Ein möglicher Gang durch das Wahlfach / Wahlpflichtfach Informatik

Stand: 29.04.2009

Beschreibung der Unterrichtsreihe	Std.
Information und ihre Darstellung	16
Einführung in das algorithmische Problemlösen	22
Digitaltechnik	17
Einführung in die imperative Programmierung	18
Datenbanken	23

Information und ihre Darstellung

Stand: 29.04.2009

Zeitrichtwert: 16 Unterrichtsstunden

Inhaltliche Schwerpunkte	Bemerkungen
	Vorbemerkung Dieser Vorschlag orientiert sich an den Ausführungen auf den Webseiten inf-schule.
Einstieg (1) • Analyse eines XHTML-Dokuments • Strukturierte Darstellung von Information	Einstieg in den Unterricht durch Betrachtung einer XHTML-Seite im Internet. Diese (möglichst online verfügbare) Seite sollte bereits valide, gut strukturiert und einfach gestaltet sein. Ebenso sollte die Darstellung in einer separaten CSS-Datei stattfinden. Durch Ansicht des Quelltextes werden erste Beobachtungen getätigt. Hierbei ist herauszuarbeiten, in welchem Bereich der eigentliche Inhalt steht.
Erstellen eines einfachen XHTML-Dokuments (3) • Strukturierung • Validierung	Die Schülerinnen und Schüler kopieren den verwendeten Quelltext in einen einfachen Editor und entfernen alle Inhalts-Elemente, so dass eine Seitenvorlage entsteht. Diese nutzen sie, um ein erstes "Hello World"-Dokument zu erstellen. Die Schülerinnen und Schüler lernen erste Strukturierungselemente (Absätze, überschriften, Hervorhebungen) kennen und erweitern ihr Dokument. Die Strukturierungselement sind im Laufe der Unterrichtseinheit geeignet zu ergänzen. Insgesamt sollten aber nur wenige ausgewählte Elemente verwendet werden. Durch Validierung des Dokuments wird zum

	einen die Notwendigkeit syntaktische Korrektheit andererseits das Lesen und Interpretieren von Fehlermeldungen eines Systems kennen gelernt. Editoren: • Notepad++ • phase5 Links: • Original Online-Validierer des W3C (Fehlermeldung in englischer Sprache) • Online-Validierer von Validome (Fehlermeldung in deutscher Sprache) • Validome mit SelfHTML • Den W3C-Validator lokal installieren • Tipp: Firefox mit Add-on "HTML-Validator" • SelfHTML - eine Referenz zu (X) HTML und CSS
Barrierefreiheit (2) • Die Notwendigkeit der Validierung von Dokumenten - auch im Hinblick auf Barrierefreiheit - diskutieren.	Zitat aus der BITV: "Markup-Sprachen [...] sind entsprechend ihrer Spezifikationen und formalen Definitionen zu verwenden." BITV (Barrierefreie Informationstechnik-Verordnung) insbesondere Ausschnitte aus der Anlage der BITV lassen sich gut im Unterricht verwenden.

Trennung von Struktur und Darstellung (3) • Vorteile der Trennung von Struktur und Inhalt herausarbeiten	Zitat aus der BITV: "Es sind Stylesheets zu verwenden, um die Text- und Bildgestaltung sowie die Präsentation von mittels Markup-Sprachen geschaffener Dokumente zu beeinflussen." Mit Hilfe des title-Elements können in einem XHTML-Dokument mehrere CSS-Dateien eingebunden und je nach Bedarf ausgewählt werden. Dies ist z.B. mit den Browsern Firefox oder Opera möglich.
Codierung von Zeichen (1)	Eine XHTML-Datei mit Umlauten und anderen Sonderzeichen wird in einem Browser mit unterschiedlichen Zeichenkodierungs-Einstellungen betrachtet (z.B. westeuropäisch und kyrillisch). Dabei werden die selben Daten verschieden dargestellt. Damit wird deutlich, dass zwischen Daten und Information (im Sinne von interpretierten Daten) unterschieden werden muss.
Einbinden von Bildern (2) • Binärdarstellung von Bildern	Die Lehrkraft fotografiert alle Schülerinnen und Schüler und stellt diese Bilder zur Verfügung. Die digitalen Bilder sind zu groß und müssen noch bearbeitet werden. Es ist über mögliche Grafikformate (jpeg, gif, png, ...) und ihr Verwendungszweck zu sprechen. Darüber hinaus müssen die Bilder bearbeitet werden, um Größe und Komprimierungsgrad anzupassen. In diesem Zusammenhang wird auch die Binärdarstellung von Bildern thematisiert. Zusatz-Information: Entsprechende Verfahren sind auf Ton- und Filmdaten anwendbar. Links: • Irfanview ein kleiner aber mächtiger

	Bildbetrachter, der die wesentlichen Dinge kann, die für den Unterricht benötigt werden. • GIMP ein mächtiges Bildbearbeitungsprogramm für die Bearbeitung von Bilder mit Hilfe besonderer Effekte
Erstellung einer XHTML-Seite, auf der sich die Schülerinnen und Schüler jeweils vorstellen (4) • Persönlichkeitsrechte, Urheberrecht	Die Menge der (mindestens) zu verwendeten Elemente sollte vorgegeben werden. Die Veröffentlichung des eigenen Bildes - vielleicht auf der Schulhomepage - wirft Fragen der Rechte und Pflichten im Zusammenhang mit Bildern und sonstigen Werken auf (Recht am eigenen Bild, Urheberrecht) - auch im Hinblick auf die Veröffentlichung von Kursfotos oder das beliebte Kopieren von fremden Inhalten in die eigene Webseite. Links: • Datenschutz bei wikipedia • remus - Rechtsfragen von Multimedia und Internet in Schule und Hochschule • Der Schutz der Persönlichkeit im Internet • Datenschutz in Rheinland-Pfalz • Bundesdatenschutzgesetz • Bundesamt für Sicherheit in der Informationstechnik • Klicksafe.de

Ausblick	Im weiteren Unterricht sollen die Schülerinnen und Schüler auf diese Weise ihre Dokumente erstellen (Zusammenfassungen, Handouts für Referate etc.)

Algorithmisches Problemlösen

Stand: 29.04.2009

Zeitrichtwert: 22 Unterrichtsstunden

Inhaltliche Schwerpunkte	Bemerkungen
	Vorbemerkung Der Lehrplan für das Wahl(pflicht)fach setzt folgende Rahmenbedingungen: In allen Bereichen der Informatik spielen Algorithmen eine zentrale Rolle. Jede automatisierte Verarbeitung von Daten mithilfe des Computers erfolgt auf der Grundlage präziser Verarbeitungsvorschriften bzw. Algorithmen. Ziel des Informatikunterrichts ist es, ein Grundverständnis für diese Zusammenhänge zu entwickeln. Eine Auseinandersetzung mit Algorithmen darf nicht als „Trockenkurs“ gestaltet werden. Der Unterricht lebt davon, interessante und altersgemäße Probleme bis hin zu einer lauffähigen Lösung zu bearbeiten. ... Dieser Vorschlag nutzt einen eher spielerischen Zugung zum algorithmischen Problemlösen mit Hilfe der Programmiersprache und -umgebung Scratch und orientiert sich an den Ausführungen auf den Webseiten inf-schule. Scratch wird von einem Forschungsteam der Lifelong Kindergarten Group am MIT Media Lab mit den Ziel entwickelt, Kindern und Jugendlichen einen einfachen und motivierenden Zugang zu Konzepten der Programmierung zu verschaffen. Scratch erlaubt es, interaktiv und ohne Erlernen programmiersprachlicher Details vielfältige algorithmische Problemlösungen zu

	entwickeln. Im Anschluss an das spielerische Erlernen von Konzepten mit Scratch wird der allen Problemlösungen zu Grunde liegende Algorithmusbegriff erarbeitet.
Erkundung der Scratch-Welt (2) • Objekte als Akteure • Anweisungen / Programme zur Steuerung der Akteure • Ereignisse als Auslöser von Programmen	Die einfache Bedienbarkeit von Scratch ermöglicht einen experimentellen und spielerischen Zugang. Mit wenigen Hinweisen können Schülerinnen und Schüler schnell erste Programme entwickeln, mit deren Hilfe vorgegebene (oder selbst erstellte) Figuren animiert werden können, auf der sogenannten Bühne bestimmte Aktionen durchzuführen. Scratch unterstützt das Erstellen von Programmen, indem es vorgefertigte Programmierkacheln zur Verfügung stellt, die wie bei einem Puzzle nur richtig aneinandergefügt werden müssen. Nach einem ersten Austesten der Programmierumgebung wird die Vorgehensweise reflektiert und mit neuen Begriffen beschrieben. Figuren - das sind die Bausteine einer Scratch-Welt - werden als Objekte mit spezifischen Eigenschaften dargestellt. Mit Hilfe von Anweisungen und Programmen lassen sich die Aktionen der Figuren festlegen. Ausgelöst werden diese Aktionen durch Ereignisse.

Kontrollstruktur Fallunterscheidung (2) • Aufbau einer (einseitigen / zweiseitigen) Fallunterscheidung • Ausführung einer Fallunterscheidung • Darstellung einer Fallunterscheidung	Zunächst wird ein Problem gelöst, bei dem Aktionen nur unter einer bestimmten Bedingung durchgeführt werden. Beispiel: Eine Figur soll sich nur dann weiterbewegen, wenn sie sich nicht am Rand der Bühne befindet. Dadurch, dass Scratch passende Programmierkacheln zur Lösung des Problems vorsieht, können Schülerinnen und Schülern selbstständig (experimentierend) beim Problemlösen vorgehen. In einer nachfolgenden Besprechung wird die Kontrollstruktur Fallunterscheidung eingeführt. Der Aufbau einer (einseitigen / zweiseitigen) Fallunterscheidung ist direkt aus den von Scratch zur Verfügung gestellten Programmierkacheln ersichtlich. Dieser Aufbau wird programmiersprachenneutral mit Hilfe entsprechender Struktogramme beschrieben. Die Ausführung von Fallunterscheidungen wird (einmalig) mit einem Flussdiagramm verdeutlicht. Flussdiagramme dienen hier also nur dazu, die Semantik der Fallunterscheidungsanweisung zu beschreiben.
Kontrollstruktur Wiederholung (4) • Wiederholung mit fester Anzahl • Wiederholung mit Abbruchbedingung • Aufbau und Ausführung • Darstellung mit Struktogrammen • Endlosschleife	Zunächst wird ein konkretes Problem gelöst, bei dem Aktionen wiederholt durchgeführt werden müssen. Beispiel: Eine Figur bewegt sich solange weiter, bis sie den Rand der Bühne erreicht. Auch hier sollen die Schülerinnen und Schüler selbstständig die passende Programmierkachel zur Lösung des Problems aufsuchen und erkunden. In einer nachfolgenden Besprechung wird die Kontrollstruktur Wiederholung dann intensiv durchdrungen. Zunächst wird der Aufbau einer Wiederholung mit fester Anzahl sowie einer Wiederholung mit Abbruchbedingung geklärt

	und jeweils mit einem Struktogramm beschrieben. Der von Scratch zur Verfügung gestellte Anweisungstyp `wiederhole bis ...` wird intensiv besprochen. Mit Hilfe eines Flussdiagramms wird die Abarbeitung dieses Schleifentyps transparent gemacht werden. Dies ist insofern von Bedeutung, da es in anderen Programmierumgebungen ähnliche Anweisungskonstrukte gibt, die jedoch einer etwas anderen Abarbeitungslogik folgen. Die Verwendung der Kontrollstruktur Wiederholung wird angemessen geübt. Dabei wird auf auch das Phänomen der Endlosschleife angesprochen.
Komplexere Ablaufstrukturen (3) • Schachtelung von Kontrollstrukturen • Zusammengesetzte Bedingungen	Die Ablaufmodellierung mit Kontrollstrukturen wird vertiefend geübt. Es werden Probleme aus verschiedenen Kontexten bearbeitet, zu deren Lösung die verschiedenen Kontrollstrukturen in überschaubarer Weise verschachtelt werden müssen. Einen gesonderten Schwerpunkt bilden zusammengesetzte Bedingungen. Die logischen Operatoren `nicht`, `und`, `oder` werden mitsamt ihrer Wertetabellen eingeführt und zum Aufbau komplexerer Bedingungen genutzt.

Variablenkonzept (4) • Variablen • Zuweisungen • Trace-Tabelle	Über Probleme, zu deren Lösung man sich etwas merken muss oder man zählen können muss, gelangt man zum Variablenkonzept. Scratch macht das Umgehen mit Variablen und Zuweisungen recht transparent, indem es den Wert einer Variablen (auf Wunsch) auf der Bühne anzeigt. Bei der Besprechung und Erkärung des Variablenkonzepts werden Variablen als Namen für Datenobjekte (wie Zahlen, ...) eingeführt. Zuweisungen werden als spezielle Anweisungen eingeführt, mit deren Hilfe man Variablen neue Datenobjekte zuweisen kann. Diese Sichtweise erlaubt es, in einer späteren Phase auch Situationen zu erfassen, bei denen Variablen veränderbare Datenobjekte (wie Listen) referenzieren, deren interne Struktur mit Zuweisungen verändert werden können. In einer Übungs- und Vertiefungsphase wird der sichere Umgang mit Variablen angestrebt. Mit Hilfe von Trace-Tabellen werden die Veränderungen von Variablenwerten bei (sinnvollen) Zuweisungssequenzen transparent gemacht. Typische Zuweisungsmuster (wie sie beim Vertauschen von Werten oder beim Zählen benutzt werden) werden herausgestellt. Auch komplexere Terme innerhalb von Zuweisungen werden beim Problemlösen benutzt. Zudem wird der Umgang mit Variablen und mit Kontrollstrukturen verknüpft.

	Bemerkung Mit Scratch lassen sich auch weitere Fachkonzepte wie das EVA-Prinzip oder das Prozedurkonzept erarbeiten. Es hängt von der Lerngruppe ab, inwieweit die Motivation anhält, Scratch weiterhin als Problemlösewerkzeug zu erkunden. In der hier skizzierten Unterrichtsreihe soll jetzt ein Übergang zu mehr allgemeinen Fragestellung folgen, bei denen der Algorithmusbegriff im Zentrum der Betrachtungen steht. Für weiterführende Informationen siehe inf-schule.
Algorithmen (3) • Algorithmusbegriff • Bedeutung von Algorithmen	Der Übergang vom Problemlösen mit Scratch zum etwas allgemeineren algorithmischen Problemlösen knüpft an eine Scratch-Lösung zu einem vorgegebenen Problem an. Die Idee des Übergangs besteht darin, die Scratch-Lösung auf ihren algorithmischen Kern zu reduzieren. Beispiel: Zunächst wird ein Scratch-Programm zum Spiel Zahlenraten entwickelt, bei dem binäres Raten als Strategie benutzt wird. Der Ratealgorithmus wird jetzt extrahiert und programmiersprachenunabhängig formuliert. Beispiel: Zunächst werden Scratch-Programme zur Simulation von Zufallsexperimenten entwickelt. Anschließend wird der Simulationsalgorithmus programmiersprachenunabhängig beschrieben. Ausgehend vom Beispiel-Algorithmus werden die charakteristischen Eigenschaften dieser Verfahrensbeschreibung herausgearbeitet: Ein Algorithmus ist eine Verarbeitungsvorschrift, die so präzise

	formuliert ist, dass sie (zumindest im Prinzip) auch von einer Maschine abgearbeitet werden kann. Zudem werden Kriterien wie Eindeutigkeit, Ausführbarkeit, Allgemeinheit und Endlichkeit besprochen, die Algorithmen (in der Regel) erfüllen. Hierbei werden dann auch Verfahrensbeschreibungen aus dem Alltag wie Kochrezepte oder Bauanleitungen eingeordnet, die algorithmische Züge aufweisen. Anhand von Beispielen wird die Bedeutung von Algorithmen für uns Menschen und unsere Gesellschaft aufgezeigt. Beispiel: Automatisierung von Vorgängen mit Hilfe von Robotern und deren gesellschaftliche Folgen Beispiel: Simulation komplexer Vorgängen als Grundlage für Prognosen wie im Fall der globalen Klimaveränderungen
Aufbau und Darstellung von Algorithmen (4) • Bausteine von Algorithmen • Darstellung von Algorithmen	Anhand klassischer Algorithmen wird das Verständis für das Fachkonzept Algorithmus vertieft. Als Beispiele eignen sich Algorithmen aus der Mathematik wie der Algorithmus zur ägyptischen Multiplikation, der euklidischer Algorithmus zur Bestimmung des ggT, ein Algorithmus zur Primfaktorzerlegung, ... Es können aber auch Mathematik-fernere Algorithmen betrachtet werden wie Suchalgorithmen. Die Auswahl richtet sich auch nach den Implementierungsmöglichkeiten. In der hier skizzierten Unterrichtsreihe werden vorerst nur solche Algorithmen betrachtet, die in Scratch implementiert werden können. Weitere Algorithmen werden behandelt, wenn eine Umsetzung in Python möglich ist. Wesentlich in dieser Phase ist es, die

	Bausteine von Algorithmen (Kontrollstrukturen Sequenzbildung, Fallunterscheidung und Wiederholung sowie Elementaranweisungen) herauszustellen und Verfahren zur Darstellung von Algorithmen einzuüben.

Digitaltechnik im Wahlfach

Stand: 25.02.2007

Zeitrichtwert: 12 +3 +2 Unterrichtsstunden

Inhaltliche Schwerpunkte	Bemerkungen
Einstieg (1) • Innenaufbau eines realen Rechners • technische Realisierung von Bitwerten	Ziel ist es, wenigstens in Ansätzen zu verstehen, wie ein Rechner funktioniert. In einem ersten Schritt wird ein Rechner geöffnet und die Hauptbestandteile (Motherboard, Prozessor, Busse, Grafikkarte, Speicher, Netzteil,...) werden gezeigt. Man kann im Unterricht hier nicht analytisch top-down vorgehen, um die Funktionsweise zu verstehen, sondern muss einen anderen Ansatz (bottom-up) wählen. Daher wird anschließend besprochen, wie ein einzelnes Bit technisch dargestellt werden kann (z. B. durch einen Kondensator oder durch einen magnetisierbaren Nagel).
analog-digital (1) • kontinuierliche Zwischenwerte • Restaurierbarkeit • 'Verrechenbarkeit'	Hier bietet sich eine vertiefende Wiederholung der binären Speicherung von Zeichen, Bildern und Tönen an. Beim Abwägen gegen analoge Verfahren werden Vorteile und Nachteile gegeneinander abgewogen. A-D- und D-A-Wandlung werden im Grundprinzip angesprochen. Fallstudie: Tonumwandlung

Grundlagen (1) • Logische Interpretation von Bit-Werten • Grundgatter • Wahrheitstafeln	Die logische Interpretation der Bitwerte 0 und 1 und die Funktion der Grundgatter kann anhand von anschaulichen Beispielen (siehe Digitaltechnik aus Weiterbildungsmaterial) wie Aufzugssteuerung (Taste gedrückt **und** Tür zu), Türöffner (Taste im EG **oder** im OG gedrückt) oder Kühlschrankbeleuchtung (Tür **nicht** zu, Licht an) erläutert werden. Hier könnten auch einfache Schaltkreise (an Modellen) aufgebaut werden.
Realisierung der Gatter (2) • Relais • Feldeffekttransistoren	Es ist eine naheliegende Frage, wie Gatter technisch realisiert werden. Hier bietet sich das Relais als eine einfach zu verstehenden Antwort an. Mit dem Simulationsprogramm Hades lassen sich dann auch Grundgatter in Relaistechnik simulieren. Die ersten Computer, wie Zuses Z3 arbeiteten mit Relais. Heutige Hardware beruht auf Feldeffekttransistoren (FETs), deren Funktion einfacher zu verstehen ist als die 'gewöhnlicher' Transistoren. Hades kann auch FETs darstellen.
Simulation (1-2) • Grundgatter simulieren • NAND, NOR, EXOR	Im praktischen Umgang mit einer Digitalsimulation (z.B. Hades oder LogicSim) stellt sich eine gewisse Vertrautheit mit den Grundgattern ein. Weitere Gatter wie NAND, NOR und EXOR können hier unaufwändig untersucht werden.

NAND als Grundgatter (1) • Darstellung der Grundgatter • EXOR aus NANDs	Es ist theoretisch interessant, dass sich alle Gatter aus NANDs aufbauen lassen. Davon unabhängig stellen sich die hier erworbenen Kenntnisse später bei der Herleitung des R-S-Flip-Flops als sehr hilfreich heraus.
Rechengesetze (1-2) • Gesetze von de Morgan • Distributivgesetze • ...	Schon einfache aussagenlogische Probleme wie die Negation der Aussage "Es regnet und die Straße ist nass" führen zu Gesetzen wie denen von de Morgan. Mit Hilfe der Digitalsimulation lassen sich Gesetze der Booleschen Algebra verifizieren. Sie werden z.B. bei der Formulierung logischer Bedingungen in Schleifen und Alternativen benötigt.
Addierer (2) • Halbaddierer • Volladdierer	Zu den Grundfunktionen eine Computers gehört das maschinelle Rechnen. Aus der Analyse des üblichen binären Additionsschemas ergeben sich die Wahrheitstafeln zu Halbaddierer und Volladdierer. Es empfiehlt sich, den Volladdierer aus zwei Halbaddieren aufzubauen. Auf diese Weise wird nahegelegt, komplexe Schaltungen aus einfacheren aufzubauen. Das Simulationsprogramm Hades unterstützt dieses Vorgehen.

optional: **Speicher (3)** • R-S-Flip-Flop • Daten-Flip-Flop	Wie das Rechnen ist das Speichern binärer Werte eine Grundfunktion des Computers. Grundlegend für Schaltwerke ist das Prinzip der Rückkopplung eines Ausgangs auf einen Eingang. Hier kann über das Spiel 'Sichere Hand' eine erprobte Herleitung des R-S-FlipFlops eingesetzt werden. Aus dem R-S-Flip-Flop lässt sich dann die Schaltung eine 1-Bit-Speichers herleiten.
optional: **Real-Experimente (2)** • EXOR aus NAND • R-S-Flip-Flop • Daten-Flip-Flop	Wenn man die Möglichkeit hat bzw. die Investition nicht scheut (Steckboard, einige ICs, Draht, USB-Netzteil, USB-Kabel zusammen weniger als 10 Euro), so bietet das Experimentieren mit realen 'Chips' eine schöne Abrundung der Reihe.

Einführung in die imperative Programmierung

Stand: 29.04.2009

Zeitrichtwert: 18 Unterrichtsstunden

Inhaltliche Schwerpunkte	Bemerkungen
	Vorbemerkung Diese Unterrichtsreihe setzt die Unterrichtsreihe Algorithmisches Problemlösen fort, in der Grundkonzepte der imperativen Programmierung spielerisch mit Schratch erarbeitet wurden. Ziel ist es, das erworbene Wissen und Können zu vertiefen, indem jetzt eine richtige Programmiersprache beim Problemlösen eingesetzt wird. Diese Vorgehensweise erlaubt es, grundlegende Konzepte der imperativen Programmierung in einem neuen Kontext reaktivierend zu wiederholen. Als Programmiersprache nutzen wir Python. Python ist in erster Linie ... eine Sprache für professionelle Programmierer. Aber in den letzten Jahren habe ich begriffen, dass es ebenso eine der besten Sprachen ist, um Programmieren zu lernen. So beschreibt der "Erfinder" Guido von Rossum in der Einleitung zum Buch Python für Kids, wie er die von ihm entwickelte Sprache Python heute sieht. Die Programmierwelt Python stellt Lernenden keine großen Verständnisbarrieren in den Weg. Sie ermöglicht sehr einfach strukturierte Programme und erlaubt es, diese interaktiv zu testen. Zudem sind alle Wege zu modernen, weiterführenden Konzepten offen. Die folgende Unterrichtsreihe ist problemorientiert angelegt. Im Mittelpunkt steht der Problemkontext Ver- und Entschüsselung von Nachrichten. Bei der Bearbeitung spezieller Problemstellungen kommen schrittweise zentrale Konzepte der imperativen

	Programmierung zum Einsatz. Zudem wird ihre Umsetzung in Python aufgezeigt. Im Vordergrund steht dabei stets die Vermittlung allgemeiner Strukturen, die Behandlung programmiersprachlicher Details orientiert sich an der Umsetzung der algorithmischen Lösungen. Für weitergehende Informationen siehe auch inf-schule.
Variablen (2) • Variablen • Zuweisungen	Zum Einstieg wird das Verschlüsselungsverfahren nach Caesar erarbeitet. Anschließend wird Python als Werkzeug zur Verschlüsselung einzelner Zeichen eingeführt: Mit Variablen und Zuweisungen können einfache Zeichenumwandlungen interaktiv vorgenommen werden. Dabei wird ein erster Umgang mit der Programmierumgebung von Python erlernt.
Programme (2) • Programm • EVA-Struktur	Das interaktive Ausführen von Anweisungen ist mühevoll, da man die Anweisungen immer wieder neu eintippen muss. Im nächsten Schritt erfahren die Schülerinnen und Schüler, wie man Programme, die beliebig oft ausgeführt werden können, zur Durchführung von Anweisungsfolgen nutzt. Zudem lernen Sie Möglichkeiten kennen, Benutzereingaben zu verarbeiten und auszugeben und Programme entsprechend dem EVA-Prinzip zu strukturieren.
Datentypen (2) • Einfache Datentypen	Schnell stößt man auf das Problem, dass Operationen mit bestimmten Daten nicht ausgeführt werden können, wenn der Datentyp nicht passt. Dies ist Anlass, sich etwas intensiver mit Datentypen zu beschäftigen. Die im Problemkontext benötigten Datentypen werden intensiver besprochen, weitere Datentypen werden nur kurz angesprochen.

Fallunterscheidungen (2) • if-Anweisung	Zunächst wird ein Problem gelöst, bei dem Anweisungen nur unter einer bestimmten Bedingung durchgeführt werden: Beim Verschieben von Zeichen muss man den Fall, dass über das Ende des Alphabets veschoben wird, gesondert behandeln. Die Schülerinnen und Schüler kennen Fallunterscheidungen als Kontrollstruktur zur adäquaten Beschreibung einer Problemlösung. Zur Umsetzung in Python benötigen sie nur Informationen darüber, wie Fallunterscheidungen hier dargestellt werden. Inwieweit bei dieser Gelegenheit noch einmal wesentliche Aspekte der Kontrollstruktur Fallunterscheidung angesprochen und geübt werden müssen, hängt von der Lerngruppe ab.
Wiederholungen (3) • while-Anweisung • for-Anweisung	Zunächst wird ein konkretes Problem gelöst, bei dem Anweisungen wiederholt durchgeführt werden müssen: Beim Verschlüsseln eines Textes nach der Caesar-Methode muss wiederholt jedes Zeichen einzeln codiert werden. Die Schülerinnen und Schüler kennen Wiederholungen als Kontrollstruktur zur adäquaten Beschreibung einer Problemlösung. Zur Umsetzung in Python benötigen sie nur Informationen darüber, wie Wiederholungen hier dargestellt werden. Inwieweit bei dieser Gelegenheit noch einmal wesentliche Aspekte der Kontrollstruktur Wiederholung angesprochen und geübt werden müssen, hängt von der Lerngruppe ab.
Bedingungen (1) • logische Operatoren	Das Vorbereiten von Texten (Umwandlung von Umlauten, ...) ist Anlass, sich mit komplexeren Bedingungen auseinanderzusetzten. Hierbei werden die logischen Grundoperatoren nicht, und und oder wiederholt und deren Implementierung in Python aufgezeigt.

Unterprogramme (6) • Funktionen • Prozeduren • Parameter	Wir gehen hier davon aus, dass Unterprogramme noch nicht im Rahmen der Unterrichtsreihe Algorithmisches Problemlösen behandelt wurden. Ausgangspunkt bildet die Analyse umfangreicher Programme zum Vorbereiten und Verschlüsseln von Texten. Durch einen Vergleich eines additiv aus vielen Anweisungen zusammengesetzten Programms und eines mit Unterprogrammen strukturierten Programms werden die Vorteile dieser Art der Strukturierung erarbeitet. Schwerpunkt der weiteren Erarbeitung bildet dann das Funktionskonzept und dessen Implementierung in Python. Das Funktionskonzept ist von besonderer Bedeutung, da es die EVA-Sichtweise (Eingaben werden verarbeitet und als Ausgaben zur weiteren Verarbeitung zurückgegeben) unterstützt. Prozeduren als Unterprogramme ohne Rückgaben bilden einen Spezielfall, der nur dann zum Tragen kommt, wenn das Unterprogramm Seiteneffekte (wie Ausgaben auf dem Bildschirm) produziert. Die Übergabe von Daten mit Hilfe von Parametern wird von Beginn an genutzt und thematisiert. Im Wahlfach reicht es, ein erstes Verständnis für das Unterprogrammkonzept zu entwickeln. Auf spezifische Probleme wie Seiteneffekte, Sichtbarkeit von Variablen, Parameterübergabemechanismen ... sollte man nur bei Bedarf eingehen. Die Verwendung von Funktionen (und Prozeduren) muss vielfältig geübt werden. Die betrachteten Beispiele sollten gut gewählt werden und als Muster für weitere Programmierversuche dienen können. Es bietet sich auch an, Algorithmen, die bereits in der Unterrichtsreihe Algorithmisches Problemlösen bearbeitet wurden, noch einmal aufzugreifen.

Nutzung und Modellierung von Datenbanken

Stand: 29.04.2009

Zeitrichtwert: 23 Unterrichtsstunden

Inhaltliche Schwerpunkte	Bemerkungen
Bemerkungen	Die dargestellte Unterrichtsreihe lehnt sich an den Vorschlägen von Thomas Mohr an (Information und ihre Darstellung). Weitere Bemerkungen zu technischen Voraussetzungen und fachliche Hinweise für die Lehrkraft finden Sie ganz unten.
Einstieg (2) • Bedeutung von Datenbanksystemen • Datenmodellierung mit Tabellen	Eine Internetrecherche durchführen lassen: "Nachbarländer Deutschlands – Fläche, Einwohner, Hauptstadt". Hierbei wird einerseits bereits ein Informationssystem genutzt, andererseits werden die von den Schülerinnen und Schülern recherchierten Daten unterschiedlich dargestellt. Eine Darstellung in Form von Tabellen wird als hilfreich erkannt. In diesem Zusammenhang wird die Struktur eines Tabellenmodells diskutiert. Beispiele aus dem Alltag heranziehen, um die Bedeutung von Datenbanken aufzuzeigen (Telefonbuch, Schulverwaltung, ...)

Erste Abfragen an eine Datenbank entwerfen (3)
 	
• Technische Vorteile und Risiken von Datenbanksystemen
• Datenbanksystem als Mehrbenutzersystem
• Grundoperationen zur Beschreibung von Abfragen
• Umsetzung in einer Abfragesprache
	Hinführung zu einem mehrbenutzerfähigen DB-Management-System, wie dem einzusetzenden mySQL. Jede Schülerin bzw. jeder Schüler hat dabei einen eigenen Zugang zu einer mySQL-Datenbank.
	
	Den Schülerinnen und Schülern werden Beispiel-DB vorgegeben. (z.B. eine DB, die verschiedene Informationen über Kontinente, Länder, Sprachen, Orte und Flüsse enthält, wie etwa "miniterra")
	
	Die Schülerinnen und Schüler können diese vorgegebenen DB (zunächst "terra1.sql") in ihre Datenbank importieren. Anhand dieser Tabelle werden die Grundbegriffe ("Datensatz", "Spalte/ Attribut", "(Attribut-)Wert") erläutert und erste SQL-Abfragen entworfen:
	
	`SELECT [Spalten]`
	`FROM [Tabelle]`
	`WHERE [Bedingung]`
	`ORDER BY [Attribute];`
	
	Absichtliches (oder unabsichtliches) Löschen von Feldern, Datensätzen oder der gesamten Datenbank sind hier unproblematisch, führen aber zur Diskussion über z.B. ökonomische Konsequenzen bei unwiederbringlichem Datenverlust.

Ausdifferenzierung der Modellierung (6) • Grundoperationen zur Beschreibung von Abfragen • Umsetzung in einer Abfragesprache • Datenmodellierung mit Tabellen • Aufteilung in Tabellen,	Die Unzulänglichkeit der Datenhaltung in der ersten Stufe von "miniterra" wird diskutiert und erste Entwurfsprinzipien ("Vermeidung von Redundanz") herausgestellt. Im Beispiel führen unterschiedliche Schreibweisen des Kontinent-Namens "Europa" zur Erkenntnis, dass hier eine Auslagerung in eine eigene Tabelle Vorteile bringt. Dabei tauchen die Begriffe "Schlüsselattribut", "Primärschlüssel" und "Fremdschlüssel" auf. Nach dem Importieren der nächsten Stufe von "miniterra" ("terra2.sql") entsteht die Notwendigkeit die beiden Tabellen

Verknüpfung von Tabellen	"Land" und "Kontinent" bei Abfragen wieder miteinander zu verbinden ("Join"). Anhand der weiteren Stufen von "miniterra" (Erweiterung der Datenbank um die Tabellen "Ort" und "Fluss" in "terra3.sql" und "terra4.sql") wird das Einüben der Datenmodellierung vertieft (Beziehungen zwischen Tabellen werden durch eigenene Tabellen repräsentiert). Zudem werden weitere Möglichkeiten zur Bildung von SQL-Abfragen untersucht. Für detailierte Beispiele zu SELECT-Abfragen in "miniterra" s. Präsentation in den oben angegebenen Materialien.
optional: **Daten einer Miniwelt modellieren (3)** • Konzepte der ER-Modellierung	Die zuletzt betrachtete DB ("terra4.sql") wird hinsichtlich ihrer Modellierung dargestellt. Anhand dieses Modells werden Begriffe der ER-Modellierung erläutert: • Objekt / Entity • Attribut • Klasse / Entity-Set • Beziehung / Relationship Übung: Am Beispiel der Schulverwaltung werden weitere Teile einer Miniwelt modelliert.
Modelle von Miniwelten in relationale Datenbanken abbilden (6) • Implementierung von Tabellenmodellen	Eine einfache Mini-Welt (z.B. mp3-Sammlung) modellieren und implementieren. Jeder Schüler implementiert für sich seine eigene Datenbank und füllt sie mit Daten.

Datenerhebungen unter dem Aspekt Datenschutz bewerten (mind. 3) • Sammlung personenbezogener Daten • Missbrauch personenbezogener Daten • Schutz personenbezogener Daten	Datenerhebungen unter Datenschutzaspekten bewerten • Anhand eines Projekts (z.B. ????) • Fallbeispielen (z.B. Bericht des Datenschutzbeauftragten in RLP zum Jahr 2005 ("Was gehört ins Klassenbuch?")) • Materialien von klicksafe.de • Flash "Paul denkt": http://panopti.com.onreact.com/swf/index.htm
Anhang 1: **XAMPP Installation (0)**	Es wird von folgender Konfiguration ausgegangen: • Im lokalen Schulnetz ist auf einem Rechner XAMPP installiert. • Auf diesen Rechner kann von allen Schülerrechner aus zugegriffen werden. • Apache und mySQL laufen als Dienste. • Der Hauptbenutzer von mySQL ("root") ist Passwort-geschützt. • Jeder Schüler kann eine eigene Datenbank erstellen und bearbeiten • Evtl.: Für jeden Schüler ist ein eigener Zugang eingerichtet mit SELECT auf einige oder alle Datenbanken des Servers - so können die Schüler ihre Ergebnisse gegenseitig betrachten. Die Installation des XAMPP-Systems ist nicht Teil dieser Unterrichtsreihe. Evtl. wurde das XAMP-System auch schon im Vorfeld benutzt, um eigene (X)HTML-Dateien in das lokale Netzwerk zu stellen.

Anhang 2: **Grundlagen für die Lehrkraft (0)**	**Datenbanken - wozu?** Einstiegsbeispiel: Datenverwaltung kleiner Datenmengne in Delphi Nachteil: Nur mein Programm kann die Daten verwalten, schlecht zu erweitern, immer wieder gleiche auftretende Probleme Lösung: Datenbank-System Datenbank übernimmt Standardaufgaben, Mehrbenutzerfähig, leicht erweiterbar; Nachteil: Extra-Sprache, jeder Client braucht eigenes Programm für Zugriff Heute: 3-schichtige Architektur: Auf dem Client läuft nur noch ein Webclient=> keine separates Client-Programm notwendig Nachteil: http-Protokoll hat keine Sessionverwaltung => Verfolgung des authentifizierten Benutzers schwierig Folge: Ein DB-Server (MySQL) stellt seine Daten dem Web-Server (Apache) und der den Clients (bel. Browser) zur Verfügung stellt Begriffe Informationssystem und Datenbasis unterscheiden (also Unterscheidung der Begriffe Information und Daten) Relationale Datenbanken: Die Daten werden in Form von Tabellen gespeichert (Begriffe: Datensatz, Spalte) ACID-Prinzip: Paradigma für Datenbanken • ***A****tomicity*: Transaktionen (änderungen der Datenbank) finden ganz oder gar nicht statt • ***C****onsistency*: Eine Transaktion führt wieder zu einem konsistenten (gültigen) Zustand der DB (z.B. positiver Wert) • ***I****solation*: Transaktionen beeinflussen sich

	nicht gegenseitig • ***D**urability*: Eine Transaktion ist dauerhaft gespeichert, auch gegen Systemabstürze gesichert.