FAKULTÄT FÜR !NFORMATIK

Didaktische Entwürfe zum Pflichtgegenstand „Angewandte Programmierung“ des ersten Jahrganges einer HTL für Informationstechnologie

MASTERARBEIT

zur Erlangung des akademischen Grades

Magister der Sozial- und Wirtschaftswissenschaften

im Rahmen des Studiums

Informatikmanagement

ausgeführt von

Andreas Prein
Matrikelnummer 0627376

am:
Institut für Rechnergestützte Automation

Betreuung:
Betreuerin: Ass. Prof. Dipl.-Ing. Dr. techn. Monika Di Angelo

Wien, 18.08.2008

(Unterschrift Verfasser) (Unterschrift Betreuerin)

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen nicht benützt und die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Spielberg, Juni 2008 .....

Danksagung

An dieser Stelle möchte ich mich bei all jenen bedanken, die mich während der gesamten Zeit meines Studiums unterstützt haben. Ganz besonderer Dank gilt meinen Eltern, Gerda und Robert Prein, die all meine Entscheidungen immer mitgetragen haben.

Auch bei meiner Masterarbeit-Betreuerin Ass. Prof. Dipl.-Ing. Dr. techn. Monika Di Angelo, bei Prof. Mag. Franz Gansterer der Technischen Universität Wien sowie bei Prof. Mag. Wolfgang Rettl der HTL Zeltweg möchte ich mich recht herzlich für die fachliche Unterstützung bedanken.

Weiters möchte ich mich bei meinem Abteilungsleiter Brigadier Dipl.-Ing. Rudolf Truxa und meinem Dienstellenleiter Brigadier Dipl.-Ing. Andreas Knoll meiner Dienststelle Materialstab Luft des Bundesministeriums für Landesverteidigung für ihre Unterstützung bedanken.

Vielen Dank!

Kurzfassung

Der Begriff Didaktik beschreibt ganz allgemein die Unterrichtslehre, also die Kunst des Lehrens aber auch die Wissenschaft von der Methode des Unterrichts. Die zentrale Frage einer Fachdidaktik lautet allgemein formuliert:

„Was soll wann, wie und mit welchem Ziel gelehrt werden?“

Der Begriff Informatik wurde aus „Information“ und „Mathematik“ als zusammengesetztes Kunstwort in die deutsche Sprache übernommen. Die Programmierung, ein Teilbereich der Informatik, soll den Schülern mittels Fachdidaktik vermittelt werden. Die Problematik besteht darin, die Lerninhalte, welche aus dem Lehrplan der höheren Lehranstalt für Informationstechnologie – Schulformkennzahl 8490 – des ersten Jahrgangs aus dem Pflichtgegenstand 11 „Angewandte Programmierung“ entnommen wurden, schülergerecht aufzubereiten und den Schülern mittels praxisnaher Beispiele verständlich zu machen.

Beim Vermitteln der Inhalte spielt die vom Vortragenden gewählte Unterrichtsmethode eine entscheidende Rolle. Der Lehrer sollte bestrebt sein, seine Methode so zu wählen, dass der Unterricht von den Schülern stets als spannend und interessant und nicht als langweilig wahrgenommen wird.

Ziel dieser Arbeit ist es, den Lehrplan für den Pflichtgegenstand „Angewandte Programmierung“ so zu entwickeln, dass der größtmögliche Lernerfolg bei den Schülern erzielt wird. Dabei wird versucht, verschiedene Methoden zu verwenden, und somit die Schüler zu motivieren und aktiv in den Unterricht mit einzubinden.

Die 15 Unterrichtseinheiten des ersten Semesters werden mittels Stundenbilder in einer praxiserprobten Form detailliert dargestellt. Im Anschluss an jede Einheit werden die Inhalte nochmals detailliert beschrieben, und gegebenenfalls mit Praxisbeispielen untermauert. Weiters sind im Anhang Prüfungsfragen, die das gesamte Stoffgebiet umfassen, aufgelistet.

Da ich selbst die Absicht verfolge, in naher Zukunft als Vortragender an einer HTL tätig zu sein, hatte und habe ich stets größtes Interesse an diesem Thema.

Abstract

The term „didactics“ paraphrases “the teaching of teaching”, that is not only the art of teaching but also pedagogy, that is the methodology of teaching.
A specialized didact usually asks the following basic question:

„What do I teach how, and when, and which aims do I want to achieve?“

The term „informatics“ is an artificial word made up from information and mathematics. An appropriate methodology of teaching must be developed to convey the concepts of programming – as a part of informatics – to the students.
The issue is to update the objectives of course 11 “Applied Programming” from the initial curriculum of the Höhere Technische Lehranstalt (HTL) für Informationstechnologie, Schulformkennzahl 8490 (Higher Technical Institute of Information Technology, type of school code 8490). The objectives of course 11 “Applied Programming” must be edited most suitably for students and communicated to them using practice-related examples.
The teaching method is of outstanding importance for the proper communication of the course contents. The teacher should be anxious to choose a method that attracts the students’ interest and keeps them at it.
The aim of this work was to compile a curriculum for course “Applied Programming” that ensures a durable, successful learning process. It employs different didactic techniques that are determined to motivate the students and encourage them to actively participate in the class.
I summarized the contents of the 15 first semester lessons in the approved form of “Stundenbilder”. In addition, I described the contents of each lesson in detail and, if necessary, explained them with practice-related examples. Finally, the appendix lists examination questions comprehending the entire subject matter.
Since I myself would like to be a HTL teacher in the near future, I am very much interested in this topic.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

AK	Abkürzungsbeschreibung
etc.	et cetera
HTL	Höhere technische Lehranstalt
IT	Informationstechnik
PAP	Programmablaufplan
Pkt	Punkte
PPT	Microsoft Power Point Präsentation
VB	Microsoft Visual Basic
VBA	Microsoft Visual Basic for Applications

I Einleitung

Der Computer ist in unserer heutigen Zeit fast nicht mehr wegzudenken. Es gibt so gut wie keine Bereiche mehr, wo nicht der Computer den Menschen in seinem Tun unterstützt. Zu diesen Bereichen zählt nicht nur die Arbeitswelt, sondern auch die Freizeit aber auch ein simulierter Blick in die Zukunft was zum Beispiel die Wetterprognose betrifft.
Aber es ist nicht nur der Computer alleine, der uns bei unseren Arbeiten unterstützt. Ein genauso wichtiger Teil ist die Software die auf den verschiedensten Rechnersystemen ihren Dienst versieht.
Doch diese Software muss von Experten ausgedacht, programmiert und auf den Rechnern installiert und zum Arbeiten gebracht werden.
Aber auch diese so genannten Experten müssen klein anfangen und zu Beginn ihrer Karriere die Schulbank drücken. Aus diesem Grund ist es sehr wichtig, eine solide und fachlich kompetente Schulausbildung für angehende Programmierer zur Verfügung zu stellen.
In dieser Arbeit wird versucht, einen Weg zur Ausbildung von zukünftigen Programmentwicklern aufzuzeigen. Dabei kann aber nur der Pflichtgegenstand „Angewandte Programmierung“ des ersten Jahrgangs einer HTL dargestellt werden.
Auf Basis dieser Stoffauflistung wird das erste Semester in einer groben Wochenübersicht dargestellt. Danach werden die 15 Doppelstunden des ersten Semesters detailliert dargestellt und beschrieben.
Diese detaillierte Darstellung sowie die Programmcode Beispiele dienen dazu, den Schülern den Lehrstoff bestmöglich zu vermitteln.
Natürlich kann diese Art und Weise der Wissensvermittlung nur als mögliche Variante gesehen werden.

Die Motivation dieser Arbeit liegt der Vorstellung zu Grunde, in naher Zukunft selbst als Lehrer einer HTL den Unterricht für die Schüler gestalten zu dürfen.

Abschließend wird festgehalten, dass sich die Darstellung der Personen immer auf beide Geschlechter, männliche wie weibliche, bezieht.

II Grundlagen

2.1 Grundlagen der Informatik

2.1.1 Informationstechnik – Informationstechnologie

Die Informationstechnik, kurz IT, stellt eine zusammenfassende Bezeichnung für Informationstechnologie, Computer- und Kommunikationstechnik dar und ist an die traditionelle Nachrichtentechnik angelehnt. Die IT wird als Bindeglied zwischen der traditionellen Elektrotechnik und der noch relativ jungen Informatik gesehen. In der Darstellung von [Reit03] können zur IT Netzwerke, Speicherung und Verarbeitung von Texten, Sprachen, Töne und Musik, Standbildern, Videos aber auch die Interaktion gezählt werden.

2.1.2 Informatik

Das Wort „informatique“ wurde von Philippe Dreyfus, Ingenieur bei der Firma Bull in Frankreich geprägt. Das Wort „Informatik“ wurde als Kunstwort von Information und Mathematik in die deutsche Sprache übernommen und sollte dabei auf die Verbindung der Bereiche Information und Automatik hinweisen. Wie in [Reit03] dargestellt, ist Informatik mehr als nur Programmieren. Es werden zwar Programmierkenntnisse vorausgesetzt, aber wie die nachfolgende Tabelle zeigt, gliedert sich die Informatik in mehrere Teilbereiche auf.

Informatik			
Kerninformatik			Angewandte Informatik
Theoretische Informatik	Theoretische Informatik	Theoretische Informatik	Medizinische Informatik
Formale Sprachen	Multimedia Computergrafik	Computer Organisation	Wirtschaft Informatik
Computer Modelle	Informations-Modelle	Logischer Rechenentwurf	Rechts Informatik
Automaten Theorie	Mensch-Maschine-Kommunikation	Mikro Programmierung	Geo-Informatik
…	…	…	…

Tabelle 1: Klassische nicht abgeschlossene Einteilung der Informatik
Quelle: vgl. [Reit03]

Wie in [Schu04] beschrieben, unterscheidet man in der Informatik sechs Teilgebiete:

- Kerninformatik
    - die Theoretische
    - die Praktische
    - die Technische
    - die Anwendungen der Informatik
- gesellschaftliche Bezüge der Informatik
- Didaktik der Informatik

Theoretische Informatik:

Dabei werden die Formulierungen und Untersuchungen von Algorithmen aber auch die Rechnerkonstruktion und deren Methoden und Modelle aus der Mathematik verstanden. Da die Informatiksysteme ständig an Komplexität zunehmen, wird auch der Abstraktionsgrad einer angemessenen Beschreibung aufwändiger.

Beispiele: Formale Sprachen, Theorie der Netze und Prozesse, Automatentheorie, Semantik und Komplexitätstheorie

Praktische Informatik:

Um Algorithmen auf einer Rechneranlage bearbeiten zu lassen, muss der Computer zu einem komfortablen Werkzeug gemacht werden. Programme welche in maschinenunabhängigen Programmiersprachen geschrieben werden, müssen von speziellen Übersetzungsprogrammen in eine computerverständliche und ausführbare Form gebracht werden.

Beispiele: Übersetzerbau, Software – Engineering, Informationssysteme, Betriebssysteme, Simulation und künstliche Intelligenz

Technische Informatik:

Diese befasst sich mit dem funktionellen Aufbau von Computern, den zugehörigen Geräten sowie den logischen Entwurf und die konkrete Entwicklung von Rechnern. Die Schnittstelle zu Betriebssystem und die Zusammenstellung von Computern spielen eine wichtige Rolle.

Beispiele: Rechnerarchitektur, Prozessdatenverarbeitung, Fehlertoleranz, Leistungsmessungen

Anwendungen der Informatik:

Unter Anwendung der Informatik werden die Anwendungen von Methoden der Kerninformatik in anderen Wissenschaften und die Entwicklung spezieller Verfahren und Darstellungstechniken verstanden.

Beispiele: Wirtschaftsinformatik, Rechtsinformatik, Bioinformatik, medizinische Informatik.

Gesellschaftliche Bezüge der Informatik:

Informatik hat starke Auswirkungen auf die Gesellschaft und wird von der Informatik und Informationsflüssen geprägt. Der Einsatz von Computern beeinflusst die Arbeitswelt und auch den Freizeitbereich. Auch die Regelungen über den Umgang mit schutzwürdigen Daten sind zu treffen. Der Computer kann zur Steuerung, Informationssammlung und Informationsauswertung auf fast allen Gebieten von Wirtschaft, Wissenschaft, öffentlichen und privaten Leben eingesetzt werden. Aufgrund der hohen Arbeitsgeschwindigkeit können immer neue und immer komplexere Probleme gelöst werden.

Didaktik der Informatik:
Fachdidaktik stellt den Bezug zwischen der Fachwissenschaft und der Lebenswelt her. Darauf wird im Kapitel 2.2 Grundlagen der Didaktik näher eingegangen.

Von vielen Experten wird „Informatik“ als „vierte Kulturtechnik“ bezeichnet. Davon kann aber in vielen Teilen der Erde noch keine Rede sein, da weder Informationstechnologie noch Informatik und auch Internet nicht fest in die berufliche Bildung integriert sind vgl. [Reit03].

2.2 Grundlagen der Didaktik

Nachfolgend werden einige theoretische Ansätze der Didaktik erläutert.

2.2.1 Bildungstheoretischer Ansatz – Göttinger Schule:

Zentrales Leitbild dieses Ansatzes nach [Hubw07] ist die bildende Begegnung des Menschen mit der kulturellen Wirklichkeit. Daraus resultiert eine doppelseitige Erschließung: Die Wirklichkeit erschließt sich dem Menschen und umgekehrt.

> *Die Aufnahme und Aneignung von Inhalten ist stets verbunden mit der Formung, Engwicklung und Reifung von körperlichen, seelischen und geistigen Kräften.*

Diese Auffassung steht im krassen Gegensatz zur historischen Trennung von materialer Bildung auf der Objektseite (Erwerb von Wissen, Erschließung der Wirklichkeit) und formaler Bildung auf der Subjektseite (Formung, Reifung von Kräften, Aneignung von Methoden).
Das vorrangige Ziel der Göttinger Schule ist die Allgemeinbildung und sollte daher nur besonders allgemeine Themen enthalten.

Fünf Fragen bestimmen die Auswahl der Unterrichtsgegenstände:

- Welche exemplarische Bedeutung hat der Unterrichtsgegenstand?
- Wie bedeutend ist er für die Gegenwart?
- Welche Bedeutung für die Zukunft lässt sich vermuten?
- Wie ist die Struktur des Inhalts?
- Wie steht es mit der unterrichtlichen Zugänglichkeit?

Für die Informatikdidaktik leistet dieser Ansatz vor allem eine solide Fundierung der Allgemeinbildung.

2.2.2 Lerntheoretischer Ansatz – Berliner Didaktik

Unter diesem Ansatz der auch oft als Berliner Didaktik bezeichnet wird, wird nach [Hubw07] die Theorie des Lehrens und des Lernens verstanden. Im Gegensatz zur Göttinger Schule, die sich ausschließlich mit der Frage „Was ist zu lehren?“ beschäftigt, ist hier auch die Frage „Wie soll man den Stoff vermitteln?“ von Bedeutung.
Der Lehrende hat unter Berücksichtigung der Elemente verschiedener Bedingungsfelder seine Entscheidungen in den jeweiligen Entscheidungsfeldern so zu treffen, dass die gewünschten Resultate erreicht werden können.

1. soziokulturelle Voraussetzungen		
sozio-ökonomische	Finanzielle und wirtschaftliche Rahmenbedingungen	Klassenstärken, Ausstattungen mit Lehrmittel, etc.
sozio-ökologische	Einlagerung des Unterrichts in ein räumliches Umgebungsgeflecht	Stadt- oder Landschule, Verkehrverbindungen, Lärmbelästigung, etc.
sozio-kulturelle	Aus der geschichtlich- geistigen Situation erwachsende Strömungen, Einstellungen, etc	Tabus, Kommunikationsweisen, Sprachformen, Symbole, etc.
ideologisch-normbildende	Aus Interessenlagen einzelner gesellschaftlicher Gruppen und Mächte stammende Einflüsse	Politische Richtziele im Wandel der Parteilandschaften, Einfluss der Umweltschutzbewegungen

2. anthropologisch-psychologische Voraussetzungen		
Schülerseite	Lernfähigkeit	Lernstand: Wissen, Können, Haltung, Lernstil
	Lernbereitschaft	Lerntempo
Lehrerseite	Lehrfähigkeit	Lehrstand: Wissen, Können, Haltung, Lehrstil
	Lehrbereitschaft	

Tabelle 2: Bedingungsfelder der Berliner Didaktik
Quelle: vgl. [Hubw07]

Intentionen	Welche Zielsetzung hat der Unterricht?
Lerninhalte	Was wird gelehrt?
Methoden	Wie wird der Stoff vermittelt?
Medien	Womit wird der Lernstoff transportiert?
Folgen des Unterrichts	Welche soziokulturellen und anthropologisch-psychologischen Auswirkungen wird der Lernvorgang haben?

Tabelle 3: Entscheidungsfelder
Quelle: vgl. [Hubw07]

2.2.3 Informationstheoretisch-kybernetischer Ansatz

Nach [Hubw07] wurde unter der Anwendung von damals sehr aktuellen Methoden der Informationstheorie und der Kybernetik der Prozess des Lernens, die Didaktik, auf reine Methodik reduziert. Es erfolgte so gut wie keine Auswahl von Lehrstoff mehr.

Entscheidende Faktoren sind dabei einerseits der pädagogische Raum, der sich in die Dimensionen Lehrziel, Lehrstoff, Medium, Soziostruktur und Psychostruktur gliedern lässt, und andererseits die Struktur des Lernprozesses.

Eine derart komplexe Domäne wie den Lernprozess mit so einfachen Modellen behandeln zu wollen, erscheint aus heutiger Sicht geradezu naiv.

2.2.4 Kommunikative Didaktik

Bei dem kommunikativen Ansatz nach [Hubw07] wird der Unterricht als ein kommunikativer und edukativer, bildungsbezogener Prozess behandelt. Zwei Aspekte sind dabei entscheidend:

- die Inhaltsdimension, die durch das Curriculum weitgehend festgelegt ist
- die Beziehungsdimension, die das Verhältnis zwischen Lehrer und Schüler beschreibt

Besonderer Wert wird auf die fortschreitende Emanzipation der Lernenden gegenüber dem Lehrer und gegenüber der Umgebung gelegt. Neben der Kommunikation im Unterrichtsprozess wird hier auch die Bedeutung der Metakommunikation, als die Kommunikation über die Kommunikation, betont.
Der Verdienst dieses Ansatzes liegt vor allem in der Klarstellung der Bedeutung des sozialen Wechselwirkungsprozesses zwischen Lehrer und Schüler.

2.2.5 Didaktik der Informatik

Didaktik ist vgl. [Schu04] im Allgemeinen die Unterrichtslehre, die Kunst des Lehrens, Wissenschaft von der Methode des Unterrichts. Dabei sind aber weitere unterschiedliche Schwerpunkte möglich:

- Didaktik als Wissenschaft und Lehre von Lehren und Lernen
- Didaktik als Wissenschaft vom Unterricht beziehungsweise der Theorie des Unterrichts
- Didaktik als Theorie der Steuerung von Lernprozessen
- Didaktik als Theorie der Lehrinhalte, Bildungsinhalte, ihrer Struktur, Auswahl und Zusammensetzung
- Didaktik als Theorie der Unterrichtsformen und Unterrichtsverfahren

Die Zentrale Frage einer Fachdidaktik lautet allgemein formuliert:

„Was soll wann, wie und mit welchem Ziel gelehrt werden?“

Die Fachdidaktik, vgl. [Schu04], stellt einen Bezug zwischen der Fachwissenschaft und der Lebenswelt her. Es wird versucht, die gewonnen Erkenntnisse der Fachwissenschaften für die Schule aber auch für die allgemeine Aus-, Fort- und Weiterbildung von Kindern und Erwachsenen verfügbar zumachen.

Zu diesem Prozess gehören:

- Definition der Ziele des Fachunterrichts
- Entwicklung von Konzepten zur Methodik und Organisation des Unterrichts
- Festlegung, welche Ideen, Methoden und Erkenntnisse der Fachwissenschaft im Unterricht vermittelt werden sollen
- Reihung der Unterrichtsinhalte zu Lehrplänen und ihre fortlaufende Aktualisierung

Die Didaktik der Informatik ist keine in sich ruhende Wissenschaft. Mehrere andere Wissenschaften wie zum Beispiel die Psychologie aber auch die Institutionen wie Schule, Behörden und dergleichen, wirken auf die Fachdidaktik ein.

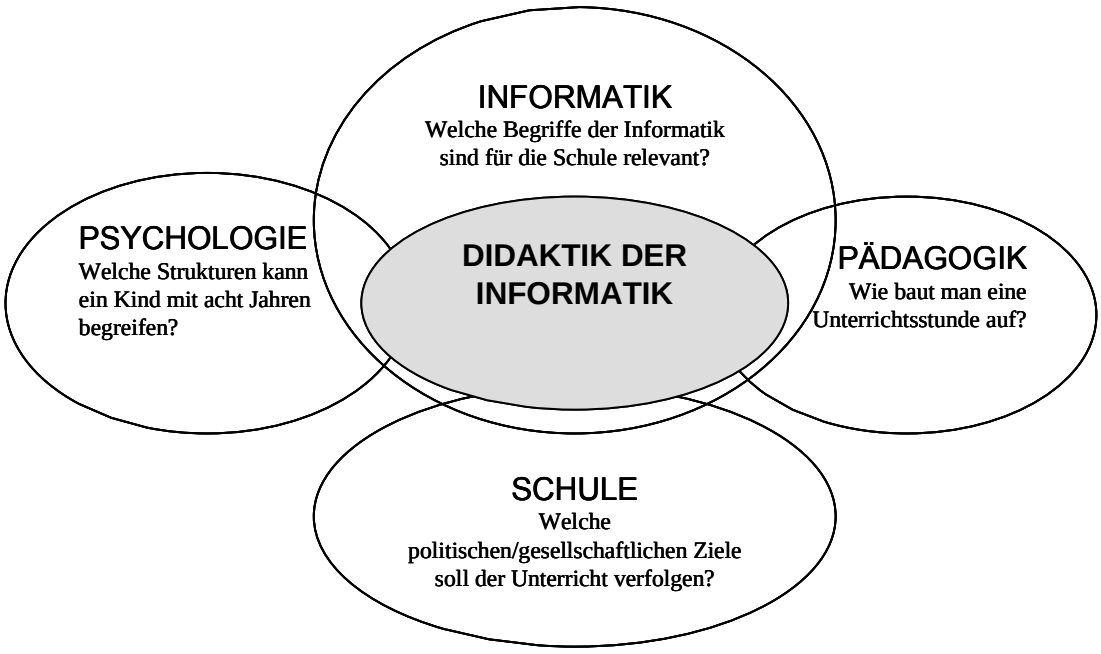


Abbildung 1: Einbettung der Didaktik der Informatik
Quelle: vgl. [Schu04]

2.2.6 Motivation

Wie in [Hubw07] dargestellt, ist ohne den durch eine angemessene Motivierung erzeugten Lernwillen jedes unterrichtliche Bemühen sinnlos. Deshalb ist Motivierung das vordringlichste Ziel didaktischen Handelns.
Unter Motivierung versteht man einen kurzen andauernden Zustand des Angetriebenseins. Unter Motivation dagegen das aktive Bemühen um die Herstellung eines solchen Motivationszustandes. Länger andauernde Initiierungs- und Lenkungsfaktoren wie Funktionslust, Schaffensfreude, Erfolg, Ehrgeiz, usw. bezeichnet man als Motive vgl. dazu Abbildung 2: Motivierung, Motive und Motivation.

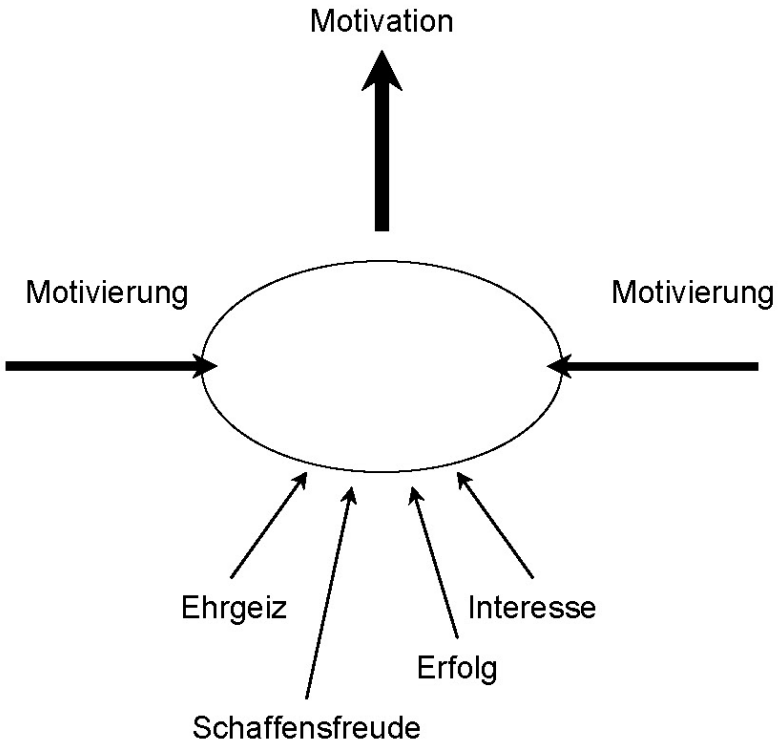


Abbildung 2: Motivierung, Motive und Motivation
Quelle: vgl. [Hubw07]

Der zu erreichende Endzustand, Motivation, kann nach verschiedenen Aspekten unterschieden werden:

- Lernmotivation: Auseinandersetzten mit der Welt
- Leistungsmotivation: Beweis der eigenen Tüchtigkeit
- intrinsische Motivation: zielt auf die Beschäftigung mit der Sache selbst
- extrinsische Motivation: zielt auf äußere Begleitumstände wie Lob, guten Noten
- Eingangsmotivierung: am Anfang der Unterrichtseinheit
- Verlaufsmotivierung: im weitern Verlauf der Unterrichtseinheit

Begriffserklärung nach [@Trim] Intrinsische/extrinsische Motivation
Verhalten ist entweder

- von innen heraus; um seiner selbst willen – intrinsisch
- von außen her; durch Zurückführen auf externale Verhaltensfolgen in Gestalt von Bekräftigungen – extrinsisch

motiviert.

Die Beziehung zwischen Motivation und Lernen ist nach [Hubw07] immer wechselseitig, das heißt Motivation ist Voraussetzung, aber auch Ergebnis von Lernprozessen. Erfolg beim Lernen kann über Motivationsverstärkung die Lernleistung mehr als verdoppeln.

Praxistipps von [Hubw07]:

- Erwartungshaltung der Schüler beachten
- Nähe zum Erfahrungsbereich der Lernenden suchen
- Überraschungsmomente anbieten
- keine überhöhten Erwartungen stecken
- möglichst intrinsisch, nur in unbedingt nötigen Fällen extrinsisch motivieren
- Freude am Tun vermitteln
- spontane Interessen berücksichtigen
- Entscheidungssituationen schaffen
- Hygienefaktoren berücksichtigen – Luft, Temperatur, Licht, soziales Klima

Für ein Nachlassen der Motivation gibt es nach [Hubw07] eine Vielzahl von Ursachen. Das Fragebedürfnis des Lernenden spielt dabei eine wichtige Rolle. Wenn Fragen nur zum Nachweismittel für bereits Erlerntem herabgesetzt werden, kann die Motivation stark absinken. Ähnlich ist es auch mit Antworten auf Lehrerfragen, die sich der Schüler zurechtgelegt hat, ohne diese jemals anbringen zu können.
Einen großen Beitrag zum Motivationsverlust kann aber auch der Lehrer selbst beitragen. So zum Beispiel wenn er negative Persönlichkeitsmerkmale oder eine schlechte Tagsverfassung aufweist, einen stark dominanten oder intoleranten Führungsstiel hat oder seine Schüler ständig unter- oder überfordert.
Weitere Ursachen für Motivationsverlust können methodische Eintönigkeit, fehlender Neuigkeitsgehalt, unbefriedigte Primärbedürfnisse wie Hunger, Müdigkeit oder störende Umwelteinflüsse wir Hitze oder Lärm sein.

Ein Beispiel, wie Motivation erzeugt werden kann ist in [Leem07] beschrieben.
Da das Programmieren bei Schülern und Lehrern generell als schwierige Angelegenheit angesehen wird, kämpfen viele Schüler von den Einführungsmodulen bis hin zu höheren Semestern mit vielen Problemen. Dabei ist die richtige Sprache der Lehrenden gegenüber den Schülern aber auch die richtige Motivation von großer Bedeutung.
Um die Motivation zu fördern, kann der Programmierunterricht zum Beispiel mittels eines programmierbaren Roboters unterstützt werden. Es handelt sich bei der Programmiersprache zwar um Java, also eine Objektorientierte Programmiersprache, aber die Art und Weise des Unterrichts kann auch auf andere Programmiersprachen angewendet werden. Die Textbooks sowie die benötigte Software aber auch Beispiele dieses Java Roboters können von der Homepage, siehe [@Beck], kostenfrei bezogen werden.
Die simulierte Roboterumgebung beinhaltet ein einfaches Interface welches speziell entwickelt wurde, um Schülern das Programmieren in den ersten Jahrgängen zu lehren. Der „Roboter“ kann angewiesen werden, sich zu bewegen, sich zu wenden, Dinge aufzuheben etc. Die einfache Programmierung der verschiedenen Aktionen des Roboters und der sofortige sichtbare Erfolg ist dafür mit verantwortlich, dass die Schüler Spaß am Unterricht haben und motiviert an die Programmierung herangehen.

2.2.7 Veranschaulichung

Als reine Datenmenge gesehen, kann nach [Hubw07] die Fülle der auf einen Schüler einströmenden Reize nicht verarbeitet werden.
Dabei kommt folgendes Zitat zur Anwendung:

> *Veranschaulichung ist das Bemühen des Lehrenden, einen Lehrinhalt so aufzubereiten, dass bei aller Wahrung der Sachgemäßheit die Vorstellungsfähigkeit des Lernenden unterstützt wird, um zu intendierten Begriffsbildung zu gelangen. Die Anschaulichkeit der Unterrichtsgestaltung zielt auf Anschauung. Dies ist ein aktiver Prozess der nur vom Lernenden vollzogen werden kann.*
> *[Hubw07]*

Ganzheitliche Lernkonzepte sollten nach [Hubw07] dabei möglichst viele verschiedene Sinnesorgane einsetzen. Zur Veranschaulichung von Lerninhalten können auch Medien eingesetzt werden. Den höchsten Grad der Anschaulichkeit hat dabei die originale Begegnung. Falls diese nicht eingesetzt werden kann, muss man auf andere Medien zurückgreifen. Zahlreiche Medien werden im Erfahrungskegel vgl. Abbildung 3: Erfahrungskegel (von unten nach oben zu lesen) dargestellt. Beim Einsatz von Medien gilt es aber ein Überangebot an Reizen zu vermeiden.

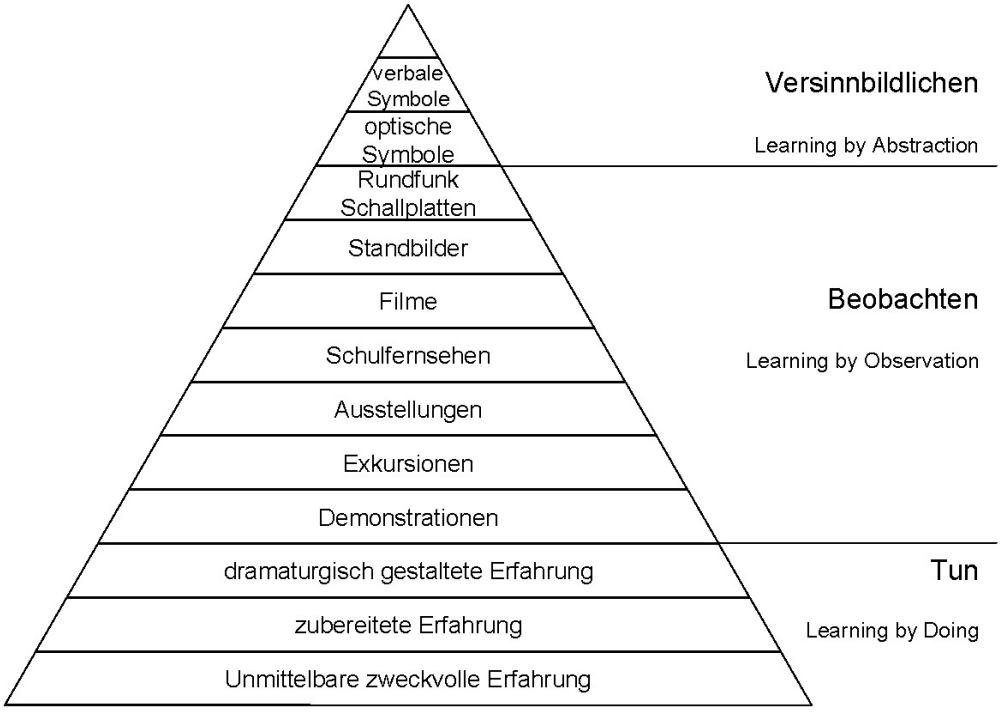


Abbildung 3: Erfahrungskegel
Quelle: vgl. [Hubw07]

2.3 Unterrichtsgestaltung

2.3.1 Was ist Unterricht

Unterricht ist ein hochkomplexer Prozess mit zahlreichen, auch zyklischen Wechselwirkungen, bei dem Lehrende und Lernende unter gewissen gesellschaftlichen, bürokratischen und materiellen Vorgaben und Rahmenbedingungen im Hinblick auf eine bestimme Zielsetzung interagierten. Langfristig hat dieser Prozess wiederum Auswirkungen auf die gesamte Gesellschaft und die von ihr formulierten Vorgaben.
[Hubw01]

2.3.2 Zeitliche Planung

Der Lernstoff wir auf bestimmte Zeiteinheiten aufgeteilt. Die verschiedenen Planungsstufen sind von unterschiedlichen Zuständigkeiten abhängig.

	verantwortlich	**zeitlicher Umfang**	**Inhalt**
Lehrplan	Kommissionen	Ausbildungs-abschnitt	Lernziele und Inhalte
Jahresplan	Lehrender	Jahr	Lernziele und Inhalte
Mittelfristige Planung	Lehrende	Woche	Inhalte und Methoden
Unterrichtsentwurf	Lehrender	Unterrichtseinheit	Inhalte, Methoden, Sozialformen und Medien

Tabelle 4: Stufenmodell der zeitlichen Unterrichtsplanung
Quelle: vgl. [Hubw01]

2.3.3 Lernziele

Die Lernziele, vgl. [Hubw01] sollten die wichtigste Frage „Was will ich mit dieser Unterrichtseinheit genau erreichen?“ beantworten.
Die Lernziele können in vier verschiedenen Stufen dargestellt werden:

Problemlösung	Selbstständiges Kombinieren verschiedener erlernter Kenntnisse und Fähigkeiten zu Lösung noch nicht behandelter Probleme
Transfer	Übertragen von Gelerntem auf einen anderen Anwendungskontext
Reorganisation	Wiedergabe in gegenüber dem Lernvorgang veränderter Form
Reproduktion	Wiedergabe in derselben Form wie beim Lernvorgang

Tabelle 5: Stufen der Lernziele
Quelle: vgl. [Hubw01]

2.3.4 Lehr- und Lernmethoden

Bei der Artikulation, vgl. [Hubw01] wird die Unterrichtseinheit in weitere kleinere Abschnitte unterteilt. Diese Stufen können unter anderem Motivation, Schwierigkeiten, Lösung, Tun und Ausführen, Behalten und Einüben, Bereitstellen, Übertragen und Integration sein.
Die Vielzahl der verschiedenen Lehr- und Lernformen lassen sich wie folgt in drei entsprechende Kategorien einordnen:

Lernmethode	Lehrmethode	Beispiele
rezeptiv	Darbietend	Vortag, Anschreiben von Beweisen, Demoversuch
geleitet produktiv	Anleitend	Gesprächsführung, Begutachtung, Richtigstellung, Beispiele geben
selbstständig produktiv	Anregend	Aufgabenstellung, Aufzeichnung eines Problems, Vermittlung von Informationsquellen und Arbeitshilfen

Tabelle 6: Lehr- und Lernformen
Quelle: vgl. [Hubw01]

Die Sozialform beschreibt die äußere Form der Interaktion zwischen Lehrer und Schüler. Dabei kann folgendermaßen Unterschieden werden:

Unterricht im Klassenverband	Lehrerzentriert	Frontalunterricht, Unterrichtsgespräch mit einem Lehrer
	Schülerzentriert	Schülervortrag, Unterrichtsgespräch ohne Lehrer, Schüler als Gesprächsleiter
differenzierter Unterricht	Gruppenunterricht	arbeitsgleich, arbeitsteilig
	Einzelunterricht	Programm-, lehrer-, schülergesteuert

Tabelle 7: Sozialformen
Quelle: vgl. [Hubw01]

2.3.5 Medien

Die Medien, wie in [Hubw01] beschrieben, dienen zur Veranschaulichung von Lerninhalten. Bei der Auswahl der Medien sollten folgende Fragen gestellt werden:

- Ist dieses Medium eindeutig genug, um das Lernziel unmissverständlich und klar darzustellen?
- Repräsentiert dieses Medium den Inhalt sodass eine optimale Erfahrungsquelle für den Lernprozess geschaffen ist?
- Ist dieses Medium attraktiv genug, um Aufmerksamkeit zu erzeugen?

Die Gestaltung eines Informatikfachraumes sollte nach [Humb06] wenn möglich folgendermaßen aussehen:

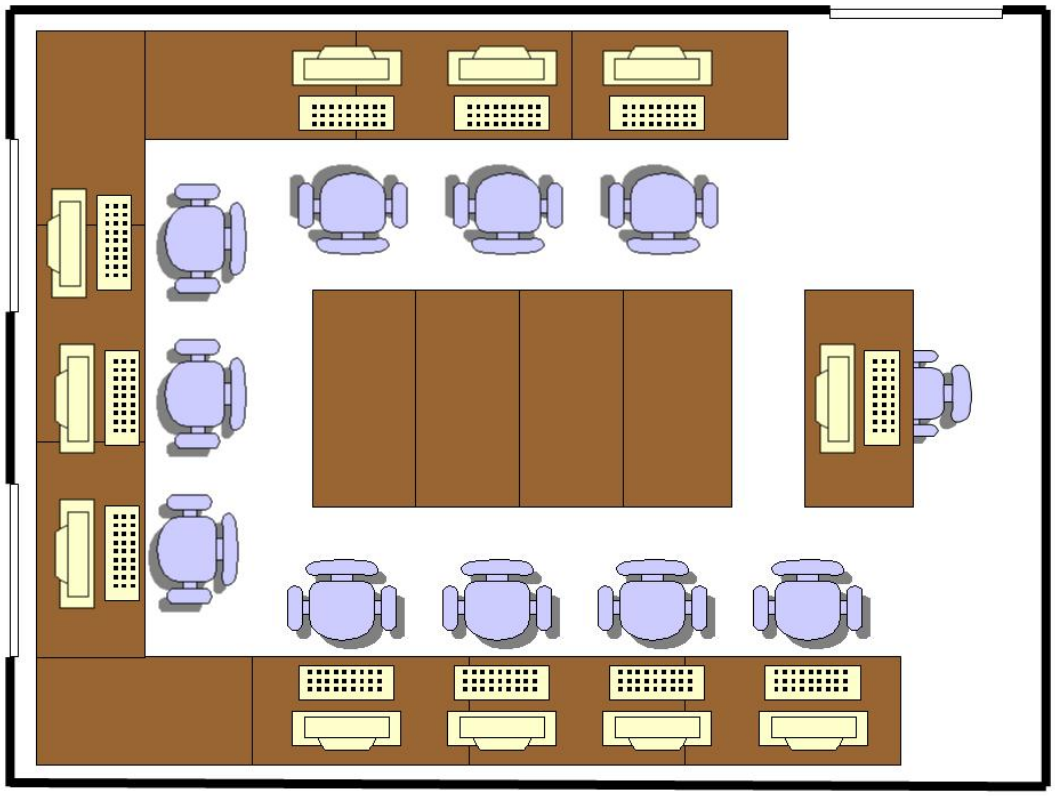


Abbildung 4: Gestaltung eines Informatikfachraumes
Quelle: vgl. [Humb06]

Die Computerarbeitsplätze sind an der Wand entlang angeordnet. In der Mitte befinden sich weitere Arbeitstische.

2.3.6 Programmiersprachen

Nach [Hubw07] zwingt die Implementierung der erstellten Modelle meist zur Verwendung von Programmiersprachen. Dabei stellt sich die Frage, welche der zahlreichen Sprachen dabei verwendet werden soll. Nachfolgende werden die Vor- und Nachteile einiger Sprachparadigmen angeführt. Dabei ist die Zuordnung einer Sprache zu einem bestimmten Paradigma nicht immer eindeutig.

Funktionale Sprachen (z.B.: Haskell, Gofer, ML)**:**
Diese Sprachen haben meist einen sehr knappen Syntax mit der Möglichkeit einer kompakten Darstellung rekursiver Datenstrukturen. Ein Programm ist ein Term, in dem selbst definierte Funktionen vorkommen können. Wiederholungen von Befehlen können nur über die Rekursion gesteuert werden. Dadurch könnten vor allem jüngere Schüler überfordert sein.

Imperative Sprachen (z.B.: Basic, Pascal, C, Modula 2, Oberon)**:**
Mit dieser Sprache kann ein Grundverständnis für die Funktionsweise der Von-Neumann-Rechner Architektur erzeugt werden. Weniger positiv ist die „Ad-hoc Codierung“ sowie die oft schwer durchschaubare Zustandssemantik.

Objektorientierte Sprachen (z.B.: C++, Java, Smalltalk)**:**
Diese Sprachen liegen nahe am menschlichen Weltbild. Schwierigkeiten bereiten dagegen erfahrungsgemäß häufig die höheren Konzepte der Objektorientierung wie dynamische Bindung, Vererbung und Polymorphie.

Prädikative Sprachen (z.B.: Prolog)**:**
Für spezielle Anwendungen wie die Modellierung von Wissensbasen sind diese Sprachen sehr gut geeignet. Die Syntax ist minimal. Problematisch ist das Grundverständnis der Semantik, das zumindest elementare Einblicke in die Prädikatenlogik voraussetzt.

Derzeit gibt es noch kein optimales Sprachenparadigma für schulische Zwecke. Daher sollten im Laufe der Schulausbildung mehrere verschiedene Sprachen eingesetzt werden. Dabei ist allerdings zu berücksichtigen, dass Schüler sehr viel Zeit zur Einarbeitung in eine neue Sprache benötigen.

2.4 Unterrichtsplanung

Um einen Unterricht konkret, vgl. [Humb06] vorzubereiten bedarf es einer Planungsstruktur. Diese Sequenzierung der Planungsprozesse läuft Gefahr, eine unzulässige Vereinfachung zu erzeugen, oder die falsche Reihenfolge zu erstellen.
Ein Vorgehensmodell zur Unterrichtsvorbereitung gliedert den Prozess der Vorbereitung in verschiedene, strukturierte Phasen. Aufgabe und Ziel eines solchen Vorgehensmodells besteht darin, die in diesem Gestaltungsprozess auftretenden Fragen, Problemstellungen und Aktivitäten in einer begründeten und plausiblen Ordnung darzustellen und die Methoden und Techniken zur konkreten Unterrichtsvorbereitung bereitzustellen.

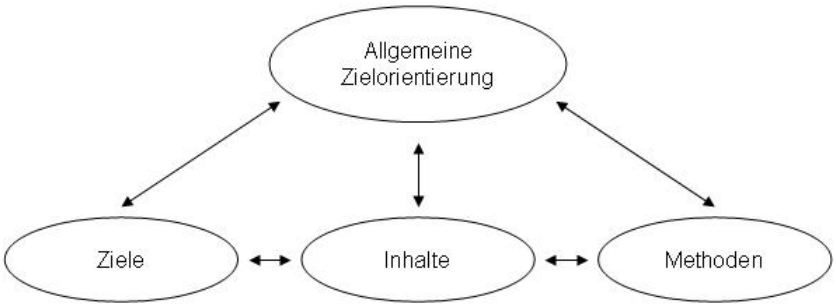


Abbildung 5: Interdependenz der Planungsentscheidungen
Quelle: vgl. [Humb06]

In Abbildung 5 wird das Zusammenspiel der verschiedenen allgemeindidaktischen Planungsdimensionen dargestellt. Für die konkrete Unterrichtsplanung sollten folgende Fragen beantwortet werden:

- Was soll ich unterrichten? – Ziele, Inhalte
- Wie soll ich unterrichten? – Methoden
- Welche Rahmenbedingungen muss ich berücksichtigen?

Für den Informatikunterricht existiert bisher keine fachdidaktisch anerkanntes Vorgehensmodell zur Unterrichtsplanung, vgl. [Humb06]. Daher orientiert sich die Unterrichtsplanung an gängigen Unterrichtsplanungsmodellen die je nach Orientierung den verschiedenen Planungsmodellen zugeordnet sind.
Die von HARTMANN zur Vorbereitung des Informatikunterrichts angegebene Strukturierungshilfe, siehe Abbildung 6, liefert eine pragmatisch ausgerichtete Sicht auf die Planungssituation.

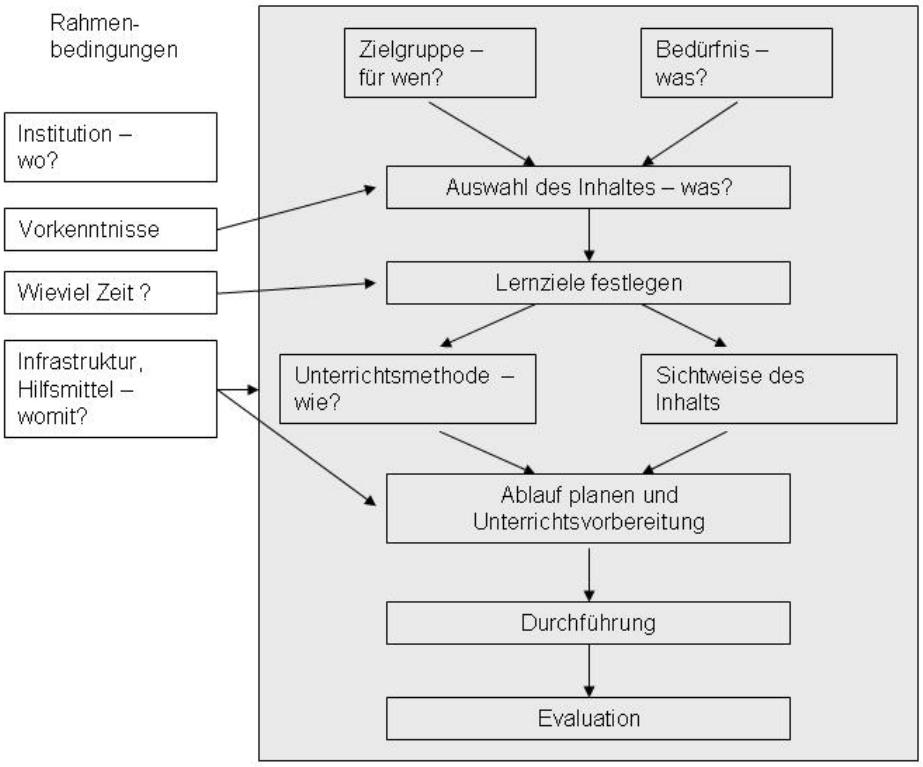


Abbildung 6: Bedingungsgefüge für den Informatikunterricht
nach HARTMANN
Quelle: vgl. [Humb06]

In dem grau hinterlegten Teil des Planungsmusters finden wir ein gängiges didaktisches Modell, das eine nützliche Grundlage für die Planung darstellt. Die Reihenfolge der Planungsschritte wird durch das Schema von oben nach unten sequenziert. Ausgehend von den inhaltlichen Entscheidungen über die Zieldimensionen, methodische und auf die Wissensstruktur abzielende Fragen, anschließende Verlaufsplanung, Durchführung bis hin zur Evaluation.

2.5 Unterrichtsmethodik

Nach [Hubw07] sollte der Unterricht folgende Forderungen erfüllen:

- Erzeugung einer entspannten Arbeitsatmosphäre, in der die Schüler ohne Stoffdruck Zeit haben, auch spezielle Interessen und Bedürfnisse zu befriedigen. Damit sollen Motivation und Aufmerksamkeit der Schüler gefördert und dauerhaft aufrechterhalten werden
- Einordnung der Lerninhalte in größere Sinnzusammenhänge sowie eine deutliche Strukturierung des Stoffes, um den Schülern die Bildung präpositionaler Netzwerke zu ermöglichen
- Förderung einer aktiven Auseinandersetzung mit dem Stoff, wobei unbedingt vor der Präsentation von Lösungen ein ausreichendes Problembewusstsein erzeugt werden muss. Diese Forderungen entstammen den gemeinsamen Erkenntnissen aller gemäßigten konstruktivistischen Lernansätzen, die betonen, dass Wissen vom Lernenden aktiv konsumiert wird
- Anbieten verschiedener Perspektiven und Zugängen zum selben Thema im Sinne von Cognitive Flexibility
- Erzeugung möglichst authentischer Problemsituationen, um mit den Schülern Problemlöseverhalten in einer Umgebung zu trainieren, in der dieses Verhalten tatsächlich auf benötigt wird
- Altersgemäße Darbietung der Lerninhalte nach den Erkenntnissen der Entwicklungspsychologie

2.5.1 Problemorientierung

Eine erfolgversprechende Vermittlung von abstrakten informatischen Lerninhalten erscheint nach [Hubw07] nur dann sinnvoll, wenn durch konkrete, anschauliche Problemstellungen eine erhöhte Aufnahmebereitschaft der Schüler geschaffen wird.

Der Begriff „Problem“ wird folgendermaßen beschrieben:
Ein Problem ist durch drei Komponenten gekennzeichnet:

- Unterwünschter Anfangszustand
- erwünschter Zielzustand
- Barriere, die die Überführung des Anfangzustandes in den Zielzustand im Augenblick verhindert

Die Aufgabe ist folgendermaßen beschrieben:
Bei einer Aufgabe verfügen wir über Regeln (Wissen, Know-how), wie die Lösung erreichbar ist.
Das Hauptgewicht sollte nicht auf das Kennenlernen des technischen Aufbaus von Rechneranlagen oder auf das Erlernen von Programmiersprachen gelegt werden, sondern die Schüler sollten vielmehr eingeführt werden in

- Methoden und Strukturierung, Mathematisierung und Algorithmisierung von Problemkreisen aus verschiedensten Gebieten sowie
- vor allem die Methoden des systematischen Programmierens und
- Möglichkeiten des Einsatzes von Datenverarbeitungssystemen zur Behandlung komplexer Aufgaben (Informationssysteme, heuristische Programmierung, Simulation)

Das Problemlösen mit Informatiksystemen ist nach [Hubw07] mittlerweile eine der allgemein anerkannten Leitlinien der informatischen Bildung. Die Struktur des Problemlöseprozesses stellt deshalb für den Informatikunterricht ein geeignetes Mittel zur Artikulation des Informatikunterrichts dar. Jeder neue Stoff sollte anhand von Problemen aus dem Erfahrungsbereich der Schüler eingeführt werden. Einer Motivationsstufe folgt eine Stufe der Schwierigkeiten, die dann in einer der nächsten Stufen gelöst werden.
Die didaktischen Fähigkeiten des Lehrers zeigen sich dabei in der Auswahl von geeigneten Problemen, deren Komplexität einerseits so hoch sein sollte, dass sie von den Schülern ohne die zu erlernenden Konzepte nicht oder nur unter erheblich höherem Aufwand gelöst werden können, andererseits darf aber der intellektuelle Horizont der Schüler nicht überschritten werden. Probleme die sich durch Anwendung von Strategien oder durch Systemdenken lösen lassen wären optimal.

Eine strikte Problemorientierung kann den Informatikunterricht davon bewahren, in eine reine Produktschulung abzufallen. Die Frage der Schüler sollte daher nicht sein: „Jetzt hab ich ein schönes Werkzeug kennen gelernt, was soll ich nun damit anfangen?“.

2.6 Unterrichtsmethoden

Die nachfolgend angeführten Unterrichtsmethoden, vgl. [@Meth], sind nur ein Auszug einer Vielzahl von verschiedenen Methoden. In dieser Arbeit wurde versucht, traditionelle Methoden aber auch die Methoden mit dem meisten Mehrwert für den Lernenden eines Informatikunterrichts zu finden.

2.6.1 Frontalunterricht, Fragendentwickelnde Methode

Dem **Frontalunterricht** lastet zurzeit ein eher negatives Image an. Wird diese Unterrichtsmethode überwiegend eingesetzt, dann sind die Lernergebnisse eher als unzureichende für die meisten Schüler einzuschätzen.
Der schnellen Informationsdarstellung und der Informationsübermittlung mittels Präsentationen kann diese Methode aber sehr hilfreich sein. Dabei sollte aber darauf geachtet werden, dass diese Präsentationen nicht in einem Frontalunterricht enden.
Präsentationen sollten zeitlich begrenzt sein und eine möglichst offene Form der Informationsweitergabe besitzen welche die Mitarbeit und ein Mitdenken aller Beteiligten anregen und fördern soll.
Bei Präsentationen ist darauf zu achten, dass die Aufmerksamkeit der Teilnehmer besonders bei abstrakten Inhalten sehr schnell nachlassen kann.

Die **darstellend-entwickelnde Methode** setzt sich aus einem Frontalunterricht mit Zwischenfragen zusammen und ist stark lehrerzentriert. Diese Zwischenfragen sollten den Zuhörer über die langen Vortragsphasen etwas aktiver in den Vortrag einbinden.

Ein **fragend-entwickelnder Unterricht** baut auf Fragen der Lehrkraft auf, die von möglichst vielen Schülern beantwortet werden können um im Unterricht voranzukommen. Dabei sind die Ziele und Fragestellungen in der Hand des Vortragenden welcher den gesamten Prozess stark dominiert und kontrolliert.

All diese Methoden sollten sich nur auf eine zeitlich begrenzte Phase des Unterrichts beziehen.

2.6.2 Einzelunterricht, Partnerarbeit, Gruppenarbeit

Der **Einzelunterricht** ist primär eine Methode des individuellen Lernens und Übens und ist vom Lernenden selbst in eigenen Handlungen zu realisieren. Dabei sollte in bestimmten Phasen des Übens und Wiederholens der Schüler selbsttätig werden, um individuell die Inhalte zu verarbeiten.
Die Einzelarbeit sollte folgende Funktion ermöglichen:

- das Behalten von Informationen
- das Sichern und Festigen von Gelerntem
- den Transfer auf Vergleichbares oder auf neue Probleme und Aufgaben

Die **Partnerarbeit** wird hauptsächlich bei Übungszwecken angewandt. Dabei können sich die Partner gegenseitig im Lernprozess unterstützen, helfen, motivieren aber auch kontrollieren. Dabei sollte darauf geachtet werden, dass leistungsheterogene Partnerzusammensetzungen für den Lernerfolg günstiger sind, als leistungshomogene Partnerschaften. Für alle Zusammensetzungen sollten sich aber immer alle Mitglieder an die Teamregeln halten.

In einer **Gruppenarbeit** welche effektiv und qualitätsvoll sein soll, sollten nach [@Blik] nachfolgende Punkte berücksichtigt werden:

- eine eindeutige Zielsetzung, wenn möglich schriftlich oder auf Folie
- klar formulierte Arbeitsaufträge, wobei der Auftrag lauten kann, dass die Kleingruppe eine Entscheidung über ihr Teil-Thema selbst zu treffen hat

- vorbereitete Arbeitshilfen wie zum Beispiel Literatur, Handwerkszeuge, Werkzeuge und Neue Medien
- klar formulierte Vereinbarungen zur "Berichterstattung“ wie zum Beispiel Vortrag, ausgearbeitete Folien, Computer-Präsentationen, Markt der Möglichkeiten, usw.
- klare Vorgaben über Zeit, Raum, Gruppengröße und Gruppenbildung
- Information über die Tätigkeit des Moderators während der Gruppenarbeit

Mit der Gruppenbildung nach [@Blik] wird in der Regel auch eine Entscheidung darüber getroffen, ob die Arbeiten arbeitsgleich oder arbeitsteilig erfolgen sollen. Die Gruppenbildung dazu kann auf ganz verschiedene Weise erfolgen:

- angeordnet – Moderator ordnet an, in welchen Gruppen an den Themen gearbeitet werden soll
- nach Zufall – die Teilnehmer ziehen Karten, zählen ab, …
- nach Interesse oder Neigung – die Teilnehmer organisieren sich bezogen auf ein Thema, ein Problem, ...
- nach Sympathie – die Teilnehmer organisieren sich nach Sympathie
- heterogen oder homogen – (Moderator oder Teilnehmer ordnen nach Leistung, Geschlecht, Jahrgang, Schulform, ...

Die jeweiligen Vorteile und Nachteile sind auf den Zweck der Gruppenarbeit hin zu bedenken. Handelt es sich im Unterricht zum Beispiel um eine Maßnahme die fachliche Leistung zu fördern, so kann eine Einteilung in leistungshomogene Gruppen sinnvoll sein. Soll aber etwa die Sozialkompetenz oder die Kommunikations- und Kooperationsfähigkeit geschult werden, so kann eine heterogene Einteilung viel nützlicher sein.

Bei der Durchführung einer Gruppenarbeit nach [@Blik] in Kleingruppen kann es hilfreich sein, wenn die Rollen der einzelnen Mitglieder festgelegt werden. So besteht die Möglichkeit, dass zwar alle gleichberechtigt an dem Projekt arbeiten, was aber nicht ausschließt, dass ein Gruppenmitglied die Gesprächsführung übernimmt, ein anderes Mitglied das Protokoll und wieder ein anderer die Präsentationsarbeit

übernimmt. Weiters bietet es sich an, dass jedes einzelne Gruppenmitglied ein Log-Buch oder Lerntagebuch führt und die Arbeiten dokumentieren.
In diesem Buch können unter anderem die fachlichen Erkenntnisse, der Erkenntnisprozess sowie die dabei erlebten Gefühle niedergeschrieben werden. Weiters könnten dabei folgende Fragen gestellt und beantwortet werden:

- Was habe ich in dieser Stunde gemacht?
- Was habe ich erreicht bzw. Neues gelernt?
- Wie ist es mir dabei ergangen?
- Bin ich mit dem heutigen Ergebnis zufrieden?

Während der Gruppenarbeit entsteht in den Kleingruppen ein spezifisches Wissen. Daher sollte nach der Gruppenarbeit das gemeinsame Wissen zwischen den einzelnen Kleingruppen ausgetauscht werden. Dieser Austausch kann durch Gruppenberichte oder Gruppenpräsentationen erfolgen. Dabei sollte aber darauf geachtet werden, dass bei zu vielen Berichten oder Präsentationen in einer Lerneinheit, diese schnell zu Langeweile bei den Zuhörern führen kann.
Nach der Gruppenarbeit sollte hin und wieder insbesondere über die Vorbereitung und Durchführung der Gruppenarbeit kommuniziert werden. Dabei sollte auf die Inhalts- aber auch auf die Beziehungsebene geachtet werden. So kann unter anderem die Kompetenz zur Teamarbeit gesteigert werden. Diese Art der Kommunikation soll bewusst machen, wo Defizite liegen oder was beim nächsten Mal verbessert werden kann oder worauf beim nächsten Mal besonderer Wert gelegt werden sollte.

Bei der individuellen Leistungsbewertung nach [@Blik] der einzelnen Schüler, sind diese in der Regel nicht damit zufrieden, dass nach einer Gruppenarbeit alle mit derselben Note bewertet werden. Die Schüler selbst, wie auch der Lehrer, wissen, dass nicht alle Mitglieder der Kleingruppe denselben fachlichen Arbeitseinsatz geleistet haben und nicht alle in derselben positiven Weise kommunikativ und kooperativ waren.

In [Trot06] werden weitere Strategien beschrieben wie in einem Team gearbeitet werden kann. Dabei gibt es die Möglichkeiten, dass die Gruppenmitglieder einzeln arbeiten, dass eine Gruppe von drei von einander unabhängigen Mitgliedern Projekte umsetzt, oder als Paar abwechselnd arbeiten. Die Anzahl von drei Gruppen-Mitgliedern hat sich in der Praxis als besonders günstig herausgestellt da sich dadurch nur schwer Untergruppen in der Gruppe bilden können.

Pure Teamwork:

Nur ein Problem wird vom Team in Angriff genommen. Die drei Team-Mitglieder arbeiten ausschließlich als Team. Es gibt verschiedene Möglichkeiten die Arbeit unter den Team-Mitgliedern aufzuteilen. Eine Möglichkeit wäre es, zwei Mitglieder zusammen am Computer das Programm erstellen zu lassen wobei das dritte Mitglied den Überblick über das Projekt halten und zum Testen des Programms eingesetzt werden könnte.

No-Teamwork:

In dieser Art der Zusammenarbeit löst jedes Team-Mitglied individuell seine Probleme. Die parallele Problemlösung ist effizienter als wenn die Probleme sequenziell abgearbeitet werden. Nur wenn es die Problemstellung erfordert, arbeiten mehrere Mitglieder zusammen. So kann Effizienz gesteigert und die Kommunikation und etwaige Debatten über das Programm reduziert werden.

Paired Methods:

Diese Art der Zusammenarbeit ist eine mögliche Alternative zu den beiden zuvor angeführten Methoden der Teamarbeit. Zwei Probleme können zur selben Zeit gelöst werden, wobei ein Team-Mitglied das eine Problem am Computer arbeitet, während die anderen Beiden den Lösungsweg des anderen Problems am Papier vorskizzieren.

2.6.3 Erkundung

Die Erkundung hat nach [@Meth] das Ziel, den Lernenden aus dem gewohnten Lernumfeld herauszuführen. Durch das Erkunden an nicht-schulischen oder künstlich gestalteten Lernorten wird versucht, die Wirklichkeit, so wie sie in einer Praxis oder Lebenswelt tatsächlich ist, direkt und möglichst mit allen Sinnen zu erfahren.
Ein zuvor theoretisch erlerntes Wissen kann vom Schüler vor Ort eigenständig überprüft und mit Erkundungserfahrungen verknüpft werden. Der Unterschied zur Exkursion besteht darin, dass keine vorgegebene Aufgabenstellung abgearbeitet wird. Der Schüler bestimmt bei der Erkundung selbständig den Erkundungsinhalt und übernimmt alle anfallenden organisatorischen Aufgaben. Die Methode ist für alle Altersstufen geeignet.

2.6.4 Freiarbeit

Die Freiarbeit basiert nach [@Meth] auf den Gedanken eines offenen Unterrichts. Bei dieser Lernmethode setzt sich der Schüler ein Lernziel, das er in Eigenleistung und mit einem hohen Maß an persönlicher Freiheit zu erreichen versucht. Er selbst trifft die Entscheidung über die Auswahl der Arbeitsformen, Inhalte sowie die Planung der Aktivitäten. Somit ist nicht mehr der Lehrer für Planungs- und Entscheidungsprozesse verantwortlich, sondern übernimmt die Rolle eines Beraters und Begleiters für den Schüler. Die hierarchischen Strukturen der Beziehung Lehrer-Schüler können dadurch teilweise aufgelöst werden.
Ziel der freien Arbeit ist es, das selbstverantwortliche Lernen und die sozialen Interaktionen zu fördern. Sie ermöglicht es dem Schüler, eigene Lernwege zu finden und die eigene Lernbiographie selbst zu gestalten.
Freies Arbeiten bedarf einer langsamen und gut geleiteten Einführung, um den Schüler, aber auch den Lehrer, nicht mit der neuen Verantwortung zu überfordern. Um die Ziele dieser Lernmethode erreichen zu können, ist eine sorgfältige Vorbereitung hinsichtlich der Her- und Bereitstellung von Arbeitsmitteln, Inhalten, der Gestaltung des Klassenraums sowie ein kooperatives Verhalten im Kollegium unbedingt notwendig.

2.6.5 Stationslernen

Beim Stationslernen nach [@Meth] sind an verschiedenen Positionen im Raum, den so genannten „Lernstationen“, Arbeitsaufträge unterschiedlicher Art bereitgestellt, die nacheinander vom Schüler bearbeitet werden. Diese Aufträge stehen in einem thematischen Zusammenhang, können aber in der Regel unabhängig voneinander und in unterschiedlicher Reihenfolge bearbeitet werden. Dadurch erhält der Schüler die Möglichkeit, seinen Lernweg entsprechend seiner Interessen und Fähigkeiten selbst zu steuern. Unterschiede im Lernverhalten einzelner Schüler können so leichter miteinander vereinbart werden.
Im Stationslernen kann durch die Art und Weise der verschiedenen Arbeitsaufträge die unterschiedlichen Zugänge zum Lernstoff ermöglicht werden. Alle Sinneskanäle lassen sich durch die verschiedenen Aufgabenstellungen ansprechen. Auch direktes Handeln kann durch gezielte Aufforderungen für Entscheidungen bei der Aufgabenbearbeitung gefördert werden. Diese Lernmethode weist dem Schüler eine aktive und verantwortungsvolle Rolle innerhalb des Lernprozesses zu.
Stationslernen wird besonders empfohlen zur Vertiefung von nicht ganz neuem Wissen (Lernziel „Kennenlernen“), zur Einübung (Lernziel „Beherrschen“) und im Rahmen von fächerübergreifendem Unterricht. Die vielen Vorzüge werden allerdings nur durch einen hohen Material- und Vorbereitungsaufwand erreicht.

2.6.6 Blitzlicht

Die Lernmethode Blitzlicht ist nach [@Meth] ein rasches Feedback, das die Stimmung, Meinung, den Stand bezüglich der Inhalte und Beziehungen in einer Gruppe ermitteln kann. Die Schüler äußern sich kurz mit ein, bzw. wenigen Sätzen zu einem klar abgegrenzten Thema. Das sich daraus ergebene Bild der Blitzlichtrunde kann helfen, die Arbeitssituation positiv zu gestalten und lösungsorientiert zu verändern.

2.6.7 Mind-Mapping

Mind-Mapping nach [@Meth] ist eine Arbeitsmethode, die ein flexibles, kreatives und gehirngerechtes Arbeiten ermöglichen soll. Diese Methode wurde von Tony Buzan in den 1970er Jahren auf der Grundlage von gehirnphysiologischen Hypothesen entwickelt.

Mind-Mapping versucht Notizen handschriftlich und unter Verwendung von Symbolen, Skizzen und Grafiken bunt und übersichtlich darzustellen. Im Gegensatz zur klassischen linearen Struktur der Aufzeichnungen, ist die Mind-Map eine auf den ersten Blick übersichtliche „Karte“, die das zentrale Thema sofort erkennbar machen soll. Im Zentrum der Mind-Map steht das Thema woraus sich anschließend alle anderen Gedanken verzweigen.

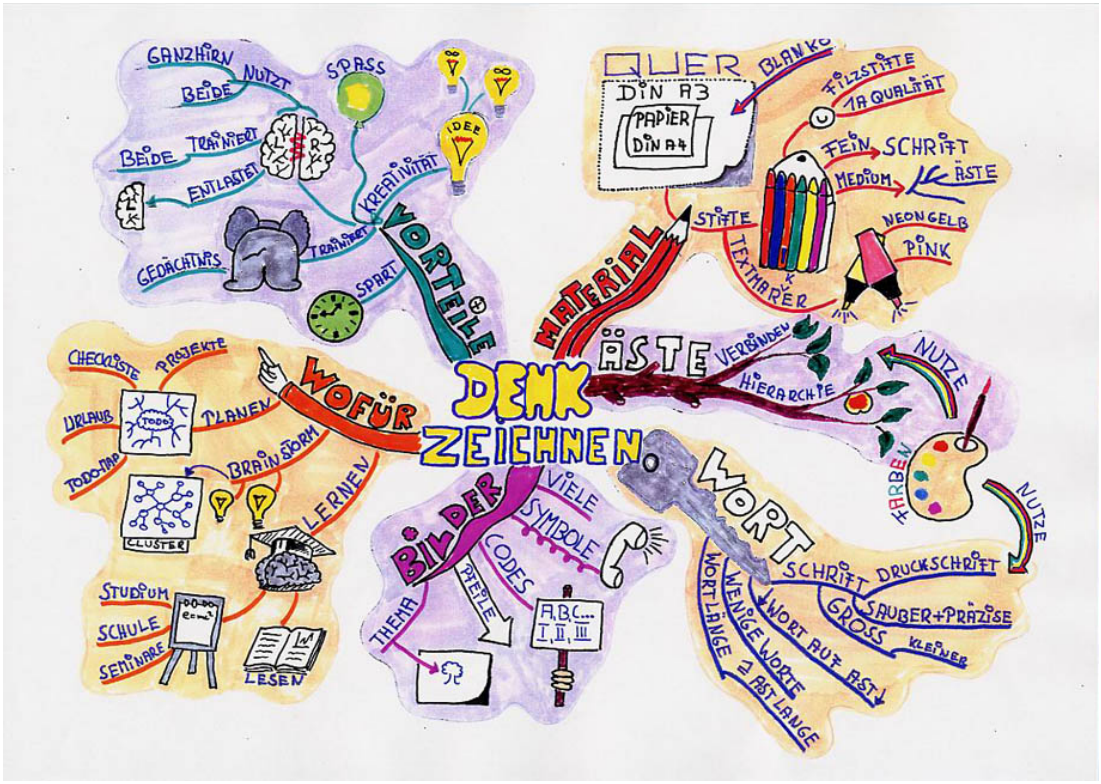


Abbildung 7: Mind-Map
Quelle: vgl. [@Denk]

2.6.8 Quiz und Rätsel

Die Methode Quiz und Rätsel sind nach [@Meth] nicht nur auf die Didaktik beschränkt. Mit dem Rätsel verbinden sich Jahrhunderte alte Traditionen und diente überwiegend der belustigenden Beschäftigung und hatte nichts mit Unterricht zu tun. Im strikten Frontalunterricht war dies lange Zeit nicht erwünscht. Erst in jüngerer Zeit kamen Quiz und Rätsel zum Einsatz und sollten noch besser für den Unterricht entwickelt werden.
Eine Definition zu Rätsel und Quiz:

> *„Das Rätsel ist die meist umschreibende Bezeichnung von Begriffen oder Inhalten, die der Hörer oder Leser finden soll.*
> *[@Meth]*

2.6.9 Wandzeitung

Die Wandzeitungen nach [@Meth] sind neben großen Ausstellungen eine Möglichkeit, Arbeitsprozesse im Lernen zu dokumentieren, wobei der Arbeitsprozess selbst sichtbar gemacht werden kann, aber auch dessen Ergebnisse präsentiert werden können. Mit der Wandzeitung lassen sich inhaltliche Arbeiten aber auch Beziehungsarbeiten darstellen. Weiters ist es möglich, Zwischen- und Endergebnisse eines Lehrprozesses zu dokumentieren.

2.6.10 Team Teaching

Das Team Teaching ist nach [@Meth] eine Lehrmethode, bei der zwei oder auch mehr Personen gemeinsam Schüler unterrichten. Die Methode ist besonders geeignet, um den Unterricht mit mehr Perspektivenvielfalt, größerer Methodenvielfalt und unterschiedlichen Anregungen zu erweitern, da die Fixierung auf einen Vortragenden verhindert wird.
Für Lehrer bietet Team Teaching die Möglichkeit, den Unterricht gemeinsam mit anderen zu planen und zu gestalten. Während des Unterrichts können Erfahrungen ausgetauscht, und die Vortragenden im Unterricht entlastet werden.

Team Teaching bezieht sich nicht nur auf Lehrer-Lehrer Interaktion, sondern auch auf die Lehrer-Schüler-Interaktionen. Dabei sollen die Lehrer und die Schüler im Team handeln.

2.7 Erlernschwierigkeiten beim Programmieren

Programmieren ist nach [Teag07] eine komplexe und schwierige Aufgabe in der sich Schüler in den ersten Anfängen immer wieder besonders abmühen. Dabei spielen mehrere verschiedene Faktoren zusammen, wie zum Beispiel schwerverständliche Programme, das Zusammenspiel zwischen Programm und Computer, der Syntax und die Semantik der Programmiersprache, eigenes Unwissen Probleme lösen zu können oder bestehenden Programmcode lesen zu können.
Ein Ansatz ist das kollaborativ, gemeinsame lernen. Dabei sind Schüler nicht nur für sich selbst verantwortlich, sondern auch für die anderen Schüler in der Gruppe. Dabei lernen Schüler von und mit anderen Schülern, sind interaktiv beschäftigt, können andere Schüler beim Lösen der Probleme beobachten und davon lernen und achten gegenseitig darauf, dass die beim Thema bleiben.
Ein Beispiel für kollaboratives Lernen ist das Pair Programming (Paarprogrammierung) welches ein Bestandteil des Extrem Programming ist. Dabei arbeiten immer zwei Programmierer an einem Rechner zusammen. Der eine Programmierer schreibt den Code, während der andere koordiniert, die Problemstellung im Auge behält und etwaige Fehler sofort anspricht. Die beiden Programmierer sollten immer wieder die Rollen abwechseln sowie auch die Zusammensetzung der Paare sollte immer wieder getauscht werden.
Weitere Ausführungen zur Gruppenarbeit siehe Punkt 2.6.2 Einzelunterricht, Partnerarbeit, Gruppenarbeit.

Nach [Pric07] wird versucht, eine besonders umfangreiche Lernumgebung zur Verfügung zu stellen. Dabei sollen die Schüler ihr eigenes Wissen und ihr eigenes Verständnis für das Erstellen von Programmen entwickeln können.

Der dynamische Prozess des Lernens wird von einer Vielzahl von Ansätzen beeinflusst, im Speziellen ist der „learning cycle“, ein interaktiver Kreis von

- experimentieren
- erhalten einer konkreten Erfahrung / Sachkenntnis
- Überlegungen aus diesen Erfahrungen
- Wissen zu abstrahieren, zu verallgemeinern

Nach [Ahma07] können Schüler schon zu Beginn eines Programmierunterrichts entmutigt sein, da sie erwarten, eine neue Art zu denken und eine neue Art zu sprechen lernen zu müssen. Weiters empfinden die Schüler den Programmierunterricht und die Erklärungen des Lehrers über Programmcode als eine Art Fremdsprache. Die Terminologie, die Fachausdrücke zu verstehen ist schwierig aber auch Analogien bei den Schülern zu bestehenden Wissen herzustellen ist nicht einfach, da die Schüler nur wenig Erfahrungen mit ähnliche Szenarien haben.
Es gibt zwei Möglichkeiten, den Programmierunterricht noch effektiver und effizienter zu gestalten:

- Lernen aus Fehlern
- Debugging

Beim Lernen aus Fehlern wird dem Schüler aufgezeigt, was an ihrem Programmcode nicht richtig ist. Dabei werden die Fehler welche beim Schreiben des Programms entstanden sind, also Syntax Fehler, aufgelistet. Der Schüler kann nun den Syntaxfehler beheben und daraus lernen.
Beim Debugging wird versucht, den Schülern den Programmablauf Schritt für Schritt zu zeigen. Der Debugger ist ein Werkzeug mit dem der Schüler sehr gut sehen kann, wie der Computer durch die Programmcodezeilen springt.

An „The Open University, Walton Hall, Milton Keynes, England“ wird nach [Grif07] versucht, die zu programmierenden Objekte zuvor als grafische Elemente darzustellen. Dabei wird zwar die objektorientierte Programmierung angesprochen, die Art und Weise des Unterrichtens kann aber auch auf die sequenzielle Programmierung angewendet werden.

Um Objekte besser verstehen zu können, werden diese als Objekte des realen Lebens dargestellt. Damit sollten den Schülern die Schwierigkeiten mit den abstrakten Objekten und Begriffen der Programmiersprache genommen werden.
Auch in der sequenziellen Programmierung sollte der Lehrer immer wieder versuchen, Bezüge zu bereits für die Schüler bekanntem herzustellen. Damit können die Schüler die neuen Informationen besser aufnehmen und besser mit Ihrem eigenen Wissen vernetzen.

III Der Lehrplan

3.1 Lehrplan der höheren Lehranstalt für Informationstechnologie

Die nachfolgende Darstellung der Stundentafel liegt dem Lehrplan vgl. [@Lehr] der höheren Lehranstalt für Informationstechnologie, Schulformkennzahl 8490, Jahrgang I zugrunde.

LEHRPLAN DER HÖHEREN LEHRANSTALT FÜR INFORMATIONSTECHNOLOGIE

Schulformkennzahlen:
8490 I. und II. Jahrgang
8491 III. bis V. Jahrgang – Netzwerktechnik
8492 III. bis V. Jahrgang – System- und Informationstechnik
8493 III. bis V. Jahrgang – Internet- und Medientechnik

3.2 Stundentafel

I. STUNDENTAFEL
(Gesamtstundenzahl und Stundenausmaß der einzelnen Unterrichtsgegenstände)

A.	**Pflichtgegenstände**	**Wochenstunden**						**Lehrverpflichtungsgruppe**
		Jahrgang						
		I.	**II.**	**III.**	**IV.**	**V.**	**Σ**	
1.	Religion	2	2	2	2	2	10	(III)
2.	Deutsch	2	2	2	2	2	10	(I)
3.	Englisch	2	2	2	2	2	10	(I)
4.	Geschichte und politische Bildung	-	-	-	2	2	4	III
5.	Leibesübungen	2	2	2	1	1	8	(IVa)

6.	Geographie und Wirtschaftskunde	2	2	-	-	-	4	(III)
7.	Wirtschaft und Recht	-	-	-	3	2	5	III
8.	Angewandte Mathematik	4	3	3	2	2	14	(I)
9.	Angewande Physik	2	2	-	-	-	4	(II)
10.	Angewandte Chemie und Ökologie	2	2	-	-	-	4	II
11.	Angewandte Programmierung	4(2)	3(2)	2(2)	-	-	9	I
12.	Grundlagen der Elektrotechnik und der Elektronik	2	2	2	-	-	6	I
13.	Grundlagen der Informatik	3(3)	2	-	-	-	5	I
14.	Medientechnik	2(2)	2(2)	2(2)	-	-	6	I
15.	Betriebswirtschaft	-	2	2	-	-	4	II
16.	Betriebssysteme und Computerarchitektur	-	2	4(2)	-	-	6	I
17.	Netzwerktechnik	-	2	2	3(2)	-	7	I
18.	Datenbanksysteme	-	-	2	2(2)	-	4	I
19.	Projekt und Projektmanagement	-	-	4(2)	4(4)	5(5)	13	II
20.	Qualitätsmanagement	-	-	-	-	2	2	II
21.	Computerpraktikum	4(4)	4(4)	-	-	-	8	IVa
	Pflichtgegenstände der schulautonomen Ausbildungsschwerpunkte gemäß Abschnitt B	-	-	5	13	14	32	
	Gesamtwochenstundenzahl	33	36	36	36	34	175	

Tabelle 8: Stundentafel
Quelle: vgl. [@Lehr]

3.3 Pflichtgegenstand „11. Angewandte Programmierung“

Aus dem Lehrplan der höheren Lehranstalt für Informationstechnologie, Schulformkennzahl 8490 wird aus dem Jahrgang I der Pflichtgegenstand 11. Angewandte Programmierung [@Lehr] entnommen und die Lehrinhalte dargestellt.

Punkt 11. ANGEWANDTE PROGRAMMIERUNG

Bildungs- und Lehraufgabe:

Der Schüler/die Schülerin soll

- den Computer als Werkzeug für fachspezifische Anwendungen einsetzen und mit Hilfe einer
- höheren technischen Programmiersprache Aufgaben seines Fachgebietes lösen können;
- komplexe Probleme analysieren und für das Programmieren aufbereiten und möglichst in Projektform (Gruppenarbeit) lösen können;
- systematisch an eine Problemlösung herangeführt werden und die Bedeutung von Algorithmen kennen lernen;
- algorithmisches Denken lernen und die daraus resultierenden Darstellungen (Pseudocode) in eine höhere Programmiersprache codieren können;
- Programmbausteine (Komponenten) aus externen Bibliotheken in eigene Programme integrieren können;
- die fachbezogenen Vorschriften und Normen anwenden.

Lehrstoff:

I. Jahrgang:

Einstieg in die Programmierung:

Einsatz einer visuellen Oberfläche, Anwendungen

Algorithmus:

Trennung Algorithmus, Programmierung, Codierung; schrittweise Verfeinerung, Darstellung von Algorithmen

Systemprogramme:

Interpreter, Compiler, Fehlerarten

Programmiersprachenelemente:

Anweisungsfolge, Verzweigung, Wiederholung

Einfache Datentypen:

Ganzzahl, Gleitkommazahlen und Zeichen

Modularisierung:

Eingabe, Verarbeitung und Ausgabe mit den jeweiligen Übergabeparametern

Zusammengesetzte Datentypen:

Felder, Zeichenketten, Strukturen, Klassen

Dateiverarbeitung:

Ein-/Ausgabe

IV Lehrstoffverteilung und Stundenbilder

4.1 Arbeitsumgebung

Für die erstellten Übungsbeispiele wird Microsoft Visual Basic for Applications (VBA) der Produkte von Microsoft Office 2003 verwendet.
Nach [Pool07] leitet sich die Skriptsprache VBA von der Programmiersprache Visual Basic (VB) ab und dient zur Steuerung von Programmabläufen. VBA ist der Nachfolger zu den verschiedenen Makro-Sprachen in den jeweiligen Microsoft Office Produkten.

Wie in [Whit07] beschrieben, produzieren Programmieranfänger mit Prozeduraler Programmierung die besseren Ergebnisse als mit einer objektorientierten Programmierung. Mit einer Prozeduraler Programmierung kann das Problem besser von oben nach unter, also top – down, in kleine Schritte aufgeteilt werden und somit vom Schüler besser verstanden werden. Damit können auch die Programme von oben nach unter erstellt werden, was für die Anfänger mehr Sinn ergibt.
Der objektorientierte Ansatz mit seinen Objekten, Attributen, Events ect. ist zwar besser an die reale Welt angelehnt, für Programmieranfänger aber eher schwieriger zu durchschauen.
Einer der Gründe, warum in dieser Arbeit die Beispiele mit VAB erstellt wurden.

Natürlich stehen andere Alternativen zur Verfügung wie zum Beispiel, Java, C++ oder C#. Bei diesen Programmiersprachen ist allerdings der Lehraufwand höher als bei VBA. Die Schüler müssten viel mehr Zeit in die Erstellung der Programme investieren und würden daher eher demotiviert, sich mit der Programmierung auseinander zu setzten. Weiters müsste den Schülern vor dem Beginn der Programmierung das Konzept der Objektorientierung vermittelt werden.

Mit der Skriptsprache VAB ist die Entwicklung von Programmen im Vergleich dazu relativ einfach. Mit simplen Mitteln können relativ komplexe Probleme gelöst werden. Daher sind die Schüler eher motiviert, sich mit dem Thema Programmieren zu beschäftigen, da mit wenig Programmieraufwand ein relativ hoher Output erreicht werden kann.

Wie in [AlFu07] dargestellt, sollte versucht werden den Stress für Lehrer aber auch für Schüler so gering wie möglich zu halten. Dabei haben die Teilnehmer in der Studie folgende Gründe für die Entstehung von Stress folgendermaßen beschrieben:

- die benötigte Zeit für die Vorbereitung, Erklärung, Installation und Reparatur der Hardware
- Probleme mit der Anwendung sowie der Zuverlässigkeit von Hard- und Software, Fehlermeldungen, Kompatibilitätsprobleme
- Fehlende Schulungen für Lehrer für die Administration von PC's
- Fehlender technischer Support durch speziell geschulte Administratoren

Ein vorrangiges Ziel sollte sein, schon die Entstehung von Stress zu vermeiden und so eine möglichst angenehme und stressfreie Arbeitsumgebung für Lehrer und Schüler zu schaffen.

4.2 Lehrstoffverteilung Semester I

Nachfolgend wird die Lehrstoffverteilung für die ersten 15 Wochen des ersten Semesters dargestellt.

Woche	**Lehrstoff**	**Anmerkung**
1	Kennen lernen der Infrastruktur – Hardware und Software der Schule	
2	Interpreter/Compiler/Fehlerarten Kennen lernen der Programmoberfläche	Visual Basic for Applications
3	Algorithmus/Programmieren Unterschied Darstellung von Algorithmen	Struktogramm / PAP
4	Ganzzahlen Zeichen Algorithmus/Programmieren	Einfache Datentypen
5	Verzweigung/Wiederholung Algorithmus/Programmieren	Kontrollstruktur IF
6	Verzweigung/Wiederholung Algorithmus/Programmieren	Kontrollstruktur Schleifen
7	Zwischenüberprüfung	
8	Verbesserung der Zwischenüberprüfung	
9	Modularisieren Algorithmus/Programmieren	Unterprogramm
10	Zeichenketten Algorithmus/Programmieren	String Funktionen
11	2 Wochenprojekt Algorithmus/Programmieren	Schüler schlagen Projekte vor
12	2 Wochenprojekt Algorithmus/Programmieren	Präsentieren der Projekte
13	Grafische Elemente Algorithmus/Programmieren	UserForms
14	Semesterprüfung	
15	Verbesserung der Semesterprüfung Semester Ausklang	

Tabelle 9: Stoffverteilung Semester I

4.3 Stundenbilder Semester I

Nachfolgend werden für die 15 Wochen des ersten Semesters des ersten Jahrgangs die Unterrichtseinheiten in den einzelnen Stundenbildern dargestellt.
Jede Unterrichtseinheit auf dem Stundenbild stellt eine Doppelstunde von 90 Minuten dar. Weiters werden die Lehreinheiten detaillierter beschrieben, Struktogramme und Programmablaufpläne sowie der Programmcode der Beispiele im Anschluss der Stundenbilder angeführt.
Die Art und Weise der Darstellung der Stundenbilder hat sich in der Praxis bewährt und wird daher in dieser Form übernommen.
Die in eckigen Klammern dargestellten Methoden können alternativ verwendet werden [alternative Methode]. Diese Alternativen werden ebenfalls im Anschluss der Stundenbilder näher beschrieben.

Semester I – Einheit 1:

Inhalt	Didaktik	Methode	Ziel	Medien	Zeit
Vorstellung des Lehrers	Lehrer stellt sich selbst vor	Frontalunterricht Rückfragen	Schüler kennen den Lehrer	PC + Beamer	10’
Infrastruktur der Schule vorstellen	Lehrer stellt die vorhandene Hard- Software Infrastruktur der Schule vor; Zeigt den Schülern die Druckerstandorte	Frontalunterricht Erkundung	Schüler kennen das Haus und den Hausgebrauch	PC + Beamer In der Schule vorhandenen Drucker – Scanner – usw.	30’
Schüler melden sich am Schulnetzwerk an und ändert das Passwort	Einzelaufgabe für den Schüler	Einzelaufgabe	Schüler können sich mit Benutzernamen und Passwort am Netwerk anmelden	Netzwerk PC + Beamer Tafel	15’
Vorstellen der Ordnerstruktur des Schülernetzwerkes	Lehrer stellt die Ordnerstruktur den Schülern vor Wo sind Vorlagen zu finden Wie erfolgt elektronische Abgabe der Übungen	Einzelaufgabe	Schüler kennen die Ordnerstruktur des Schulnetzwerkes und können damit arbeiten Können Übungen vom Serververzeichnis holen und Aufgaben am Server abgeben	Netzwerk PC + Beamer Tafel	35’

Semester I – Einheit 1:

Der Vortragende stellt sich mittelst Präsentation über Beamer bei den Schülern vor, und geht auf etwaige Rückfragen ein.
Der Vortragende stellt das Schulnetzwerk vor und zeigt den Schülern beim Rundgang durch die Schule, wo sich die einzelnen Drucker, Scanner, usw. physikalisch befinden.
Nach dem Rundgang durchs Schulgebäude werden die Schüler aufgefordert, sich mit Ihrem Benutzernamen im Netzwerk anzumelden und anschließend das Passwort zu ändern. Nach der erfolgten Anmeldung der Schüler am Schulnetzwerk, stellt der Vortragende die Ordnerstruktur am Schulserver vor. Dabei wird erläutert, in welchem Ordner sich die Übungsaufgaben befinden, und in welchen Ordnern die Schüler die erledigten Übungen am Server wieder abzulegen haben.
Kurze Übungen zur Erstellung von Ordnern sind möglich.

Bei der Einzelarbeit kann der Vortragende jeden Schüler genauer beim Umgang mit dem Computer beobachten. Der Lehrer kann daher etwaige positive aber auch negative Auffälligkeiten notieren oder gegebenenfalls korrigierend eingreifen.

Semester I – Einheit 2:

Inhalt	Didaktik	Methode	Ziel	Medien	Zeit
Interpreter	Lehrer stellt den Begriff „Interpreter“ vor	Frontalunterricht Rückfragen [Erkundung]	Schüler können den Begriff „Interpreter“ erklären	PC + Beamer	15’
Compiler	Lehrer stellt den Begriff „Compiler“ vor	Frontalunterricht Rückfragen [Erkundung]	Schüler können den Begriff „Compiler“ erläutern	PC + Beamer	15’
Unterschiede Interpreter / Compiler	Unterschiede zw. Interpreter/Compiler mittels Internet erar-beiten	Gruppenarbeit 2 Schüler je Gruppe [Erkundung]	Schüler können im Internet recherchieren und Unterschiede Erarbeiten	Netzwerk Internet	15’
Pause					5’
Unterschiede Interpreter / Compiler	Einige Schüler präsentieren Ihre Ausarbeitungen	Blitzlicht	Schüler können die Unterschiede zwischen Interpreter und Compiler aufzäh-len		15’
VBA – Programmier – Oberfläche	Lehrer stellt den VBA Editor den Schüler vor	Frontalunterricht Rückfragen	Schüler können den VBA – Editor in den Grundzügen erklären und bedienen	PC + Beamer Netzwerk	25’

Semester I – Einheit 2

Problemorientierter Ansatz: Wir wollen ein geschriebenes Programm (Programm-Code) am Computer ablaufen lassen. Welche verschiedenen Möglichkeiten gibt es, Programmcode im Prozessor auszuführen.

Interpreter:
ist vergleichbar mit einem Dolmetscher.
Ist ein Software-Programm, welches den Quellcode einliest, analysiert und ausführt. Daher erfolgt die Analyse des Quellcodes erst zur Laufzeit des Programms.
Nachteil: langsamere Ausführungsgeschwindigkeit im Gegensatz zum Compiler
Vorteil: Programme können auf verschiedenen Rechnerarchitekturen ausgeführt werden, wenn das Interpreter-Programm auf diesen Architekturen lauffähig ist
Beispiele: BASIC; Perl; Python; Ruby; PHP; Skriptsprachen (wie z. B. Javascript); Java; C#; Perl; Python

Compiler:
ist vergleichbar mit einem Übersetzter.
Ist ein Software-Programm, welches den Quellcode vor dem Ausführen in ein ausführbares Programm (Bytecode, Maschinensprache) übersetzt.
Vorteil: Fehler werden vor der Programmausführung aufgezeigt
Nachteil: Programm muss nach jeder Änderung am Quellcode neu übersetzt/kompiliert werden
Beispiel: Pascal; C

Gruppenarbeit / Blitzlicht
Schüler erarbeiten (auch mittels Hilfe des Internets) die Unterschiede zwischen Interpreter und Compiler und präsentieren Ihre Ergebnisse den Mitschülern

VBA – Programmier – Oberfläche / VBA Editor:
Oberfläche siehe 6.1 Microsoft Visual Basic for Applications Editor
Dient zum erstellen der VBA – Programme

Vorstellen der verschiedenen Fenster; Projektfenster, Eigenschaftenfenster, Direktbereich, Code-Fenster,
Beschreiben der Funktionen der einzelnen Fenster

Beim Frontalunterricht kann der Vortragende auf Rückfragen reagieren, sodass alle Schüler im Klassenraum die Beantwortung mitbekommen. Mittels der Methode Blitzlicht, können einige Schüler ihr erarbeitetes Wissen der gesamten Klasse mitteilen. Dabei kann der Vortragende relativ rasch kontrollieren, ob der Lehrstoff von den Schülern richtig verstanden wurde.

Alternative Methode:
Eine alternative Methode wäre die Erkundung des Themas Compiler/Interpreter. Die Schüler könnten verschiedene Medien wie zum Beispiel Internet aber auch die schuleigene Bibliothek nutzen, um sich mit dem Thema vertraut zu machen. Dabei sollten sich die Schüler selbstständig Wissen aneignen. Diese Wissen sollten sie anschließend in einer kurzen Präsentation den Mitschülern weitergeben.
Mit dieser Methode würden die Schüler mehr zur Selbstständigkeit hingeführt werden.
Ein Vorteil dieser Methode wäre, dass sich jeder Schüler die Art und Weise der Wissensaneignung selbst aussuchen kann. Weiters können sich die Schüler mehr oder weniger frei in der Schule bewegen um sich an den verschiedenen Lernbehelfen wie zum Beilspiel Internet oder Schulbibliothek zu bedienen.
Nachteile könnten sein, dass sich die Lernenden nicht nur mit den vorgegebenen Themen beschäftigen, sondern die Zeit für andere Beschäftigungen wie zum Beispiel Chatten im Internet oder mit ihrem Handy verschwenden. Daher solle der Lehrer die Schüler immer im Auge behalten, auch wenn diese über mehrere Räume der Schule verteilt sind.

Semester I – Einheit 3:

Inhalt	Didaktik	Methode	Ziel	Medien	Zeit
Programmieren Definition	Lehrer erklärt was „Programmieren“ ist	Frontalunterricht Rückfragen	Schüler können „Programmieren“ erklären	PC + Beamer	10’
Algorithmus Definition	Lehrer erklärt was ein „Algorithmus“ ist	Frontalunterricht Rückfragen	Schüler können „Algorithmus“ erklären	PC + Beamer	10’
Algorithmus Darstellung	Lehrer erläutert die Darstellungs-Möglichkeiten von Algorithmen	Frontalunterricht Rückfragen	Schüler kennen die Möglichkeiten der Darstellungen	PC + Beamer	10’
Struktogramm	Station I: Struktogramm Darstellungselemente	Stationslernen Wandzeitung [Gruppenarbeit]	Schüler können Struktogramm – Elemente darstellen und erklären	PC + Beamer Papier und Stifte für Wandzeitungen	30’
PAP – Programmablaufplan	Station II: PAP Darstellungselemente	Stationslernen Wandzeitung [Gruppenarbeit]	Schüler können PAP – Elemente darstellen und erklären	PC + Beamer Papier und Stifte für Wandzeitungen	30’

Semester I – Einheit 3:

Problemorientierter Ansatz: Seitenlanger Programmcode ist am Bildschirm schwer zu lesen. Welche Möglichkeiten gibt es, Programmcode komprimiert am Papier darzustellen.

Programmieren Definition:
Die Tätigkeit Computerprogramme, sprich Software, zu erstellen, zu programmieren. Der Algorithmus wird in ein Softwareprogramm umgesetzt.

Algorithmus Definition:
Der Algorithmus ist eine abstrakte, allgemeine Handlungsvorschrift zur Lösung eines Problems in endlich vielen Schritten. Beispiel: Kochrezept, Gebrauchsanweisungen

Struktogramm Darstellung:
Zum Erstellen der Struktogramme wurde das Programm „Vips“ der Webseite vgl. [@Part] verwendet.
Dieses Programm eignet sich bestens zum Erstellen und zum Ausführen kleiner Programme.
Einen Auszug der Darstellungselemente von Struktogrammen sind unter 6.2 Struktogramm – Flussdiagramm aufgelistet.

PAP Darstellung:
Zum Erstellen der Flussdiagramme wurde das Programm „Fludi“ vom Autor Thomas Schaller verwendet. Der download dieses Programms wurde von der Webseite vgl. [@Ziem] durchgeführt.

Wandzeitung:
Die Schüler erstellen eine Wandzeitung der verschiedenen Darstellungselemente von Struktogrammen und PAPs. Die Ergebnisse werden in der Klasse diskutiert. Die grafisch anspruchvollsten Wandzeitungen welche von den Schülern gewählt wurden, werden im Klassenraum ausgestellt und dienen immer wieder als Grundlage für die Darstellung weitere Programme.

Alternative Methode:
Sollte das Stationslernen nicht umsetzbar sein, wäre eine Alternative die Gruppenarbeit. Dabei können sich die Schüler selbst zu Gruppen von zwei bis drei Mitschülern zusammenfinden, und die beiden Darstellungsarten von Programmcode erarbeiten.

Ein Vorteil dieser Methode wäre, dass kein zweiter Vortragender für das Stationslernen von Nöten wäre. Ein Lehrer wäre für die Schüler ausreichend. Das eher aufwändige vorbereiten der Stationen würde entfallen.
Nachteile könnten sein, dass die Schüleranzahl der Gruppen zu groß wird und daher nicht mehr effektiv gearbeitet wird, oder sich die Schüler nicht genügend mit dem Thema auseinandersetzen.

Semester I – Einheit 4:

Inhalt	Didaktik	Methode	Ziel	Medien	Zeit
Datentypen	Lehrer erläutert die unterschiedlichen Datentypen	Frontalunterricht Rückfragen	Schüler können die Datentypen aufzählen und erklären	PC + Beamer	10’
Darstellung von Werten im PC	Exkurs in die Welt von 0 und 1	Frontalunterricht Rückfragen	Schüler kennen die interne Verarbeitungs-art des PC	PC + Beamer	10’
Ganzzahlen – Integer Operatoren	Lehrer erläutert den Datentyp „Integer“ + Operatoren	Frontalunterricht Rückfragen [Stationslernen]	Schüler können den Datentyp „Integer“ + Operatoren erklären	PC + Beamer	10’
Zeichenkette – String Operatoren	Lehrer erläutert den Datentyp „String“ + Operatoren	Frontalunterricht Rückfragen [Stationslernen]	Schüler können den Datentyp „String“ + Operatoren erklären	PC + Beamer	15’
Beispiel P01-Datentypen Anforderungen vor-stellen	Lehrer stellt die An-forderungen für das Beispielprogramm vor	Frontalunterricht Rückfragen	Schüler verstehen die Anforderungen	PC + Beamer	10’
Algorithmus erstellen für P01-Datentypen	Schüler erstellen Algorithmus	Gruppenarbeit 2 Schüler je Gruppe	Algorithmus	PC + Beamer Netzwerk	10’
Programm erstellen für P01-Datentypen	Schüler erstellen Programm	Gruppenarbeit 2 Schüler je Gruppe	Programmcode	PC + Beamer Netzwerk	15’
Musterlösung	Lehrer stellt die Musterlösung vor	Frontalunterricht Rückfragen	Schüler können die Musterlösung erläutern	PC + Beamer Netzwerk	10’

Semester I – Einheit 4:

Problemorientierter Ansatz: Wie können Informationen wie Zahlen, Texte, Bilder oder Sounddateien am Computer gespeichert oder dargestellt werden, wenn der Computer nur mit Nullen und Einsen umgehen kann.

Datentypen: Aufzählen und erläutern der wichtigsten Datentypen

Darstellung von Werten am PC: Erläutern wie Daten am PC und somit im Prozessor verarbeitet werden; Darstellung aller Werte und Texte in Nullen und Einsen

Ganzzahlen Integer: Wertebereiche definieren; Darstellung am PC
Zeichenkette String: Wertebereiche definieren; Was ist Leer-String
vgl. [@Msdn]

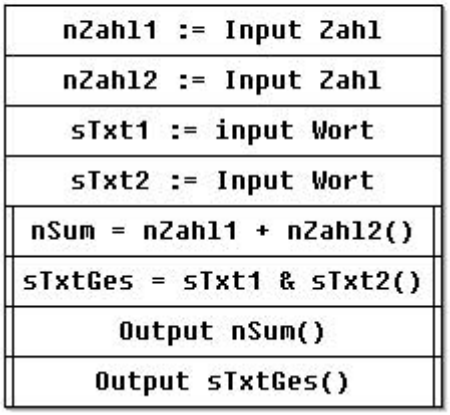
Der Frontalvortrag ist zeitlich gesehen relativ lang. Daher ist darauf zu achten, dass die Schüler mittels fragend-entwickelnder Methode immer wieder beim Thema gehalten werden. Die Schüler sollten damit angeregt werden, aktiv am Unterricht mitzuwirken.
Durch den Frontalunterricht kann der Lehrer besser auf Fachbegriffe eingehen, welche von den Schülern von Beginn an richtig verwendet werden sollen. Weiters ist sichergestellt, dass sich die Schüler bei diesem grundlegenden Thema irrtümlich nichts Falsches einlernen. Wenn möglich, sollte der Vortragende kleine Pausen einplanen.
Danach erstellen die Gruppen zu je zwei Schülern selbstständig den Algorithmus und das Programm. Dabei sollte der Vortragende berücksichtigen, dass dies die ersten Gehversuche der Schüler in Sachen Algorithmus und Programmerstellung sind. Etwaige Hilfestellungen sollten über den Beamer aber auch direkt bei den Schülergruppen am Computer erfolgen. Dabei kann sich der Lehrer erste Bilder über den unterschiedlichen Wissensstand der Schüler machen und diese bei der nächsten Gruppenzusammenstellung berücksichtigen.

Eine alternative Methode wäre das Stationslernen.
Vorteile dabei, die Gruppe der Schüler wäre kleiner und der Vortragende könnte besser auf das Thema und auf die Schüler der Gruppe eingehen.
Ein Nachteil ist, dass zwei Vortragende benötigt werden, und der eher doch themenübergreifende Stoff exakt getrennt werden müsste.

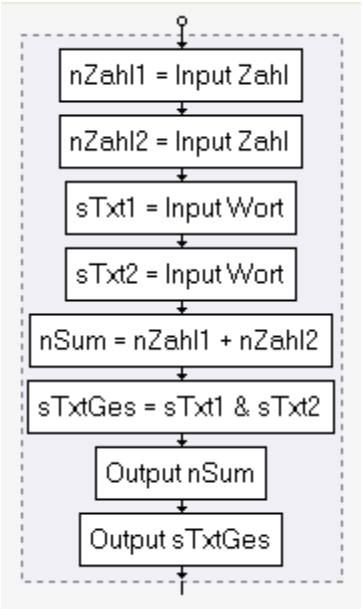
Programm P01-Datentypen Anforderung:
Schüler erstellen das Programm welches zur Eingabe zweier Zahlen und zweier Zeichenketten auffordert. Alle vier Werte werden in entsprechenden Variablen gespeichert. Nach erfolgter Dateneingabe, werden die beiden Zahlenwerte addiert, und die beiden Zeichenketten mittels Sting-Verkettung aneinander gehängt. Danach werden die beiden neuen Werte am Bildschirm ausgegeben.
Falscheingaben werden nicht berücksichtigt.

Am Ende der Lehrveranstaltung sollte der Lehrer seinen Schülern eine Musterlösung vorstellen. Dabei ist aber darauf zu achten, dass diese Lösung nur als Muster und nicht als richtige Lösung dienen soll. Es sollte ein vorrangiges Ziel jeden Vortragenden sein, dass jeder Schüler seinen eigenen Stil im erstellen von Programmen entwickeln darf.

Struktogramm:



Programmablaufplan:



Programm:

```
Option Explicit
'
'Programm: P01-Datentypen
'
Sub P01_Datentypen()
'Deklarieren der Variablen
Dim nZahl1 As Integer
Dim nZahl2 As Integer
Dim nSum As Integer
Dim sTxt1 As String
Dim sTxt2 As String
Dim sTxtGes As String

    'Dateneingabe
    nZahl1 = InputBox("erste Zahl eingeben:")
    nZahl2 = InputBox("zweite Zahl eingeben:")
    sTxt1 = InputBox("erstes Wort eingeben")
    sTxt2 = InputBox("zweites Wort eingeben")

    'Berechnung
    nSum = nZahl1 + nZahl2


    'String Verkettung
    sTxtGes = sTxt1 & " " & sTxt2

    'Datenausgabe
    MsgBox "Die Summe der beiden Zahlen ist " & nSum
    MsgBox "Die beiden Wörter ergeben " & sTxtGes

End Sub
```

Semester I – Einheit 5:

Inhalt	Didaktik	Methode	Ziel	Medien	Zeit
Wiederholung Datentypen	Lehrer wiederholt mit Schüler die Datentypen	Fragen entwickelnde Methode Quiz	Festigen des Stoffs Datentypen	PC + Beamer	15’
Kontrollstruktur Verzweigung mittels Schlüsselwort IF	Lehrer erläutert die Kontrollstruktur mittels dem Schlüsselwort IF	Frontalunterricht Rückfragen	Schüler können die Kontrollstruktur mittels dem Schlüsselwort IF anwenden	PC + Beamer	15’
Beispielprogramm P02-If Anforderungen vorstellen	Lehrer stellt die Anforderungen für das Beispielprogramm vor	Frontalunterricht Rückfragen	Schüler verstehen die Anforderungen	PC + Beamer	10’
Algorithmus erstellen für P02-If	Schüler erstellen Algorithmus	Gruppenarbeit 2 Schüler je Gruppe	Algorithmus	PC + Beamer Netzwerk	20’
Programm erstellen für P02-If	Schüler erstellen Programm	Gruppenarbeit 2 Schüler je Gruppe	Programmcode	PC + Beamer Netzwerk	20’
Musterlösung	Lehrer stellt die Musterlösung vor	Frontalunterricht Rückfragen	Schüler können die Musterlösung erläutern	PC + Beamer Netzwerk	10’

Semester I – Einheit 5:

Problemorientierter Ansatz: Wir haben in einem Programm zwei verschieden große Zahlen. Wir wollen die höhere Zahl am Bildschirm ausgegeben.

Am Beginn des Unterrichts wird mittels Quiz der Lehrstoff aus der Vorwoche wiederholt. Dabei ist es auch durchaus denkbar, dass auch Schüler Fragen an Schüler stellen und somit nicht ein reines Frage – Antwort Spiel zwischen dem Lehrer und den Schülern entsteht.

Die Kontrollstruktur Verzweigung wird in der Programmiersprachen VBA mit dem Schlüsselwort IF bzw. END IF als Anweisungsblock gebildet. Dabei wird zwischen der einseitigen und zweiseitigen Auswahl unterschieden.

Einseitig Auswahl:

```
IF <Bedingung> THEN
      <Aktion1>
END IF
```

Zweiseitige Auswahl:

```
IF <Bedingung> THEN
      <Aktion1>
ELSE
      <Aktion2>
END IF
```

Der Vortragende erarbeitet mit den Schülern gemeinsam verschiedene Beispiele, welche eine IF-Abfrage benötigen. Dabei sollte darauf geachtet werden, dass auch die Schüler ihre Beispiele der gesamten Klasse erläutern, denn dann kann der Lehrer wenn nötig korrigierend eingreifen.
Vor der Vorstellung der Programmanforderung sollten die Schüler zu Gruppen von je zwei Schülern zusammengestellt werden. Die Gruppen können heterogen nach Leistung zusammengestellt werden. Dadurch besteht die Möglichkeit, dass bessere Schüler ihren nicht so guten Gruppenpartner unterstützten.

Programm P02-Datentypen Anforderungen:

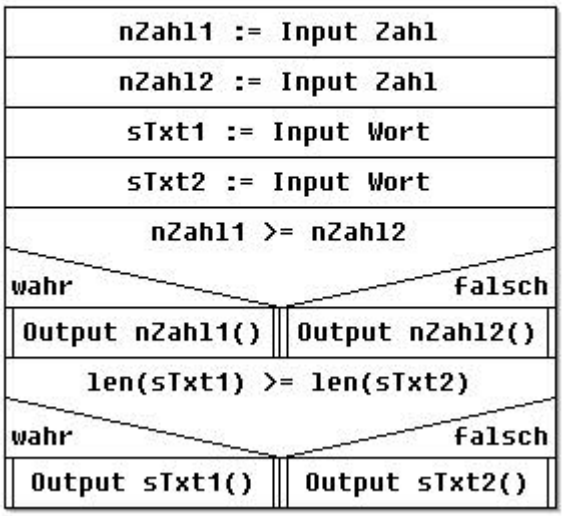
Schüler erstellen das Programm welches zur Eingabe zweier Zahlen und zweier Zeichenketten auffordert.

Die beiden Zahlen werden mittels dem Vergleichsoperator <= (kleiner oder gleich) oder >= (größer oder gleich) verglichen. Die vom Wert her größere Zahl wird am Bildschirm ausgegeben.

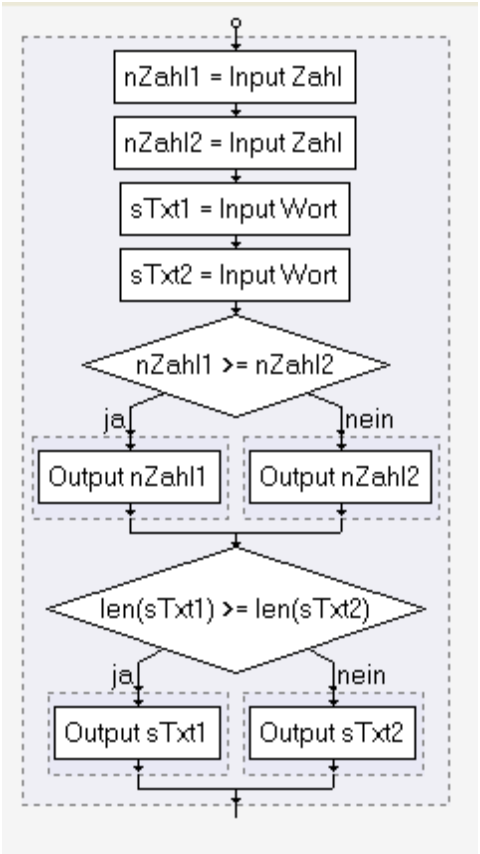
Die Zeichenanzahl beider Zeichenketten werden mittels dem Vergleichsoperator <= (kleiner oder gleich) oder >= (größer oder gleich) verglichen. Die längere Zeichenkette wird am Bildschirm ausgegeben.

Der Zustand, dass beide Zahlen denselben Wert haben, oder die beiden Zeichenketten gleich lang sind, wird in diesem Programm nicht behandelt.

Struktogramm:



Programmablaufplan:



Programm:

```
Option Explicit
'
'Programm: P01-IF
'
Sub P02_IF()
'Deklarieren der Variablen
Dim nZahl1 As Integer
Dim nZahl2 As Integer
Dim sTxt1 As String
Dim sTxt2 As String

    'Dateneingabe
    nZahl1 = InputBox("erste Zahl eingeben:")
    nZahl2 = InputBox("zweite Zahl eingeben:")
    sTxt1 = InputBox("erstes Wort eingeben")
    sTxt2 = InputBox("zweites Wort eingeben")

    'Zahlen vergleichen
    If nZahl1 >= nZahl2 Then
        MsgBox nZahl1
    Else
        MsgBox nZahl2
    End If

    'String vergleichen
    If Len(sTxt1) >= Len(sTxt2) Then
        MsgBox sTxt1
    Else
        MsgBox sTxt2
    End If

End Sub
```

Semester I – Einheit 6:

Inhalt	Didaktik	Methode	Ziel	Medien	Zeit
Wiederholung Verzweigung IF	Lehrer wiederholt mit Schüler die Verzwei-gung IF	Fragen entwickelnde Methode	Festigen des Stoffs Verzweigung IF	PC + Beamer	10’
Kontrollstruktur Schleifen FOR-NEXT DO-WHILE-LOOP DO-LOOP-WHILE	Lehrer erläutert die Kontrollstruktur Schleifen	Stationslernen [Frontalunterricht]	Schüler können die Kontrollstruktur Schleifen anwenden	PC + Beamer	25’
Beispielprogramm P03-Schleifen Anforderungen vor-stellen	Lehrer stellt die Anforderungen für das Beispielprogramm vor	Frontalunterricht Rückfragen	Schüler verstehen die Anforderungen	PC + Beamer	10’
Algorithmus erstellen für P03-Schleifen	Schüler erstellen Algorithmus	Gruppenarbeit 2 Schüler je Gruppe	Algorithmus	PC + Beamer Netzwerk	15’
Programm erstellen für P03-Schleifen	Schüler erstellen Programm	Gruppenarbeit 2 Schüler je Gruppe	Programmcode	PC + Beamer Netzwerk	20’
Musterlösung	Lehrer stellt die Musterlösung vor	Frontalunterricht Rückfragen	Schüler können die Musterlösung erläutern	PC + Beamer Netzwerk	10’

Semester I – Einheit 6:

Problemorientierter Ansatz:
Was muss ich tun, um mit einem Hammer einen Nagel einzuschlagen. Wie lange muss ich den Nagel treffen, damit er vollständig eingeschlagen ist vgl. [@IHEA].

Nach erfolgter Wiederholung sollten wenn möglich die drei Stationen für die jeweilige Schleife aufgebaut werden. Dabei sollten die Schüler versuchen, für jede Schleifenart Beispiele zu finden. Diese Beispiele werden danach mit den Schülern diskutiert. Dabei können Schüler mit einem bereits sehr gutem Grundwissen die Vortragenden unterstützen, anderen Schülern den Lehrstoff zu vermitteln.
Eine alternative Methode wäre der Frontalunterricht wenn kein weiterer Vortragender zur Verfügung steht.

Die Kontrollstruktur Schleifen werden in der Programmiersprache VBA auf drei unterschiedliche Arten dargestellt vgl. [@VbaH]:

For-Next – Schleife (Zählschleife):

```
For nSchleifenZähler = nVon To nBis
      <Aktion>
Next nSchleifenZähler
```

Die Aktion wird von der nVon-Zahl bis zur nBis-Zahl wiederholt. In der Zahl nSchleifenZähler wird die Zahl der aktuellen Wiederholung gespeichert.

Kopfgesteuerte Schleife:

```
Do While <Bedingung>
      <Aktion>
Loop
```

Die Aktion wird so oft wiederholt, bis die Bedingung Wahr ist.

Fußgesteuerte Schleife:

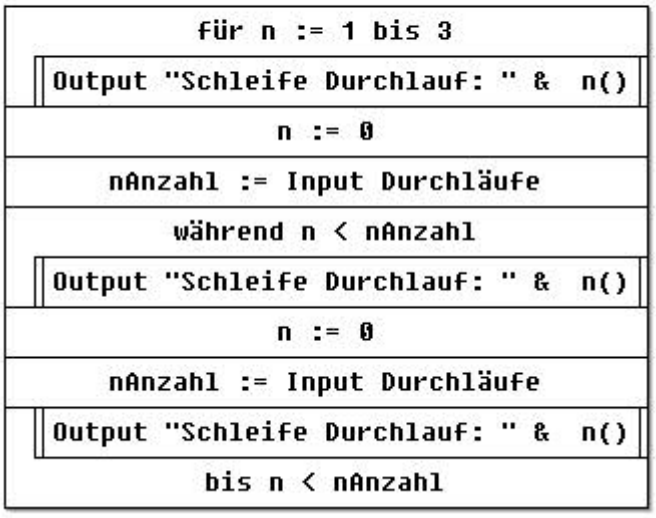
```
Do
      <Aktion>
Loop While <Bedingung>
```

Die Aktion wird mindestens einmal, und anschließend so oft wiederholt, bis die Bedingung Wahr ist.

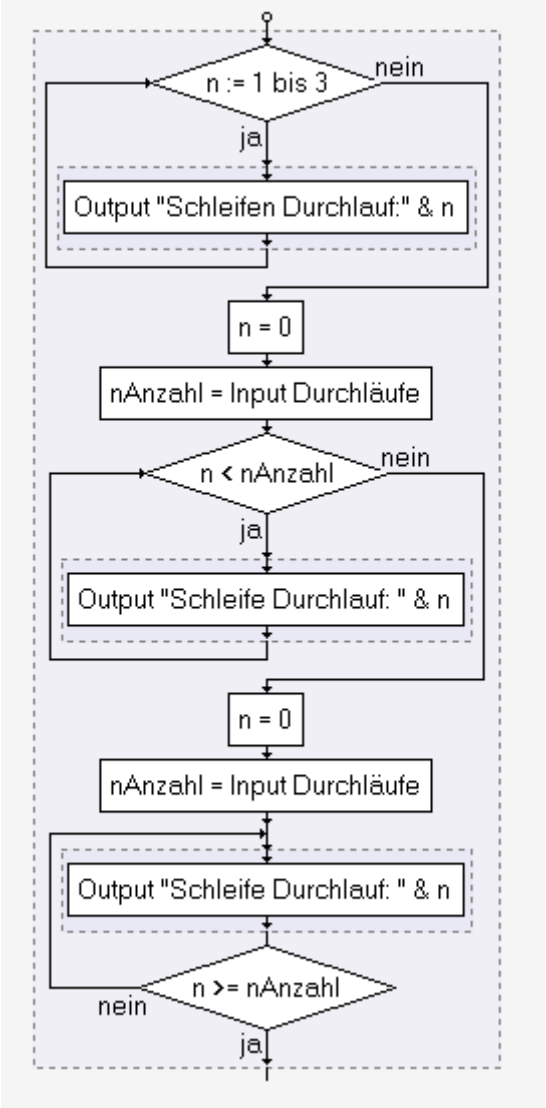
Programm P03-Schleifen Anforderungen:
Die Meldung „Schleife Durchlauf: n“ soll genau drei Mal am Bildschirm erscheinen, wobei die Variable n die Zahl des aktuellen Schleifendurchlaufes anzeigen soll.
Danach erfolgt eine Eingabeaufforderung (Inputbox) für den Benutzer, welcher die gewünschte Anzahl der Schleifendurchläufe [0-5] eingeben kann. Die Meldung „Schleife Durchlauf: n“ soll so oft angezeigt werden, wie dies der Benutzer zuvor eingegeben hat, wobei die Variable n die Zahl des aktuellen Schleifendurchlaufs anzeigen soll.
Zuletzt sollte eine Eingabeaufforderung für den Benutzer erfolgen, welcher die gewünschte Anzahl der Schleifendurchläufe [0-5] eingeben kann. Diese Meldung „Schleife Durchlauf: n“ soll so oft angezeigt werden, wie dies der Benutzer zuvor eingegeben hat, mindestens aber einmal, auch wenn der Benutzer den Wert Null eingegeben hat. Die Variable n zeigt die Zahl des aktuellen Schleifendurchlaufes.

Zu realisieren ist dieses Programm mittels Schleifen und ohne IF-Abfragen. Nicht behandelt werden Falscheingaben sowie der Abbruch der Eingabe.

Struktogramm:



Programmablaufplan:



Programm:

```
Option Explicit
'
'Programm: P03-LOOPS
'
Sub P03_LOOPS()
'Deklarieren der Variablen
Dim nAnzahl As Integer
Dim n As Integer


    'FOR Schleife
    For n = 1 To 3
        MsgBox "FOR - Schleife Durchlauf: " & n
    Next n

    'Kopfgesteuerte Schleife
    n = 0
    nAnzahl = InputBox("Wie viele Durchläufe [0-5]?")

    Do While n < nAnzahl
        n = n + 1
        MsgBox "Kopfgesteuerte Schleife - Durchlauf: " & n
    Loop

    'Fußgesteuerte Schleife
    n = 0
    nAnzahl = InputBox("Wie viele Durchläufe [0-5]?")

    Do
        n = n + 1
        MsgBox "Fußgesteuerte Schleife - Durchlauf: " & n
    Loop Until n >= nAnzahl

End Sub
```

Semester I – Einheit 7:

Inhalt	Didaktik	Methode	Ziel	Medien	Zeit
Zwischenüberprüfung Vorstellen der Fragen	Lehrer stellt die Fragen der Zwischenüberprüfung vor	Frontalunterricht Rückfragen	Schüler verstehen die Fragen der Zwischenüberprüfung	PC + Beamer	10’
Zwischenüberprüfung	Schüler werden auf Ihr Wissen überprüft	Einzelarbeit	Schüler geben Ihr erworbenes Wissen wieder	Test	80’

Semester I – Einheit 7:

Zwischenüberprüfung vorstellen der Fragen: Der Lehrer stellt die Fragen der Zwischenüberprüfung vor. Dabei wird überprüft, ob alle Schüler die gestellten Fragen verstehen.

Zwischenüberprüfung: Die Schüler erarbeiten selbstständig in Einzelarbeit die Fragen der Zwischenüberprüfung. Nach erfolgter Abgabe des Fragebogens verlassen die Schüler den Klassenraum.

Fragebogen: siehe Anhang 6.3 Fragen der Zwischenüberprüfung Semester I

Semester I – Einheit 8:

Inhalt	Didaktik	Methode	Ziel	Medien	Zeit
Rückgabe und besprechen der Zwischenüberprüfung	Lehrer gibt die korrigierte und benotete Zwischenüberprüfung an die Schüler zurück	Frontalunterricht Rückfragen	Schüler bekommen Ihre korrigierte Zwischenüberprüfung	Korrigierte und benotete Zwischenüberprüfung	20’
Zwischenüberprüfung nochmals erarbeiten	Schüler erarbeiten in Gruppen die Zwischenüberprüfung nochmals	Gruppenarbeit 2 Schüler je Gruppe	Alle Schüler verstehen alle gestellten Fragen der Zwischenüberprüfung	PC + Beamer Netzwerk	50’
Musterlösung	Lehrer stellt die Musterlösung vor	Frontalunterricht Rückfragen	Schüler können die Musterlösung erläutern	PC + Beamer Netzwerk	20’

Semester I – Einheit 8:

Rückgabe der benoteten Zwischenüberprüfung an die Schüler.

Anschließend erarbeiten die Schüler in Gruppenarbeit die Zwischenüberprüfung nochmals. Die richtigen aber auch falschen Antworten sollten in der gesamten Klasse diskutiert werden. Damit ist sichergestellt, dass jeder Schüler alle Fragen beantworten kann. Auf diese Weise sollte der Lehrstoff nochmals wiederholt und gefestigt werden.

Semester I – Einheit 9:

Inhalt	Didaktik	Methode	Ziel	Medien	Zeit
Modularisieren von Programmen SUB FUNCTION	Lehrer erläutert die Bedeutung von Unterfunktionen	Frontalunterricht Rückfragen [Erkundung]	Schüler können Unter-funktionen anwenden	PC + Beamer	20’
Beispielprogramm P04-Unterfunktion Anforderungen vorstellen	Lehrer stellt die Anforderungen für das Beispielprogramm vor	Frontalunterricht Rückfragen	Schüler verstehen die Anforderungen	PC + Beamer	10’
Algorithmus erstellen für P04-Unterprogramm	Schüler erstellen Algorithmus	Einzelarbeit [Gruppenarbeit]	Algorithmus	PC + Beamer Netzwerk	25’
Programm erstellen für P04-Unterprogramm	Schüler erstellen Programm	Einzelarbeit [Gruppenarbeit]	Programmcode	PC + Beamer Netzwerk	25’
Musterlösung	Lehrer stellt die Musterlösung vor	Frontalunterricht Rückfragen	Schüler können die Musterlösung erläutern	PC + Beamer Netzwerk	10’

Semester I – Einheit 9:

Problemorientierter Ansatz: Die Funktion für die Bildschirmausgabe sollte nur einmal im Programmcode geschrieben werden und es sollte sichergestellt werden, dass immer das gleiche Layout bei der Ausgabe am Bildschirm verwendet wird. Layout – Änderungen sollten mit möglichst geringem Aufwand durchgeführt werden können.

Für die Modularisierung von Programmen mittels Unterfunktionen stehen zwei Möglichkeiten zur Verfügung:
Unterprogramm ohne Rückgabewert

```
Sub UnterProgramm01 ([Übergabewerte])
      <Anweisungen>
End Sub
```

Unterprogramm mit Rückgabewert (Zahlenwert oder Textwert):

```
Function UnterProgramm02 ([Übergabewerte]) Rückgabewert
      <Anweisungen>
      UnterProgramm02 = Rückgabe
End Function
```

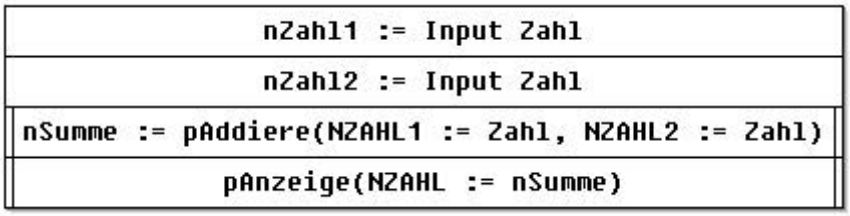
Beim Frontalunterricht kann der Vortragende auf die wichtigen Punkte bei diesem doch eher komplexen Thema der Modularisierung eingehen. Als Alternative wird hier die Erkundung angeführt. Dabei sollte der Vortragende aber eine gewisse Richtung für die Recherche der Schüler vorgeben.
Den Algorithmus und das Programm sollten die Schüler in Einzelarbeit erstellen. Bei Schülern mit noch nicht so ausgeprägten Programmierkenntnissen wäre es auch denkbar, einzelne heterogene Wissensgruppen zu bilden. Ziel aber sollte es sein, dass jeder für sich selbst vorab versucht, den Algorithmus und das Programm zu erstellen.

Programm P04-Unterprogramm Anforderungen:

Der Benutzer soll zur Eingabe zweier Zahlen aufgefordert werden. Danach werden die beiden Zahlen in einem Unterprogramm addiert und die Summe an die aufrufende Stelle im Programm wieder zurückgegeben. Danach wird die Summe der beiden Zahlen an ein Unterprogramm übergeben und anschließend dem Benutzer am Bildschirm angezeigt.

Struktogramm:

Main-Programm

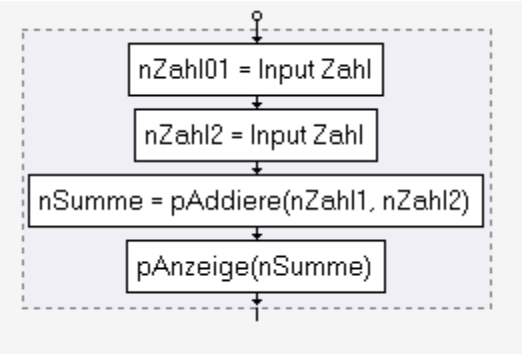


Unterprogramm pAddiere

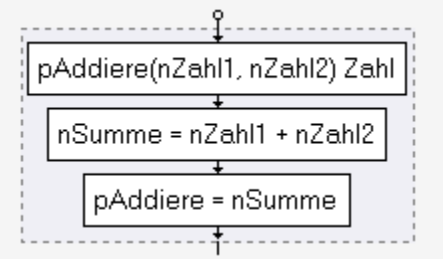
Unterprogramm pAnzeige

Programmablaufplan:

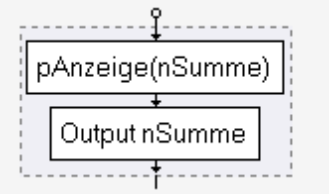
Main Programm



Unterprogramm pAddiere



Unterprogramm pAnzeige



Programm:

```
Option Explicit
'
'Programm: P04-Unterprogramm
'
Sub P04_Unterprogramm()
'Deklarieren der Variablen
Dim nZahl1 As Integer
Dim nZahl2 As Integer
Dim nSumme As Integer

    'Dateneingabe
    nZahl1 = InputBox("erste Zahl eingeben:")
    nZahl2 = InputBox("zweite Zahl eingeben:")

    'Unterprogramm pAddiere
    nSumme = pAddiere(nZahl1, nZahl2)
    'Unterprogramm pAnzeige
    pAnzeige nSumme

End Sub

'Unterprogramm
Function pAddiere(nZahl1 As Integer, nZahl2 As Integer) As Integer
Dim nSumme As Integer

    'Zahlen addieren
    nSumme = nZahl1 + nZahl2
    pAddiere = nSumme

End Function

'Unterprogramm
Sub pAnzeige(nSumme As Integer)

    'Summe anzeigen
    MsgBox nSumme

End Sub
```

Semester I – Einheit 10:

Inhalt	Didaktik	Methode	Ziel	Medien	Zeit
String-Funktionen Right, Left, Mid, Instr, UCase, Len, Trim	Lehrer erläutert die String-Funktionen zum Bearbeiten von Zeichenketten	Blitzlicht [Frontalunterricht]	Schüler können die String-Funktionen anwenden	PC + Beamer	25’
Beispielprogramm P05-String Anforderungen vorstellen	Lehrer stellt die Anforderungen für das Beispielprogramm vor	Frontalunterricht Rückfragen	Schüler verstehen die Anforderungen	PC + Beamer	10’
Algorithmus erstellen für P05-String	Schüler erstellen Algorithmus	Einzelarbeit	Algorithmus	PC + Beamer Netzwerk	20’
Programm erstellen für P05-String	Schüler erstellen Programm	Einzelarbeit	Programmcode	PC + Beamer Netzwerk	20’
Musterlösung	Lehrer stellt die Musterlösung vor	Frontalunterricht Rückfragen	Schüler können die Musterlösung erläutern	PC + Beamer Netzwerk	10’
Hausübung bis zur nächsten Einheit	Lehrer fordert Schüler auf, eine Projektthema zu überlegen	Hausübung	Schüler können in der nächsten Einheit ein Thema vorschlagen	PC + Beamer	5’

Semester I – Einheit 10:

Problemorientierter Ansatz: Wir möchten den Namen aus einer eMail-Adresse extrahieren, damit wir eine persönliche Anrede im Text erstellen können.
Es wird angenommen, dass die eMail-Adresse folgendermaßen aufgebaut ist:
Max.Mustermann@muster.at

Stringfunktionen vgl. [@VbaH]:

Right / Right$:

```
Teilstring = Right(Zeichenkette, länge)
```

Gibt einen Wert vom Typ Variant (String) zurück, der eine bestimmte Anzahl von Zeichen von der rechten Seite (dem Ende) einer Zeichenfolge enthält.

Left / Left$:

```
Teilstring = Left(Zeichenkette, länge)
```

Gibt einen Wert vom Typ Variant (String) zurück, der eine bestimmte Anzahl von Zeichen ab dem ersten (linken) Zeichen einer Zeichenfolge enthält.

Mid:

```
string = Mid(string, start[,length])
```

Gibt einen Wert vom Typ Variant (String) zurück, der eine bestimmte Anzahl von Zeichen aus einer Zeichenfolge enthält.

Instr:

```
pos = InStr(start, string1, string2, comp)
```

Gibt einen Wert vom Typ Variant (Long) zurück, der die Position des ersten Auftretens einer Zeichenfolge innerhalb einer anderen Zeichenfolge angibt.

Tim / LTrim / RTrim:

```
Zeichenkette = Trim(Zeichenkette)
```

Gibt einen Wert vom Typ Variant (String) zurück, der eine Kopie einer bestimmten Zeichenfolge enthält, die keine führenden Leerzeichen (LTrim), keine nachgestellten Leerzeichen (RTrim) sowie keine Kombination aus führenden und nachgestellten Leerzeichen (Trim) enthält.

UCase:

```
GROSS-ZEICHENKETTE = UCase(Zeichenkette)
```

Gibt einen Wert vom Typ Variant (String) zurück, der eine bestimmte, in Großbuchstaben umgewandelte Zeichenfolge enthält.

Len:

```
num = Len(Zeichenkette)
```

Gibt einen Wert vom Typ Long zurück, der die Anzahl der Zeichen in einer Zeichenfolge enthält.

Die Unterrichtsmethode Blitzlicht könnte so verwendet werden, dass Schüler in Kleingruppen, drei bis vier Schüler, eine Stringfunktion zugewiesen bekommen oder sich eine Stringfunktion aussuchen können, und anschließend diesen Betriff unter der Verwendung der VBA-Hilfe beschreiben und erklären. Danach können einzelne Gruppen den Begriff der Klasse erläutern.
Der Vorteil dieser Methode wäre, dass die Schüler den Umgang mit der VBA-Hilfe erlernen und trainieren.
Als Nachteil wäre zu sehen, dass diese Vorgangsweise sehr zeitintensiv ist. Weiters sollte darauf geachtet werden, dass jeder Schüler in der Gruppe etwas zum Gruppenergebnis beiträgt.
Als Alternative wäre hier der Frontalunterricht mit all seinen Vor- und Nachteilen zu sehen.

Programm P05-String Anforderungen:

Aus dem Beispielsatz soll der Teil der sich zwischen den beiden Beistrichen befindet herausgeschnitten werden. Dabei sollen im ersten Unterprogramm dies mit den

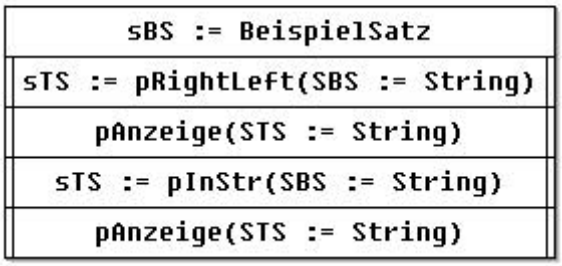
Stringfunktionen Left bzw. Right erfolgen, im zweiten Unterprogramm derselbe Teil des Beispielsatzes mit den Stringfunktionen Instr, Mid und Trim.

Beispielsatz:

„Visual Basic for Applications (VBA) ist eine zu den Microsoft-Office-Programmen gehörende Skriptsprache, VBA wurde aus dem von der Fa. Microsoft entwickelten Basic-Dialekt Visual Basic (VB) abgeleitet und zur Steuerung von Abläufen entwickelt, und wird immer noch weiter verbessert."

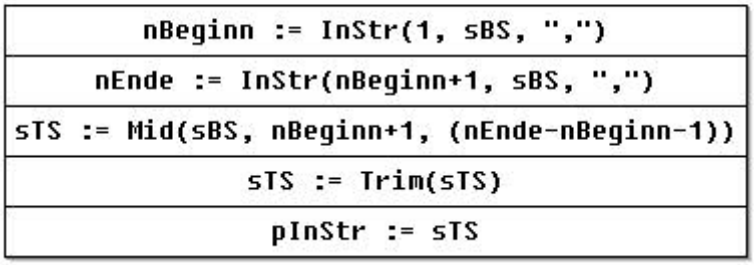
Struktogramm:

Main-Programm



Unterprogramm pRightLeft

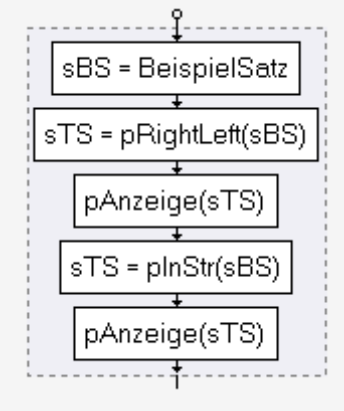
Unterprogramm pInStr



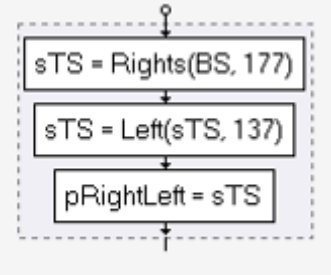
Unterprogramm pAnzeige

Programmablaufplan:

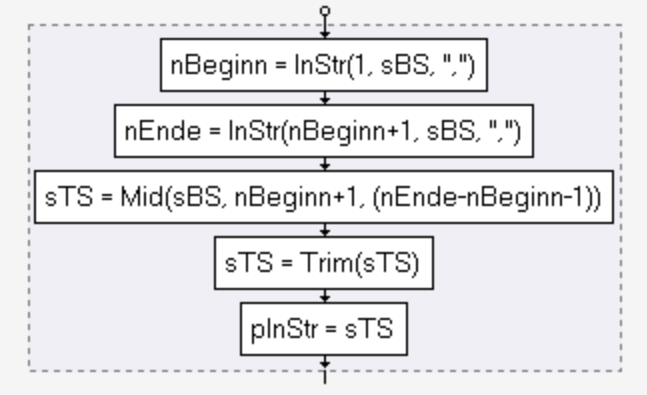
Main-Programm



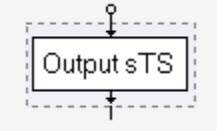
Unterprogramm pRightLeft



Unterprogramm pInStr



Unterprogramm pAnzeige



Programm:

```
Option Explicit
'
'Programm: P05-String
'
Sub P05_String()
'Deklarieren der Variablen
Dim sBS As String
Dim sTS As String

'Initialisieren der Variable mit dem Beispielsatz BS
sBS = "Visual Basic for Applications (VBA) ist eine zu den " _
    & "Microsoft-Office-Programmen gehörende Skriptsprache, " _
    & "VBA wurde aus dem von der Fa. Microsoft entwickelten " _
    & "Basic-Dialekt Visual Basic (VB) abgeleitet und zur " _
    & "Steuerung von Abläufen entwickelt, und wird immer " _
    & "noch weiter verbessert."

    sTS = pRightLeft(sBS)
    pAnzeige sTS

    sTS = pInStr(sBS)
    pAnzeige sTS

End Sub

'Teilstring mit den Stringfunktionen Right/Left erstellen
Function pRightLeft(sBS As String) As String
Dim sTS As String

    ' , an der 178 Stelle im Text
    sTS = Right(sBS, 177)
    ' , an der 138 Stelle im Text
    sTS = Left(sTS, 137)

    pRightLeft = sTS

End Function
```

```
'Teilstring mit den Stringfunktionen InStr/Mid/Trim erstellen
Function pInStr(sBS As String) As String
Dim nBeginn As Integer
Dim nEnde As Integer
Dim sTS As String

    'suche nach erstem Vorkommen im Text
    nBeginn = InStr(1, sBS, ",")

    'suche nach zweitem Vorkommen im Text
    'beginnen nach dem ersten Vorkommen
    nEnde = InStr(nBeginn + 1, sBS, ",")

    'Text vom ersten Vorkommen (+1 für das Zeichen selbst)
    'Länge = nEnde - nBeginn   (-1 für das Zeichen selbst)
    sTS = Mid(sBS, nBeginn + 1, (nEnde - nBeginn - 1))

    'Entfernen der Leerzeichen am Beginn und Ende der Zeichenkette
    sTS = Trim(sTS)

    pInStr = sTS

End Function

'Ausgabe des Teilstrings
Sub pAnzeige(sTS As String)

    MsgBox "Beispielsatz:" _
        & vbNewLine & sTS _
        & vbNewLine & "Länge: " & Len(sTS)

End Sub
```

Semester I – Einheit 11:

Inhalt	Didaktik	Methode	Ziel	Medien	Zeit
2 Wochenprojekt Themenvorschläge konkretisieren	Schüler erarbeiten Ideen zum Themen-vorschlag	Mind-Mapping 2 Schüler je Gruppe	Mind-Map zum Projektthema	PC + Beamer Netzwerk	20’
Projektvorschlag präsentieren	Schüler präsentieren Ihren Projektvorschlag	Präsentation	Projektvorschlag	PC + Beamer Netzwerk	30’
Projekttagebuch für PJ01-Projektprogramm erstellen	Schüler beginnen mit dem Projekttagebuch	Freiarbeit 2 Schüler je Gruppe	Schüler arbeiten mit einem Projekttage-buch	PC + Beamer Netzwerk	20’
Algorithmus für PJ01	Schüler erstellen Algorithmus	Freiarbeit 2 Schüler je Gruppe	Algorithmus	PC + Beamer Netzwerk	20’

Semester I – Einheit 11:

Zwei-Wochenprojekt: Die Schüler können in einem gewissen Rahmen nach freier Zeiteinteilung Ihre Projekte in einer Kleingruppe zu je zwei Schüler erarbeiten.
Nach erfolgter Themenfindung konkretisieren Sie Ihren Themenvorschlag mittels Mind-Map. Dabei sollte die Methode Mind-Map bereits bei den Schülern bekannt sein. Ist dies nicht der Fall, dann wird die Erstellung von Mind-Maps kurz erläutert. Danach werden die Projektvorschläge in Form der Mind-Maps vor der Klasse präsentiert, um anderen Schülern weitere Anregungen für das eigene Projekt geben zu können.
Nach kurzer Erläuterung wie Projekttagebücher zu führen sind, beginnen die Schüler damit, das Projekttagebuch zu ihrem Projekt zu erstellen. Dabei werden die wichtigsten Schritte des Projektes niedergeschrieben.
Im Anschluss erstellen die Schüler die Struktogramme und Programmablaufpläne für Ihr Programm. Beide Möglichkeiten der Darstellung sollten verwendet werden.
Sollten die Schüler mit den Algorithmen nicht im Unterricht fertig werden, sind diese in Heimarbeit fertig zustellen.

Semester I – Einheit 12:

Inhalt	Didaktik	Methode	Ziel	Medien	Zeit
2 Wochenprojekt Algorithmus vorstellen	Schüler präsentieren den Algorithmus	Präsentation 2 Schüler je Gruppe	Algorithmus	PC + Beamer Netzwerk	50’
Programm erstellen für PJ01	Schüler erstellen das Programm	Freiarbeit 2 Schüler je Gruppe	Programmcode	PC + Beamer Netzwerk	40’

Semester I – Einheit 12:

Die Schüler stellen in kurzen Präsentationen Ihren Algorithmus der gesamten Klasse vor. Danach werden die Programme in den Kleingruppen erstellt.
Die Projekte müssen bis zur nächsten Einheit fertig gestellt werden.

Semester I – Einheit 13:

Inhalt	Didaktik	Methode	Ziel	Medien	Zeit
2 Wochenprojekt Präsentation	Einige Schüler präsentieren Ihr Projekt	Präsentation 2 Schüler je Gruppe	PJ01 präsentieren und abschließen	PC + Beamer Netzwerk	25’
Grafische Elemente in VBA	Lehrer stellt die grafischen Elemente in VBA vor	Frontalunterricht Rückfragen	Schüler können grafische Elemente in VAB anwenden	PC + Beamer	25’
Beispielprogramm P06-UserForm Anforderungen vorstellen	Lehrer stellt die Anforderungen für das Beispielprogramm vor	Frontalunterricht Rückfragen	Schüler verstehen die Anforderungen	PC + Beamer	10’
Algorithmus und Programm P06-UserForm erstellen	Schüler erstellen Algorithmus und Programm	Freiarbeit Einzelarbeit	Algorithmus und Programm	PC + Beamer Netzwerk	20’
Musterlösung	Lehrer stellt die Musterlösung vor	Frontalunterricht Rückfragen	Schüler können die Musterlösung erläutern	PC + Beamer Netzwerk	10’

Semester I – Einheit 13:

Einige Kleingruppen präsentieren Ihr Projekt der gesamten Klasse.

Problemorientierter Ansatz: Die Ein- und Ausgabe am Bildschirm sollte optisch Verbessert werden. Bis jetzt wurden sämtliche Ein- und Ausgaben immer mittels den VBA-Funktionen InputBox und MsgBox realisiert.

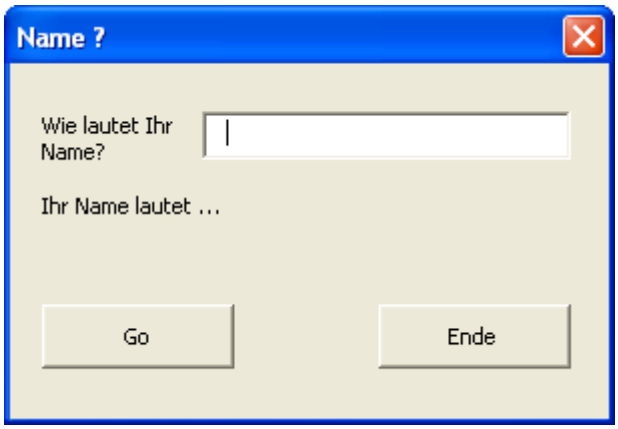
Grafische Elemente in VBA:
UserForm; Bezeichnungsfeld; Textfeld; Kombinationsfeld; Listenfeld; Kontrollkästchen; Optionsfeld; Rahmen; Befehlsschaltfläche
Wichtige Eigenschaften der grafischen Elemente in VBA

Um einen relativ umfassenden Überblick über die Vielfalt der grafischen Elemente in VBA geben zu können, wurde der Frontalunterricht gewählt. Der Vortragende ist gefordert, vor allem Elemente zu erläutern, welche häufig in der Praxis verwendet werden. Dabei ist auch auf eine sinnvolle Reihenfolge – von den einfachern bis hin zu den eher komplexen Elementen – zu achten.
Weiters wäre es sinnvoll, wenn der Lehrer die Erstellung der verschiedenen grafischen Elemente direkt über dem Beamer vorführt, damit die Schüler auch den Umgang mit dem VBA-Editor sehen können.

Programm P06-UserForm:

Zu erstellen ist eine UserForm mit einem Eingabefeld für den Namen, zwei Schaltflächen, eine zum Anzeigen des Namens im Ausgabefeld, die zweite zum Beenden des Programms.

UserForm:



Programm:

```
Option Explicit
'
'Programm: P06-UserForm
'
Sub P06_UserForm()
    F06_UserForm.Show
End Sub
'
'F06_UserForm
'
Const cTEXT = "Ihr Name lautet"

Private Sub btnGo_Click()

    If Me.txtName <> "" Then
        'Wenn txtName nicht leer ist, dann Namen ausgeben
        Me.lbOutput = cTEXT & " " & Me.txtName
    Else
        'Ausgabefeld löschen
        Me.lbOutput = ""
    End If

End Sub

Private Sub btnEnd_Click()
    End
End Sub
```

Semester I – Einheit 14:

Inhalt	Didaktik	Methode	Ziel	Medien	Zeit
Semesterprüfung Vorstellen der Fragen	Lehrer stellt die Fragen der Semesterprüfung vor	Frontalunterricht Rückfragen	Schüler verstehen die Fragen der Semester-prüfung	PC + Beamer	10’
Semesterprüfung	Schüler werden auf Ihr Wissen überprüft	Einzelarbeit	Schüler geben Ihr erworbenes Wissen wieder	Test	80’

Semester I – Einheit 14:

Semesterprüfung vorstellen der Fragen: Der Lehrer stellt die Fragen der Semesterprüfung vor. Dabei wird überprüft, ob alle Schüler die gestellten Fragen verstehen.

Semesterprüfung: Die Schüler erarbeiten selbstständig in Einzelarbeit die Fragen der Semesterprüfung. Nach erfolgter Abgabe des Fragebogens verlassen die Schüler den Klassenraum.

Fragebogen: siehe Anhang 6.4 Fragen der Semesterprüfung Semester I

Semester I – Einheit 15:

Inhalt	Didaktik	Methode	Ziel	Medien	Zeit
Rückgabe der Semesterprüfung	Lehrer gibt die korrigierte und benotete Semesterprüfung an die Schüler zurück	Frontalunterricht Rückfragen	Schüler bekommen Ihre korrigierte Semesterprüfung	Korrigierte und benotete Semesterprüfung	20’
Semesterprüfung nochmals erarbeiten	Schüler erarbeiten in Gruppen die Semesterprüfung nochmals	Gruppenarbeit 2 Schüler je Gruppe	Alle Schüler verstehen alle gestellten Fragen der Semesterprüfung	PC + Beamer Netzwerk	50’
Musterlösung	Lehrer stellt die Musterlösung vor	Frontalunterricht Rückfragen	Schüler können die Musterlösung erläutern	PC + Beamer Netzwerk	20’

Semester I – Einheit 15:

Rückgabe der benoteten Semesterprüfung an die Schüler.

Anschließend erarbeiten die Schüler in Gruppenarbeit die Semesterprüfung nochmals. Damit ist sichergestellt, dass jeder Schüler alle Fragen beantworten kann. Damit sollte der Lehrstoff nochmals wiederholt und gefestigt werden.
Semesterausklang

V Schlussfolgerungen und Ausblicke

5.1 Erkenntnisse der Arbeit

In dieser Arbeit wurden die Stundentafeln für das erste Semester des ersten Jahrganges für den Unterricht des Pflichtgegenstandes 11. Angewandte Programmierung des Lehrplans der höheren Lehranstalt für Informationstechnologie vgl. [@Lehr] erstellt. Dabei wurde nicht nur auf den Lehrstoff eingegangen, sondern auch auf die Art der Vermittlung dieses Lehrstoffs. Somit ist nicht nur die Frage nach dem Was ist zu lehren sondern auch Wie ist es zu vermitteln beantwortet worden.
Es wurde versucht, den Lehrstoff mittels mehreren verschiedenen Methoden den Schülern stets als spannend und informativ zu vermitteln. Der zeitliche Aufwand für die Erstellung der Stundentafeln aber auch für die Beispiele zum Unterricht darf auf keinen Fall unterschätzt werden. Vor allem die Programmierbeispiele sollten mit großer Sorgfalt erstellt werden damit diese auch von den Schülern nachvollzogen werden können. Dabei sollte aber auch darauf geachtet werden, dass der Lehrer zwar sein Wissen an die Schüler weiter gibt aber auch den Schülern eine gewisse Freiheit lässt, damit diese ihren eigenen Programmierstil entwickeln können.
Jeder einzelne Schüler sollte die Möglichkeit bekommen, sich selbst zu entfalten und sich selbst weiter zu entwickeln.

5.2 Eigene Erfahrungen

Am 21.04.2008 durfte ich eine Doppelstunde „Angewandte Programmierung“ einer zweiten Klasse der HTL Zeltweg gestalten.
Ich entschied mich nach Absprache mit Hr. Prof. Rettl für die Kontrollstrukturen „Schleifen“. Da es sich aber nicht um eine HTL für Informationstechnologie handelt, wurde der Inhalt auf die Zählschleifen reduziert.
Nach einer kurzen persönlichen Vorstellung meiner Person wurde von mir ein kurzer Theorieblock von cirka zehn Minuten zum Thema „Zählschleifen“ abgehalten.
Danach versuchte ich die Theorie sofort mit den Schülern in die Praxis umzusetzen. Dabei war ich stets bemüht, immer wieder Assoziationen zum realen Leben herzustellen damit sich die Schüler unter den abstrakten Begriffen der Informatik etwas vorstellen können.

Am Ende der Lehrveranstaltung bat ich um ein kurzes Feedback seitens der Schüler und bedankte mich anschließend für deren Mitarbeit.
Da ich selbst in naher Zukunft eine Lehrtätigkeit anstrebe, war diese Erfahrung sehr lehrreich für mich.

5.3 Weiterführende Ausblicke

5.3.1 Stundenbilder

Die in dieser Arbeit dargestellten Stundenbilder dienen als Vorlage oder Anregung zur Gestaltung des Unterrichts. Sie können niemals als zwingend angesehen werden, da jeder Unterrichtende seinen eigenen Input und seine eigenen Erfahrungen in den Unterricht einbringen sollte.
In dieser Arbeit sind die Stundenbilder des ersten Semesters dargestellt worden. In weiterer Folge sollten die Stundenbilder aller Semester aller Jahrgänge unter Berücksichtigung des aktuellen Lehrplans erarbeitet werden.
Ein weiterer Vorteil für die Schüler könnte sein, wenn nicht nur ein Beispiel pro Einheit angeboten wird, sondern die Schüler aus mehren Beispielvorschlägen wählen könnten. Dazu könnte es sinnvoll sein, wenn die Beispiele unter den Lehrern ausgetauscht werden würden, oder die verschiedenen Beispiele mit dazugehöriger Beschreibung auf der Schulwebseite zum Download bereitgestellt werden würden.

5.3.2 Wissensüberprüfung

Die Wissensüberprüfungen siehe Punkt 6.3 Fragen der Zwischenüberprüfung Semester I und Punkt 6.4 Fragen der Semesterprüfung Semester I sind Multiple Choice Tests. Darin enthalten sind sowohl geschlossene wie auch offene Fragen. Die Fragen und die dazugehörigen Antworten wurden in Eigenregie erstellt, und können vom jeweiligen Lehrer zur Bewertung der Schüler verwendet werden.
Die Wissensüberprüfung, und die damit verbundenen Tests sind nach [Hale07] ein wichtiger Teil im Programmierunterricht. Die Schüler lernen besser bei immer wiederkehrenden Überprüfungen als bei nur einer Gesamtprüfung.

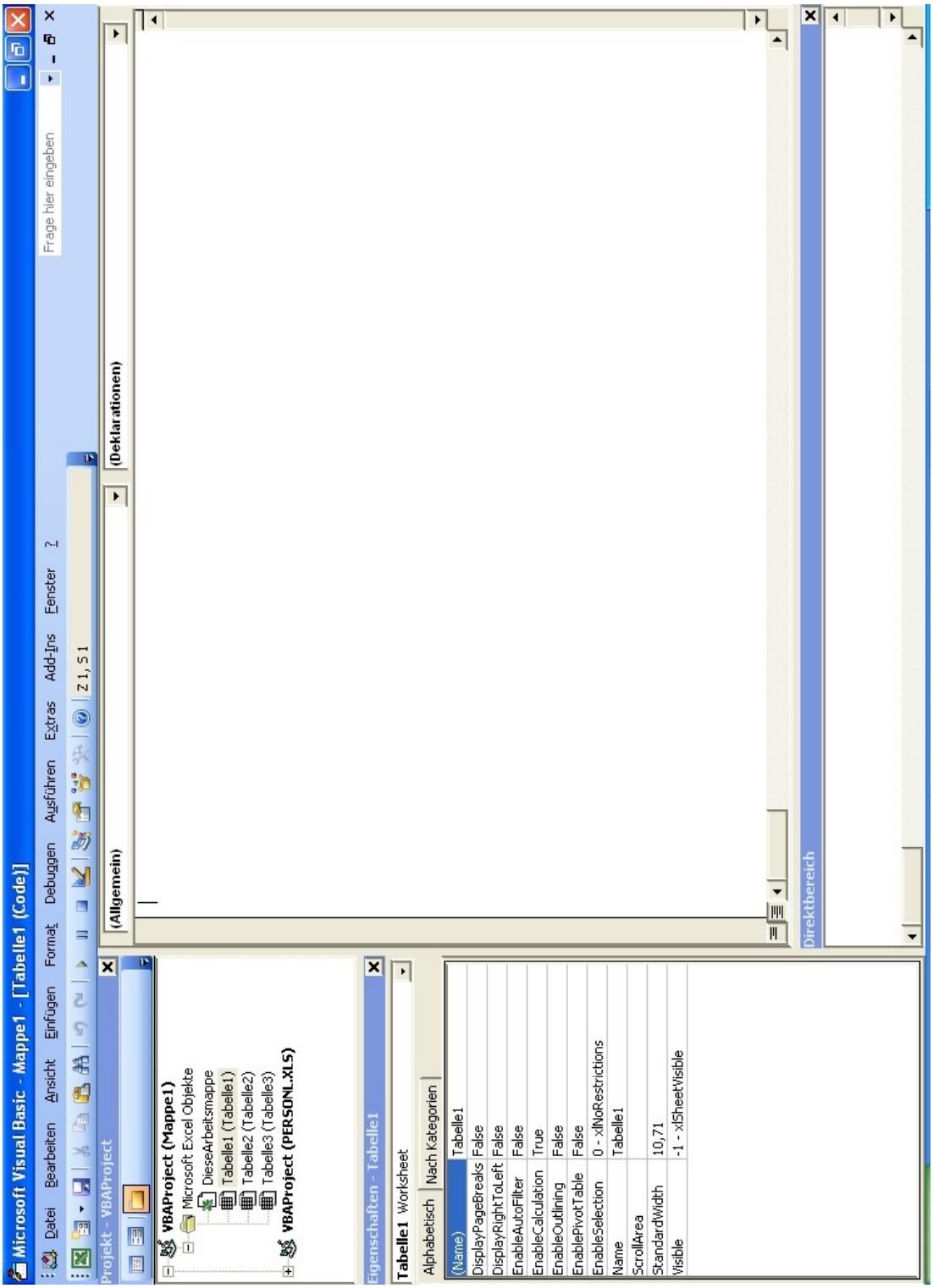
Wenn die Wissensüberprüfung mittels Computerunterstützung durchgeführt wird, kann dem Lehrer als auch dem Schüler wertvolle Zeit eingespart werden. Die einmalig erstellten Fragen können vom Computerprogramm zu Tests zusammengestellt werden, von den Schülern am PC bearbeitet und beantwortet werden und auch vom Computer ausgewertet werden. Die entstandene Zeitersparnis kann der Vortragende für andere kreativere Tätigkeiten verwenden. Auch die ungleichen Auswertungen von Fragen des Lehrers werden damit weitgehend vermieden.
Zum Erstellen solcher computergestutzten Tests kann zum Beilspiel das Programm Moodle vgl. [@Mood] verwendet werden. Diese Lernplattform auf Open-Source-Basis wird kostenlos zum Download angeboten und kann von Schulen verwendet werden. Weitere Funktionalitäten von Moodle sind das Verwalten von Schulklassen, Klassenräume, aber auch der gesamte Unterrichtsstoff und auch das erstellen von Tests. Eine genaue Beschreibungen von Moodle kann auf der Website vgl. [@Mood] entnommen werden.

5.4 Schlussworte

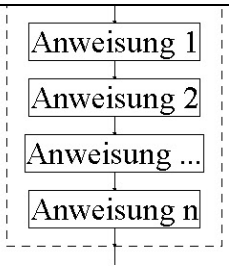
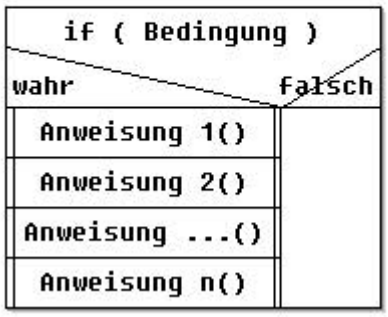
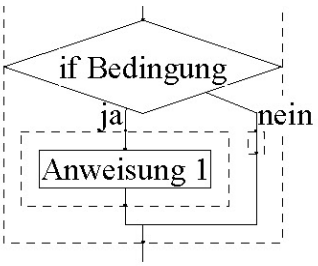
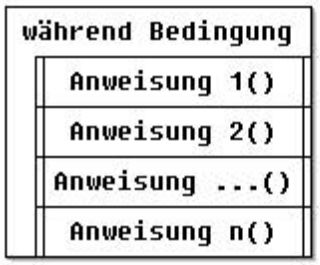
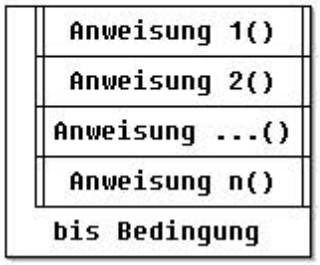
In den Jahren an der TU Wien habe ich sehr viele neue Erfahrungen sammeln dürfen. Damit meine ich nicht nur die fachliche Ausbildung, welche ich als ausgezeichnet empfunden habe, sondern auch die zwischenmenschlichen Eindrücke. Vor allem der Altersunterschied zu manch anderen Studienkollegen aber auch Vortragenden war immer wieder eine Herausforderung für mich. Gerade deshalb denke ich, haben sich immer wieder interessante Gespräche zwischen Kollegen und mir, aber auch zwischen Vortragenden und mir, ergeben.

VI Anhang

6.1 Microsoft Visual Basic for Applications Editor



6.2 Struktogramm – Flussdiagramm

Struktogramm	Flussdiagramm	Beschreibung
		Anweisung
		Block
		If – Bedingung
		While
		Until

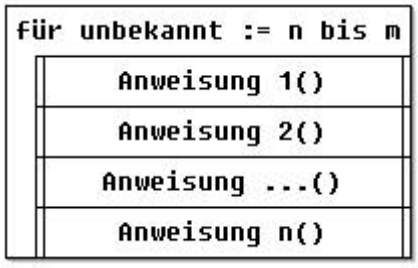
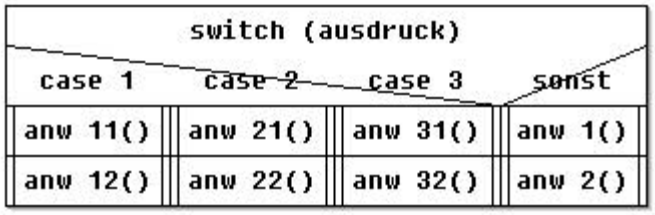
		For
		Switch - Case

Abbildung 8: Struktogramm – Flussdiagramm
vgl. [@Schm]

6.3 Fragen der Zwischenüberprüfung Semester I

Fragen der Zwischenüberprüfung (Die Antworten zu allen Fragen müssen in ihrer Reihenfolge nochmals verändert werden, da die richtigen Antworten an erste Stelle stehen; Auch die Fragenblöcke selbst sollten durchgemischt werden.):

Fragen:

Erstellen Sie ein Struktogramm und/oder einen Programmablaufplan für folgende Programmanforderung: (20 Pkt)

> Eingabe von einem Zahlenwert. Wenn der Zahlenwert größer oder gleich 10 ist, sollte er am Bildschirm ausgegeben werden. Ist der Zahlenwert größer als 10, dann multipliziere diesen Wert mit 20 und gibt das Produkt am Bildschirm aus.
>
> Danach soll eine Eingabe von einem Textwert erfolgen. Der Text soll genau zweimal am Bildschirm ausgegeben werden.

Was ist ein Interpreter? (5 Pkt)

- ☐ ist eine Art Dolmetscher (R)
- ☐ ist ein Software Programm (R)
- ☐ geschriebene Programme können auf verschiedenen Rechnerarchitekturen ausgeführt werden (R)
- ☐ ist eine Art Übersetzter
- ☐ Quellcode wird vor dem Ausführen in ein ausführbares Programm (Bytecode) übersetzt

Was ist ein Compiler? (5 Pkt)

- ☐ ist eine Art Übersetzter (R)
- ☐ ist ein Software Programm (R)
- ☐ Quellcode wird vor dem Ausführen in ein ausführbares Programm (Bytecode) übersetzt (R)
- ☐ ist eine Art Dolmetscher
- ☐ geschriebene Programme können auf verschiedenen Rechnerarchitekturen ausgeführt werden

Was ist ein Programm? (5 Pkt)

- ☐ ist Software (R)
- ☐ läuft auf Hardware (R)
- ☐ ist eine Art Kochrezept
- ☐ kann als Struktogramm / Programmablaufplan dargestellt werden

Was ist ein Algorithmus? (5 Pkt)

- ☐ ist eine Art Kochrezept (R)
- ☐ eine abstrakte Handlungsvorschrift (R)
- ☐ kann als Struktogramm / Programmablaufplan dargestellt werden (R)
- ☐ ist Software

Wie werden Ganzzahlen intern am PC dargestellt? (5 Pkt)

- ☐ in 0 und 1 (R)
- ☐ in absoluten Zahlen
- ☐ unter Anführungszeichen „123“
- ☐ in Volltext „einhundertdreiundzwanzig“

Wie werden Texte intern am PC dargestellt? (5 Pkt)

- ☐ in 0 und 1 (R)
- ☐ ausgeschrieben ohne Leerzeichen zwischen den Wörtern
- ☐ unter Anführungszeichen „das ist mein Text“
- ☐ ohne Anführungszeichen

Was ist eine Kontrollstruktur? (5 Pkt)

- ☐ dient zur Steuerung des Programmablaufes (R)
- ☐ IF-ELSE-END IF Block ist eine Kontrollstruktur (R)
- ☐ kontrolliert den Programmierer bei der Dateneingabe
- ☐ kontrolliert den PC

Welche der folgenden Aussage ist richtig? (10 Pkt)

☐ (R)	IF <Bedingung> THEN
	  <True-Teil>
	ELSE
	  <False-Teil>
	END IF

☐ (R)	IF <Bedingung> THEN
	  <True-Teil>
	END IF

☐	IF <Bedingung> THEN
	  <False-Teil>
	ELSE
	  <True-Teil>
	END IF

☐	IF THEN
	  <False-Teil>
	END IF

☐	IF THEN
	  <True-Teil>
	END IF

Die <Aktion1> soll in einem Programm genau dreimal ausgeführt werden. Welche Art von Schleife verwenden Sie? (5 Pkt)

- ☐ For-Next (R)
- ☐ Do-While
- ☐ While-Loop
- ☐ Do-Loop

Welche Kontrollstruktur verwenden Sie, wenn eine <Aktion1> mindestens einmal oder öfters durchlaufen werden soll. (5 Pkt)

- ☐ Do-Loop-While (R)
- ☐ If-Then-Else-End If
- ☐ For-Next
- ☐ Do-While-Loop

6.4 Fragen der Semesterprüfung Semester I

Fragen der Semesterprüfung (Die Antworten zu allen Fragen müssen in ihrer Reihenfolge nochmals verändert werden, da die richtigen Antworten an erste Stelle stehen; auch die Fragenblöcke selbst sollten durchgemischt werden.):

Fragen:

Wie können Sie einen Kommentar im Programm kennzeichnen? (5)

- ☐ mit dem einzelnen Hochstrich ' (R)
- ☐ mit dem Schlüsselwort REM (R)
- ☐ mit dem Schlüsselwort COMMENT
- ☐ mit dem Schlüsselwort KOMMENTAR
- ☐ gar nicht

Welche Aussagen treffen auf eine Variable zu? (5)

- ☐ ein Speicherbereich in einem Computerprogramm (R)
- ☐ der Speicherbereich wird über den Namen der Variablen angesprochen (R)
- ☐ beim Namen der Variable können Ziffern zwischen und am Ende des Namens verwendet werden (R)
- ☐ Variable können mehr als einen Zahlenwert aufnehmen
- ☐ beim Namen der Variable können Ziffern am Beginn des Namens verwendet werden

Was trifft auf den Begriff INTEGER zu? (5)

- ☐ ist ein Schlüsselwort (R)
- ☐ ist ein Datentyp (R)
- ☐ kann als Name für eine Variable verwendet werden
- ☐ ist ein Datentyp für Zeichenketten
- ☐ ist ein Datentyp für Text

Was trifft auf den Begriff STRING zu? (5)

- ☐ ist ein Datentyp für Zeichenketten (R)
- ☐ `Dim sVarName as String` (R)
- ☐ ist ein Datentyp für Ganzzahlen
- ☐ kann als Name für ein Unterprogramm verwendet werden
- ☐ ist kein Schlüsselwort

Welche Verkettungen von Textteilen sind richtig? (5)

- ☐ `sTxt = sText1 & sText2` (R)
- ☐ `sText1 = sText2 & sText1` (R)
- ☐ `sTxt = sText1 & " aber auch " & sText2` (R)
- ☐ `sTxt = sText1 & " & sText2`
- ☐ `sTxt = "Summe: " "10"`

Mit welchen Funktionen können Teile aus einer Zeichenkette herausgeschnitten werden? (5)

- ☐ Left (R)
- ☐ Left$ (R)
- ☐ Right (R)
- ☐ Mid (R)
- ☐ Len

Beschreiben Sie alle der folgenden Operatoren? (5)

- ☐ <>..... (Ungleich)
- ☐ >=..... (Größer Gleich)
- ☐ <=..... (Kleiner Gleich)
- ☐ >.....(Größer als)
- ☐ <.....(Kleiner als)

Welche der folgenden Aussagen zum Thema Übergabeparameter sind richtig? (5)

- ☐ `function pBerechne(Z1 as Integer, Z2 as Integer) as Integer` (R)
- ☐ `sub pAnzeige(sText as String)` (R)
- ☐ `sub pAusgabe()` (R)
- ☐ `function pDiff(Z1 as Integer, Z2 as Integer)`
- ☐ `sub pAdd(Z1 as Integer, Z1 as Integer) as Integer`

Welche Ergebnisse sind bei den folgenden Funktionen zu erwarten? (5)

Variable: `sT="VBA ist einfach zu erlernen!"`

- ☐ `Left(sT,3)` .....(ersten 3 Stellen von Links = VBA)
- ☐ `Len(sT)` .....(Anzahl der Zeichen = 28)
- ☐ `Right(sT,1)` .....(erstes Zeichen von Rechts = !)
- ☐ `UCase(sT)` .....(gesamter Text in Großbuchstaben)
- ☐ `Trim(sT)` .....(Text ohne Leerzeichen am Beginn und Ende)

Welche der folgenden Aussage ist richtig? (10 Pkt)

☐ (R)	`Do`
	`  <Aktion>`
	`Loop While <Bedingung>`

☐ (R)	`Do While <Bedingung>`
	`  <Aktion>`
	`Loop`

☐ (R)	`For nSchleifenZähler = nVon To nBis`
	`  <Aktion>`
	`Next nSchleifenZähler`

☐	`Do While <Bedingung>`
	`  <Aktion>`
	`Loop While <Bedingung>`

☐	`For nSchleifenZähler = nVon To nBis`
	`  <Aktion>`
	`Loop nSchleifenZähler`

Literaturverzeichnis

Working Papers:

[Ahma07] Marzieh Ahmadzadeh, Dave Elliman, Colin Higgins;
The Impact of Improving Debugging Skill on Programming Ability;
Shiraz University of Technology, Iran; 2007

[AlFu07] Mohammed Al-Fudail, Harvey Mellar;
Investigating teacher stress when using technology;
London Knowledge Lab, Institute of Education, University of London, 20 Bedford Way, London WC1H 0AL, United Kingdom; 2007

[Grif07] Rob Griffiths, Simon Holland, Marion Edwards; Rob Griffiths, Simon Holland, Marion Edwards;
Sense before syntax: a path to a deeper understanding of objects;
Computing Department, The Open University, Walton Hall, Milton Keynes, England MK7 6AA; 2007

[Hale07] Debra Trusso Haley, Pete Thomas, Anne De Roeck, Marian Petre;
Seeing the Whole Picture: Evaluating Automated Assessment Systems;
The Open University Walton Hall, Milton Keynes MK7 6AA UK; 2007

[Leem07] R. Mark Leeman and David H. Glass;
Teaching Java with Robots and Artificial Life;
School of Computing and Mathematics University of Ulster Newtownabbey, Co. Antrim, BT37 0QB, UK; 2007

[Pool07] Nigel Poole;
Activating laboratories using Visual Basic for Applications;
2007

[Pric07] Colin B. Price;
From Kandinsky to Java (The Use of 20th Century Abstract Art in Learning Programming);
University of Worcester; 2007

[Teag07] Donna Teague, Paul Roe;
Learning to Program: Going Pair-Shaped;
Queensland University of Technology Brisbane, Australia; 2007

[Trot06] Andrew Trotman, Chris Handley;
Programming contest strategy;
Department of Computer Science, University of Otago, P.O. Box 56, Dunedin, New Zealand; 2006

[Whit07] Whitfield A. K., Blakeway S., Herterich G. E., Beaumont C.;
Programming, disciplines and methods adopted at Liverpool Hope University;
School of Computer Science Liverpool Hope University, Hope Park, Liverpool L16 9JD; 2007

Bücher:

[Hubw01] Peter Hubwieser; Didaktik der Informatik; Springer-Verlag Berlin Heidelberg New York; Berlin Heidelberg 2000, 2001

[Hubw07] Peter Hubwieser; Didaktik der Informatik 3. Auflage; Springer-Verlag Berlin Heidelberg New York; Berlin Heidelberg 2000, 2001, 2004 und 2007

[Humb06] Ludger Humbert; Didaktik der Informatik mit praxiserprobtem Unterrichtsmaterial; B.G.Teubner Verlag / GWV Fachverlage GmbH; Wiesbaden 2006

[Schu04] Sigrid Schubert, Andreas Schwill; Didaktik der Informatik; Spektrum Akademischer Verlag; Heidelberg – Berlin 2004

[Reit03]: Anton Reiter, Gerhard Scheidl; Schulinformatik in Österreich; CDA-Verlag; Wien 2003

Web-Ressourcen:

[@Beck]
Java: Learning to Program with Robots
http://www.learningwithrobots.com
Stand 25.06.2008

[@Blik]
Pädagogisches Institut der deutschen Sprachgruppe - Bozen - 2003
http://www.blikk.it/angebote/schulegestalten/se855.htm
Stand 17.03.2008

[@Denk]
Denkzeichnen
http://www.denkzeichnen.de/
Stand 01.08.2008

[@IHEA]
ICS Subject Centre of the Higher Education Academy
http://www.ics.heacademy.ac.uk
Stand 30.05.2008

[@Lehr]
http://www.htl.at/fileadmin/content/Lehrplan/HTL_Informationstechnologie.pdf
Stand 25.01.2008

[@Meth]
Reich, K. (Hg.): Methodenpool
http://methodenpool.uni-koeln.de
Stand 12.03.2008

[@Mood]
http://www.edumoodle.at
Stand 17.07.2003
[@Msdn] Microsoft Developer Networt
http://msdn2.microsoft.com/de-at/default.aspx
Datentyp Single:
http://msdn2.microsoft.com/de-de/library/xay7978z(VS.80).aspx
Datentyp Double:
http://msdn2.microsoft.com/de-de/library/x99xtshc(VS.80).aspx
Zusammengesetze Datentypen:
http://msdn2.microsoft.com/de-de/library/d682h20d(VS.80).aspx
Stand 22.04.2008

[@Part]
"Homepage" der Familie Partheil aus Biedenkopf-Wallau
http://www.partheil.com/vips
Stand 31.03.2008

[@Schm]
www.schmidthammer.de – Struktogramm
http://www.schmidthammer.de/mphinf/fach/inf/c/struktogramme.jpg
Stand 31.03.2008

[@Trim]
Michael Trimmel, Intrinsische / extrinsische Motivation
http://homepage.univie.ac.at/Michael.Trimmel/mws00_schodl.htm
Stand: 15.05.2008

[@Ziem]
Peter-Michael Ziemke – Lehrer am Gymnasium in Leverkusen
http://www.ziemke-koeln.de/download/index.htm
Stand 31.03.2008

Software-Ressourcen:

[@VbaH]
Microsoft Visual Basic 6.0
Visual Basic Programm Hilfe:
Microsoft Excel Visual Basic Referenz / Microsoft Visual Basic Dokumentation
Stand: Microsoft Office Excel 2003