

Begründung:

Die Schülerinnen und Schüler haben im Fachunterricht der Sekundarstufe I vielfältige Anwendersysteme kennen gelernt, damit konkrete Anwendungsprobleme gelöst und individuelle sowie gesellschaftliche Folgen der Nutzung reflektiert. Kennzeichnend für den Umgang mit Anwendersystemen ist die interaktive Arbeit, d.h. das benutzte System führt direkt die Aktionen aus, die der Anwender per Tastatur oder Maus auslöst.

Für den kompetenten Umgang mit modernen Informatiksystemen benötigt man aber auch Hintergrundwissen über Wirkprinzipien und Möglichkeiten automatischer Informationsverarbeitung. Der Zugang zu diesen Inhalten erfolgt über das Thema *Grundlagen der Programmierung*. Da der Begriff der Programmierung sehr unterschiedlich interpretiert wird, ist eine Klärung nötig. Programmierung wird nicht als ingenieurwissenschaftliche Tätigkeit, sondern ebenso wie das elementare Rechnen in der Mathematik als eine Primärerfahrung mit der Informatik verstanden. Zentrale Begriffe der Informatik erwachsen aus den Erfordernissen des Programmierens; die Erfahrung des Programmierens spielt eine Schlüsselrolle für das Verständnis informatischer Grundbegriffe. Programmierung wird aufgefasst als Problemlösen (Modellieren und Strukturieren) unter Anwendung von Informatikprinzipien und -methoden, als Teilgebiet der Informatik, das die Methoden und Denkweisen beim Entwickeln von Programmen umfasst. Die Programmiersprache ist lediglich Mittel zum Zweck und steht nicht im Zentrum des Unterrichts. Ein Programmierkurs, der von Sprach- anstatt von Problemstrukturen ausgeht, verfehlt die Zielsetzungen eines allgemein bildenden Unterrichts.

Programmieren beginnt mit dem Modellieren der betrachteten Problemsituation. Dabei werden für den jeweiligen Zweck die wichtigsten Merkmale und Aktionsmöglichkeiten der beteiligten Objekte unter Vernachlässigung der unwichtigen herausgearbeitet, beschrieben und strukturiert. Daraus ergibt sich das informatische Modell, das die Rolle eines Bauplans für die Konstruktion eines Informatiksystems spielt. Bei der Modellierung sind äußere Aspekte in Form der Gestaltung der Benutzungsoberfläche zur Ein- und Ausgabe und innere Aspekte wie Darstellung von Informationen durch Daten sowie Verarbeitung von Daten durch Algorithmen zu beachten. Algorithmen können mit Hilfe von Struktogrammen visualisiert werden. Weitere wichtige Hilfsmittel sind Zustands- bzw. Ablaufdiagramme.

Die Konstruktion des informatischen Modells erfolgt mit den Mitteln einer objektorientierten Programmiersprache. Um an die Erfahrungswelt der Schülerinnen und Schüler anzuschließen, ist der Einsatz eines Entwicklungssystems erforderlich, mit dem Programme für grafikorientierte Benutzungsoberflächen entwickelt werden können. Insbesondere empfehlen sich hierzu Delphi oder Java. Diese beiden Entwicklungssysteme werden durch entsprechende Beispiele, Unterrichtsvorschläge und Materialien auf dem Hessischen Bildungsserver unterstützt. Sie beinhalten ein umfangreiches Hilfesystem, das zur Förderung des selbstständigen Lernens eingesetzt werden kann. Ein zusätzlicher Beitrag zur Stärkung des Sprachenlernens erfolgt durch die Nutzung der englischsprachigen Programmdokumentationen.

An die erfolgreiche Implementierung eines Informatiksystems schließt sich deren Präsentation, die Diskussion der Lösung und eventueller offener Probleme samt ihrer individuellen und gesellschaftlichen Bedeutung sowie die kritische Reflexion des gesamten Lösungsprozesses an. Damit erfahren die Schülerinnen und Schüler ihre Fortschritte in Bezug auf Fach-, Methoden-, Sozial- und Selbstkompetenz.

Verbindliche Unterrichtsinhalte/Aufgaben:

Variablen	Als benannter Behälter für Werte eines bestimmten Datentyps, Wertzuweisung
Einfache Datentypen mit deren relevanten Operationen und Relationen	Integer, Real, Char, Boolean
Strukturierte Datentypen mit ihren relevanten Operationen und Relationen	String, Feld (array)

Kontrollstrukturen	Anweisungen, Sequenzen, Schleifen, Fallunterscheidungen, Syntaxdiagramme
Struktogramme	grafische Darstellung von Algorithmen
Modularisierung	Prozeduren, Parameter
Benutzeroberfläche Mensch-Maschine-Interaktion	grundlegende Ein-/Ausgabe-Komponenten Ereignisse, Ereignisroutinen
Zustandsorientierte Modellierung	Zustände, Übergänge, Zustandsdiagramme

Fakultative Unterrichtsinhalte/Aufgaben:

Externes Speichern von Zuständen	Textdateien
Funktionale Modellierung	Aufteilung komplexer Systeme in Teilsysteme, Datenflussdiagramme, schrittweise Verfeinerung, Schnittstellen

Arbeitsmethoden der Schülerinnen und Schüler/Hinweise und Erläuterungen:

Die Unterrichtsinhalte werden anhand von Themen erarbeitet. Ein themenorientierter Zugang ist an Verstehensorientierung und auf die Initiierung und Stärkung sachbezogener Motivation ausgerichtet. Er erlaubt exemplarische Vertiefung und umgeht somit das Problem der Stofffülle. Die Themen sind so auszuwählen, dass sie einen Beitrag zur Stiftung kultureller Kohärenz, zur Weltorientierung und zur Entfaltung von Verantwortungsbereitschaft leisten. Sie sollten sich auf authentische Problemsituationen beziehen, die auf Grund ihres Realitätsgehalts und ihrer Relevanz dazu motivieren, neues Wissen oder neue Fertigkeiten zu erwerben. Beispiele für geeignete Themen sind Kalender, Kryptografie, Grafik, Spiele, Ton- oder Bildverarbeitung, Auswertung von Messergebnissen, Simulation technischer Automaten. Auf Grund des interdisziplinären Charakters der Informatik gibt es viele Bezüge zu den anderen Unterrichtsfächern und damit Gelegenheit, auch fachübergreifende und fächerverbindende Themen zu behandeln.

Im Sinne des problemorientierten Lernens sollen Phasen expliziter Instruktion durch die Lehrkraft und konstruktiver Aktivität der Lernenden in einer sinnvollen Balance stehen. Die Schülerinnen und Schüler müssen ausreichend Gelegenheit haben, praktische Erfahrungen im Umgang mit dem Programmentwicklungssystem zu erwerben und Möglichkeiten des selbstständigen Lernens zu nutzen. In der Jahrgangsstufe 11 muss von sehr heterogenen Vorkenntnissen und großen Differenzen in den Lern- und Arbeitstechniken ausgegangen werden. Problemorientiertes Lernen bietet unter diesen Umständen sinnvolle Voraussetzungen für die innere Differenzierung sowie für Partner- und Teamarbeit.

Mit der zustandsorientierten Modellierung werden typischerweise endliche Automaten beschrieben, die genauer im Kurshalbjahr 13.1 behandelt werden. Da Variablen im Grunde nichts anderes als Programmzustände beschreiben und die Wertzuweisung in imperativen Programmiersprachen im engeren Sinne eine Zustandsänderung darstellt, ist die Verwendung von Variablen immer als zustandsorientierte Modellierung zu verstehen und daher verbindlicher Unterrichtsinhalt dieses Kurshalbjahres. Beispiele für die zustandsorientierte Modellierung sind auf dem Bildungsserver dokumentiert (Ampelschaltung, POP3-Protokoll).

Am Ende der Jahrgangsstufe 11 kann ein kleines Projekt dazu dienen, in Teamarbeit und arbeitsteilige Vorgehensweise bei der Analyse von Problemen, der Erstellung und dem Test von Lösungen sowie der Dokumentation des Lösungsprozesses einzuführen.

Querverweise: Berücksichtigung von Aufgabengebieten (§6 Abs. 4 HSchG):

Massenmedien und Kultur: D, E, F, Spa, Ita, L, PoWi

Programmierung und Simulation: M, Ch, Phy, PoW

Internet und Hypertext: Bio, PoWi

Rechtserziehung

12.1	Objektorientierte Modellierung	Std.: GK 36 LK 63
-------------	---------------------------------------	----------------------------------

Begründung:

In der Jahrgangsstufe 11 wird mit den Grundlagen der Programmierung ein Fundament geschaffen, auf dem im Halbjahr 12.1 das Thema objektorientierte Modellierung aufbaut. Im Kurshalbjahr 12.1 befassen sich die Schülerinnen und Schüler mit Konzepten, Methoden und Verfahren der objektorientierten Modellbildung in den Phasen Analyse, Design und Programmierung. Die objektorientierte Analyse ist

Ausgangspunkt der Modellierung. Ein relevanter Ausschnitt der realen Welt wird untersucht und unter Vernachlässigung der unwichtigen auf die bedeutsamen Merkmale reduziert. Der Abstraktionsprozess liefert Attribute, Methoden, Objekte und Klassen, welche in der Unified-Modeling-Language (UML) dargestellt werden. Mittels objektorientiertem Design wird die Struktur eines passenden Informatiksystems entworfen. Strukturelle Beziehungen, zeitliche Abläufe und Datenrepräsentierung sind relevante Gestaltungsaufgaben, deren Lösung sich in Konstruktionsplänen dokumentiert. Die geplante Struktur des Informatiksystems wird mit der gewählten Programmiersprache in ein lauffähiges Programm übersetzt, mit dem die Lösung getestet werden kann.

Es schließt sich eine kritische Reflexion der Vorgehensweise und der erzielten Lösung an. Dabei spielen Auswirkungen des Einsatzes dieses Informatiksystems eine wichtige Rolle. Exemplarisch kann daran aufgezeigt werden, dass Informatiksysteme soziotechnische Systeme sind, die auf die Lebens- und Arbeitswelt der Betroffenen Einfluss haben.

In Bezug auf Algorithmen findet eine Vertiefung statt. Die effiziente Informationsverarbeitung erfolgt auf der Basis von Standardalgorithmen. Die Rekursion als wichtiges Programmierkonzept wird anhand von Beispielen mit der Iteration verglichen. Beispiele aus dem Bereich der Grafik ermöglichen eine unmittelbare Rückmeldung über die Korrektheit der Lösung. Im Grundkurs genügt die Behandlung elementarer Algorithmen zum Suchen und Sortieren. Im Leistungskurs sind auch effiziente Algorithmen nach dem Prinzip *Teile und Herrsche* bzw. *Backtracking* zu behandeln.

Die Verarbeitung großer Datenmengen und flexibler Datenstrukturen erfordert die Behandlung abstrakter Datentypen. Im Sinne objektorientierter Modellierung entwirft und implementiert man abstrakte Datentypen als Klassen. In Attributen werden die Daten gespeichert; Standardalgorithmen erscheinen als Methoden der jeweiligen Klasse.

Moderne Programmentwicklungssysteme bieten fertige Klassen für die zu betrachtenden abstrakten Datentypen an, so dass diese nicht selbst entwickelt werden müssen. Unter diesem Aspekt spielt eher die Orientierung in der zum Entwicklungssystem gehörenden Klassenhierarchie, die Fähigkeit, die für die eigene Problemlösung relevanten Klassen und Methoden zu finden, und die Anwendung durch Ableitung spezialisierter Klassen, z. B. für spezielle Datentypen, eine wichtige Rolle.

Demgegenüber spielt unter dem Aspekt der eigenen Konstruktion das Zeigerkonzept zur Realisierung dynamischer Datenstrukturen eine wichtige Rolle. Die explizite Arbeit mit Zeigern ist nicht unbedingt erforderlich. Ein Einblick in das Zeigerkonzept als Variablen mit dem Variablentyp Adressen genügt. Die abstrakten Datentypen Keller und Schlange können beispielsweise als spezielle lineare Listen aufgefasst und daher von einer vorgegebenen Listen-Klasse abgeleitet werden.

Verbindliche Unterrichtsinhalte/Aufgaben:

Objektmodell	Identifikation von Objekten Objekt als Exemplar einer Klasse Kommunikation über Botschaften
Klassen	Attribute als Datenstruktur zur Repräsentierung der Information über ein Objekt Methoden als Schnittstellen für den Zugriff auf Attribute und zum Nachrichtenaustausch Darstellung von Klassen, Objektbeziehungen und Vererbung mit der grafischen Modellierungssprache UML Vererbung und Klassenhierarchie
Standardalgorithmen	rekursive und iterative Verfahren einfache Such- und Sortierverfahren binäre Suche (LK)
Abstrakte Datentypen (LK, GK fakultativ)	Repräsentierung und Standardoperationen Keller, Schlange lineare Liste, binärer Suchbaum
Effiziente Algorithmen (LK, GK fakultativ)	schnelle Sortierverfahren Suchen durch Backtracking Teile-und-Herrsche-Prinzip
Komplexität von Algorithmen (LK, GK fakultativ)	polynomiale und exponentielle Zeitkomplexität
Grundkonzepte des Software-Engineerings	Problemanalyse, Modellierung, Entwurf, Implementation, Test, Revision, Dokumentation

Fakultative Unterrichtsinhalte/Aufgaben:

Abstrakter Datentyp	Abbildungen aus (Schlüssel, Wert)-Paaren Hash-Tabelle
Graphen	Datenstruktur Tiefen- und Breitensuche Graphenalgorithmen
Internetprogrammierung	Sockets, Client-Server-Konzept z. B. POP3-Explorer, Mikro-Browser
Heuristische Verfahren	z. B. Bewertung von Spielsituationen
Polymorphie	z. B. Manipulation graphischer Objekte
Suche in Texten	Volltextsuche, Mustererkennung

Arbeitsmethoden der Schülerinnen und Schüler/Hinweise und Erläuterungen:

z. B. Bewertung von Spielsituationen z. B. Manipulation graphischer Objekte Volltextsuche, Mustererkennung

Die Schülerinnen und Schüler wenden im Laufe des Kurshalbjahres die Methodik der objektorientierten Modellierung auf ein Problem an, das auf ihr Leistungsvermögen und den zur Verfügung stehenden Zeitrahmen abgestimmt ist. Je nach Erfahrung mit Projektarbeit arbeiten sie in Teams an verschiedenen Projekten oder gemeinsam an einem Projekt. Die Bearbeitung fachübergreifender Probleme bietet sich an, wobei möglichst die behandelten abstrakte Datentypen und Algorithmen zum Einsatz kommen sollen. Die Dokumentation des Projekts erfolgt sinnvollerweise in Form von HTML-Dokumenten (siehe Grundkurs 11.1). Bei Verwendung von Java kann das Projektergebnis als Applet direkt in die Dokumentation eingebunden werden. Die Vorgabe einer Struktur für die Dokumentation ist hilfreich. Die erforderlichen UML-Diagramme können mit Grafikwerkzeugen hergestellt werden. Die Qualität der Dokumentation ergibt sich unter anderem durch die Klarheit im Aufbau, die Konzentration auf das Wesentliche, die verständliche und fachlich einwandfreie Beschreibung, den expliziten Bezug zwischen Text, Bild und Diagrammen, die kritische Diskussion der Ergebnisse und die Reflexion der Auswirkungen beim Einsatz des

Informatiksystems.

Zum Abschluss der Projektphase stellen die Teams ihre Arbeit in Form einer Präsentation im Plenum vor. Die Präsentation hat in Bezug auf die Dokumentation ihren eigenen Stellenwert, der sich aus einem Vortrag in Kombination mit Präsentationsfolien oder HTML-Seiten ergibt. Für die Beurteilung der Präsentation sind auch kommunikative Fähigkeiten, angewandte Präsentationstechnik und bewusste Körpersprache heranzuziehen.

Querverweise: Berücksichtigung von Aufgabengebieten (§6 Abs. 4 HSchG): Modellierung: PoWi, Ch