Heinrich-Heine-Gymnasium Köln
Herausforderungen annehmen
Haltungen entwickeln
Gemeinschaft stärken

Schulinterner Lehrplan Informatik

Fachschaft Informatik
Stand: November 2015
Email: 184858@schule.nrw.de
Verfasst von Christian Pothmann

Inhalt

1 Die Fachgruppe Informatik des HHG

Das Heinrich-Heine-Gymnasium ist ein vierzügiges Gymnasium im Kölner Stadtteil Ostheim und bildet zusammen mit der Albert-Schweitzer-Realschule das Schulzentrum Ostheim. Das HHG hat derzeit ca. 800 Schülerinnen und Schüler und ca. 60 Lehrkräfte. Im Zusammenhang mit dem Fach Informatik ist insbesondere zu erwähnen, dass das HHG 2013 das Berufswahlsiegel erhalten hat und damit eine vorbildliche Vorbereitung der Schüler auf den Einstieg ins Berufsleben leistet, sowie die aktuelle Bemühung um das Siegel „MINT-freundliche Schule“. Seit einem Jahr wird in der jetzigen Jahrgangstufe 8 im Rahmen eines dreijährigen Pilotprojekts mit iPads unterrichtet, die von den Eltern der Jahrgangstufe finanziert wurden. Seit 2013 ist das HHG zudem „CAS-Schule“, d.h. die Medienausstattung wurde von der Stadt Köln grundlegend modernisiert und ausgebaut.

Das Fach Informatik wird an unserer Schule seit vielen Jahren unterrichtet und hat sich mittlerweile zu einem Aushängeschild des Heinrich-Heine-Gymnasiums entwickelt. Vor einigen Jahre wurden die Fünftklässler eine Stunde pro Woche von einer Schülermutter in die Benutzung von Computern und verschiedenen nützlichen Programmen eingeführt. Mittlerweile hat die Fachschaft Informatik drei Kollegen, zwei davon haben die Sek-II-Lehrbefähigung. In der Mittelstufe gibt es je eine AG in der 5. und 7. Jahrgangstufe, sowie in jedem Jahr zwei Differenzierungskurse in der 8, die dann in der 9 fortgesetzt werden. Dass unsere Schüler Interesse an Neuen Medien und Informatik haben zeigt sich bei der Wahl der Differenzierungskurse sowie der Oberstufenkurse in Informatik, bei der regelmäßig etwa die Hälfte unserer Schüler dieses Fach wählen, so dass in der Einführungsphase zwei bis drei Grundkurse zustande kommen. Ein Leistungskurs ist derzeit jedoch nicht in Planung.

In der Mittelstufe steht im Fach Informatik das Thema „Medienerziehung“ im Vordergrund. In den AGs in der 5 und 7 werden die Schüler an die Anwendung von Computern herangeführt, sie lernen hier z.B. Text- und Bildbearbeitung, die Erstellung von Präsentationen und das Zehnfingersystem, machen aber auch schon erste Schritte in der Programmierung mit dem System Scratch. In den Differenzierungskursen lernen die Schüler im Rahmen der Reihe „Rechnerstrukturen“ schon etwas über den grundlegenden Aufbau von Computern und bauen ihre Programmierkenntnisse z. B. mit dem „Java-

Hamster“ weiter aus. Im Rahmen der Erstellung einer Webseite besteht hier auch die Möglichkeit, an einem größeren Projekt zu arbeiten.

In der Oberstufe werden seit dem Schuljahr 2012/13 Informatikkurse angeboten, was vorher aufgrund zu weniger Kollegen nicht möglich war. In diesem Schuljahr 2014/15 nehmen die ersten Schüler der Q2 an der Abiturprüfung im Fach Informatik teil. Zur Zeit gibt es in allen drei Jahrgangstufen jeweils zwei Grundkurse, mit insgesamt zwischen 30 und 55 Schülern pro Jahrgangstufe.

Als Computerausstattung stehen am HHG zwei Computerräume mit jeweils 25 Rechnern, ein Laptopwagen mit 16 Laptops und 30 iPads zur Verfügung. In allen Klassenräumen steht ein Videoprojektor zur Verfügung. Das pädagogische Netzwerk wird von den Kollegen der Fachschaft Informatik in Kooperation mit der Firma NetCologne administriert. Die Schüler haben einen persönlichen Account mit 50 MB Speicherplatz auf dem Schulserver. Weiterhin steht die Lernplattform Moodle zur Verfügung.

Die Kollegen der Fachschaft sind namentlich Herr Pothmann (Fachvorsitz), Herr Kowalski und Herr Eberhardt.

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.

Im „Übersichtsraster Unterrichtsvorhaben“ (Kapitel 2.1.1) wird die für alle Lehrer gemäß Fachkonferenzbeschluss verbindliche Verteilung der Unterrichtsvorhaben dargestellt. Das Übersichtsraster dient dazu, den 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. Neuen 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,

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

2.1.1 Übersichtsraster Unterrichtsvorhaben

Einführungsphase – Grundkurs

Unterrichtsvorhaben E-I	
Thema	Überblick über das Fach Informatik, Grundlagen von Informatik-systemen, Geschichte der Informatik
Zentrale Kompetenzen	Argumentieren / Darstellen und Interpretieren Kommunizieren und Kooperieren
Inhaltsfelder	Informatiksysteme / Informatik, Mensch und Gesellschaft
Inhaltliche Schwerpunkte	Digitalisierung, Einzelrechner, Dateisysteme, Internet Einsatz von Informatiksystemen / Wirkungen der Automatisierung Geschichte der Datenverarbeitung
Zeitbedarf	6 Stunden

Unterrichtsvorhaben E-II	
Thema	Grundlagen der Programmierung mit Java und einfache Algorithmik
Zentrale Kompetenzen	Argumentieren / Darstellen und Interpretieren / Modellieren Implementieren / Kommunizieren und Kooperieren
Inhaltsfelder	Daten und ihre Strukturierung / Algorithmen / Formale Sprachen und Automaten / Informatiksysteme
Inhaltliche Schwerpunkte	Analyse, Entwurf und Implementierung einfacher Algorithmen Syntax und Semantik einer Programmiersprache
Zeitbedarf	30 Stunden

Unterrichtsvorhaben E-III	
Thema	Umgang mit Arrays sowie Entwurf und Analyse einfacher Sortierverfahren
Zentrale Kompetenzen	Argumentieren / Darstellen und Interpretieren / Modellieren Implementieren / Kommunizieren und Kooperieren
Inhaltsfelder	Daten und ihre Strukturierung / Algorithmen Formale Sprachen und Automaten
Inhaltliche Schwerpunkte	Analyse, Entwurf und Implementierung einfacher Algorithmen Algorithmen zum Suchen und Sortieren Syntax und Semantik einer Programmiersprache
Zeitbedarf	20 Stunden

Unterrichtsvorhaben E-IV	
Thema	Objektorientierte Modellierung und Implementierung von Klassen- und Objektbeziehungen im Anwendungskontext
Zentrale Kompetenzen	Argumentieren / Darstellen und Interpretieren / Modellieren Implementieren / Kommunizieren und Kooperieren
Inhaltsfelder	Daten und ihre Strukturierung / Formale Sprachen und Automaten
Inhaltliche Schwerpunkte	Objekte und Klassen Analyse, Entwurf und Implementierung einfacher Algorithmen Syntax und Semantik einer Programmiersprache
Zeitbedarf	20 Stunden

Zeitbedarf Einführungsphase insgesamt: 76 Stunden

Qualifikationsphase I – Grundkurs

Unterrichtsvorhaben Q1-I	
Thema	Implementierung einfacher Sortierverfahren, Laufzeitanalyse
Zentrale Kompetenzen	Argumentieren / Darstellen und Interpretieren / Modellieren Implementieren / Kommunizieren und Kooperieren
Inhaltsfelder	Daten und ihre Strukturierung / Algorithmen Formale Sprachen und Automaten
Inhaltliche Schwerpunkte	Analyse, Entwurf und Implementierung von Algorithmen Algorithmen zum Suchen und Sortieren Syntax und Semantik einer Programmiersprache
Zeitbedarf	10 Stunden

Unterrichtsvorhaben Q1-II	
Thema	Rekursion, Analyse rekursiver Sortierverfahren
Zentrale Kompetenzen	Argumentieren / Darstellen und Interpretieren / Modellieren Implementieren / Kommunizieren und Kooperieren
Inhaltsfelder	Daten und ihre Strukturierung / Algorithmen Formale Sprachen und Automaten
Inhaltliche Schwerpunkte	Analyse, Entwurf und Implementierung von Algorithmen Algorithmen zum Suchen und Sortieren Syntax und Semantik einer Programmiersprache
Zeitbedarf	10 Stunden

Unterrichtsvorhaben Q1-III	
Thema	Lineare Datenstrukturen Liste, Queue und Stack: Analyse und Entwurf Implementierung von Anwendungsprogrammen
Zentrale Kompetenzen	Argumentieren / Darstellen und Interpretieren / Modellieren Implementieren / Kommunizieren und Kooperieren
Inhaltsfelder	Daten und ihre Strukturierung / Algorithmen Formale Sprachen und Automaten
Inhaltliche Schwerpunkte	Objekte und Klassen Analyse, Entwurf und Implementierung von Algorithmen Algorithmen in ausgewählten informatischen Kontexten Syntax und Semantik einer Programmiersprache
Zeitbedarf	30 Stunden

Unterrichtsvorhaben Q1-IV	
Thema	Binärbaum und binärer Suchbaum: Analyse und Entwurf Implementierung von Anwendungsprogrammen
Zentrale Kompetenzen	Argumentieren / Darstellen und Interpretieren / Modellieren Implementieren / Kommunizieren und Kooperieren
Inhaltsfelder	Daten und ihre Strukturierung / Algorithmen Formale Sprachen und Automaten
Inhaltliche Schwerpunkte	Objekte und Klassen Analyse, Entwurf und Implementierung von Algorithmen Algorithmen in ausgewählten informatischen Kontexten Syntax und Semantik einer Programmiersprache
Zeitbedarf	20 Stunden

Unterrichtsvorhaben Q1-V	
Thema	Informatiksysteme (Von-Neumman-Rechner und Netzwerktopologien)
Zentrale Kompetenzen	Argumentieren / Darstellen und Interpretieren Kommunizieren und Kooperieren
Inhaltsfelder	Informatiksysteme / Informatik, Mensch und Gesellschaft
Inhaltliche Schwerpunkte	Einzelrechner und Rechnernetzwerke / Sicherheit Wirkungen der Automatisierung
Zeitbedarf	10 Stunden

Zeitbedarf Qualifikationsphase I insgesamt: 80 Stunden

Qualifikationsphase II – Grundkurs

Unterrichtsvorhaben Q2-I	
Thema	Kryptographie
Zentrale Kompetenzen	Argumentieren / Darstellen und Interpretieren Kommunizieren und Kooperieren
Inhaltsfelder	Einzelrechner und Rechnernetzwerke / Sicherheit Wirkungen der Automatisierung
Inhaltliche Schwerpunkte	Analyse und Entwurf einfacher Algorithmen Nutzung von Informatiksystemen
Zeitbedarf	5 Stunden

Unterrichtsvorhaben Q2-II	
Thema	Modellierung und Nutzung relationaler Datenbanken in Anwendungskontexten
Zentrale Kompetenzen	Argumentieren / Darstellen und Interpretieren / Modellieren Implementieren / Kommunizieren und Kooperieren
Inhaltsfelder	Daten und ihre Strukturierung / Formale Sprachen und Automaten Informatiksysteme / Informatik, Mensch und Gesellschaft
Inhaltliche Schwerpunkte	Datenbanken / Syntax und Semantik einer Programmiersprache Algorithmen in ausgewählten informatischen Kontexten Nutzung von Informatiksystemen / Sicherheit Wirkung der Automatisierung
Zeitbedarf	25 Stunden

Unterrichtsvorhaben Q2-III	
Thema	Endliche Automaten und reguläre Grammatiken
Zentrale Kompetenzen	Argumentieren / Darstellen und Interpretieren / Modellieren Implementieren / Kommunizieren und Kooperieren
Inhaltsfelder	Formale Sprachen und Automaten / Informatiksysteme
Inhaltliche Schwerpunkte	Endliche Automaten / Grammatiken regulärer Sprachen Möglichkeiten und Grenzen von Automaten und formalen Sprachen Nutzung von Informatiksystemen
Zeitbedarf	20 Stunden

Zeitbedarf Qualifikationsphase I insgesamt: 50 Stunden

2.1.2 Konkretisierte Unterrichtsvorhaben

Im Folgenden sollen die in Abschnitt 2.1.1 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 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 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 (K)
- nutzen das Internet zur Recherche, zum Datenaustausch und zur Kommunikation (K)

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.

Unterrichsvorhaben E-I

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

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

Zeitbedarf: 6 Stunden

Absprachen zur vorhabenbezogene Konkretisierung

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

Unterrichtssequenzen

1. Allgemeine Einführung
- Übersicht über das Fach
- Einführung in die Informatiksysteme der Schule, ggf. in die Lernplattform

2. Grundlagen von Informatiksystemen und Überblick über die Geschichte der Informatik

- Darstellung von Zahlen im Binärsystem
- Von-Neumann-Architektur
- Geschichte der Datenverarbeitung

Zu entwickelnde Kompetenzen

Die Schülerinnen und Schüler

- nutzen die im Unterricht eingesetzten Informatiksysteme selbstständig, sicher, zielführend und verantwortungsbewusst (D)
- nutzen die verfügbaren Informatiksysteme zu strukturierten Verwaltung und gemeinsamen Verwendung von Daten (K)
- stellen ganze Zahlen und Zeichen in Binärcodes dar (D)
- interpretieren Binärcodes als Zahlen und Zeichen (D)
- beschreiben und erläutern den strukturellen Aufbau und die Arbeits-weise singulärer Rechner am Beispiel der „Von-Neumann-Architektur“ (A)
- erläutern wesentliche Grundlagen der Geschichte der digitalen Daten-verarbeitung (A)

Beispiele, Materialien, Medien

Lernplattform Moodle

<u>Unterrichsvorhaben E-II</u>

Thema: Grundlagen der Programmierung mit Java und einfache Algorithmik

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

Zeitbedarf: 30 Stunden

Absprachen zur vorhabenbezogene Konkretisierung:

Die Grundlagen der Programmierung mit Java werden zum Beispiel mithilfe der Javabibliothek „Turtle“ und der Entwicklungsumgebung „BlueJ“ durch die SuS erarbeitet.

Alternativ kann auch die Entwicklungsumgebung „Greenfoot“ eingesetzt werden. Es werden zunächst vorgegebene Grafiken programmiert und die Reihe schließt mit einem selbstständig programmierten Projekt ab. Zu den Grundlagen zählen einfache Datentypen, Methoden und Parameter, Variablen und Wertzuweisungen, Schleifen und Verzweigungen. Die Schülerinnen und Schüler wenden ihr Wissen kontextbezogen auf graphische Problemstellungen an. Dabei durchlaufen sie den kontinuierlichen Prozess von Implementieren, Kompilieren und Testen.

Unterrichtssequenzen

- Installation der Entwicklungsumgebung (zu Hause)
- Erste Schritte mit der Entwicklungsumgebung
- Aufteilen eines Programms in Methoden
- Variablen, Datentypen und Wertzuweisung
- Nutzung von Parametern
- for-Schleife
- Verzweigungen

Zu entwickelnde Kompetenzen

Die Schülerinnen und Schüler

- analysieren und erläutern einfache Algorithmen und Programme (A)
- modifizieren einfache Algorithmen und Programme (I)
- entwerfen einfache Algorithmen und stellen sie umgangssprachlich dar (M)
- implementieren Algorithmen unter Verwendung von Variablen und Wertzuweisungen, Kontrollstrukturen sowie Methodenaufrufen (I)
- implementieren Methoden unter Nutzung dokumentierter Klassenbibliotheken (I)
- testen Programme schrittweise anhand von Beispielen (I)

Beispiele, Materialien, Medien

- Entwicklungsumgebung (z.B. BlueJ)
- Java-Bibliothek im Anwendungskontext (z.B. Turtle)
- Arbeitsblätter
- Klassendokumentation (online)

Unterrichsvorhaben E-III

Thema: Umgang mit Arrays sowie Entwurf und Analyse einfacher Sortierverfahren

Leitfragen: Welche Möglichkeiten bieten Arrays zur Speicherung von Daten, und wie kann man diese im Programm verarbeiten? Wie kann man Arrays zum Sortieren von großen Datenmengen benutzen?

Zeitbedarf: 20 Stunden

Absprachen zur vorhabenbezogene Konkretisierung:
Nach Abschluss des Einstiegs in die Programmierung lernen die SuS Programme mit Textein- und ausgabe kennen. Aufbauend auf den programmiertechnischen Grundlagen können die SuS ihr Wissen, inbesondere zu for-Schleifen und Verzeigungen, auf die Datenstruktur „Array“ anwenden. Das Unterrichtsvorhaben hat das Ziel, die Implementierung der iterativen Sortierverfahren am Anfang der Qualifikationsphase I vorzubereiten und die nötigen Fertigkeiten im Umgang mit Arrays zu erlernen. Nach vorbereitenden Aufgaben (z. B. Bildung des Mittelwertes, Umkehren der Reihenfolge) werden zwei bis drei der einfachen Sortierverfahren behandelt, indem sie simuliert werden (die Implementierung folgt am Anfang der Q1).

Unterrichtssequenzen
- Einfache Programme mit Textein- und Ausgabe
- Datenstruktur „Array“
- Einfache Berechnungen mit Arrays (z.B. Summe, Mittelwert, Maximum)
- Einfache Verschiebeoperationen (z.B. Umkehren der Reihenfolge)
- Einfache Sortierverfahren (z.B. Selection Sort, Bubble Sort, Insertion Sort)

Zu entwickelnde Kompetenzen
Die Schülerinnen und Schüler
- implementieren Algorithmen unter Verwendung von Variablen und Wertzuweisungen, Kontrollstrukturen sowie Methodenaufrufen (I)
- testen Programme schrittweise anhand von Beispielen (I)
- analysieren Such- und Sortieralgorithmen und wenden sie auf Beispiele an (D)

Beispiele, Materialien, Medien

- Entwicklungsumgebung (z.B. BlueJ)
- Lehrbuch
- Arbeitsblätter

Unterrichsvorhaben E-IV

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

Leitfragen: Was sind Objekte und Klassen? Wie programmiert man objektorientiert? Wie stellt man Klassen und Objekte graphisch, wie im Programm dar? Welche Beziehungen gibt es zwischen Klassen? Wie lassen sich Animationen und Simulationen realisieren?

Zeitbedarf: 20 Stunden

Absprachen zur vorhabenbezogene Konkretisierung:

Aufbauend auf dem Wissen über einfache Datentypen lernen die SuS, wie man komplexe Datentypen mit Attributen und Operationen (Klassen) definiert, sowie die Beziehungen zwischen Klassen (Assoziation und Vererbung). Die Darstellung erfolgt graphisch in UML (Klassen- und Objektdiagramme). Die SuS lernen dann eine bestehende Klassenbibliothek, zum Beispiel mit der graphischen Programmierumgebung Greenfoot kennen und fügen eigene Klassen hinzu, mit denen einfache graphische Aufgaben (z.B. Bewegung innerhalb eines Labyrinths) gelöst werden.

Unterrichtssequenzen

- Klassen und Objekte
- Darstellung von Klassen und Objekten mithilfe von UML
- Beziehungen zwischen Klassen (Vererbung)
- Einstieg in eine vorgegebene Klassenbibliothek, z.B. Greenfoot
- Implementierung einfacher Algorithmen mit der Klassenbibliothek

Zu entwickelnde Kompetenzen

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)
- modellieren Klassen unter Verwendung von Vererbung (M)
- ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen oder lineare Datensammlungen zu (M)
- ordnen Klassen, Attributen und Methoden ihren Sichtbarkeitsbereich zu (M)
- stellen den Zustand eines Objekts dar (D)
- stellen die Kommunikation zwischen Objekten grafisch dar (M)
- stellen Vererbungsbeziehungen in Diagrammen grafisch dar (D)
- dokumentieren Klassen durch Beschreibung der Funktionalität der Methoden (D)
- analysieren und erläutern eine objektorientierte Modellierung (A)
- implementieren Klassen in einer Programmiersprache auch unter Nut-zung dokumentierter Klassenbibliotheken (I)

Beispiele, Materialien, Medien

- Entwicklungsumgebung (z.B. Greenfoot)
- Ggf. Programm zum Erstellen von UML-Diagrammen
- Arbeitsblätter

<u>Unterrichsvorhaben Q1-I</u>

Thema: Implementierung einfacher Sortierverfahren, Laufzeitanalyse

Leitfragen: Wie werden Sortierverfahren implementiert? Wie effizient arbeitet ein Algorithmus bezüglich Laufzeit und Speicherplatzbedarf?

Zeitbedarf: 10 Stunden

Absprachen zur vorhabenbezogene Konkretisierung

Aufbauend auf der Modellierung einfacher Sortierverfahren in der Einführungsphase implementieren die SuS unter Anleitung zwei bis drei der einfachen Sortierverfahren. Sie lernen Methoden kennen, die Effizienz dieser Sortierverfahren bezüglich ihrer Laufzeit und ihres Speicherplatzbedarfs zu analysieren.

Unterrichtssequenzen

- Wiederholung der einfachen Sortierverfahren
- Implementierung der Sortierverfahren in Java
- Analyse der Laufzeit bei wachsender Größe der zu sortierenden Liste
- Begriff der „quadratischen Laufzeit“

Zu entwickelnde Kompetenzen

Die Schülerinnen und Schüler

- implementieren Algorithmen unter Verwendung von Variablen und Wertzuweisungen, Kontrollstrukturen sowie Methodenaufrufen (I)
- testen Programme schrittweise anhand von Beispielen (I)
- implementieren Such- und Sortieralgorithmen und wenden sie auf Beispiele an (D)
- beurteilen die Effizienz von Algorithmen am Beispiel von Sortierverfahren hinsichtlich Zeitaufwand und Speicherplatzbedarf (A)

Beispiele, Materialien, Medien

- Entwicklungsumgebung (z.B. BlueJ)
- Arbeitsblätter

<u>Unterrichsvorhaben Q1-II</u>

Thema: Rekursion, Analyse rekursiver Sortierverfahren

Leitfragen: Wie arbeitet ein rekursiver Algorithmus? Wie beendet man eine Rekursion? Wie effizient arbeitet ein rekursiver Algorithmus bezüglich Laufzeit und Speicherplatz-bedarf?

Zeitbedarf: 10 Stunden

Absprachen zur vorhabenbezogene Konkretisierung

Rekursion kann z.B. aufbauend auf der graphischen Programmierung in der Einführungsphase anhand rekursiver Grafiken erarbeitet werden. Die SuS analysieren rekursive Programme und implementieren selbstständig weitere Programme, um z.B. rekursive Grafiken zu zeichnen. Auf diesem Basiswissen baut die Behandlung eines rekursiven Sortierverfahrens, z.B. Merge Sort oder Quicksort auf, den die SuS bezüglich seines Laufzeit- und Speicherplatzbedarfs analysieren.

Unterrichtssequenzen

- Analyse eines rekursiven Programms
- Implementierung weiterer rekursiver Programme
- Analyse eines rekursiven Sortierverfahrens
- Begriff der „logarithmischen Laufzeit“

Zu entwickelnde Kompetenzen

Die Schülerinnen und Schüler

- implementieren Algorithmen unter Verwendung von Variablen und Wertzuweisungen, Kontrollstrukturen sowie Methodenaufrufen (I)
- testen Programme schrittweise anhand von Beispielen (I)
- implementieren Such- und Sortieralgorithmen und wenden sie auf Beispiele an (D)
- beurteilen die Effizienz von Algorithmen am Beispiel von Sortierverfahren hinsichtlich Zeitaufwand und Speicherplatzbedarf (A)

Beispiele, Materialien, Medien

- Entwicklungsumgebung (z.B. BlueJ)
- Arbeitsblätter

<u>Unterrichsvorhaben Q1-III</u>

Thema: Lineare Datenstrukturen Liste, Queue und Stack
Analyse und Entwurf sowie Implementierung von Anwendungsprogrammen

Leitfragen: Wie können die Nachteile der Datenstruktur „Array“ durch dynamische Datenstrukturen „behoben“ werden? Wie funktioniert die rekursive Verkettung von

„Knoten“? Welche besonderen Anwendungssituationen gibt es für die Strukturen „Queue“ und „Stack“?

Zeitbedarf: 30 Stunden

Absprachen zur vorhabenbezogene Konkretisierung

Aufbauend auf den Konzepten der objektorientierten Modellierung (Assoziation und Vererbung) lernen die SuS die dynamischen Datenstrukturen kennen, die auf der rekursiven Verkettung von Objekten basiert. Sie lernen die Vor- und Nachteile gegenüber der Datenstruktur Array kennen. Da die Datenstrukturen „Queue“ und „Stack“ einfacher zu verstehen sind als die allgemeinere „Liste“ sollten diese Strukturen zuerst behandelt werden. Die SuS lernen die Methoden von Queue und Stack kennen und implementieren typische Anwendungen, sowie im Anschluss auch Anwendungen für die allgemeine Liste. Da die Java-Klassen, die für das Zentralabitur in NRW vorgegeben werden, generisch sind, wird die Anwendung generischer Klassen behandelt (soweit das für die Anwendung der dynamischen Datenstrukturen notwendig ist).

Unterrichtssequenzen

- Bildung dynamischer Datenstrukturen durch rekursive Verkettung von Knoten
- Implementierung von Anwendungen für Queue und Stack
- Implementierung von Anwendungen für die Liste

Zu entwickelnde Kompetenzen

Die Schülerinnen und Schüler

- ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M)
- stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D)
- modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M)
- ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie lineare und nichtlineare Datensammlungen zu (M)
- stellen lineare Strukturen grafisch dar und erläutern ihren Aufbau (D)
- erläutern Operationen dynamischer (linearer) Datenstrukturen (A),
- implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I)

- analysieren und erläutern Algorithmen und Programme (A)
- modifizieren Algorithmen und Programme (I)
- stellen iterative Algorithmen umgangssprachlich und grafisch dar (D)
- implementieren iterative Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I)

Beispiele, Materialien, Medien

- Entwicklungsumgebung (z.B. BlueJ)
- Ggf. Programm zum Erstellen von UML-Diagrammen
- Arbeitsblätter

Unterrichsvorhaben Q1-IV

Thema: Binärbaum und binärer Suchbaum
Analyse und Entwurf sowie Implementierung von Anwendungsprogrammen

Leitfragen: Wie können sortierte Daten mithilfe von Baumstrukturen effizienter dargestellt werden als mit linearen Datenstrukturen? Wie sind Baumstrukturen aufgebaut? Wie können Baumstrukturen mit rekursiven Methoden durchsucht, erweitert und traversiert werden? Welche Möglichkeiten der rekursiven Traversierung gibt es?

Zeitbedarf: 20 Stunden

Absprachen zur vorhabenbezogene Konkretisierung

Aufbauend auf den Konzepten der linearen Datenstrukturen lernen die SuS die rekursive Datenstruktur des Binärbaums kennen sowie die Möglichkeit, einen binären Suchbaum anhand eines Vergleichskriteriums aufzubauen. Sie analysieren den Zeitaufwand der Suche im binären Suchbaum im Vergleich zur Suche in der linearen Liste. Es werden Anwendungen für den Binärbaum und den binären Suchbaum implementiert, bei denen die Suche, das Hinzufügen von Elementen in einen Binärbaum und die Traversierung (z.B. zur Ausgabe aller Elemente eines Binärbaums) behandelt werden.

Unterrichtssequenzen

- Aufbau eines Binärbaums

- Implementierung von Anwendungen für Binärbaum und binären Suchbaum (Einfügen von Elementen und Suche)
- Analyse der Laufzeit für Einfügen und Suchen im binären Suchbaum
- Traversieren eines Binärbaums (pre-, in- und postorder-Reihenfolgen)

Zu entwickelnde Kompetenzen

Die Schülerinnen und Schüler

- ermitteln bei der Analyse von Problemstellungen Objekte, ihre Eigenschaften, ihre Operationen und ihre Beziehungen (M)
- stellen lineare und nichtlineare Strukturen grafisch dar und erläutern ihren Aufbau (D)
- modellieren Klassen mit ihren Attributen, Methoden und ihren Assoziationsbeziehungen unter Angabe von Multiplizitäten (M)
- ordnen Attributen, Parametern und Rückgaben von Methoden einfache Datentypen, Objekttypen sowie/oder lineare und nichtlineare Datensammlungen zu (M)
- ordnen Klassen, Attributen und Methoden ihre Sichtbarkeitsbereiche zu (M)
- stellen Klassen und ihre Beziehungen in Diagrammen grafisch dar (D)
- analysieren und erläutern objektorientierte Modellierungen (A)
- implementieren Klassen in einer Programmiersprache auch unter Nutzung dokumentierter Klassenbibliotheken (I)
- stellen iterative und rekursive Algorithmen umgangssprachlich und grafisch dar (D)
- entwickeln iterative und rekursive Algorithmen unter Nutzung der Strategien „Modularisierung“, „Teilen und Herrschen“ und „Backtracking“ (M)
- implementieren iterative und rekursive Algorithmen auch unter Verwendung von dynamischen Datenstrukturen (I)
- erläutern Operationen dynamischer (nicht-linearer) Datenstrukturen (A)

Beispiele, Materialien, Medien

- Entwicklungsumgebung (z.B. BlueJ)
- Ggf. Programm zum Erstellen von UML-Diagrammen
- Arbeitsblätter

Unterrichsvorhaben Q1-V

Thema: Informatiksysteme (Von-Neumman-Rechner und Netzwerktopologien)

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

Zeitbedarf: 10 Stunden

Absprachen zur vorhabenbezogene Konkretisierung

Anhand einer von-Neumann-Architektur und einem maschinennahen Programm wird die prinzipielle Arbeitsweise von Computern verdeutlicht. Anschließend werden Vor- und Nachteile der Grenzen der Automatisierbarkeit angesprochen und der Einsatz von Informatiksystemen hinsichtlich prinzipieller Möglichkeiten und prinzipieller Grenzen beurteilt.
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. Verschiedene Netzwerk-Topologien werden entwickelt. Über die Sicherheit von Netzwerkanwendungen wird das Augenmerk auf verschiedene symmetrische und asymmetrische kryptografische Verfahren gelenkt, welche dann am Anfang der Q2 analysiert und erläutert werden.

Unterrichtssequenzen

- Modell des von-Neumann-Rechners
- Beispiele für maschinennahe Programme
- Struktur von Rechner-Netzwerken und verschiedene Topologien
- Schichtenmodell der Netzwerkkommunikation

Zu entwickelnde Kompetenzen

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)
- beschreiben und erläutern Netzwerk-Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken (A)
- beschreiben und erläutern Netzwerk-Topologien, die Client-Server-Struktur und Protokolle sowie ein Schichtenmodell in Netzwerken (A)
- untersuchen und bewerten anhand von Fallbeispielen Auswirkungen des Einsatzes von Informatiksystemen sowie Aspekte der Sicherheit von Informatiksystemen, des Datenschutzes und des Urheberrechts (A)
- untersuchen und bewerten Problemlagen, die sich aus dem Einsatz von Informatiksystemen ergeben, hinsichtlich rechtlicher Vorgaben, ethischer Aspekte und gesellschaftlicher Werte unter Berücksichtigung unterschiedlicher Interessenlagen (A)

Beispiele, Materialien, Medien

- Ggf. Modellrechner für maschinennahe Programme
- Ggf. Simulationsprogramm für Computernetzwerke
- Arbeitsblätter

Unterrichsvorhaben Q2-I

Thema: Kryptographie

Leitfragen: Warum ist es sinnvoll, Daten zu verschlüsseln? Was ist ein symmetrisches Verschlüsselungsverfahren? Was ist das Problem der „Schlüsselverteilung“, und wie wird es durch asymmetrische Verschlüsselungsverfahren gelöst?

Zeitbedarf: 5 Stunden

Absprachen zur vorhabenbezogene Konkretisierung

Anhand von Beispielen wird motiviert, warum kryptographische Verfahren wichtig sind und welche Rolle sie in der Geschichte gespielt haben. Es werden beispielhaft symmetrische Verfahren (z.B. Vigenère) behandelt und an diesen Beispielen das Problem der Schlüsselverteilung erläutert. Den Abschluss bildet ein asymmetrisches Verfahren (z.B.

RSA), das mithilfe eines Schlüsselpaares (öffentlich bzw. privat) das Problem der Schlüsselverteilung löst.

Unterrichtssequenzen

- Symmetrische Verschlüsselungsverfahren
- Asymmetrische Verschlüsselungsverfahren

Zu entwickelnde Kompetenzen

Die Schülerinnen und Schüler

- analysieren und erläutern Algorithmen und Programme (A)
- analysieren und erläutern Eigenschaften, Funktionsweisen und Einsatzbereiche symmetrischer und asymmetrischer Verschlüsselungsverfahren (A)
- untersuchen und bewerten anhand von Fallbeispielen Auswirkungen des Einsatzes von Informatiksystemen sowie Aspekte der Sicherheit von Informatiksystemen, des Datenschutzes und des Urheberrechts (A),
- untersuchen und bewerten Problemlagen, die sich aus dem Einsatz von Informatiksystemen ergeben, hinsichtlich rechtlicher Vorgaben, ethischer Aspekte und gesellschaftlicher Werte unter Berücksichtigung unterschiedlicher Interessenlagen (A)

Beispiele, Materialien, Medien

- Simulationsprogramme für kryptographische Verfahren (z.B. Cryptool)
- Arbeitsblätter

<u>Unterrichsvorhaben Q2-II</u>

Thema: Modellierung und Nutzung relationaler Datenbanken in Anwendungskontexten

Leitfragen: Was ist eine Datenbank? Welche Beziehungen gibt es zwischen Datensätzen? Wie kann man Daten mithilfe eines Entity-Relationship-Modells modellieren? Wie funktioniert die Abfragesprache SQL, und wie kann sie genutzt werden, Informationen aus einer Datenbank abzufragen? Wie normalisiert man eine Datenbank?

Zeitbedarf: 25 Stunden

Absprachen zur vorhabenbezogene Konkretisierung

Die Nutzung eines Datenbanksystems (z.B. Access oder Libre Office Base) wird anhand von Beispiel-Datenbanken eingeführt. Die Begriffe „Entität“ und „Beziehung“ werden anhand von Beispielen erarbeitet und dann zur Modellierung von Datenbanken eingesetzt. Die Abfragesprache SQL wird erläutert, insbesondere das Abfragen von Daten aus mehreren Tabellen mithilfe von Joins. Abschließend wird die Normalisierung einer Datenbank schrittweise erarbeitet und die Vor- und Nachteile der Normalisierung erarbeitet.

Unterrichtssequenzen

- Aufbau einer Datenbank und Nutzung eines Datenbanksystems
- Entity-Relationship-Modelle
- Die Abfragesprache SQL
- Normalisierung einer Datenbank

Zu entwickelnde Kompetenzen

Die Schülerinnen und Schüler

- ermitteln für anwendungsbezogene Problemstellungen Entitäten, zugehörige Attribute, Relationen und Kardinalitäten (M)
- stellen Entitäten mit ihren Attributen und die Beziehungen zwischen Entitäten mit Kardinalitäten in einem Entity-Relationship-Diagramm grafisch dar (D)
- modifizieren eine Datenbankmodellierung (M)
- modellieren zu einem Entity-Relationship-Modell ein relationales Datenbankschema (M)
- implementieren ein relationales Datenbankschema als Datenbank (I)
- bestimmen Primär- und Sekundärschlüssel (M)
- analysieren und erläutern eine Datenbankmodellierung (A)
- ermitteln Ergebnisse von Datenbankabfragen über mehrere verknüpfte Tabellen (D)
- analysieren und erläutern die Syntax und Semantik einer Datenbankabfrage (A)
- verwenden die Syntax und Semantik einer Datenbankabfragesprache, um Informationen aus einen Datenbanksystem zu extrahieren (I)
- erläutern die Eigenschaften normalisierter Datenbankschemata (A)
- überprüfen Datenbankschemata auf vorgegebene Normalisierungseigenschaften (D)
- überführen Datenbankschemata in die 1. bis 3. Normalform (M)

- erläutern Eigenschaften und Aufbau von Datenbanksystemen unter dem Aspekt der sicheren Nutzung (A),
- untersuchen und bewerten anhand von Fallbeispielen Auswirkungen des Einsatzes von Informatiksystemen sowie Aspekte der Sicherheit von Informatiksystemen, des Datenschutzes und des Urheberrechts (A)
- 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)

Beispiele, Materialien, Medien

- Datenbanksystem (z.B. Access oder Base)
- Ggf. Programm zum Erstellen von ER-Modellen
- Arbeitsblätter

Unterrichsvorhaben Q2-III

Thema: Endliche Automaten und reguläre Grammatiken

Leitfragen: Wozu benötigt man formale Sprachen? Wie sind formale Sprachen aufgebaut? Wie kann ein Computer einen Text auf die Einhaltung bestimmter Regeln überprüfen? Was sind reguläre Sprachen, und wie ist der Zusammenhang mit endlichen Automaten und regulären Grammatiken?

Zeitbedarf: 20 Stunden

Absprachen zur vorhabenbezogene Konkretisierung

Ausgehend vom Compiler, der den Quelltext eines Programms auf Fehler analysiert und in Maschinensprache übersetzt, werden formale Sprachen motiviert. Die SuS lernen das Modell des endlichen Automaten mit seinen Zuständen und Zustandsübergängen kennen sowie seine Anwendung zum „Erkennen“ von Wörtern einer formalen Sprache. Im Anschluss werden regluäre Grammatiken erarbeitet, sowie die Möglichkeit, die gleichen Wörter mithilfe von Ableitungsbäumen zu erzeugen. Abschließend wird die Umwandlung eines endlichen Automaten in eine regluäre Grammatik, und umgekehrt, erarbeitet.

Unterrichtssequenzen

- Endliche Automaten und formale Sprachen
- Reguläre Grammatiken und Ableitungsbäume
- Umwandlung eines endlichen Automaten in eine regluäre Grammatik, und umgekehrt

Zu entwickelnde Kompetenzen

Die Schülerinnen und Schüler

- analysieren und erläutern die Eigenschaften endlicher Automaten einschließlich ihres Verhaltens bei bestimmten Eingaben (A)
- ermitteln die Sprache, die ein endlicher Automat akzeptiert (D)
- entwickeln und modifizieren zu einer Problemstellung endliche Automaten (M)
- stellen endliche Automaten in Tabellen oder Graphen dar und überführen sie in die jeweils andere Darstellungsform (D)
- entwickeln zur Grammatik einer regulären Sprache einen zugehörigen endlichen Automaten (M)
- analysieren und erläutern Grammatiken regulärer und kontextfreier Sprachen (A)
- modifizieren Grammatiken regulärer und kontextfreier Sprachen (M)
- ermitteln die formale Sprache, die durch eine Grammatik erzeugt wird (A)
- entwickeln zu einer regulären Sprache eine Grammatik, die die Sprache erzeugt (M)
- entwickeln zur akzeptierten Sprache eines Automaten eine zugehörige Grammatik (M)
- beschreiben an Beispielen den Zusammenhang zwischen Automaten und Grammatiken (D)
- zeigen / erläutern die Grenzen endlicher Automaten und regulärer Grammatiken / Sprachen im Anwendungszusammenhang auf (A)

Beispiele, Materialien, Medien

- Ggf. Programm zur Modellierung endlicher Automaten
- Arbeitsblätter

2.2 Fachdidaktische und fachmethodische Grundsätze

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 SuS
9. Die SuS 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 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

2.3.1 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.

2.3.2 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.

2.3.3 Formen der Leistungsüberprüfung

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

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

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

Punkte	Note
0 – 19	6
20 – 26	5-
27 – 32	5
33 – 38	5+
39 – 44	4-
45 – 49	4
50 – 54	4+
55 – 59	3-
60 – 64	3
65 – 69	3+
70 – 74	2-
75 – 79	2
80 – 84	2+
85 – 89	1-
90 – 94	1
95 – 100	1+

Die Fachkonferenz legt die Dauer der Kursarbeiten und Klausuren fest. Am Heinrich-Heine-Gymnasium Köln gelten für die Sekundarstufe II folgende Regelungen. In der Qualifikationsphase I kann die erste Klausur im 2. Halbjahr durch eine Facharbeit ersetzt werden.

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

Mitarbeit im Unterricht

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
- Führen eines Lerntagebuchs
- Projektarbeit (oft in Form von Gruppenarbeit)
- Lernerfolgsüberprüfungen und schriftliche Übungen

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.

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.

2.4 Lehr- und Lernmittel

Eingesetzte Lehrbücher und Arbeitsmaterialien:
- Lehrbuch „Schoeningh Informatik I“ für die Einführungsphase
- Arbeitsblätter und Programmvorlagen der Lehrkräfte

Eingesetzte Software (jeweils in der aktuellen Version):
- Java SDK
- BlueJ
- Bibliothek „Java-Turtle“
- JavaEditor
- StructEdit
- UMLet
- Greenfoot
- xampp
- LibreOffice
- Lernplattform Moodle

3 Entscheidungen zu fach- und unterrichtsübergreifenden Fragen

Die Informatikfachschaft nutzt die sich flexibel ergebenden Gelegenheiten, mit anderen Fächern zu kooperieren.
Falls es sich anbietet, werden Exkursionen durchgeführt.

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 2017 für den ersten Durchgang durch die gymnasiale Oberstufe nach Erlass des Kernlehrplanes verbindlich. Jeweils vor Beginn eines neuen Schuljahres, d.h. erstmalig nach Ende der Einführungsphase im Sommer 2015 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 2017 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.