Der Informatikunterricht kann in der gymnasialen Oberstufe neu beginnen, kann aber auch auf dem Unterricht im Wahlpflichtbereich der Klassen 9 und 10 (Gesamtschule, Realschule) bzw. der Klasse 10 (Gymnasium) aufbauen.

Der Kurs in-B für den Fundamentalbereich der Einführungsphase ist für Schüler ohne Vorkenntnisse aus der Sekundarstufe I konzipiert. Er ist im wesentlichen inhaltsgleich mit dem Unterricht im Wahlpflichtfach der Sekundarstufe I. Schüler, die bereits in Klasse 9 oder 10 Informatikunterricht besucht haben, nehmen am Kurs in-B nicht teil.

Schüler, die entweder am Informatikunterricht im Wahlpflichtfach der Sekundarstufe I oder in der gymnasialen Oberstufe im Basiskurs in-B der Einführungsphase teilgenommen haben, können den Unterricht in der Kursphase mit den Kursen in-1.1, in-2.1, in-3.1 und in-4.1 fortsetzen.

Schüler können den Unterricht im Fach Informatik auch in der Kursphase neu beginnen. Sie können dann die Kurse in-1.2, in-2.2, in-3.2 und in-4.2 besuchen. Diese Kurse sind inhaltsgleich mit dem einjährigen Basiskurs in-B und den Kursen

Das Fach Informatik kann von Schülern nur dann zum Abiturprüfungsfach gewählt werden, wenn sie mindestens drei Jahre am Informatikunterricht teilgenommen haben. Sie müssen in der Kursphase die Kurse in-1.1, in-2.1, in-3.1 und in-4.1 besuchen.

Übersicht über die Kursfolgen

	Klasse 11	**Klasse 12**	**Klasse 13**
einjähriger Unterricht			
	in-B	**-**	**-**

	-	**in-1.2/in-2.2**	**-**
	-	**-**	**in-1.2/in-2.2**
zweijähriger Unterricht			
	in-B	**in-1.1/in-2.1**	**-**
	-	**in-1.2/in-2.2**	**in-3.2/in-4.2**
dreijähriger Unterricht *)			
	in-B	**in-1.1/in-2.1**	**in-3.1/in-4.1**

Anmerkungen: ***) mit der Möglichkeit der Wahl zum 3. oder 4. Prüfungsfach im Abitur**

Der Kurs in-B kann durch den Wahlpflichtunterricht der Klassen 9 und 10 (Gesamtschule) oder der Klasse 10 (Gymnasium) ersetzt werden.

Die Kurse in-1.2/in-2.2 sind inhaltsgleich mit in-B.
Die Kurse in-3.2/in-4.2 sind inhaltsgleich mit in-1.1/in-2.1.

Damit in den Kursen ausreichende Teilnehmerzahlen erreicht werden können, wird es in der Regel erforderlich sein, daß sich eine Schule für jeweils eine der möglichen Kursfolgen entscheidet und entweder in der Sekundarstufe I bzw. der Einführungsphase mit dem Informatikunterricht beginnt oder ihn erst in der Kursphase anbietet. Nur im ersten Fall ist es möglich, Informatik als Prüfungsfach

Grundsätzlich ist es jedoch möglich, in der Kursphase inhaltsgleiche Kurse organisatorisch zusammenzufassen. Danach können Schüler der Kurse in-1.1 und in-3.2 bzw. in-2.1 und in-4.2 gemeinsam unterrichtet werden. Die grundsätzliche Kurszugehörigkeit der Schüler bleibt davon unberührt. In besonders gelagerten Fällen ist es auch möglich, den Basiskurs in-B mit den Kursen in-1.2/in-2.2 organisatorisch zusammenfassen.

Die Kurse in-B bzw. in-1.2/in-2.2 sind als Kurse für Schüler ohne Vorkenntnisse von anderen Kursen in jedem Fall getrennt zu führen. Schüler, die bereits in der Sekundarstufe I oder in der Einführungsphase am Informatikunterricht teilgenommen haben, dürfen am Unterricht der Kurse in-1.2/in-2.2 nicht teilnehmen.

Das Fach Informatik darf in der gymnasialen Oberstufe nur mit Zustimmung der Senatsverwaltung für Schule, Berufsbildung und Sport angeboten und eingerichtet werden.

Die Genehmigung für Informatikunterricht wird in drei Stufen erteilt:

1. Stufe: Informatik über ein Unterrichtsjahr

2. Stufe: Informatik über zwei Unterrichtsjahre

3. Stufe: Informatik über drei Unterrichtsjahre mit der Möglichkeit der Wahl zum 3. oder 4. Prüfungsfach im Abitur

Jede Genehmigung einer Stufe schließt die vorhergehende Stufe mit ein.

Anträge auf Genehmigungen müssen u. a. folgende Angaben enthalten:
(1) Angabe der beantragten Genehmigungsstufe
(2) Angaben über die verwendete apparative Ausstattung (Rechnertyp, Zahl der Arbeitsplätze, Art der Vernetzung)

(3) Nennung der vorgesehenen Lehrkräfte unter Beifügung von Unterlagen (Kopien) über ihre Qualifikation

Die Genehmigungen werden befristet (in der Regel dann für zwei Jahre) oder unbefristet erteilt. Unbefristete Genehmigungen gelten solange, wie sich die Genehmigungsbedingungen nicht ändern. Insbesondere muß eine Erweiterung der Genehmigungsstufe oder der Einsatz anderer Lehrkräfte neu beantragt werden.

Anträge auf Genehmigung bzw. einer Veränderung der Genehmigung sind jeweils bis zum 1. April des vorangehenden Schuljahres einzureichen.

Der Rahmenplan läßt für die Wahl der inhaltlichen Beispiele so viel Raum, daß das erforderliche Einbringen berufsfeldspezifischer Aspekte leicht möglich ist. Allerdings ist die notwendige Breite der Anwendungsbeispiele sicherzustellen. Eine Beschränkung auf berufsfeldspezifische Anwendungen ist daher zu vermeiden.

Berufsfeld I (Wirtschaft und Verwaltung) und Berufsfeld XII (Ernährung und Hauswirtschaft):

Es gilt der Rahmenplan. Trotz des in der Einführungsphase nur zweistündigen Unterrichts sind dort alle Abschnitte des ersten Unterrichtsjahres zu behandeln; notwendige Konzentrationen sind im zweiten Unterrichtsjahr auszugleichen.
Der bisher im Berufsfeld I verwendete Arbeitsplan ist nicht mehr anzuwenden.

Berufsfeld II (Metalltechnik) und Berufsfeld III (Elektrotechnik):

In der Einführungsphase findet kein eigenständiger Basiskurs Informatik statt. Die Kurse "Grundlagen des Maschinenbaus"(im Berufsfeld II) bzw."Elektrotechnik mit Labor" (im Berufsfeld III)sehen in der Einführungphase 40 Unterrichtsstunden zu Themen der Informatik vor.

Im ersten und zweiten Halbjahr der Kursphase gilt der Rahmenplan mit der Einschränkung, daß einerseits Teile des Basiskurses Informatik noch berücksichtigt werden müssen, andererseits die obligatorischen Leistungsfächer "Maschinenbau" bzw. "Elektrotechnik mit Labor" Beiträge zur Informatik leisten. Am Ende des zweiten Kurshalbjahres müssen die Ziele der Kurse in-B und in-1.1/in-2.1 voll erreicht sein.

Im dritten und vierten Kurshalbjahr gilt der Rahmenplan für das Fach Informatik ohne Modifikationen.

Berufsfeld VII (Chemie, Physik, Biologie):

Es gilt der Rahmenplan Informatik ohne Modifikationen.

Die Kurse sind (mit Ausnahme der Kurse in-3.1 und in-4.1) jahresweise beschrieben.

Die beschriebenen Lernziele und Lerninhalte sind verbindlich. Die Zeitangaben sind Richtzeiten und als Empfehlung zu verstehen. Dabei wird von 80 Unterrichtsstunden im Schuljahr ausgegangen; die tatsächlich zur Verfügung stehende Zeit wird also nicht voll ausgeschöpft, um individuelle Schwerpunktsetzungen zu ermöglichen.

Die Beschreibung der Kurse orientiert sich an der fachlichen Systematik der jeweiligen Lernziele und Lerninhalte. Es ist aber im Sinne eines integrativen Informatikunterrichts erforderlich, Lernziele und Lerninhalte verschiedener Abschnitte miteinander zu kombinieren; bei der Beschreibung der Kurse wird an mehreren Stellen ausdrücklich darauf hingewiesen. Die Verteilung des Stoffes auf einzelne Unterrichtseinheiten und ihre zeitliche Abfolge ist daher vom Lehrer selbst

Bei dem Kurs in-4.1 im dritten Unterrichtsjahr stehen dem Lehrer vier Alternativen zur Verfügung, von denen er eine zu wählen hat. Es ist aber auch möglich, ein anderes Thema zu wählen; in diesem Fall muß rechtzeitig vor der Einrichtung ein entsprechender Antrag bei der Senatsverwaltung für Schule, Berufsbildung und Sport gestellt werden. Dieser Antrag muß die inhaltliche Konzeption darstellen und die Lerninhalte analog zu den beschriebenen Alternativen des Kurses in-4.1 enthalten.

Benutzung, Analyse und Konstruktion komplexer Programmsysteme
(kursiv: fundamentale Ideen)

Weitere Inhalte
(nach Möglichkeit integrativ unterrichten)

Erstes Unterrichtsjahr: Einführung in die Informatik

1.1. Benutzung und Analyse eines dokumentierten Systems (10 Stunden)

Komplexes System benutzen und analysieren
- ***Programmsystem benutzen***
- ***System- und Programmstruktur erkennen***
- ***wiederverwendbare Bausteine finden***
- ***kleine Programmänderungen mit Lehrerhilfe durchführen können***

1.2. Konstruktion von Teilalgorithmen zu Anwendungsfällen (35 Stunden)

In einem mäßig komplexen System Teilalgorithmen unter Mithilfe des Lehrers konstruieren
- ***algorithmische Elemente kennenlernen***
- ***Elemente der Programmiersprache erlernen***
- ***Verwendung von Bausteinen***
- ***Steuerung und Mithilfe des Lehrers***

2. Anwendungen und Auswirkungen der Datenverarbeitung (20 Stunden)

3. Rechnerorganisation (10 Stunden)

4. Entwicklungsgeschichte der Datenverarbeitung (5 Stunden)

Zweites Unterrichtsjahr: Grundlagen großer Programmsysteme

1. Konstruktion eines Programmsystems zur Dateiverwaltung (30 Stunden)

<u>Ein komplexes System mit Lehrerhilfe konstruieren</u>
- ***Dateien***
- ***Prozedur- und Modulkonzept***
- ***Verwendung und Konstruktion von Bausteinen***
- ***Überblick über den Software-Life-Cycle***
- ***arbeitsteilige Konstruktion***

2. Anwendung eines relationalen Datenbanksystems (10 Stunden)

3. Auswirkungen des Einsatzes von Datenverarbeitungsanlagen (10 Stunden)

4. Spezielle Algorithmen in typischen Anwendungssituationen (30 Stunden)

Drittes Unterrichtsjahr: Projekt (1. Hj.)

Mögliche Projekttypen:
- **A. Neues Projekt**
- **B. Fortführung eines früheren Projektes**
- **C. Wartung eines Softwareproduktes**

<u>Ein komplexes System weitgehend selbständig konstruieren</u>
- ***Methoden und Probleme des Software-Engineering (-Reengineering)***
- ***Software-Life-Cycle***

Vertiefungsgebiet (2. Hj.)

Mögliche Alternativen:
- **A. Theoretische Informatik (Automaten)**
- **B. Nicht-prozedurale Sprachkonzepte**
- **C. Datenbanken**
- **D. Computergrafik**

Struktureller Aufbau des Unterrichts

Jeder Schüler sollte durch den Informatikunterricht befähigt werden, die Möglichkeiten und Risiken der Anwendung von Computern und die Grenzen ihres Einsatzes zu kennen und zu erkennen. Dabei soll der Schüler bereits im Anfangsunterricht an die komplexe Struktur von Problemlösungen mit Computern

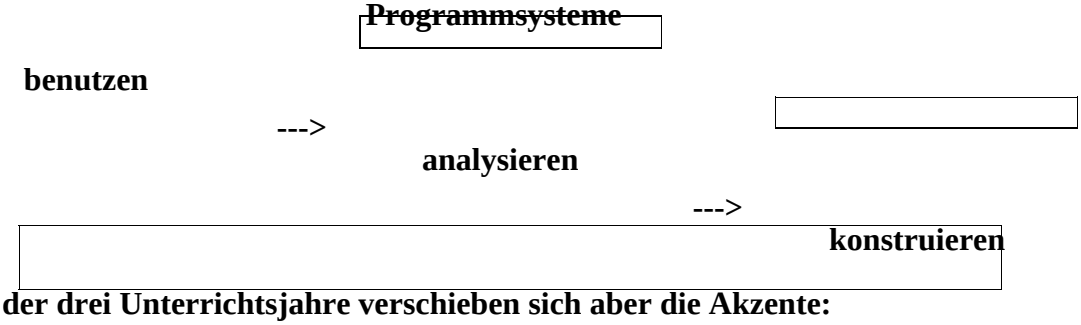
Bei der Erstellung eigener Programme zur Einführung der üblichen Daten- und Kontrollstrukturen sollen nur wenige, aber etwas umfangreichere Anwendungsfälle behandelt werden, die typische Einsatzbereiche des Computers repräsentieren. Dabei ist es nicht erforderlich, daß das gewählte Problem von jedem Schüler vollständig gelöst wird; arbeitsteilige Vorgehensweise, vom Lehrer vorgegebene Programmteile oder auch bewußte Lücken in der Lösung sind möglich und nötig.

Bei der Problemlösung sollen auch im Anfangsunterricht alle Phasen, dem jeweiligen Stand des Unterrichts angepaßt, behandelt werden; die Problemlösung darf nicht auf die reine Programmierung reduziert werden. Im zweiten Unterrichtsjahr sollen so bereits ein oder zwei Kurzprojekte durchgeführt werden, um die Begriffe Prozedur, Modul und Datei ausführlich vorzustellen. Die eigentliche Projektarbeit bleibt aber dem ersten Halbjahr des dritten Unterrichtsjahrs vorbehalten.

Die Aspekte der Anwendungen und ihre Auswirkungen sollen integrativ behandelt werden, das heißt in unmittelbarem Zusammenhang mit dem jeweils gewählten Programmbeispiel. Dasselbe gilt für Themen der Rechnerorganisation.

Im letzten Unterrichtshalbjahr bietet die vertiefende Behandlung eines auswählbaren Themenbereichs die Gelegenheit, einen besonderen Schwerpunkt zu setzen.

Der analytische und konstruktive Umgang mit komplexen Programmsystemen zieht sich durch den gesamten Unterricht. Die folgenden Stufen werden dabei mehrfach durchlaufen:



Im Laufe der drei Unterrichtsjahre verschieben sich aber die Akzente:

Komplexitätsgrad der Programme nimmt zu
Vorgaben, Steuerung und Hilfestellung durch den Lehrer nehmen ab

Algorithmik und Programmiersprache

Im Vordergrund des Informatikunterrichts steht die Algorithmik. Dabei soll die Fähigkeit zum selbständigen Finden und Formulieren algorithmischer Lösungen komplexer Probleme systematisch entwickelt werden. Dazu trägt auch die gezielte Behandlung von grundlegenden algorithmischen Aufgaben wie Suchen und Sortieren, von rekursiven Verfahren und von dynamischen Datenstrukturen bei.

Die für den Unterricht gewählte Programmiersprache dient lediglich zur Umsetzung der algorithmischen Lösungen in ablauffähige Programme. Daher steht nicht das Erlernen einer bestimmten Programmiersprache im Vordergrund, sondern das Erfassen der Methoden der Datenstrukturierung und der Algorithmenkonstruktion.

Anwendungen und Auswirkungen

Der Schüler soll über die tatsächlichen und absehbaren Anwendungen einschließlich ihrer Grenzen zuverlässig informiert sein. Er soll die Veränderungen durch den Computer-Einsatz in den verschiedenen gesellschaftlichen Bereichen kennen, und er soll die Möglichkeiten der Gestaltung und Einflußnahme beim Einsatz von Computern kennen, sei es als Betroffener oder aktiv Beteiligter. Insbesondere die Auswirkungen auf die Arbeitswelt und die

Rechnerorganisation und Rechnerbedienung

Themen der Rechnerorganistion und Rechnerbedienung haben primär die Aufgabe, Einblick in die Funktionsweise von Rechnersystemen zu geben und das Verständnis im Umgang mit ihnen zu vertiefen. Bei der Beschäftigung mit den Komponenten eines Rechnersystems muß stets darauf geachtet werden, daß vor allen Dingen die grundlegenden Aspekte und nicht nur die einer ständigen Änderung unterworfenen speziellen Kenntnisse über die der Lerngruppe zur Verfügung stehende Hard- und Software vermittelt werden. Die Bedienung des Rechners sollte schrittweise und nur in dem für den Unterricht jeweils erforderlichen Umfang thematisiert werden.

Text- und Datenbanksysteme

Neben dem für die Programmierarbeit verwendeten Editor sollen die Schüler auch ein Textverarbeitungsprogramm und seine typischen Funktionalitäten kennenlernen und auch im Informatikunterricht, z. B. für Dokumentationsarbeiten, einsetzen. Ferner wird eine Datenbank kurz behandelt; darüberhinaus sollen auch noch andere Werkzeuge wie z. B. die Tabellenkalkulation vorgestellt werden. Der Umgang mit Datenbanken und deren Grundlagen kann im Rahmen des Projektunterrichts oder der Schwerpunktsetzung im letzten Unterrichtshalbjahr vertieft werden.

Softwareprojekte

In den beiden ersten Unterrichtsjahren werden bereits die Phasen der Programmentwicklung eingeführt, aber die eigentliche Projektarbeit bleibt dem dritten Unterrichtsjahr vorbehalten. Das Projekt erstreckt sich nur über ein Halbjahr. Diese zeitliche Beschränkung ist bei der Wahl des Projektthemas und der Art der Durchführung zu berücksichtigen.

Neben der Behandlung eines neuen Projektthemas ist auch die Fortsetzung eines früheren Projekts oder seine Überarbeitung möglich; in jedem Falle werden nicht alle Projektphasen mit gleicher Intensität behandelt werden können, sondern es müssen Schwerpunkte gesetzt werden. Außerdem sollten vorgefertigte Programmbausteine eingesetzt werden, um die Programmierarbeit auf ein sinnvolles Maß zu beschränken.

Vorgefertigte Programme und Programmteile

Für den Einstieg in den Informatikunterricht ist der Einsatz eines geeigneten Programmsystems erforderlich. Dabei ist daran gedacht, daß hier Programme zum Einsatz kommen, die dem Projektunterricht der eigenen Schule oder einer anderen Schule entstammen. Unter Umständen eignen sich auch Programme, die als Kurzprojekte im zweiten Unterrichtsjahr erstellt wurden.

Um auch schon im Anfangsunterricht komplexere Probleme mit größerer Realitätsnähe behandeln zu können, müssen vom Lehrer vorgefertigte Programmteile oder Programmbausteine zur Verfügung gestellt werden. Dadurch kann sich der Unterricht stärker auf die jeweils neu zu behandelnden Inhalte konzentrieren und auch den für Übungen notwendigen Raum schaffen. Der geschickte Einsatz solcher Programmbausteine ist ebenfalls Anliegen des Unterrichts und bereitet eine effektive Projektarbeit vor.

Bei der Formulierung der Lernziele und Lerninhalte wird davon ausgegangen, daß eine prozedurale Programmier sprache verwendet wird. Sie muß eine einfache Umsetzung der dargestellten algorithmischen Bausteine und Konzepte gestatten und insbesondere die Modularisierungstechnik angemessen unterstützen.

Als Programmiersprachen, die unter diesen Voraussetzungen für den Unterricht geeignet sind, haben sich PASCAL, MODULA-2 und ELAN bewährt.

Während der Unterricht in den ersten beiden Jahren mit einer dieser drei Programmiersprachen durchgeführt werden sollte, ist für den Kurs in-3.1 auch der Einsatz einer Datenbank sprache denkbar.

Nicht-prozedurale Programmiersprachen können in der Regel nur im Kurs in-4.1 im Rahmen der Alternative B (Nicht-prozedurale Sprachkonzepte) verwendet werden. Es ist aber auch möglich, bei der Behandlung spezieller Algorithmen (Abschnitt 4 im zweiten Unterrichtsjahr) bereits geeignete nicht-prozedurale Sprachkonzepte einzuführen und sie ggf. im Projekt zu verwenden.

urse: in-B

in-1.2/in-2.2

Thema: Einführung in die Informatik (I und II)

Analyse und Konstruktion von Algorithmen

1.1. Benutzung und Analyse eines dokumentierten Systems 10 Stunden

1.2. Konstruktion von Teilalgorithmen zu Anwendungsfällen

35 Stunden

2. Anwendungen und Auswirkungen der Datenverarbeitung 20 Stunden

3. Rechnerorganisation 10 Stunden

4. Entwicklungsgeschichte der Datenverarbeitung 5 Stunden

Der Anfangsunterricht im Fach Informatik ist bislang häufig bestimmt durch die Bearbeitung vieler kleiner Aufgaben, die das benötigte grundlegende Rüstzeug für die verwendete Programmiersprache bereitstellen sollen. Diese Aufgaben haben in der Regel keine praktische Relevanz und werden, nachdem sie diesen Zweck erfüllt haben, schnell wieder beiseitegelegt. Auch die Qualität dieser Programme muß gering bleiben, z. B. schon falsche Eingaben zum Absturz des Programms führen.

Diese und andere Nachteile des bisherigen Anfangsunterrichts führten zu einer neuen Konzeption, die durch folgende grundlegende Ideen gekennzeichnet ist.

<u>Einstieg über Analyse und Wartung eines fertigen Softwareprodukts</u>

Durch diesen Einstieg wird schon frühzeitig das Denken in komplexen Zusammenhängen gefördert. Das verwendete Softwareprodukt soll in compilierter Form und als Quellcode vorliegen. Es soll aus einem Anwendungsbereich stammen, der den Schülern leicht zugänglich ist. Um die bei der Analyse eines dokumentierten Systems (Abschnitt 1.1.) angestrebten Lernziele zu erreichen, sind

- **klare Benutzerführung**
- **gute Strukturierung mit deutlich erkennbarer Modulhierarchie**
- **relativ begrenzter Funktionsumfang, der sich z. B. im wesentlichen in den Menüs widerspiegelt**
- **gut verständliche Dokumentation, die auch im Programm integriert sein kann**
- **einige Programmteile, die für eine Überarbeitung leicht zugänglich sind**

<u>Verwendung von Prozeduren aus Bibliotheken von Anfang an</u>

Kleine Hilfsprogramme, Prozeduren, Funktionen und Datentypen, die in bereits compilierter Form in Bibliotheken vorliegen, stellen mächtige Hilfsmittel bei der Konstruktion von Teilalgorithmen (Abschnitt 1.2.) dar. Ihre Verwendung ermöglicht es, auch schon im Anfangsunterricht relativ praxis- und realitätsnahe Problemstellungen zu bearbeiten.

Eine wesentliche Hilfe bieten z. B. Routinen, die eine absturzsichere und komfortable Eingabe von Zeichen, Zeichenketten und Zahlen an bestimmten Bildschirmpositionen ermöglichen.

Routinen zur Gestaltung der Benutzeroberfläche unterstützen benutzerorientierte Programm entwürfe. Diese Sichtweise schließt auch an den Unterricht in der informationstechnischen Grundbildung an, in der die Schüler fertige Software benutzt und dabei verschiedene Benutzeroberflächen kennengelernt haben.

Diese Hilfsmittel sollten in einer verständlich dokumentierten Form vorliegen und in einem Modul zusammengefaßt sein. Dadurch werden dem Schüler die Bedeutung der Dokumentation und das Prinzip der Modularisierung von Anfang an nahegebracht.

<u>Komplexe Programme von Anfang an</u>

Wesentliches Merkmal von Problemlösungen mit Hilfe der Informatik ist in der Regel

die hohe Komplexität der Programme, in der sich die Komplexität der entsprechenden Aufgabenstellung widerspiegelt. Insofern bieten sehr kleine Programme, wie sie typischerweise bei der Einführung von Daten- und Kontrollstrukturen einer Programmiersprache verwendet werden, ein verzerrtes Bild der Informatik.

Die Programmbeispiele sollten daher auch schon im Anfangsunterricht realitätsnäher und damit auch hinreichend komplex sein. Außerdem sollten sie typische Einsatzbereiche repräsentieren. Dabei ist es durchaus zulässig, daß der Lehrer Programmteile vorgibt oder auch bewußt Lücken in der Lösung gelassen werden, da die Einführung algorithmischer Bausteine das wesentliche Ziel des Anfangsunterrichts ist.

Eine arbeitsteilige Vorgehensweise, wie sie im späteren Projektunterricht unumgänglich ist, bietet sich auch hier schon insbesondere bei der Programmierarbeit an.

Integrative Behandlung von Themen zur Anwendung des Computers, zu Auswirkungen seines Einsatzes und zur Rechnerorganisation

Die Themenbereiche "Anwendungen und Auswirkungen der Datenverarbeitung" und "Rechnerorganisation" werden aus systematischen Gründen gesondert aufgeführt. Um ihre zeitliche Bedeutung zu bestimmen, sind sie zwar mit einem eigenen Stundenansatz versehen, jedoch sollten sie möglichst bei den Anwendungsfällen (Abschnitt 1.2.) integrativ behandelt werden. Dadurch wird die tatsächliche Verzahnung von Anwendungsfall, Rechnerorganisation, Problemlösung und Auswirkung besser verdeutlicht. Diese Vorgehensweise wird durch die oben dargestellte Konzentration auf wenige, komplexe Programmbeispiele unterstützt.

Benutzung von Anwendersoftware

Angesichts der weiten Verbreitung von Standardwerkzeugen in der beruflichen Praxis im Büro wird bei den Anwendungen und Auswirkungen der Datenverarbeitung (Abschnitt 2) der Anwendungsfall "Bürokommunikation" betrachtet. Hierbei ist darauf zu achten, daß die Lernziele auf einem höheren Niveau anzusiedeln sind als in einer möglicherweise früher behandelten Unterrichtseinheit in der informationstechnischen Grundbildung der Sekundarstufe I.

Hinweise: Die folgenden Lerninhalte (insbesondere die des Abschnitts 1.2.) sollen an einigen komplexen und für die Informatik repräsen tativen Beispielen erarbeitet werden. In Abhängigkeit von den gewählten Beispielen sollten einige Inhalte aus den Abschnitten 2 und 3 schon hier integrativ behandelt werden. Dies gilt insbesondere für die Auswirkungen auf die Arbeitswelt, den Datenschutz im Anwendungsfeld Schule, die Hardwarekomponenten und die Betriebssoftware eines

(10 Stunden)

Hinweise: Bei dem einführenden Beispiel ist ein Rückgriff auf ein vorhandenes System inklusive Dokumentation erforderlich.
(siehe Konzeption des ersten Unterrichtsjahres)

Lernziele	**Lerninhalte**	**Hinweise**
- bei dem dokumentierten Anwendungs beispiel erkennen, welche Teilbereiche des Problems vom Menschen bzw. vom Computer erledigt werden können		
	- Bestimmung automati - sier barer Arbeitsschritte	
- die automatisierbaren Arbeitsschritte in Teilprobleme zerlegen, ihre gegenseitige Abhängigkeit erkennen und den erwarteten Leistungsumfang des Systems beschreiben		
	- Zerlegung in geeignete Teilprobleme und Formulierung der Anforderungen an das System	
- Arbeitsaufträge mit dem Softwareprodukt bearbeiten können		

	- elementare Bedienung des Computers und Benutzung des Systems	
		Hier geht es insbesondere um Reaktionen des Systems auf unterschiedliche Eingaben.
- die Systemstruktur feststellen können		
	- Mensch-Maschine-Schnitt stellen, Funktionen des Systems	
- Abweichungen zwischen den erwarteten Anforde - rungen und der System realität sowie ggf. Systemschwächen benennen können		
	- Vergleich der von den Schülern formulierten Anforderungen mit der Realisierung durch das System	
- die zum System gehö - rende Dokumentation auswerten, insbesondere mit der Realisierung in Beziehung setzen können und lokale Änderungen vornehmen können		
	- Umgang mit der Benutzer - dokumentation des Systems	

Lernziele	**Lerninhalte**	**Hinweise**
- wissen, daß das Software produkt aus verschiedenen Teilen besteht, welche Bedeutung diese haben und wie man auf sie zugreifen kann		
	- Bestandteile des Systems: ausführbares Programm, Quellcode, zugehörige Dateien,	
- grundlegende Elemente der Programmstruktur kennen lernen		
	- Programm, Hauptprogramm, Prozeduren, Deklarationsteil, Blockstruktur, Eingaben, Ausgaben, Verarbeitung	
- die Arbeitsweise im Editor kennenlernen und kleine Programm - änderungen durchführen können		
	- kleine Programm - odifika tionen, elementare Arbeiten im Editor, compilieren	
- die übergeordnete Bedeutung einiger Programm bausteine erkennen können		
	- Wiederverwendbarkeit von Programmbausteinen	

(35 Stunden)

Hinweise:
- *Die elementaren Daten- und Kontrollstrukturen sollen an Anwendungsfällen eingeführt werden. Es sollten etwa drei hinreichend komplexe Anwendungsfälle behandelt werden, wobei jeweils weitere Strukturen zur Verfügung gestellt werden sollten. Jeder Anwendungsfall soll weitere Sprachelemente zur Verfügung stellen. Bei bekannten Sprachelementen kann arbeits teilig vorgegangen werden.*
- *Bei der Behandlung der Anwendungsfälle ist nicht an die Durchführung eines Projekts gedacht. Sinnvolle Programmergänzungen und erforderliche Hilfs routinen sollten vom Lehrer zur Verfügung gestellt werden.*

Lernziele	**Lerninhalte**	**Hinweise**
- ausgehend von einem kon kreten Anwendungs beispiel erkennen, welche Teil bereiche des Problems vom Menschen bzw. vom Computer erledigt werden können	**- Bestimmung automati - sier barer Arbeitsschritte**	
- die automatisierbaren Arbeitsschritte soweit ein grenzen und detaillieren, daß der Umfang der Problem - lösung erkennbar ist	**- Präzisierung der Aufgaben stellung**	
- ein Problem durch Zerle gung in Teilprobleme ein - schließ lich des Erkennens ihrer Abhängigkeiten analysieren können	**- Zerlegung in geeignete Teilprobleme**	
- beschreiben, welche Tätigkeiten das System vom Benutzer erwartet, bzw. wie die Kommunikation mit dem System aussehen soll		

- **Beschreibung der Schnitt stelle Mensch-Maschine**

- **Algorithmen in einer programmier - sprachen unab - hängigen Form darstellen können**

- **Entwurf von Lösungen für die Teilprobleme mit geeigneten Darstellungsformen**

Die Darstellung kann z. B. verbal oder durch Struktogramme erfolgen.

- **die Elemente einer proze duralen Programmier sprache kennenlernen und die Prinzipien des struktu rierten Programmierens auf einfache Algorithmen anwenden können**

- **Umsetzung des Entwurfs in die verwendete Programmier spra che unter Einführung der benötigten Sprach - elemente**

Hierbei sollte auch auf elementare Qualitäts - anforderungen geachtet werden, z. B.
- geeignete Bezeichnerwahl
- Strukturierung des Programmtextes
- Kommentare

Lernziele	**Lerninhalte**	**Hinweise**
	- elementare Anweisungen: Wertzuweisung, Eingabe, Ausgabe	
	- Kontrollstrukturen: Auswahl (ohne Alternative, mit Alternative, Mehrfachauswahl), Wiederholung (vorprüfende Schleife, nachprüfende Schleife, Zählschleife)	
	- Unterprogrammtechnik: parameterlose Prozeduren; verständiger Umgang mit parametrisierten Prozeduren und Funktionen	
		Ein vollständiges Prozedurkonzept ist erst für das folgende Unterrichtjahr vorgesehen.
	- elementare Datentypen mit den zugehörigen Operationen: ganze Zahlen (Integer), Dezimalzahlen (Real), Zeichen (Char), Zeichenketten (String), logische Werte (Boolean)	
		Hier kann auch die rechnerinterne Darstellung der elementaren Datentypen behandelt werden.

	- zusammengesetzte Daten typen mit den zugehörigen Zugriffs - operationen: Feld (Array), Verbund (Record); elementarer Umgang mit Dateien (File)	
		Eine Ver tiefung des Umgangs mit Dateien, ins besondere die Behand lung index-sequen tieller Dateien, ist für ein späteres Unterrichtsjahr vorgesehen.
	- weitere Datentypen der verwendeten Programmiersprache wie z. B. Menge (Set), benutzer definierte Typen	
- mit den auf dem Rechner verfügbaren Dienst programmen zur Text bearbei tung (Editor) und Übersetzung (Compiler) ein lauffähiges Programm erstellen können		
	- Implementierung des Entwurfs auf dem Rechner	
		Die zur Verfügung stehenden Betriebssystem- und Edi tor kommandos sind nur in dem Maße zu behan deln, wie sie zur Bearbeitung der Pro - gramme erforderlich sind.
- Programmtests durch führen und bewerten, sowie eventuell auftretende inhaltliche Fehler des Algorithmus und Notationsfehler beseitigen können		
	- Validierung des	

Programms durch Überprüfung der Testdaten

Lernziele	**Lerninhalte**	**Hinweise**
- das fertige Programm im Hinblick auf die ursprüng liche Aufgaben stellung unter Berück - sichtigung der tatsäch lichen Einsetzbar keit beurteilen können		
	- beispielhafter Einsatz des fertigen Programms und kritische Würdigung seiner Leistungsfähigkeit	
- die Notwendigkeit einer begleitenden Dokumen tation einsehen und die erforder lichen Dokumente kennen und erstellen können		
	- Begleitende Erstellung einer Dokumentation - Problemstellung - präzisierte Aufgabe - programmiersprachen unabhängiger Lösungs entwurf - Programmlisting - Testdokumentation - Bedienungsanleitung	
		In zeitlichem Zusammenhang ist bei jedem Arbeitsschritt eine schriftliche Fixierung der Ergebnisse vorzunehmen.

(20 Stunden)

Lernziele	**Lerninhalte**	**Hinweise**
- mit einem Textverarbei tungs - system im Unterricht anfallende Aufgaben erledigen können		
- typische Leistungs - merkmale von Textver arbeitungs - systemen kennen		
	- Einsatz von Standardwerk zeugen im Büro: Textver - arbeitung	
		Ein Text ver ar- beitungssystem kann zur Erstellung der Programm - dokumentationen benutzt werden. Durch Vergleich des Textverarbeitungs - systems mit dem bekannten Editor lassen sich Funktions merkmale herausarbeiten.
- weitere Software im Anwen dungs bereich Büro kennen		
	- weitere Ele mente der Bürokommuni kation, z. B. Tabellen kal- kulation, Geschäfts - graphik, integrierte Systeme	
		Die Behandlung von Datenbanken ist erst für spätere Unterrichtsjahre vorgesehen.
- moderne Formen der Informations - übermittlung kennen		
	- DFÜ, BTX, ISDN,...	

Sofern möglich, sollten hier über die eigene Schule hinaus Möglichkeiten der Datenkommunikation praktisch benutzt werden.

- **typische Konsequenzen kennen, die mit dem Einsatz der EDV in Betrieben verbunden sind**

  - **Auswirkungen auf die Arbeitswelt**

    - **Veränderung von Arbeitsabläufen**

    - **Veränderung von Qualifikations- und Ausbildungs anforde rungen**

Die Auswirkungen sollten in Abhängigkeit von den unter 1. gewählten Beispielen möglichst dort integrativ behandelt werden. Weitere Aspekte können je nach Anwendungsfall ergänzt werden.

    - **Soziale Folgen: Entlohnung, Überwachung des Mitarbeiters, Betriebsklima**

    - **Veränderung von Berufsbildern**

    - **Veränderung von Betriebsstrukturen**

    - **Umverteilung der Arbeitsplätze**

    - **Ergonomie am Computer arbeitsplatz**

    - **Prinzipien der Datensicherung**

Lernziele	**Lerninhalte**	**Hinweise**
- den Begriff des Daten schutzes und die für die Schule relevanten Regelungen kennen	**- Datenschutz im Anwendungs feld Schule**	***Dieser Aspekt des Daten schutzes kann bei Auswahl eines geeigneten Beispiels unter 1. integrativ behandelt werden.***
- Rechtmäßigkeit der Verarbeitung von Daten in der Schule und Schutz maß - nahmen beurteilen können		

(10 Stunden)

Lernziele	**Lerninhalte**	**Hinweise**
- die Hauptbestandteile eines Computers und ihr grundsätzliches Zusammenwirken kennen		
	- Komponenten einer Rechen anlage einschließlich ihrer Peripherie	
		Die Hardwarekomponenten eines Rechner systems sollten in Abhängigkeit von den unter 1. gewählten Beispielen möglichst dort integrativ behandelt werden.
- wesentliche Teile der zum Betrieb eines Computers notwendigen Software kennen		
	- Betriebssoftware (Betriebs systeme, Dienstprogramme)	
		Die Betriebssoftware eines Rechner systems sollte in Abhängigkeit von den unter 1. gewählten Beispielen möglichst dort integrativ behandelt werden.
- grundlegende Betriebsarten eines Computers kennen		
	- Betriebsarten (Ein- und Mehrbenutzerbetrieb, Echt zeit betrieb, Stapelbetrieb, Dialogbetrieb)	
- Merkmale verbreiteter Rechnerklassen		

kennen

- **Rechnerklassen (Personal-Computer, Netze, Mehrplatz - anlagen)**

(5 Stunden)

Lernziele	**Lerninhalte**	**Hinweise**
- wesentliche Stationen der Entwicklungsgeschichte des Computers kennen	**- Historische Entwicklung der Informatik und ihrer Anwendungen**	
- charakteristische Bau- und Leistungsmerkmale der Computer - generationen kennen		
- Entwicklung der Rechnertechnologie im 20. Jahrhundert		

Kurse: in-1.1/in-2.1
in-3.2/in-4.2

Thema: Grundlagen großer Programmsysteme (I und II)

1. Konstruktion eines Programmsystems zur Dateiverwaltung 30 Stunden
- **1.1. Analyse eines dokumentierten Systems**
- **1.2. Dateiverwaltung**
- **1.3. Gliederungselemente von umfangreichen Softwaresystemen**
  - **1.3.1. Prozeduren**
  - **1.3.2. Module**
  - **1.3.3. Qualitätsanforderungen**

2. Anwendung eines relationalen Datenbanksystems 10 Stunden

3. Auswirkungen des Einsatzes von Datenverarbeitungsanlagen 10 Stunden

4. Spezielle Algorithmen in typischen Anwendungssituationen 30 Stunden
- **4.1. Suchen und sortieren**
- **4.2. Dateien**
- **4.3. Rekursionen und Backtracking-Verfahren**
- **4.4. Dynamische Datenstrukturen**

Während im ersten Unterrichtsjahr nach der Analyse eines fertigen Softwareprodukts der Schwerpunkt auf der Konstruktion kleiner Programmteile innerhalb eines größeren Anwendungszusammenhanges lag, treten jetzt die strukturellen Aspekte bei der Konstruktion von Programmsystemen in den Vordergrund. Der Schwerpunkt liegt auf der Vermittlung des Prozedur- und Modulkonzeptes. Dabei ist die ausführliche Behandlung von Dateien als entscheidendes Instrument zur Informationsspeicherung vorgesehen.

Die notwendigen Kenntnisse sollen durch die Erstellung eines geeigneten Programms zur Dateiverwaltung (z. B. Büchereiverwaltung, Partnervermittlung, Lagerhaltung) vermittelt werden. In diesem Zusammenhang sollen auch die Auswirkungen des Programmeinsatzes aus unterschiedlicher Sicht (z. B. Chef, Angestellter, Verkäufer, Kunde) und die Gesetzgebung zum Datenschutz behandelt werden (Abschnitt 3).

Die Thematik der Informationsspeicherung wird mit der Einführung des Datenbankkonzepts fortgeführt. Hierzu kann derselbe Anwendungsfall wie bei der Dateiverwaltung untersucht werden. Im vorgegebenen Stundenrahmen kann nur ein elementarer Einblick in Datenbanksysteme vermittelt werden, eine Vertiefung ist im dritten Unterrichtsjahr (Projekt oder Vertiefungsgebiet) möglich.

Die Untersuchung spezieller Algorithmen in typischen Anwendungssituationen (Abschnitt 4) dient auch zur gezielten Vorbereitung des Projekts im dritten Unterrichtsjahr. Es ist möglich, eine Thematik jetzt nur kurz zu behandeln, weil sie im Projekt vertieft wird; oder sie kann besonders gründlich behandelt werden, weil sie dort vorausgesetzt wird. Auf jeden Fall sollten daher das Thema des Projekts und seine Schwerpunkte schon vor der Untersuchung der speziellen Algorithmen festgelegt werden.

(30 Stunden)

Hinweise: *Im ersten Unterrichtsjahr ist den Schülern die Datenverarbeitung bei der "Benutzung und Analyse eines dokumentierten Softwaresystems" und bei der "Konstruktion von Teilalgorithmen zu Anwendungsfällen" bereits in komplexen Situationen begegnet. Dabei sind auch vorbereitete Bausteine mehrfach verwendet worden.*
Aufbauend auf diesen Grundlagen sollen die Schüler ihre Kenntnisse nun in einem "Kurzprojekt" anwenden und vertiefen. Schwerpunkt wird nun die Konstruktion des Produkts , wobei überwiegend arbeitsteilig vorzugehen ist.

Für die Durchführung des "Kurzprojekts" sind insbesondere folgende Aspekte wichtig:

(1) Es soll sich um einen Anwendungsfall handeln, in dem Dateiverwaltung benötigt wird.
(2) Im Verlaufe des Projekts sollen Prozeduren und Module benutzt und entwickelt werden.

Lerninhalte, die im engeren Sinne mit der Projektarbeit verknüpft sind, bleiben dem dritten Unterrichtsjahr vorbehalten und sollen hier nur ansatzweise behandelt werden. Der Umfang des Projektthemas ist so zu wählen, daß eine Bearbeitung unter Beachtung der genannten Aspekte in ca. 30 Stunden möglich ist. Diese zeitliche Begrenzung wird u. a. durch folgende Maßnahmen erreicht.

(3) Vorgabe des Projektthemas und enge Projektführung durch den Lehrer
(4) Auswahl eines Themas, bei dem die notwendigen Vorkenntnisse schnell erworben werden können
(5) In dem Projekt sollen u. a. bereits im ersten Unterrichtsjahr benutzte und ggf. weitere vorbereitete Bausteine verwendet werden.
(6) Stärkere Hilfen und Vorgaben des Lehrers bei der Spezifikation, Entwurfsarbeit und Programmierung
(7) Beschränkung der Testfälle
(8) Lehrerhilfe bei der Systemintegration
(9) Beschränkung der Dokumentation auf das unbedingt notwendige Maß (Darstellung der gewählten Zerlegung des Problems und der grundlegenden Algorithmen, kurze Benutzeranleitung)
(10) Die Schüler sollen bei dem "Kurzprojekt" noch nicht mit den besonderen Organisationsformen (z. B. Formen des Management) der Projektarbeit vertraut gemacht werden.

Für die mit der Arbeit am "Kurzprojekt" verbundenen Aspekte (1), (2) sind die in 1.1. bis 1.3. zusammengestellten Lernziele und Lerninhalte verbindlich. Sie sind im Unterricht integrativ zu behandeln.

Lernziele	**Lerninhalte**	**Hinweise**
- das gewählte Problem analysieren und eine Anforderungsdefinitio n erstellen können		
	- Problemanalyse, Anforderungsdefinitio n	
		Für die im folgenden genannten Lernziele und Lerninhalte beachte man die oben formu - lierten Hinweise (3) bis (10) zur zeitlichen Beschränkung der Unterrichtseinheit.
- die Zerlegung des geplanten Systems in Module begründen können		
	- Modularisierung, abstrakter Datentyp	
- Module unter Beachtung der Schnittstellen - problematik programmieren und testen können		
	- Modulprogrammierung , Schnittstellen, Modultest	
- Module integrieren können		
	- Modulintegration	
- Dokumentationsteile erstellen können		
	- Dokumentation - Darstellung der gewählten Zerlegung des Problems und der grundlegenden Algorithmen - kurze	

	Benutzeranleitung	
		siehe obiger Hinweis (9)
- das System einsetzen können und notwendige Wartungsarbeiten erkennen		
	- Einsatz des Systems, Systemschwächen	

Lernziele	**Lerninhalte**	**Hinweise**
- Erzeugen, Ändern, Löschen von Dateien und Datensätzen als Programm realisieren		
	- Dateiverwaltung mit sequentiellen Dateien und mit Direktzugriff	
- Suchen, Sortieren von Datensätzen programmieren		
	- je ein einfaches Such- und Sortierverfahren	
		Der Sortiervorgang soll auf Feldern erfolgen.

Lernziele	**Lerninhalte**	**Hinweise**
- verschiedene Prozedurarten als syntaktische Varianten des Prozedurkonzepts kennen und entscheiden, welche Prozedurart für eine Problemlösung geeignet ist		
	- syntaktische Varianten des Prozedurbegriffs **- Prozedur im engeren Sinne (Aktionsprozedur)** **- Funktion**	
- vorgegebene Spezifikationen beurteilen und eigene erstellen		
	- Prozedurspezifikation **- Prozedurart** **- Namensgebung** **- Parametrisierung** **- Effektbeschreibung, auch in Sonder- und Fehlerfällen**	
- mit Prozeduren verbundene algorithmische Konzepte kennen und bei der Konstruktion von Prozeduren verständig anwenden		
	- Konstruktion **- lokaler Datenraum (Lebensdauer und Überdeckung von Datenobjekten)** **- Parametrisierung (Eingangs-, Ausgangs-, Durchgangsparameter)** **- Rekursivität**	

		Rekursivität sollte nur beispielhaft behandelt werden.
- geeignete Testfälle konstruieren		
	- Verifikation - Testfälle zur Erfüllung der spezifizierten Effekte und der internen Korrektheit	
		Bei der Durchführung von Tests ist auf den Stichprobencharak ter und die damit verbundene Problematik hinzuweisen.
- Testumgebungen anfertigen		
	- Testumgebungen (Testtreiber, erforderliche Dummyprozeduren)	
- Tests durchführen und dokumentieren		
	- Testdokumentation (einschließlich der Dokumentation nicht beseitigter Fehler)	

Lernziele	**Lerninhalte**	**Hinweise**
- abstrakte Datentypen problemgerecht konstruieren		
	- abstrakte Datentypen (ADT)	
- Modulbegriff kennen		
	- Module als - Zusammenfassung von Typdefinition und zugehörigen Operationen (abstrakter Datentyp) und als Sonderfälle - Datenmodul (z. B. Sammlung von Konstanten) - Funktionsmodul (Prozedursammlung)	
- Modulspezifikation erstellen		
	- Spezifikation von Modulen - Namensgebung - Import-, Exportschnittstelle - Effektbeschreibung, auch in Sonder- und Fehlerfällen	
- Datenobjekte und zugehörige Operationen zu einem Modul einkapseln		
	- Konstruktion von Modulen	
- Module testen		
	- Test von Modulen	

Lernziele	**Lerninhalte**	**Hinweise**
- **Qualitätsanforderungen an Spezifikation, Entwurf und Konstruktion von Prozeduren und Modulen kennen und berücksichtigen**	- **Qualitätsanforderungen aus Benutzersicht** - **Problemangemessenheit (Exaktheit)** - **Benutzerschnittstelle (Benutzerführung, Oberfläche, Begriffswahl)**	***Die Qualitätsanforderungen sollen schon bei der Einführung der entsprechenden Konzepte einbezogen werden. Qualitätsanforderungen aus Programmierersicht und die Bewertung von Software produkten werden im Projektsemester behandelt.***

(10 Stunden)

Hinweise: ***Die im folgenden formulierten Ziele und Inhalte des Unterrichts setzen die Verwendung einer relationalen Datenbank voraus. Im Vordergrund steht ein pragmatischer Umgang mit einer Datenbank. Es ist nicht beabsichtigt, den (mathematischen) Begriffsapparat des Relationenkalküls ausführlich zu behandeln; ebenso ist nicht intendiert, auf semantische Integritätsbedingungen einzugehen. Allerdings sollten im Unterricht nur konkrete Datenbanken verwendet werden, die die üblichen Normalformbedingungen erfüllen. Als Datenbank abfragesprache sollte sql bevorzugt werden, denn***

- ***sql ist die einzige Datenbanksprache, die standardisiert ist,***
- ***sql bietet als nicht-prozedurale Sprache die Möglichkeit, einen Einblick in andere Programmiersprachenkonzepte zu erhalten.***

Lernziele	**Lerninhalte**	**Hinweise**
- wissen, daß Datenbanken zur Modellierung von Realitätsausschnitten benutzt werden		
	- Elementare Begriffe im Umgang mit relationalen Datenbanken: Entity, Attribut, Schlüssel; Tabelle (Relation)	
- mit einfachen Tabellen elementare Datenbank - operationen durchführen		
	- Operationen Selektion, Projektion, Join	
- in einfachen Fällen Operationen auf Tabellen in der Datenbanksprache formulieren		
	- Umsetzen von Tabellenoperationen in Datenbankabfragen	
- Konzepte kennen, die den Datenbankzugriff benutzerabhängig gestalten, und sie in einfachen Fällen in der Datenbanksprache formulieren		

- **Vergabe von Rechten bei der Definition der Tabellen, Benutzersichten**

In der Datenbanksprache sql: grant und view

(10 Stunden)

Hinweise: ***Die folgenden Punkte sollen integrativ, wenn möglich unter Rückgriff auf das Kurzprojekt (1.) und die Anwendung des Datenbanksystems (2.) behandelt werden. Dabei können sich je nach Anwendungsfall unterschiedliche Schwerpunkte ergeben. Es sollten aber immer die Auswirkungen auf verschiedene Betroffene untersucht werden (Anwender, Nutzer, Kunden o. ä.).***

Lernziele	**Lerninhalte**	**Hinweise**
- Datenschutzproblematik in unterschiedlichen Anwendungsbereichen kennen und beurteilen	**- Datenschutzgesetze**	
- Auswirkungen und Gefahren bei Fehlern (z. B. Stromausfall, Softwarefehlern) kennen und einschätzen	**- Auswirkungen großer Softwaresysteme**	***Als Beispiel kann ein in Kaufhäusern genutztes Warenwirtschaftssystem gewählt werden, bei dem ein Einkaufen bei Stromausfall nicht möglich ist. Andere tagesaktuelle Formen des Fehlverhaltens von Computer- oder Software systemen sollte je nach Auftreten heran gezogen werden.***

(30 Stunden)

Hinweise: *Dieser Unterrichtsabschnitt bietet Gelegenheit, anhand von Beispielen aus typischen Anwendungsbereichen des Computers spezielle Themen aus der Algorithmik zu vertiefen. Dabei sollen einerseits bereits behandelte Themen vertiefend aufgearbeitet werden, andererseits können hier auch schon durch Schwerpunktsetzung gezielt Vorbereitungen für das Projekt getroffen werden.*
Bei der Analyse oder Konstruktion spezieller Algorithmen können zur zeitlichen Straffung und Konzentration auf das Wesentliche auch vorgefertigte Programme, Programmrahmen oder Programmteile verwendet werden.

Lernziele	**Lerninhalte**	**Hinweise**
- ein höheres Sortier - verfahren analysieren können und begründen können, warum es zum Erfolg führt		
	- ein höheres Sortierverfahren	
		Quicksort ist ein geeignetes Verfahren, das auch ein gutes Beispiel für die Rekursivität ist. Effizienzuntersuchungen sollten sich auf Plausibilitätsbetrachtungen beschränken.
- ein höheres Such - verfahren anlysieren können		
	- ein höheres Suchverfahren	
		Geeignete Suchverfahren sind z. B. binäres Suchen, Suchen in Bäumen, Suchen mit Hash-Algorithmen. Effizienzuntersuchungen sollten sich auf Plausibiltätsbetrachtungen beschränken.

Lernziele	**Lerninhalte**	**Hinweise**
- die speziellen Zugriffsmechanismen index-sequentieller Dateien kennen und anwenden können		
	- indexsequentielle Dateien	
		Es ist angemessen, in dem "Kurzprojekt" nur ent sprechende Modifikationen vorzunehmen.
- die Besonderheiten im Umgang mit Dateien zur elementaren Verarbeitung von Texten kennen und anwenden können		
	- Textdateien	
		Es sollten nur einige der elementaren Funktionali täten (z. B. Zeile einfügen, löschen, zentrieren) arbeitsteilig behandelt werden.

Hinweise: ***Rekursive Verfahren sollten auch im Zusammenhang mit der Behandlung dynamischer Datenstrukturen eingesetzt werden.***

Lernziele	**Lerninhalte**	**Hinweise**
- einen ausgewählten Backtracking-Algorithmus analysieren können		
	- Backtracking-Algorithmen	

Hinweise: ***Eine geeignete visuelle Darstellung von Zeigern, Listen und Bäumen ist ein wichtiger Beitrag zum besseren Verständnis dynamischer Datenstrukturen.***

Lernziele	**Lerninhalte**	**Hinweise**
- das Zeigerkonzept als Grundlage dynamischer Datenstrukturen kennen		
	- Zeiger	
- mit einfach verketteten Listen umgehen können		
	- einfach verkettete Listen	
		Bei dem Umgang mit Listen und Bäumen lassen sich auch rekursive Verfahren gut einsetzen.
- einen sortierten Baum erzeugen, ausgeben und durchsuchen können		
	- sortierte Bäume	

Kurs: **in-3.1**

Thema: **Software-Projekt**

Mögliche Projekttypen:

A. Neues Projektthema

B. Fortführung eines früheren Projektthemas

C. Wartung eines Softwareproduktes (Reengineering)

Die Alltags-Bedeutung der Informatik besteht in erster Linie in ihren Anwendungen. In den häufigsten Fällen ist der Einsatz von Computersystemen erst oberhalb einer gewissen Komplexität der Anwendungen wirtschaftlich sinnvoll. Für die Bewältigung dieser komplexen Probleme hat die Informatik eine Reihe von Methoden und Verfahren entwickelt, u. a.

- **Analysemethoden,**
- **Spezifikationsmethoden,**
- **Gliederung in Abstraktionsebenen,**
- **Modularisierung und**
- **Test- und Implementierungsverfahren.**

Diese dienen der Steigerung der Arbeitseffizienz, der Übersichtlichkeit und Korrektheit der Software. Gleichzeitig ermöglichen sie eine leichtere Identifizierung und rationellere Beseitigung von Fehlern bei der Implementierung und Wartung.

Im Projektsemester sollen die Schüler daher an die Methoden, aber auch an die Probleme der Bearbeitung von Softwareprojekten mit realitätsnaher Komplexität herangeführt werden. Die Komplexität äußert sich in dreifacher Hinsicht:

- **bei der inhaltlichen Analyse des Sachproblems,**
- **bei der Konstruktion des Softwaresystems und**
- **bei der Organisation des aus arbeitsökonomischen Gründen notwendigen arbeitsteiligen Vorgehens.**

Um der beabsichtigten Zielsetzung nahezukommen, steht der eigene konstruktive Umgang mit einem Softwaresystem für das Projektsemester im Vordergrund. Hierbei ist es erforderlich, daß das bearbeitete Projekt eine gewisse Mindestkomplexität aufweist, um die angesprochenen Ziele überhaupt erfahrbar machen zu können, andererseits muß der Umfang so gewählt werden, daß mehrere Phasen der Softwarekonstruktion in der zur Verfügung stehenden Zeit durchlaufen werden können.

Die Ziele der Projektarbeit können mit verschiedenen Projekttypen erreicht werden.

Projekttypen

- **Neues Projektthema**
  In diesem Fall werden alle Phasen vom Projektauftrag bis zum fertigen Produkt erstmalig durchlaufen. Gegebenenfalls müssen Schwerpunkte gesetzt werden, um den Umfang des Projekts einzuschränken; dies kann auch durch eine Begrenzung der Ausbaustufe erreicht werden. Man vergleiche hierzu die Ausführungen für das zweite Unterrichtsjahr.

- **Fortführung eines früheren Projektthemas**
  Hierbei kann es sich z.B. um die Vervollständigung bereits vorliegender Entwürfe oder den Ausbau existierender Software handeln.

- **Wartung eines fertigen Softwareprodukts (Reengineering)**
  Die Anpassung fertiger Software an veränderte Gegebenheiten und die Fehlerbeseitigung sind Aufgaben, die in der Datenverarbei tungspraxis häufig auftreten. Angesichts der sehr unterschiedlichen neuen Anforderungen sind verschiedene Projektabläufe bzw. Schwerpunktsetzungen denkbar.
  Auch die Softwarewartung läuft über (ausgewählte) Projektphasen des Software-Life-Cycle. Hierbei sind jeweils die anschließend formulierten speziellen Ziele der Projektarbeit zu beachten. Diese zeigen auch an, ob das gewählte Wartungsprojekt einen angemessenen Umfang hat.

Unabhängig vom Projekttyp ist bei der Auswahl und Durchführung des Projekts auf die Einhaltung folgender Lernziele und Kriterien zu achten:

Lernziele (unabhängig vom Projekttyp)

Die Schüler sollen

- **einen Überblick geben können**
  - über die Funktionalität und Bedienung des Gesamtsystems (Anwendersicht)
  - über den internen Aufbau des Gesamtsystems (Entwicklersicht)

- **Fragestellungen und Entscheidungen bestimmten Projektphasen zuordnen können**

- **erkennen, daß sich zu frühe Festlegungen innerhalb des Projektablaufs möglicherweise nachteilig auf die folgenden Arbeiten und auf die Qualität des Endprodukts auswirken können**

- **die durchlaufenen Projektphasen angemessen dokumentieren können**

- **einen Überblick über den gesamten Software-Life-Cycle erhalten.**

Kriterien zur Wahl von Art und Umfang des Projekts

- **Das behandelte Problem muß mindestens so komplex sein, daß die durch den Software-Life-Cycle implizierten Methoden als notwendig erlebt werden; andererseits ist der Projektumfang so einzuschränken, daß mindestens ein Prototyp erstellt werden kann.**

- **Die Projektarbeit darf sich nicht nur auf eine Phase der Softwareentwicklung erstrecken, sie darf sich insbesondere nicht auf reine Programmiertätigkeiten beschränken.**

- **In mindestens einer der Projektphasen muß die Anwendersicht behandelt werden.**

Phasen des Software-Life-Cycle

Lernziele (phasenbezogen) der Tätigkeiten Phasen)	**Aktivitäten**	**Dokumente** (schriftliche Festlegung Ergebnisse der in den einzelnen
- die Umstände, unter denen das zu erstellende System eingesetzt werden soll, analysieren können - Anforderungen aus Anwendersicht formulieren und strukturieren können	- Untersuchung - des Einsatzfeldes - des Istzustandes - der Rahmenbedingungen Ergebnis ist die - Aufgabenstellung aus Anwendersicht mit Anforderungen an - Leistungsumfang - Arbeitsweise des zu entwickelnden Systems - Qualität - Datenschutz	Anforderungsdefinition mit Angabe der vom Anwender gewünschten - Leistungen - Organisation der Benutzung - Art und Umfang der zu erstellenden Dokumente
- Anforderungen der Anwender auf Realisierbarkeit überprüfen können - geeignete Datentypen finden können		

- **geeignete Zerlegung und programmiersprachen- unabhängige Formulie rung von Teilfunktionen des Systems aus Entwicklersicht vornehmen können**

- **Anforderungen aus Anwender- und Entwicklersicht unterscheiden können**

- **Planung des Systems aus Entwicklersicht mit**
  - Zerlegung in Teilaufgaben
  - Realisierbarkeitsprüfung
  - Festlegung der Datenstruktur
  - Beschreibung der Benutzerschnittstellen
  - Festlegung der Test- strategien und Testfälle
  - Zeitplan für den Anwender (wann ist welcher Teil fertig, Endabnahme - termin, usw.)

Funktionelle Spezifikation (Pflichtenheft) mit Beschreibung des verbindlich zugesagten Leistungsumfangs
- vorgesehene Funktionen
- Benutzerschnittstelle
- Datenein/ausgaben
- Testdaten
- Reaktion auf Fehleingaben
- Abnahmekriterien

Lernziele (phasenbezogen) der Tätigkeiten Phasen)	**Aktivitäten**	**Dokumente** (schriftliche Festlegung Ergebnisse der in den einzelnen
- Modulare Techniken zur Arbeitserleichterung anwenden können - Absprachen zum Ablauf des Projekts treffen und einhalten können - insbesondere die Schnitt - stellenfestlegung als notwendige Voraussetzung für arbeitsteiliges Vorgehen erkennen	- Zerlegung des Systems in Module und Festlegung von - Import- und Exportschnittstellen - Ein- und Ausgabeform, Ein -und Ausgabeformat - zu verwendender Software, Tools und Peripherie - Festlegung von Zuständigkeiten bei den Entwicklern - Zeitplan für die Entwickler (Aufeinanderfolge einzelner Module u. ä.)	Entwurfsspezifikation mit Darstellung der Modulhierarchie und Modulspezifikationen; Vereinbarung von Dokumentationsrichtlinien und Programmierkonventionen

- mit vorgegebenen Schnittstellen und Datentypen einzelne Module funktionsgerecht programmieren und testen können		
	- Programmierung und Test einzelner Module	
		Programmlisten und Testprotokolle
- Module sinnvoll schrittweise zum System zusammenstellen können		
- geeignete Testfälle für das System finden und anwenden können		
	- Zusammenstellung der Einzelmodule zum Gesamtsystem und Test des Systems	
		Testprotokolle Benutzerhandbuch
- in sinnvollen Zusammenhängen die Tauglichkeit des Systems überprüfen und erforderliche Anpassungen erkennen können		
	- Einsatz des Systems unter "Alltagsbedingungen" - Fehlerbeseitigungen	

- Anpassungen an veränderte Problemstellungen

Wartungshandbuch

Gelegentlich wird man ein Projekt aufsetzen können auf Teillösungen oder Software, die von anderen Kursen oder im früheren Unterricht erstellt wurden. Nach dem Verstehen dieser Ansätze mündet das Projekt je nach existierenden Vorlagen in eine der Phasen des Software-Life-Cycle ein.
Es gelten die entsprechenden phasenbezogenen Lernziele aus Abschnitt A.

nweise: *Mögliche Wartungsaufgaben und ein Beispiel für einen Wartungsfall sind weiter unten aufgeführt.*

prechenden phasenbezogenen Lernziele aus Abschnitt A.

(1) Die Wartung eines Softwareprodukts beginnt in der Regel mit der Analyse der vorliegenden Software :
- **Programmbenutzung**
- **Funktionsumfang erkennen**
- **Dokumentation (Handbücher) inspizieren**
- **Datenstruktur verstehen**
- **Schnittstellen erkennen**
- **änderungsrelevante Programmteile ausfindig machen.**

(2) Nun erfolgt die Veränderung des Systems gemäß den neuen Anforderungen. Im folgenden werden einige mögliche Wartungsaufgaben genannt. Diese treten in der Regel kombiniert miteinander auf und führen zu einer neuen Programmversion.

(3) Ein Wartungsdurchgang endet mit der Implementierung des überarbeiteten Systems in einer neuen Form oder einer neuen Umgebung.

<u>Mögliche Wartungsaufgaben</u>

a) **Beseitigung von Fehlern; dabei können die Fehler ganz unterschiedliche Ursachen haben, z. B.**
- **fehlerhafte Analyse**
- **unvollständige oder widersprüchliche funktionale Spezifikation**
- **ungeeignete oder fehlerhafte Modularisierung**
- **Programmierfehler**

b) **Erweiterung der Funktionalität**
- **Hinzufügen neuer Module**
- **Ersetzen von Modulen durch andere mit vergrößertem (verkleinertem) Funktionsumfang**
- **Einfügen graphischer Elemente**

c) **Verbesserung der Benutzerfreundlichkeit**
- **Veränderung der Benutzeroberfläche**
- **integrierte Hilfen**
- **Einfügen graphischer Elemente**
- **Umstellung von Tastenbedienung auf Mausbedienung**

d) **Nachprogrammierung eines bekannten Systems mit gesteigerten Anforderungen**
- **an die interne Programmstruktur (Modularisierung)**
- **unter Verwendung von Standardmodulen**

e) **Verbesserung und Ergänzung der Dokumentation**

f) **Standardisierung**
- **Vereinheitlichung der Benutzeroberfläche**
- **Vereinheitlichung bei der Strukturierung**
- **Vereinheitlichung der Namensgebung**

g) **Erhöhung der Sicherheit und Zuverlässigkeit**

h) **Steigerung der Effizienz**

i) Verbesserung der Portabilität
- Umstellung auf Betrieb im Netz
- Beseitigung von Abweichungen vom Standardsprachumfang

Beispiel eines Wartungsfalls

Von einem älteren Informatikkurs ist ein Softwareprodukt für den Einsatz im Schulfach Erdkunde konstruiert worden. Das System wurde inzwischen von einem Fachlehrer im Unterricht benutzt. Dabei sind einige Schwächen in der Benutzerführung, Dokumentationsfehler und Programmierfehler festgestellt und notiert worden.
Der Fachlehrer wendet sich mit diesen Unterlagen an den Informatiklehrer und bittet um Nachbesserung. Der Informatikkurs soll diese Aufgabe im Rahmen von Projektarbeit bearbeiten.

In diesem Wartungsfall werden vermutlich folgende oben genannte Punkte relevant: a), c), e), g). Der Informatikkurs muß bei der Realisierung der Wünsche alle Phasen der Projektarbeit von der Systemanalyse bis zur neuen einsatzfähigen Version abarbeiten, jedoch in sehr eingeschränktem Umfang, da im wesentlichem auf das vorliegende System aufzubauen ist.

Kurs: **in-4.1**

Thema: **Vertiefungsgebiet**

Mögliche Alternativen:

A. Theoretische Informatik: Automaten

B. Nicht-prozedurale Sprachkonzepte

C. Datenbanken

D. Computergrafik

Das letzte Halbjahr des Unterrichts dient dazu, einen speziellen Themenbereich zu vertiefen. Von den vier vorgeschlagenen Themen ist eines auszuwählen.

Es ist aber auch möglich, ein anderes Thema zu wählen; in diesem Fall muß rechtzeitig vor der Einrichtung ein entsprechender Antrag bei der Senatsverwaltung für Schule, Berufsbildung und Sport gestellt werden. Dieser Antrag muß die inhaltliche Konzeption darstellen und die Lerninhalte analog zu den beschriebenen Alternativen des Kurses in-4.1 enthalten.

Die Beschreibung der Kurse beschränkt sich auf die wesentlichen Inhalte. Die Aufteilung in konkrete Unterrichtseinheiten, ihr zeitlicher Umfang und die Schwerpunktsetzung sind vom Lehrer vorzunehmen.

Bei der Planung ist zu berücksichtigen, daß das Kurshalbjahr durch die Abiturprüfung zeitlich eingeschränkt ist. Sofern im Fach Informatik eine mündliche Abiturprüfung durchgeführt wird, muß diesem Kurs eine der beiden Prüfungsaufgaben entstammen.

1) Endliche Automaten mit Ausgabe (Transduktoren)
- **Beispiele: Fahrscheinautomat, Getränkeautomat, ...**
- **Simulation von Automaten, z. B. in Pascal**
- **Beschreibung durch Zustandsgraphen und Zustandstabellen**
- **formale Beschreibung (endliche Eingabemenge, endliche Zustandsmenge, Übergangsfunktion, Ausgabefunktion)**

2) Endliche Automaten ohne Ausgabe (Akzeptoren)
- **Beispiele: Codeschloß, Parser, ...**
- **Simulation von Automaten, z. B. in Pascal**
- **formale Sprachen (Grammatik, Syntaxdiagramme, Bezug zur Programmiersprache Pascal)**
- **Kellerautomat als Parser**

3) Turingmaschine
- **Aufbau und Arbeitsweise**
- **Beispiele: Kopieren einer Zeichenkette, Umkehren, ...**
- **ggf. Verwendung eines Modells (CUM-Programm)**

4) Vertiefung des Algorithmusbegriffs, Berechenbarkeit

Die in der Schule am häufigsten verwendeten Programmiersprachen Pascal, Modula-2 oder Elan sind imperative Sprachen. Daher sind die algorithmischen Teile des Informatikrahmenplans so angelegt, daß die Frage, mit welcher Anweisungsfolge ein Problem zu lösen ist, im Vordergrund steht. Demgegenüber bestehen deklarative Entwurfsmethoden darin, daß die Objekte, auf denen ein Problem gelöst werden soll, ihre Eigenschaften und das Problem selber so genau wie möglich beschrieben werden. Diese Methoden lassen sich z. B. in Prolog realisieren.

1) Beschreibung von Objekten und ihren Eigenschaften
- **einfache Objekte, Verknüpfung von Objekten**
- **Abfragen von Eigenschaften und Existenzen**
- **Listen**

2) Problemlösungsstrategien
- **Rückverfolgung (Backtracking)**
- **Vorwärtsverkettung**

3) Anwendung in einem Expertensystem
- **Aufbau eines Expertensystems**
- **Wissensbasis und Inferenzmaschine**
- **Benutzerschnittstelle**
- **Problematik des Einsatzes von Expertensystemen**

stelle der hier beschriebenen deklarativen Entwurfsmethode kann auch ein funktionales Sprachkonzept behandelt werden.

Während im zweiten Unterrichtsjahr der pragmatische Umgang mit einer Datenbank im Vordergrund steht, werden hier die zugrundeliegenden theoretischen Konzepte vertieft behandelt. Die Datenmodellierung nach dem Entity-Relationship-Modell ist unabhängig von dem verwendeten Datenbankkonzept. Die weitere Unterrichtsreihe konzentriert sich aber bewußt auf relationale Datenbanken.

Datenmodellierung mit dem Entity-Relationship-Modell
- **vertiefte Behandlung der Begriffe Entität (entity), Attribut, Beziehung (relationship) und ihre graphische Darstellung**
- **Klassifizierung der Beziehungsarten (eineindeutig (1:1), eindeutig (1:n), komplex (n:m), zwingend oder freigestellt, ...)**

Umsetzung des Entity-Relationship-Modells in das relationale Datenmodell
- **vertiefte Behandlung des relationalen Datenmodells (mathematischer Relationenbegriff: Kartesisches Produkt, Teilmenge, n-Tupel, Tabelle, Spalte, Zeile)**
- **Umsetzung von Entitäten und Beziehungen in Relationen (Tabellen), 1. Normalform, Schlüssel**
- **Operationen zum Einfügen, Ändern, Löschen und Wiedergewinnen von Daten als Manipulation von Relationen (Vereinigung/Durchschnitt, Teilmenge, Projektion, Verbund, ...)**
- **Normalisierung (Beseitigung von Mehrdeutigkeiten und Inkonsistenzen), funktionale Abhängigkeit, 2. und 3. Normalform, Boyce-Codd-Normalform**

Implementierung des Datenmodells mit der Standard-Datenbanksprache SQL (Structured Query Language)
- **Datendefinitionsbefehle und Datenmanipulationsbefehle von SQL**
- **select-Befehl für komplexere Abfragen**
- **Vergleich verschiedener Benutzerschnittstellen (SQL-Interpreter, Masken für formularorientierte Abfragen, Menüs zum Anstoßen von Berichten, ggf. Schnittstelle zu prozeduraler Programmiersprache oder 4GL-Erweiterung von SQL)**

Datenschutz und Datensicherheit in Datenbanksystemen
- **Steuerung von Zugriffsrechten und Benutzersichten**
- **Sicherung der Integrität der Datenbank "im Betrieb", z. B. durch Indizierung der Schlüsselfelder**
- **Datenbank-Transaktionen, Wiederherstellung im Fehlerfall (Sicherheitskopien, Protokoll-Datei)**

Grundlagen der Computergraphik
- **Beziehungen zur analytischen Geometrie und zur Abbildungsgeometrie**
- **rechnerinterne Darstellung graphischer Objekte (z. B. Pixel- oder Vektordarstellung)**
- **Darstellungsprobleme auf dem Bildschirm (z. B. hidden line)**

2) CAD-Programme
- **Benutzung eines CAD-Programms**
  - **Einsatzbereiche**
  - **Funktionsumfang**
  - **Konstruktionsaufgaben (z. B. Entwurf von Schaltungen, Wohnungsgrundrissen, Tapetenmustern,...)**
- **Simulation grundlegender Elemente eines CAD-Systems durch kleine Programme (z. B. Koordinatensystem, Spiegelung an einer Geraden, Verschiebung von Figuren, ...)**

) Weitere Anwendungen von Computergraphik
- **Beispiele (zur Auswahl)**
  - **Geschäftsgraphik**
  - **Funktionenplotter**
  - **Malprogramme**
  - **Computergraphik in der Medizin**
  - **Fraktale**
  - **Animation**
- **Programmierung einiger Teilaufgaben**

.Anfang Verzeichnis V.

Inhalt

.Ende Verzeichnis V.