Ein kleiner Schritt für LehrerInnen, ein großer Schritt für SchülerInnen.

Peter Micheuz

Institut für Informatik Systeme
Universität Klagenfurt
Universitätsstraße 65-67
A-9020 Klagenfurt
Österreich
peterm@isys.uni-klu.ac.at

Abstract: Die Schulinformatik an den allgemein bildenden höheren Schulen Österreichs ist geprägt von einer nahezu unüberschaubaren Vielfalt an Organisationsformen, die auf schulautonome Profilbildungen zurückzuführen sind. Dennoch gibt es an der Schnittstelle zwischen Sekundarstufe I und Sekundarstufe II ein Pflichtfach Informatik in der 9. Jahrgangsstufe im Ausmaß von zwei Wochenstunden, für das es ab dem Schuljahr 2004/2005 einen neuen zentral verordneten Lehrplan gibt. In diesem Beitrag wird beleuchtet, ob und wie ein Teil dieses sehr offenen und abstrakten Lehrplans, nämlich das Thema Programmierung, der konkreten Unterrichtsarbeit zugänglich gemacht werden kann. Einerseits werden die entsprechenden Formulierungen dieses Lehrplanabschnittes kommentiert, andererseits führt eine empirische Untersuchung über elementare Algorithmen auf dieser Altersstufe zu einigen interessanten Schlussfolgerungen und Thesen.

Nichts ist so gefährlich wie das „Allzu modern Sein".
Man gerät in Gefahr, plötzlich aus der Mode zu kommen.
Oscar Wilde

1 Einleitung

„Natürlich kommt man ohne eine Programmiersprache nicht aus. Aber mehrmals habe ich feststellen müssen, dass Schüler angeblich eine ganze Programmiersprache gelernt hatten und die einfache Aufgabe, die Werte zweier Variablen zu vertauschen, nicht lösen konnten. Das ist erschütternd: Man lernt Rekursion, aber der Schüler kann die Werte zweier Variablen nicht vertauschen! Zu diesem Kapitel gehören noch andere Dinge, z. B. die Unterscheidung zwischen der Gleichheitsrelation und der Zuweisungsoperation." Ist dieser Tausch der Werte zweier Variablen wirklich so wichtig, dass dieses Bedauern von Peter Rechenberg [Re94] gerechtfertigt wäre? Diese Aussage wirft Fragen auf, die in vieler Hinsicht den Kern der Informatik-Fachdidaktik treffen: Was soll aus welchen Gründen in welcher Altersstufe wie vermittelt werden? Der Trend in Österreich in Richtung Output-Steuerung legt außerdem nahe, die Frage nach Standards und Evaluationen und damit der tatsächlichen Schülerleistungen mit einzubeziehen.

Das mangelhafte Wissen und Können der SchülerInnen im Grundlagenbereich betrifft

derzeit (in Österreich) weniger die Informatik[1] als vielmehr bereits etablierte Fächer wie Mathematik und naturwissenschaftliche Fächer, wie es eine einschlägige Studie[2] belegt. Diese tradierten Fächer haben im Unterschied zum Schulfach Informatik ihren weitest gehend gesicherten Platz im Fächerkanon und zeichnen sich durch einen über Jahrhunderte gewachsenen, systematischen und soliden Aufbau, verteilt über die einzelnen Jahrgangsstufen, aus. Zweifellos sind das notwendige, aber noch lange nicht hinreichende Bedingungen für nachhaltige Kompetenzen der SchülerInnen. Die Klagen von (üblicherweise nicht larmoyanten) Mathematik-LehrerInnen, dass angehende AbsolventInnen nicht gut rechnen können und mit Bruch- und Prozentrechnungen auf Kriegsfuß stehen, hört man immer öfter. Wenn die SchülerInnen bereits in einem Fach wie Mathematik, das im Fächerkanon ab der Grundschule mit mindestens 3 Wochenstunden dotiert ist, gravierende Defizite aufweisen, wie sieht es dann erst mit der Nachhaltigkeit informatischer Kompetenzen[3] aus? Um dieser Frage nachzugehen, müssen wir uns zunächst die organisatorischen Rahmenbedingungen ansehen.

2 Informatikunterricht an den AHS Österreichs

<table>
<tr><td></td><td></td><td colspan="5">Informatik - Reifeprüfung</td></tr>
<tr><td rowspan="4">Oberstufe</td><td>8. Klasse</td><td rowspan="3">Wahlpflichtfach Informatik (2 / 3-stufig), Kursysteme</td><td rowspan="3">E-Learning Initiativen Notebook-klassen (E-Cluster)</td><td rowspan="3">Integration in anderen Fächern</td><td rowspan="3">Schul-versuche</td><td rowspan="3">?</td></tr>
<tr><td>7. Klasse</td></tr>
<tr><td>6. Klasse</td></tr>
<tr><td>5. Klasse</td><td colspan="5">Informatik als Pflichtfach</td></tr>
<tr><td rowspan="4">Unterstufe</td><td>4. Klasse</td><td rowspan="4">Schul-autonomes Pflichfach, Ubungen</td><td rowspan="4">E-Learning Initiativen (ELSA), Notebook-klassen</td><td rowspan="4">Integration in anderen Fächern</td><td rowspan="4">Schul-versuche</td><td rowspan="4">?</td></tr>
<tr><td>3. Klasse</td></tr>
<tr><td>2. Klasse</td></tr>
<tr><td>1. Klasse</td></tr>
</table>

Abbildung 1: Informatik an den Gymnasien Österreichs

Die einzige Invariante informatischer Unterrichtserteilung in Österreichs Gymnasien seit nunmehr 20 Jahren ist das zweistündige Pflichtfach Informatik [Re03] in der 9. Jahrgangsstufe (5. Klasse der AHS[4]). Im schlechtesten Fall besteht das minimale Lehrangebot innerhalb der gymnasialen Laufbahn aus zwei Wochenstunden, das sind 2 von insge-

[1] In Österreich ist die (ministeriell abgesegnete) Entwicklung von Standards derzeit auf die Fächer Mathematik, Englisch und Deutsch beschränkt. Sehr wohl aber definieren die weit verbreiteten und in Österreichs Schulen gut angenommenen Zertifikate wie der ECDL Standards, die unabhängig von der Altersstufe erworben werden können.

[2] Die Ergebnisse der PISA-Studie 2003 waren für Österreich ernüchternd.

[3] Auf die Kompetenzen der Informatik-Lehrer wird in diesem Beitrag nicht eingegangen. Nebenbei sei vermerkt, dass es seit 2000 in Österreich ein vollwertiges Informatik-Lehramtstudium an vier Universitäten gibt.

[4] In Österreich besuchen ca. 20% eines Jahrgangs die 5. Klasse eines Gymnasiums (AHS).

samt ungefähr 250 Wochenstunden. Allerdings ist dieses worst-case Szenario nicht mehr oft anzutreffen, zumal die Schulautonomie in Österreich seit 1995 eine Vielzahl von Schwerpunkten und Schulprofilbildungen gerade im Bereich der Schulinformatik bewirkt hat. Es liegen derzeit allerdings (noch) keine exakten Zahlen über diese Entwicklung im zuständigen Ministerium vor.

So bedauerlich aus Sicht der Protagonisten der Schulinformatik der Zustand des minimalen verpflichtenden Angebots nur in der 9. Jahrgangsstufe ist, so erfreulich ist die Tatsache, dass es auch SchülerInnen gibt, die (in Ausnahmefällen) im Gymnasium das Fach Informatik durchgehend von der 1. bis zur 8. Klasse besuchen und aus diesem Fach auch maturieren (können). Zwei Beispiele aus dem Gymnasium, an dem ich unterrichte und das in einem Schultyp in einer Klasse seit Jahren einen Informatikschwerpunkt setzt, mögen die enorme Bandbreite informatischer Kompetenz verdeutlichen. Eine Schülerin der 6. Klasse (10. Jg.) wollte ein halbes Jahr nach dem Pflichtfach Informatik in der 5. Klasse in einem anderen Fach eine Präsentation vorführen, die sie auf einer CD von zu Hause mitbrachte. Sie konnte diese Präsentation nicht öffnen, weil sie das CD-Laufwerk über das Betriebssystem nicht ansprechen konnte. Dies ist ein extremes Negativbeispiel eines wenig nachhaltigen Unterrichts. Das andere Extrem auf der fast unendlichen Informatik-Kompetenzskala repräsentieren SchülerInnen[5], die bei der Informatik-Matura[6] mit außerordentlichen Leistungen brillieren und anschließend das Informatik-Studium mit Bravour absolvieren.

An Österreichs Gymnasien findet Schulinformatik in verschiedensten Ausprägungsformen statt (Abbildung 1). Die Ausnahme ist die Regel. Es ist nicht Ziel dieses Beitrages darüber zu reflektieren, ob diese Disparitäten in der Schulinformatik an Österreichs Gymnasien als Ergebnis schulautonomer Profilbildungen gut oder schlecht sind. Ebenso gebe ich an dieser Stelle auch kein Werturteil über diesen "Fleckerlteppich" (euphemistisch ausgedrückt: "bunte blühende Wiese") ab, solange keine aktuellen quantitativen und qualitativen Studien vorliegen.[7]

3 Vom abstrakten Lehrplan zum konkreten Unterricht

Im Zuge einer inhaltlichen Reform des Gymnasiums sind ab dem Schuljahr 2004/2005 neue Lehrpläne für die Oberstufe der AHS, also für die 9.-12. Jahrgangsstufe, in Kraft gesetzt worden. Damit ist eine neue gesetzliche Grundlage, warum und wie welche Inhalte im Fach Informatik unterrichtet werden sollen, Teil österreichischer Schulrealität, zumindest auf dem Papier.

In diesem Beitrag liegt der Schwerpunkt in der Interpretation des Lehrplanabschnittes über Algorithmen und Programmierung in der 9. Jahrgangsstufe[8]. Für die Lehrplanauto-

[5] In Österreich ist der Anteil der SchülerInnen an den AHS, die im Fach Informatik maturieren, klein.
[6] In Deutschland ist es das Abitur.
[7] Ein Überblick über die Informatik an den AHS kann im CDA-Sonderheft [CD05] nachgelesen werden.
[8] Bezüglich des Lehrplanes des Wahlpflichtfaches Informatik wird auf http://www.gemeinsamlernen.at verwiesen.

ren gab es von ministerieller Seite aus die Vorgabe, diesen - nicht zuletzt wegen der Rücksichtnahme auf das unterschiedliche Vorwissen der SchülerInnen aus der Sekundarstufe I (gymnasiale Unterstufe) - so offen wie möglich zu halten. Dieses gelang auch, allerdings auf Kosten von jedweden konkreten stofflichen Vorgaben. Die Freiräume für engagierte InformatiklehrerInnen sind dementsprechend groß. Für weniger einfallsreiche LehrerInnen sind die Vorgaben dieses Lehrplanes zu abstrakt. Studien hinsichtlich der unterschiedlichen Interpretationen und Exekutionen dieser gesetzlichen Vorgabe können erst nach diesem Schuljahr durchgeführt werden. So wird das Thema Programmierung in einem Absatz kurz abgehandelt:

"Die Schülerinnen und Schüler sollen** Einblicke in wesentliche Begriffe und Methoden der Informatik, ihre typischen Denk- und Arbeitsweisen, ihre historische Entwicklung sowie ihre technischen und theoretischen Grundlagen gewinnen und **Grundprinzipien von Automaten, Algorithmen und Programmen kennen lernen."

Ein wenig konkreter, aber für unbedarfte Informatik-Lehrkräfte nicht wirklich hilfreich, wird es in einem Kommentar zu diesem Lehrplanauszug von Caba [CD05]: "Eine Behandlung dieser Themen im Rahmen von schülerrelevanten Problemstellungen stellt aber sicher eine ganz besondere didaktische Herausforderung dar, zumal im Lehrplan der 5. Klasse die Programmierung bzw. Programmiersprachenkonzepte als Lehrstoff nicht explizit vorgesehen sind. Es sei … auf die in [Hu00] vorgestellten Unterrichtsbeispiele … hingewiesen. Dort werden Aspekte von Konzeptwissen (z. B. Objektorientierung) bzw. von Methoden (wie Modellierungstechniken) und Begriffen (wie Algorithmus oder Automat) im Rahmen der Verwendung von Anwendersoftware aufgegriffen."

Eine weitere Konkretisierung in Form von operationalisierten[9] Lernzielen tut Not. Unterrichtsarbeit bedeutet, mühsam einen Fuß vor den anderen zu setzen. Im fachdidaktischen Elfenbeinturm ist es zwar angenehm, doch profitieren können die Kolleginnen und Kollegen nur davon, dass sich manche durch den ebenso altersgemäßen wie fordernden Unterricht quälen. Insofern unterscheidet sich der Informatikunterricht nicht vom Unterricht in den tradierten Fächern.

4 Programmierung als wichtiger Teil des Informatikunterrichts

Sehen wir uns eine mögliche Umsetzung und Evaluation zum Thema Programmierung in der 9. Jahrgangsstufe an. Im November 2004 wurde im Rahmen einer Diplomarbeit an der Universität Klagenfurt eine diesbezügliche Studie durchgeführt.

Der Anlass, dieses Thema zu beforschen, ergab sich aus den Lehrplanvorgaben, die Programmierung (im weitesten Sinn) auf dieser konkreten Alterstufe als Notwendigkeit vorsehen. Eine immanente Aufgabe jeder Fachdidaktik besteht darin, mit geeigneten fachspezifischen Lernzielen den Nachweis zu erbringen, dass kein anderes Fach diesen spezifischen Beitrag zur Allgemeinbildung zu erbringen vermag [Re03]. In diesem Zu-

[9] Operationalisierte Lernziele beschreiben das erwünschte beobachtbare Verhalten des Schülers möglichst eindeutig [vgl. den Syllabus des ECDL auf http://www.ecdl.at].

sammenhang könnten leichte Zweifel auftauchen, das Thema Programmierung und Algorithmen den Mathematikern zu überlassen. Im neuen österreichischen Mathematik-Lehrplan ist Programmierung nicht explizit vorgesehen. Wir können die Kirche also im Dorf lassen. Bei Ottmann [OW96] finden wir: „Sie [Algorithmen] sind das zentrale Thema der Informatik. Die Entwicklung und Untersuchung von Algorithmen zur Lösung vielfältiger Probleme gehört zu den wichtigsten Aufgaben der Informatik.“ Und sogar bei Hentig [He93] ist nachzulesen: "Schwer schließlich ist das Programmieren - aber das müssen nicht sehr viele Leute in dieser Gesellschaft tun, es müssen sich nur alle einmal an einem Programm versucht haben, um zu verstehen, was da vor sich geht." War diese Aussage gar Grundlage der sanften Formulierung im Lehrplan "Die Schüler sollen ... Grundprinzipien von Automaten, Algorithmen und Programmen kennen lernen". Dies impliziert folgende Fragen und einen großen Interpretationsraum.

- Wie soll die Primärerfahrung der SchülerInnen mit Programmierung gestaltet werden? Welche Zugänge und Unterrichtsmethoden sind adäquat?
- Gibt es einen Katalog allgemein bildender und altersstufen-adäquater Aufgabenstellungen für die 5. Klasse (9. Jg.)?
- Welche (Programm)-Entwicklungsumgebung ist besonders zu empfehlen?
- Wie lange und in welchem Umfang soll das Thema Programmierung in einem geschätzten Rahmen von ca. 60-70 Unterrichtstunden behandelt werden?
- Was darf von den SchülerInnen am Beginn dieser Altersstufe vorausgesetzt werden?
- Welche Schülerleistungen sind im Programmierunterricht erwartbar?

Die eingangs erwähnte fachdidaktische Diplomarbeit der Lehramtskandidatin Marina Glatz aus Informatik unter dem Titel: "Visualisierung von elementaren Algorithmen im Informatikunterricht", die an zwei Schulen in vier 5. Klassen durchgeführt wurde, beschäftigt sich mit der Forschungsfrage, inwieweit die Visualisierung von grundlegenden Programmieraufgaben den Lernerfolg positiv beeinflusst. Ergebnisse aus dieser Studie werden in diesem Beitrag nur insoweit dargestellt, als sie in direktem Zusammenhang mit Schülerleistungen in grundlegender Programmierung stehen.

Der Ausdruck "grundlegende" Algorithmen ist mehr denn je gerechtfertigt, da bei dieser Studie keine Vorerfahrungen der SchülerInnen im Programmieren vorausgesetzt wurden. Es hat sich bei den Voruntersuchungen gezeigt, dass die TestschülerInnen in nur ganz wenigen Ausnahmefällen diesbezügliche Vorkenntnisse aus der gymnasialen Unterstufe (Sekundarstufe I) hatten. Dies ist nicht sonderlich überraschend, da bis auf wenige Ausnahmen der Programmierunterricht in der Sekundarstufe I, das ist die Altersstufe der

10 bis 14-jährigen SchülerInnen, an den AHS aus verschiedenen Gründen[10] fast nicht (mehr) existent ist. Die Frage, ob das ein wünschenswerter Zustand ist, wird an dieser Stelle, obwohl von Interesse, nicht erörtert. Um welchen programmiertechnischen Kern geht es in dieser Studie?

- Sichere Handhabung von Wertzuweisungen in Sequenzen
- Beherrschung von Vergleichsoperatoren in einfachen Abfragen
- Elementare Ereignissteuerung

Nicht viel, würde man meinen. Ein informatischer "Substandard", bestenfalls vergleichbar mit dem großen Einmaleins in der Mathematik. Die Ergebnisse allerdings, soviel kann bereits als wesentlichstes Ergebnis vorweg genommen werden, waren ernüchternd. Die bei den Tests gestellten Aufgaben wurden, nach einer - im Nachhinein betrachtet - viel zu kurzen Instruktionsphase (eine Doppelstunde), wenig zufrieden stellend gelöst.

- Was bedeutet X = Y?
- Ist Y=X das gleiche wie X=Y?
- Ergänze den Algorithmus, welcher die Inhalte der Zellen B2 und D2 vertauscht:

```
x = cells(2,2)     y = cells(2,4)
.....  .....  .....
cells(2,2) = x     cells(2,4) = y
```

- Ein verwirrter Tierhändler bietet seine Tierchen online an. Er hat die Fotos leider falsch zugeordnet. Die Datei vogel.jpg repräsentiert einen Fisch, die Datei fisch.jpg einen Vogel. Wie bringst du das in Ordnung?
- Ein einfacher Automat (Wechselschalter) besteht aus einem einzigen Button mit der Beschriftung AUS. Nach Betätigen dieses Buttons soll die Beschriftung auf EIN gesetzt werden usw. Wie realisierst du diesen Automaten?

Diese repräsentative Auswahl der Aufgabenstellungen soll einen Eindruck über das angepeilte Niveau vermitteln, um die anschließenden Ausführungen und Thesen einschätzen zu können.

Ist die scheinbar so triviale Wertzuweisung und der Tausch der Werte von Variablen überhaupt noch zeitgemäß und die klassische Trennung von Daten und Algorithmen in Zeiten von Objektorientierung für die Schulinformatik überhaupt noch ein Thema? Hartmann [HN02] bemerkt zu der überzogenen Erwartungshaltung im Hinblick auf

[10] Ein Grund ist der in den 90-er Jahren erfolgte Paradigmenwechsel vom algorithmenorientierten zum an Anwendungen orientierten Unterricht und einer beobachtbaren "ECDLisierung" nicht nur im Bereich der Sekundarstufe I.

realistische Schülerleistungen: "Es wird heute den Schulen nahegelegt, von Anfang an die Objektorientierung ins Zentrum der Ausbildung zu stellen. Dabei wird übersehen, dass Objektorientierung als Entwurfsmethode für große Softwaresysteme kaum für den Einstieg in die Informatik geeignet ist."

Es geht hier dezidiert um Vorbehalte bei der allzu raschen Einführung in objektorientierte Software-Entwicklung und nicht um die legitime Forderung nach moderat objektorientierten Sichtweisen bereits im Anfangsunterricht. Ich möchte an dieser Stelle in Zweifel ziehen, ob die objektorientierte Modellierung im Anfangsunterricht in Form von UML-Diagrammen soweit gehen darf, Anweisungen wie z. B. i++; ohne eine vorangegangene Vermittlung von Grundlagen über Daten und Wertzuweisungen stehen zu lassen[11]. Wurde im Mathematikunterricht jemals Analysis unterrichtet, ohne vorher algebraische Umformungen ausführlich behandelt und geübt zu haben? Übung macht den Meister, heißt ein Sprichwort. Der österreichische Komponist Anton Bruckner drückt das so aus: "Will man hohe Türme bauen, so muss man lange beim Fundament verweilen." Damit soll ausgedrückt werden, dass ohne ausgiebige "Trainingsphasen" mit einer einhergehenden Festigung grundlegender Programmierkenntnisse das ehrgeizige Vorhaben, SchülerInnen selbst an bescheidenen Softwareentwicklungsprozessen konstruktiv zu beteiligen, zum Scheitern verurteilt ist.

Auch der noch so trivial anmutende Tausch des Wertes zweier Variablen, in der informatischen Literatur auch als Ring- oder Dreieckstausch bezeichnet, bedeutet für die SchülerInnen eine nicht zu unterschätzende kognitive Leistung. Für die gesetzte, über den Dingen des informatischen Schulalltags stehende routinierte Informatik-Lehrkraft, die dies bereits verinnerlicht hat, weil sie ja bereits Jahre unterrichtet, mag dieses Fundament des Wertetausches als atomares Modul von Sortieralgorithmen ein kleiner Schritt im Unterricht sein. Für die SchülerInnen ist es in der Regel ein sehr großer.

5 Ein Blick zurück in das Jahr 1992

Um die Programmierung im gegenwärtigen Informatikunterricht besser verorten zu können, ist es lehrreich, einen kurzen Blick in die Vergangenheit zu wagen. Wer erinnert sich noch an die COMPED[12] – Studie 1992, die von der IEA[13] durchgeführt wurde? Es war die letzte groß angelegte Studie im Rahmen der internationalen vergleichenden Bildungsforschung zum Informatikunterricht [Ha94].

Diese Studie liefert wertvolle nationale und internationale Informationen zu den Rahmenbedingungen und die Praxis des Computereinsatzes an Schulen, Einstellungen von SchülerInnen und LehrerInnen sowie die Leistungen der SchülerInnen in "Informationstechnischer Grundbildung", in "Textverarbeitung" und im "Programmieren". Aus dieser

[11] Der Autor erlebte dies bei einem Vortrag eines Informatiklehrers anlässlich der fachdidaktischen Tagung in Königstein im März 2004.
[12] COMPED: Computers in Education Study
[13] IEA: International Association for the Evaluation of Educational Achievement, seit 1960. Als letztes OECD-Land ist Österreich 1989 der IEA beigetreten.

Studie möchte ich nur den letzten Teil herausgreifen, der für diesen Artikel Relevanz hat, nämlich die Schülerleistungen im "Programmieren" und hier auch nur das Segment der 14-jährigen SchülerInnen. Man lese und staune: Ein internationaler Vergleichstest, der Programmierkenntnisse für SchülerInnen bereits in dieser Altersstufe abfragt! Ich rufe in Erinnerung, dass es sich um das Jahr 1992 handelt. Aus den 20 gestellten Aufgaben seien hier sechs ausgewählt:

- Welcher der folgenden Ausdrücke wäre mathematisch korrekt, aber in einem Computerprogramm ist ein Fehler?

```
a)  x = y + z
b)   x + 2 = y
c)  x = x +z – 2
d)  x = x + 1
```

- Der folgende Programmteil wurde entworfen, um die Summe einer Anzahl von Werten zu ermitteln. Welche Reihenfolge der Anweisungen ist richtig?

```
1. s = 0                    a) 3,1,2,4
2. s = s + x                b) 1,3,4,2
3. for x = 1 to n           c) 1,3,2,4
4. next x                   d) 1,2,3,4
```

- Den Variablen p, q, r werden beliebige Werte zugewiesen. Dann wird folgender Algorithmus vom Computer durchgeführt:

```
Wenn p größer ist als q, dann                          Welche Zahl wird ausgegeben?
        Wenn p größer ist als r, dann setze t gleich p  a) die größte Zahl von p,q,r
        Sonst setze t gleich r                          b) die kleinste Zahl von p,q,r
Sonst                                                   c) die größere Zahl von p,q
        Wenn r kleiner ist als q, dann setze t gleich q d) die kleinere Zahl von q,r
        Sonst setze t gleich r
Drucke t
```

- Was ist der effizienteste Weg, die Summe von vorgegebenen Zahlen in einem Feld a(1), a(2), a(3), ... a(n) zu berechnen, wenn n sehr groß ist?

```
a) mit einem Ausdruck s = a(1)+a(2)+a(3)+ ... +a(n)   b) mit einer Auswahl-Anweisung
c) mit einer Schleifen-Anweisung                      d) mit einer Input-Anweisung
```

- Wähle den richtigen Output für die unten aufgelistete Prozedur.

```
a = 2    b = 3              a) 2,3,5
b = a + b                   b) 2,6,8
c = a + b                   c) 2,3,8
write a,b,c                 d) 2,6,5
```

- Die Praxisaufgabe: Schreibe ein Computerprogramm (BASIC, PASCAL, LOGO, usw.), das dich zuerst um die Eingabe von Länge und Breite (in Zentimetern) fragt und daraus die Fläche und den Umfang berechnet.

Der Informatikunterricht in Österreichs Gymnasien fand in den frühen 90-er Jahren für 13 bis 14-jährige SchülerInnen bis auf wenige Ausnahmen in unverbindlichen Übungen mit einem Anteil von ca. 60% statt, wobei in wiederum ca. zwei Drittel der 4. Klassen (8. Jg.) programmiert wurde und zwar im Mittel mehr als 20 Stunden im Schuljahr. Die Testaufgaben wurden international akkordiert. Allerdings wurde wegen Terminnot der Test nicht in allen an der COMPED-Studie teilnehmenden Staaten durchgeführt. Aber auch ein nur auf Salzburg beschränkter nationaler Test gab interessante Aufschlüsse. Diese Studie über die Programmierleistungen war im gleichen Maße interessant, wie das Ergebnis schlecht. Bedenkt man, dass die getesteten SchülerInnen angaben, bis zum Testzeitpunkt im Unterricht bis zu 40 Stunden programmiert zu haben, so muss der in das Testergebnis direkt eingeflossene Unterrichtsertrag als insuffizient bezeichnet werden. Dies umso mehr, als die wenigen guten Testleistungen das Ergebnis zusätzlicher Programmier-Übungen der SchülerInnen zu Hause waren. Die ernüchternde Quintessenz dieser nostalgischen Retrospektive ist wohl, dass damals Programmierung im Unterschied zu heute zwar einen beträchtlichen Teil des Informatikunterrichts ausmachte, die Erfolgserlebnisse und nennenswert nachhaltige Programmierkenntnisse bei den SchülerInnen aber nicht in korrespondierendem Ausmaß nachgewiesen werden konnten.

6 Ringtausch und Umfüllaufgabe

Nun könnte man bei großzügiger Auslegung des Lehrplanes der 5. Klasse interpretieren, auf Programmierung und das Variablenkonzept überhaupt verzichten zu können und es dabei bewenden zu lassen, SchülerInnen frei nach Hentig nur zu zeigen, wie schwierig das ist. Außerdem würden später die wenigsten vom "Programmieren" leben müssen. Das wäre allerdings zu kurz gegriffen. Dieser Ringtausch, auch Dreieckstausch, findet sich fast in jedem Lehrbuch über die Einführung in die Programmierung. So auch in [HU04, S 48], wo als Beispiel Zähler und Nenner eines Bruches unter Zuhilfenahme einer Hilfsvariablen getauscht werden. Viele SchülerInnen werden im Laufe ihrer schulinformatischen Ausbildung mit den drei Zuweisungen **h ← a; a ← b; b ← h** konfrontiert. Ich habe bewusst Zuweisungspfeile gewählt, um die Dynamik der Zustandsänderung auszudrücken. Die in den gängigen Programmiersprachen verwendete Darstellung der Zuweisung durch ein statisches " = "[14] erschwert das Verständnis in einer nicht zu unterschätzenden Weise. In einem allfälligen Pseudocode sollte das Gleichheitszeichen auf jeden Fall vermieden werden.

[14] Sowohl BASIC, als auch die gesamte C-Familie wie C++, Javascript, Java, PHP, Python verwenden dieses Zuweisungssymbol. Pascal bzw. Delphi bildet hier mit dem Operator " := " eine Ausnahme.

Abbildung 2: GW-Basic

```
Python 2.4 (#60, Nov 30 2004
Type "copyright", "credits"

IDLE 1.1
>>> a = 2
>>> b = 5
>>> a,b = b, a
>>> print a, b
5 2
```

Abbildung 3: Python-Source

Ist es denn überhaupt notwendig, sich über diese Zuweisungsfolge Gedanken zu machen, wo es doch bereits vor über 20 Jahren in BASIC die im Sprachumfang implementierte "SWAP-Prozedur" gab (Abbildung 2). Es gehört zu den Rätseln der "Entwicklungsumgebungsentwickler", warum dies in Visual Basic nicht mehr der Fall ist. Vielleicht aber waren Didaktiker am Werk, um Informatiklehrkräften mit diesem Beispiel das Modularkonzept förmlich aufzudrängen, in dem die häufig benötigte Routine des Wertetausches, als Modul (Funktion, Prozedur) implementiert, für weitere Algorithmen als Bibliotheksmodul zur Verfügung steht? Eine raffinierte Zuweisungsvariante ist in der Sprache Python implementiert. Die Zuweisung a, b = b, a vertauscht die Werte der Inhalte der Variablen a und b (Abbildung 3). So einfach geht das, wenn man es weiß.

Aber was lernt der Schüler daraus? Soll ihm der beschwerlichere Weg, der auch in Python gegangen werden kann, den Ringtausch vorzuenthalten, erspart werden? Das wirft natürlich die grundsätzliche Frage auf, wie weit im Informatikunterricht Black-box-Denken erlaubt sein soll. Wo ist es legitim, auf einer höheren Abstraktionsstufe anzusetzen und wo erforderlich, "black boxes" zu öffnen und die zu Grunde liegenden Algorithmen zu verstehen?

Muss es überhaupt ein Ringtausch sein? Kann das Problem des Platztausches der Werte zweier (numerischer) Variablen ausschließlich mit einer Hilfsvariablen gelöst werden? Bei dieser Frage betreten wir die Domäne der Denksportaufgaben.

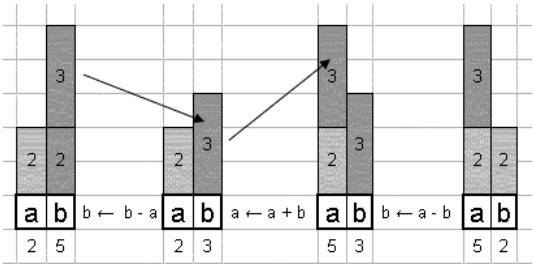

Abbildung 4: Tausch der Inhalte zweier numerischer Variablen

Die Zuweisungsfolge b ← b – a; a ← a + b; b ← a – b tut es, wie aus Abbildung 4 ersichtlich, für numerische Variablen auch.

Ein weiterer reizvoller Weg, sich die Algorithmik des "Platztausches" jenseits des Variablenkonzepts zu erschließen, ist der in den neuen bayerischen Informatik-Lehrbüchern für den 6./7. Jg. empfohlene Karel, der Roboter" [Ka05], der die (allerdings nicht triviale) Aufgabe lösen muss, zwei verschieden hohe Ziegelhaufen zu vertauschen, der in Abbildung 4 gestellten "Umfüllaufgabe" nicht unähnlich. Schließlich könnte diese Umfüllaufgabe auch mit dem legendären "Knowhow-Computer" [Ba97] von Back und Rohde realisiert werden. Dieser Modellcomputer (Abbildung 5) ist ein Beispiel für eine so genannten Registermaschine mit ganzen fünf Anweisungen (Sprungbefehl, Vergleich, In- und Dekrementierer, Stop-Anweisung).

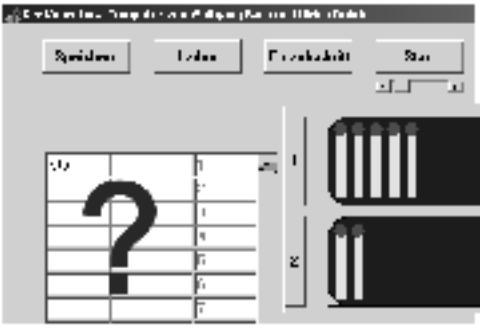

Abbildung 5: Knowhow-Computer

In diesem Kapitel ist nur angedeutet worden, welch reichhaltiges, didaktisch wertvolles Potential bereits in (einfachen) Aufgabenstellungen, wie dem Tausch der Inhalte zweier Variablen steckt. Die Legitimation, den "Platztausch" im Informatikunterricht zu thematisieren, ist aber auch durch unzählige lebensweltliche Anwendungen und die Tatsache gegeben, dass er das Fundament vieler Algorithmen darstellt, in denen es um Misch- sowie Sortierverfahren geht.

7 Resümee

Die 5. Klasse an den AHS Österreichs ist an der Schnittstelle zwischen Unterstufe und Oberstufe angesiedelt. Mit dieser Klasse endet auch die offizielle Schulpflicht und beginnt die Frage, was von einem 15-Jährigen an informatischer Bildung und Kompetenz erwartet werden darf. Neben der Schulung von Standardsoftware, die in vielen Fällen auf den ECDL (Europäischer Computerführerschein) ausgerichtet ist und auch einen (derzeit noch nicht absehbaren) Einfluss auf die Inhalte des Informatikunterrichts an den AHS hat, sollen im Rahmen von weniger als 70 Unterrichtsstunden auch noch Grundprinzipien über Automaten, Algorithmen und Programme vermittelt werden. Gibt der ECDL in Form eines detaillierten Syllabus mit weit über hundert operationalisierten Lehrzielen und den darauf ausgerichteten Tests ganz klar vor, was an Schülerleistungen für eine allfällige Zertifikatserreichung einzufordern ist, so stehen wir in den kerninformatischen Bereichen, an der Schnittstelle zwischen Anwenderschulung und Anwendungsprogrammierung, auch nach mehr als 20 Jahren vor offenen Problemen.

Es besteht österreichweit kein Konsens darüber, in welchem Ausmaß Algorithmen und Programmierung im Informatikunterricht der 5. Klasse eine Rolle spielen sollen[15]. Wenn man sich auf dieser Altersstufe auf Programmierung im weitesten Sinne einlässt, muss sich der/die Informatik-Lehrende über folgende Thesen und Forderungen im Klaren sein:

- Informatikunterricht, in dem aktive Programmierleistungen und Problemlösestrategien der SchülerInnen eingefordert werden, ist schwer.

- Es muss überlegt werden, welche diesbezüglichen Lehrziele im Sinne von Reproduktion und Transferleistungen der/die (durchschnittliche) Schüler/in erreichen soll.

- In der Unterrichtsmethodik gibt es keinen Königsweg. Die oben genannte Studie hat gezeigt, dass der Unterrichtsertrag unabhängig von der gewählten Unterrichtsmethode[16] in Sinne der Erreichung konkreter Testziele unbefriedigend ist.

- Daher müssen in der Unterrichtsplanung unbedingt entsprechende Übungs- und Festigungsphasen in Form von lösbaren Aufgabenstellungen berücksichtigt werden. Das kostet Zeit.

- Informatiklehrende stehen je nach persönlichem Interesse vor dem Problem der adäquaten Werkzeugwahl. Das Spektrum vieler mehr oder weniger geeigneter Entwicklungsumgebungen, die von Pascal, Delphi, Visual Basic, Javascript bis Python reichen, macht diese Wahl nicht einfacher.

- Der Informatikunterricht darf sich nicht in einseitiger Werkzeug-Schulung erschöpfen.

Die Dagstuhler Empfehlung [Da04] betont die Bedeutung von Standards für Ziele, Inhalte und Methoden eines verpflichtenden Informatikunterrichts und fordert die Einbeziehung der Informatik in Schulleistungsstudien wie z. B. PISA oder einer an das 21. Jahrhundert angepassten aktuellen COMPED-Studie. Die Definition dieser Standards in der Alterstufe der 15-Jährigen auf internationaler Ebene könnte sich nachhaltig und normativ auf die Unterrichtsgestaltung im Hinblick auf Algorithmen und Programmierung auswirken. Das Erarbeiten der Standards ist im Hinblick auf die enorme Bandbreite an Software-Werkzeugen kein einfaches Unterfangen, die Destillation konzeptueller informatischer Fundamente und deren Überprüfbarkeit ein noch schwierigeres. Allein im Bereich der elementaren Algorithmen wie dem Wertetausch zweier Variablen steckt (noch immer) ein großes didaktisches Potential. Der im Titel dieses Beitrags sug-

[15] Die Extreme, dass fast ein ganzes Schuljahr lang in der 5. Klasse einer österreichischen AHS nur in einer Programmiersprache programmiert wird, sollten eigentlich der Vergangenheit angehören. Auszuschließen ist es aber nicht. Anders sieht es in den höheren technischen Lehranstalten (HTBLA) in dieser Altersstufe aus, wo z. T. die Lehrpläne konkreten Programmierunterricht in einer dezidierten Sprache vorgeben.

[16] Die SchülerInnen wurden in drei Gruppen mit jeweils unterschiedlichen Methoden (Animation, interaktive Konstruktion, Arbeitsblätter) unterrichtet.

gerierte kleine Schritt für LehrerInnen könnte sich dann allerdings leicht zu einem Riesenschritt auswachsen.

Literaturverzeichnis

[Ba97] http://www.wolfgang-back.com/knowhow_home.php (geprüft am 20.2.2005).

[CD05] CDA-Sonderheft 5/2005, Micheuz, P. (Hrsg.), CDA-Verlag, Linz, 2005.

[Da04] Zweite Dagstuhler Empfehlung vom 24.9.2004 (geprüft am 20.2.2004) http://ddi.uni-paderborn.de/didaktik/gi/dagstuhl_2004/dagsthuhl_empfehlung_2004l.pdf.

[EX03] Deutsches Institut für Internationale Pädagogische Forschung, Frankfurt a. M., 2003.

[Ha94] Haider G., Schule und Computer, Informationstechnische Grundbildung in Österreich, Österreichischer Studienverlag, Innsbruck, 1994.

[HN02] Hartmann, Nievergelt: Informatik und Bildung zwischen Wandel und Beständigkeit, Informatik-Spektrum, Dez. 2002.

[He93] von Hentig H., Die Schule neu denken, Beltz 1993, S. 69.

[Hu00] Hubwieser P, Didaktik der Informatik, Springer, Berlin 2000, S. 105.

[Hu04] Hubwieser P. Fundamente der Informatik, Oldenburg, München, 2004, S. 47.

[Ka05] http://www.schule.bayern.de/karol (geprüft am 20.2.2005).

[ILP04] http://www.bmbwk.gv.at/medienpool/11866/lp_neu_ahs_14.pdf - Informatik-Lehrplan AHS-Oberstufe Österreich (geprüft am 20.2.2005).

[NB03] Expertise zur Entwicklung nationaler Bildungsstandards, 2003.

[OW96] Ottmann T. Widmayer P., Algorithmen und Datenstrukturen, 3. Auflage. 1996, Spektrum Akademischer Verlag, Heidelberg u. a.

[Re94] Rechenberg P., Was ist Informatik, Hanser Verlag, S. 264.

[Re03] Reiter, Schulinformatik in Österreich, Ueberreuter-Verlag, Wien, S. 150.