukturen

Unterrichtssequenzen	zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Analyse von Baumstrukturen in Verschiedenen Kontexten • Grundlegende Begriffe (Grad, Tiefe, Höhe, Blatt, Inhalt, Teilbaum, Ebene, Vollständigkeit) • Aufbau und Darstellung von binären Bäumen anhand von Baumstrukturen in verschiedenen Kontexten	Die Schülerinnen und Schüler • erläutern Operationen dynamischer (linearer oder nicht-linearer) Datenstrukturen (A). • analysieren und erläutern Algorithmen und Programme (A). • beurteilen die syntaktische Korrektheit und die Funktionalität von Programmen (A). • ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M). • ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M). • modellieren abstrakte und nicht ab- strakte Klassen unter Verwendung von Vererbung durch Spezialisieren und Generalisieren (M). • verwenden bei der Modellierung geeigneter Problemstellungen die Möglichkeiten der Polymorphie (M). • entwickeln iterative und rekursive Algorithmen unter Nutzung der Konstruktionsstrategien „Modularisierung“ und „Teilen und Herrschen“ (M). • implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I). • modifizieren Algorithmen und Programme (I). • nutzen die Syntax und Semantik einer Programmiersprache bei der Implementierung und zur Analyse von Programmen (I).	**Projekt: Suchbäume** Alle Inhalte, die nach einer Ordnung vor dem Inhalt im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Inhalt im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. (Dies gilt für alle Teilbäume.) **Projekt: Codierungsbäume für Codierungen, deren Alphabet aus genau zwei Zeichen besteht** Morse hat Buchstaben als Folge von Punkten und Strichen codiert. Diese Codierungen können in einem Binärbaum dargestellt werden, so dass ein Übergang zum linken Teilbaum einem Punkt und ein Übergang zum rechten Teilbaum einem Strich entspricht. Wenn man im Gesamtbaum startet und durch Übergänge zu linken oder rechten Teilbäumen einen Pfad zum gewünschten Buchstaben sucht, erhält man die Morsecodierung des Buchstabens. **Projekt: Termbaum** Der Aufbau von Termen wird mit Hilfe von binären Baumstrukturen verdeutlicht. **Projekt: Entscheidungsbäume** Um eine Entscheidung zu treffen, werden mehrere Fragen mit ja oder nein beantwortet. Die Fragen, die möglich sind, wenn die Antwort auf eine Frage mit „ja“ beantwortet wird, befinden

	• interpretieren Fehlermeldungen und korrigieren den Quellcode (I). • testen Programme systematisch anhand von Beispielen (I). • stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D). • stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D). • implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I).	sich im linken Teilbaum, die Fragen, die möglich sind, wenn die Antwort „nein“ lautet, stehen im rechten Teilbaum.
2. Die Datenstruktur Binärbaum im Anwendungskontext unter Nutzung der Klasse BinaryTree • Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen im Anwendungskontext • Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramms • Erarbeitung der Klasse BinaryTree und beispielhafte Anwendung der Operationen • Implementierung der Anwendung oder von Teilen der Anwendung • Traversierung eines Binärbaums im Pre-, In- und Postorderdurchlauf		**Durchlaufarten**: • Suchbaum • Termbaum: In-Order • Termbaum: UPN • Termbaum: Post-Ordner
3. Die Datenstruktur binärer Suchbaum im Anwendungskontext unter Verwendung der Klasse BinarySearchTree • Analyse der Problemstellung, Ermittlung von Objekten, ihren Eigenschaften und Operationen • Modellierung eines Entwurfsdiagramms und Entwicklung eines Implementationsdiagramm, grafische Darstellung eines binären Suchbaums und Erarbeitung der Struktureigenschaften • Erarbeitung der Klasse BinarySearchTree und Einführung des Interface ComparableContent • zur Realisierung einer geeigneten Ordnungsrelation • Implementierung der Anwendung oder von Teilen der Anwendung inklusive einer sortierten Ausgabe des Baums		**Projekt: Informatikerbaum als Suchbaum** In einem binären *Suchbaum* werden die Namen und die Geburtsdaten von Informatikern lexikographisch geordnet abgespeichert. Alle Namen, die nach dieser Ordnung vor dem Namen im aktuellen Teilbaum stehen, sind in dessen linkem Teilbaum, alle die nach dem Namen im aktuellen Teilbaum stehen, sind in dessen rechtem Teilbaum. Folgende Funktionalitäten werden benötigt: • Einfügen der Informatiker-Daten in den Baum

		• Suchen nach einem Informatiker über den Schlüssel Name • Ausgabe des kompletten Datenbestands in nach Namen sortierter Reihenfolge
4. Übung und Vertiefung der Verwendung von Binärbäumen oder binären Suchbäumen anhand weiterer Problemstellungen		**Projekt: Huffman-Codierung** Aufgeteilt in mehrere Teilprojekte mit BinaryTree und BinarySearchTree

Unterrichtsvorhaben Q1-V: Sicherheit und Datenschutz in Netzstrukturen

Unterrichtssequenzen	zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Daten in Netzwerken und Sicherheitsaspekte in Netzen sowie beim Zugriff auf Datenbanken • Beschreibung eines Webserverzugriffs im Netz anhand eines Anwendungskontextes und einer Client-Server-Struktur zur Klärung der Funktionsweise einer Website-Abfrage • Netztopologien als Grundlage von Client-Server-Strukturen und TCP/IP-Schichtenmodell als Beispiel für eine Paketübermittlung in einem Netz • Vertraulichkeit, Integrität, Authentizität in Netzwerken sowie symmetrische und asymmetrische kryptografische Verfahren (Cäsar-, Vigenère-, RSA-Verfahren) als Methoden Daten im Netz verschlüsselt zu übertragen	Die Schülerinnen und Schüler • beschreiben und erläutern Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken (A). • analysieren und erläutern Eigenschaften und Einsatzbereiche symmetrischer und asymmetrischer Verschlüsselungsverfahren (A). • untersuchen und bewerten anhand von Fallbeispielen die Auswirkungen des Einsatzes von Informatiksystemen, die Sicherheit von Informatiksystemen sowie die Einhaltung der Datenschutzbestimmungen und des Urheberrechts (A). • untersuchen und bewerten Problemlagen, die sich aus dem Einsatz von Informatiksystemen ergeben, hinsichtlich rechtlicher Vorgaben, ethischer Aspekte und gesellschaftlicher Werte unter Berücksichtigung unterschiedlicher Interessenlagen (A). • nutzen bereitgestellte Informatiksysteme und das Internet reflektiert zum Erschließen, zur Aufbereitung und Präsentation fachlicher Inhalte (D).	Materialien: Protokolle: HTTP, HTTPS, POP3 Dabei Analyse eines fertigen POP3-Clients
2. Fallbeispiele zur Datenschutzproblematik und zum Urheberrecht		Materialien: Materialblatt Datenschutzgesetz

Unterrichtsvorhaben Q2-I: Modellierung und Nutzung von relationalen Datenbanken in Anwendungskontexten

Unterrichtssequenzen	zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Nutzung von relationalen Datenbanken (a) Aufbau von Datenbanken und Grundbegriffe • Entwicklung von Fragestellungen zur vorhandenen Datenbank • Analyse der Struktur der vorgegebenen Datenbank und Erarbeitung der Begriffe Tabelle, Attribut, Datensatz, Datentyp, Primärschlüssel, Fremdschlüssel, Datenbankschema (b) SQL-Abfragen • Analyse vorgegebener SQL-Abfragen und Erarbeitung der Sprachelemente von SQL (SELECT (DISTINCT) ... FROM, WHERE, AND, OR, NOT) auf einer Tabelle • Analyse und Erarbeitung von SQL- Abfragen auf einer und mehrerer Tabelle zur Beantwortung der Fragestellungen (JOIN, UNION, AS, GROUP BY,ORDER BY, ASC, DESC, COUNT, MAX, MIN, SUM, Arithmetische Operatoren: +, -, *, /, (...), Vergleichsoperatoren: =, <>, >, <, >=, <=, LIKE, BETWEEN, IN, IS NULL)	Die Schülerinnen und Schüler • erläutern die Eigenschaften und den Aufbau von Datenbanksystemen unter dem Aspekt der sicheren Nutzung (A). • analysieren und erläutern die Syntax und Semantik einer Datenbankabfrage (A). • analysieren und erläutern eine Datenbankmodellierung (A). • erläutern die Eigenschaften normalisierter Datenbankschemata (A). • bestimmen Primär- und Sekundärschlüssel (M). • ermitteln für anwendungsbezogene Problemstellungen Entitäten, zugehörige Attribute, Relationen und Kardinalitäten (M). • modifizieren eine Datenbankmodellierung (M). • modellieren zu einem Entity-Relation- ship-Diagramm ein relationales Datenbankschema (M). • überfuhren Datenbankschemata in die 1. bis 3. Normalform (M). • verwenden die Syntax und Semantik einer Datenbankabfragesprache, um Informationen aus einen Datenbanksystem zu extrahieren (I). • ermitteln Ergebnisse von Datenbankabfragen über mehrere verknüpfte Tabellen (D). • stellen Entitäten mit ihren Attributen und die Beziehungen zwischen Entitäten in einem Entity-Relationship-Diagramm grafisch dar (D). • überprüfen Datenbankschemata auf vorgegebene Normalisierungseigenschaften (D).	*Beispiel:* VideoCenter VideoCenter ist die Simulation einer Online-Videothek für den Informatik- Unterricht mit Webfrontends zur Verwaltung der Kunden, der Videos und der Ausleihe. Außerdem ist es möglich direkt SQL-Abfragen einzugeben.

2. Modellierung von relationalen Datenbanken (a) Entity-Relationship-Diagramm • Ermittlung von Entitäten, zugehörigen Attributen, Relationen und Kardinalitäten in Anwendungssituationen und Modellierung eines Datenbankentwurfs in Form eines Entity-Relationship-Diagramms • Erläuterung und Modifizierung einer Datenbankmodellierung (b) Entwicklung einer Datenbank aus einem Datenbankentwurf • Modellierung eines relationalen Datenbankschemas zu einem Entity-Relationship-Diagramm inklusive der Bestimmung von Primär- und Sekundärschlüsseln (c) Redundanz, Konsistenz und Normalformen • Untersuchung einer Datenbank hinsichtlich Konsistenz und Redundanz in einer Anwendungssituation • Überprüfung von Datenbankschemata hinsichtlich der 1. bis 3. Normalform und Normalisierung (um Redundanzen zu vermeiden und Konsistenz zu gewährleisten)		

Unterrichtsvorhaben Q2-II: Endliche Automaten und formale Sprachen

Unterrichtssequenzen	zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Endliche Automaten • Vom Automaten in den Schülerinnen und Schülern bekannten Kontexten zur formalen Beschreibung eines endlichen Automaten • Untersuchung, Darstellung und Entwicklung endlicher Automaten • Umwandlung von nichtdeterministischen Akzeptoren in deterministische Akzeptoren • Simulation eines Akzeptors in der Programmiersprache	Die Schülerinnen und Schüler • analysieren und erläutern die Eigenschaften endlicher Automaten einschließlich ihres Verhaltens auf bestimmte Eingaben (A). • analysieren und erläutern Grammatiken regulärer Sprachen (A). • zeigen die Grenzen endlicher Automaten und regulärer Grammatiken im Anwendungszusammenhang auf (A). • ermitteln die formale Sprache, die durch eine Grammatik erzeugt wird (A). • entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M). • entwickeln zur akzeptierten Sprache eines Automaten die zugehörige Grammatik (M). • entwickeln zur Grammatik einer regulären Sprache einen zugehörigen endlichen Automaten (M). • modifizieren Grammatiken regulärer Sprachen (M). • entwickeln zu einer regulären Sprache eine Grammatik, die die Sprache erzeugt (M). • stellen endliche Automaten in Tabellen oder Graphen dar und überführen sie in die jeweils andere Darstellungsform (D), • ermitteln die Sprache, die ein endlicher Automat akzeptiert (D). • beschreiben an Beispielen den Zusammenhang zwischen Automaten und Grammatiken (D).	Beispiel: Entwicklung eines Taschenrechners für Terme
2. Untersuchung und Entwicklung von Grammatiken regulärer Sprachen • Erarbeitung der formalen Darstellung regulärer Grammatiken • Untersuchung, Modifikation und Entwicklung von Grammatiken • Entwicklung von endlichen Automaten zum Erkennen regulärer Sprachen die durch Grammatiken gegeben werden • Entwicklung regulärer Grammatiken zu endlichen Automaten		Beispiel: reguläre Grammatik für Wörter mit ungerader Parität, Grammatik für Wörter, die bestimmte Zahlen repräsentieren, Satzgliederungsgrammatik
3. Grenzen endlicher Automaten		Beispiel: Klammerausdrücke $a^n\ b^n$ im Vergleich zu $(ab)^n$

4. Kontextfreie und kontextsensitive Grammatiken, Chomsky-Hierarchie • Syntaxbäume • Eindeutigkeit • Grenzen kontextfreier Grammatiken		Grammatiken für Klammerterme, Satzgliederungsgrammatiken, Grammatik für if-Abfragen
5. Compilerbau • Stufen eines Compilers • Realisierung des Parsers und des Codierers in der Programmiersprache		Beispiel: Termcompiler

Unterrichtsvorhaben Q2-III: Prinzipielle Arbeitsweise eines Computers und Grenzen der Automatisierbarkeit

Unterrichtssequenzen	zu entwickelnde Kompetenzen	Beispiele, Medien, Materialien
1. Von-Neumann-Architektur und die Ausführung maschinennaher Programme • prinzipieller Aufbau einer von Neumann-Architektur mit CPU, Rechenwerk, Steuerwerk, Register und Hauptspeicher • maschinennahe Befehle und ihre Repräsentation in einem Binär- Code, der in einem Register gespeichert werden kann • Analyse und Erläuterung der Funktionsweise eines einfachen maschinennahen Programms	Die Schülerinnen und Schüler • erläutern die Ausführung eines einfachen maschinennahen Programms sowie die Datenspeicherung auf einer „Von- Neumann-Architektur“ (A). • untersuchen und beurteilen Grenzen des Problemlösens mit Informatiksystemen (A).	Beispiel: Addition von 4 zu einer eingegebenen Zahl mit einem Rechnermodell Verwenden eines Von-Neumann-Simulators
2. Grenzen der Automatisierbarkeit • Vorstellung des Halteproblems • Unlösbarkeit des Halteproblems • Beurteilung des Einsatzes von Informatiksystemen hinsichtlich prinzipieller Möglichkeiten und prinzipieller Grenzen		Beispiel: Halteproblem

3.2 Grundsätze der fachmethodischen und fachdidaktischen Arbeit

In Absprache mit der Lehrerkonferenz sowie unter Berücksichtigung des Schulprogramms hat die Fachkonferenz Informatik 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 Schülerinnen und Schüler
3. Die Unterrichtsgestaltung ist auf die Ziele und Inhalte abgestimmt.
4. Medien und Arbeitsmittel sind schülernah gewählt.
5. Die Schülerinnen und Schüler erreichen einen Lernzuwachs.
6. Der Unterricht fördert eine aktive Teilnahme der Schülerinnen und Schüler
7. Der Unterricht fördert die Zusammenarbeit zwischen den Schülerinnen und Schülern und bietet ihnen Möglichkeiten zu eigenen Lösungen.
8. Der Unterricht berücksichtigt die individuellen Lernwege der einzelnen Schülerinnen und Schüler
9. Die Schülerinnen und Schüler 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 in der Regel 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 gegenwarts- und zukunftsorientiert und gewinnt dadurch für die Schülerinnen und Schüler an Bedeutsamkeit.

19. Der Unterricht ist meist 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.

3.3 Grundsätze der Leistungsbewertung und Leistungsrückmeldung

3.3.1 Transparenz der Leistungsbeurteilung

Schulische Leistungsbewertung steht im Spannungsfeld pädagogischer und gesellschaftlicher Zielsetzung.

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

Für die Erziehungsberechtigten sind Noten eine einfache und zentrale Information zum Leistungstand ihre Kinder. Sie bieten den Anlass, über die Ursache von Defiziten und über die Beseitigung von Lernschwierigkeiten verschiedenster Art Rücksprache zu halten. Noten sind zudem Grundlage und Anlass, in den halbjährlich stattfindenden pädagogischen Konferenzen über die Schwierigkeiten und besonderen Probleme einzelner Schüler wie auch Klassen zu beraten und Maßnahmen zur Verbesserung zu beschließen.

Schulische Leistungsbewertung ist eingebettet in die durch das Schulgesetz § 48 (Grundsätze der Leistungsbewertung), APO - GOSt §13 bis §17 sowie Kapitel 3 des Kernlehrplans Informatik für die gymnasiale Oberstufe vorgegebene Grundsätze und Verfahren. Daraus erwächst für die Schulen konkret die Aufgabe, sowohl die individuellen Schwächen und Stärken der Schüler zu diagnostizieren und gegebenenfalls die Defizite durch gezielte Maßnahmen zu beseitigen sowie besondere Begabungen zu fördern.

Die gesellschaftliche Funktion von Noten zu erfüllen ist der Schule aufgegeben. Noten entscheiden mit über Schullaufbahnen, Versetzungen und Abschlüsse. Zeugnisse sind mit entscheidender Parameter bei der Zuteilung von Berufs- und Lebenschancen. Daraus erwachsen für die Beurteilenden eine besondere Verantwortung und die Pflicht einer größtmöglichen Objektivität bei der Notenfindung.

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

3.3.2 Grundsätze der Leistungsbeurteilung

Es gelten folgende Grundsätze der Leistungsbewertung:

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

3.3.3 Formen der Leistungsüberprüfung

3.3.3.1 Beurteilungsbereich Kursarbeiten bzw. Klausuren

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

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

Am Städt. Gymnasium Wermelskirchen werden die Kursarbeiten bzw. Klausuren in der Regel nach einem vorab festgelegten Punkteschema bewertet.

In der Qualifikationsphase wird dabei in der Regel folgendes Schema angewendet. Dabei ist eine glatt ausreichende Leistung bei 45% der Punktzahl erreicht worden. Die übrigen Notenstufen ergeben sich dann dadurch, dass für jede Notenstufe Intervalle der erreichten Punkte gebildet werden, die in der Regel gleich groß sind:

ab %	Punkte	Note
0,00%	0	6
20,00%	1	5-
26,67%	2	5
33,33%	3	5+
40,00%	4	4-
45,00%	5	4
50,00%	6	4+
55,00%	7	3-

ab %	Punkte	Note
60,00%	8	3
65,00%	9	3+
70,00%	10	2-
75,00%	11	2
80,00%	12	2+
85,00%	13	1-
90,00%	14	1
95,00%	15	1+

Von dem Zuordnungsschema kann abgewichen werden, wenn sich z.B. besonders originelle Teillösungen nicht durch Hilfspunkte gemäß den Kriterien des Erwartungshorizonts abbilden lassen oder eine Abwertung wegen besonders schwacher Darstellung angemessen erscheint.

Die Fachkonferenz legt die Dauer der Kursarbeiten und Klausuren fest. Am Gymnasium Wermelskirchen gelten für die Sekundarstufe folgende Regelungen:

Klasse	1. Klausur, 1. HJ	2. Klausur 1. HJ	1. Klausur 2. HJ	2. Klausur, 2. HJ
Eph	90 min		90 min	90 min
Q1 GK	95 min	95 min	135 min	135 min
Q2 GK	160 min	160 min	225 min	225 min

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

Die Abiturvorklausur wird – was den formalen Rahmen betrifft – unter Abiturbedingungen geschrieben. Insbesondere wird spätestens in der Abiturvorklausur die im Zentralabitur gemäß unten aufgeführter Tabelle vorgegebene Zuordnung der erreichten Punkte (maximale Punktzahl: 100 im GK, 150 im LK) zur Note als Grundlage der Notenfindung genutzt.

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

3.3.3.2 Beurteilungsbereich Sonstige Mitarbeit

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

Bei den mündlichen Leistungen im Unterricht sind zu bewerten:

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

Neben der Richtigkeit, Vollständigkeit und Komplexität der Gedankengänge sind die der Altersstufe angemessene sprachliche Darstellung und die Verwendung der Fachsprache von Bedeutung.

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

- praktische Leistungen am Computer als Werkzeug im Unterricht,
- Protokolle und Referate,
- Projektarbeit (oft in Form von Gruppenarbeit),
- Lernerfolgsüberprüfungen und schriftliche Übungen.

3.3.3.3 Individuelle Förderung

Die Lehrerinnen und Lehrer beobachten die individuellen Leistungen in allen Bereichen der Informatik über einen längeren Zeitraum, um auf dieser Grundlage ein Leistungsbild zu erhalten. Neben der Orientierung an den Kompetenzstandards der jeweiligen Jahrgangsstufe kann bei der Leistungsbewertung auch die jeweilige Entwicklung des Schülers bzw. der Schülerin, gemäß der zu beobachtenden Lern- und Denkfortschritte, berücksichtigt werden.

Der Informatikunterricht lebt von der verantwortungsvollen und selbständigen Arbeit der Schülerinnen und Schüler, so dass die Lehrperson die nötige Zeit hat, bei Bedarf gezielt und individuell zu fördern.

Leistungsstärkere Schülerinnen und Schüler können ihr Wissen anhand von vertiefenden Problemstellungen erweitern.

3.3.3.4 Bildung der Zeugnisnote

In die Note gehen alle im Unterricht erbrachten Leistungen ein. Dabei nehmen die Beurteilung der Kursarbeiten bzw. Klausuren den gleichen Stellenwert wie die Leistungen im Bereich der Mitarbeit im Unterricht ein. Zudem ist bei der Notenfindung die individuelle Lernentwicklung der Schülerinnen und Schüler angemessen zu berücksichtigen.

3.3.3.5 Mündliche Abiturprüfungen

Auch für das mündliche Abitur (im 4. Fach oder bei Abweichungs- bzw. Bestehensprüfungen im 1. bis 3. Fach) wird ein Kriterienraster für den ersten und zweiten Prüfungsteil vorgelegt, aus dem auch deutlich wird, wann eine gute oder ausreichende Leistung erreicht wird.

3.3.4 Grundsätze der Leistungsrückmeldung

Den Schülerinnen und Schülern werden die verbindlichen Vereinbarungen zur Leistungsbewertung im Informatikunterricht am Städt. Gymnasium Wermelskirchen zugänglich gemacht. Zu Beginn eines Schul- bzw. Halbjahres wird durch die Lehrkraft auf die einheitlichen Grundsätze zur Leistungsbewertung hingewiesen.

Jede Lehrerin / jeder Lehrer dokumentiert die Leistungen der Schülerinnen und Schüler regelmäßig. Für Präsentationen, Arbeitsprotokolle, Dokumentationen und andere Lernprodukte der sonstigen Mitarbeit erfolgt eine Leistungsrückmeldung, bei der inhalts- und darstellungsbezogene Kriterien angesprochen werden. Hier werden zentrale Stärken als auch Optimierungsperspektiven für jede Schülerin bzw. jeden Schüler hervorgehoben.

Leistungsrückmeldungen bezogen auf die mündliche Mitarbeit erfolgen mündlich in regelmäßigen Abständen, zumindest am Ende eines jeden Quartals oder auf Nachfrage der Schülerinnen und Schüler außerhalb der Unterrichtszeit.

Eltern können sich im Rahmen von Elternsprechtagen sowie der Sprechstunden über den Leistungsstand ihrer Kinder informieren. Dabei wird auch eine individuelle Beratung im Hinblick auf Stärken und Verbesserungsperspektiven angeboten.

3.4 Lehr- und Lernmittel

Eingesetzte Software

- Java SDK
- BlueJ
- Java-Editor
- Cryptool

4 Qualitätssicherung und Evaluation

Das schulinterne Curriculum stellt keine starre Größe dar, sondern ist als „lebendes Dokument“ zu betrachten. Dementsprechend sind die Inhalte stetig zu überprüfen, um ggf. Modifikationen vornehmen zu können. Die Fachkonferenz (als professionelle Lerngemeinschaft) trägt durch diesen Prozess zur Qualitätsentwicklung und damit zur Qualitätssicherung des Faches bei.

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 2023 für den ersten Durchgang durch die Einführungsphase der gymnasialen Oberstufe nach Erlass des Kernlehrplanes verbindlich. Jeweils vor Beginn eines neuen Schuljahres, d.h. erstmalig nach Ende der Einführungsphase im Sommer 2023 werden in einer Sitzung der Fachkonferenz für die nachfolgenden Jahrgänge zwingend erforderlich erscheinende Veränderungen diskutiert und ggf. beschlossen, um erkannten ungünstigen Entscheidungen schnellstmöglich entgegenwirken zu können.

Nach Abschluss des Abiturs 2025 wird eine Arbeitsgruppe aus den zu diesem Zeitpunkt in der gymnasialen Oberstufe unterrichtenden Lehrkräften auf der Grundlage ihrer Unterrichtserfahrungen eine Gesamtsicht des schulinternen Curriculums vornehmen und eine Beschlussvorlage für die erste Fachkonferenz des folgenden Schuljahres erstellen.