universität wien

DIPLOMARBEIT / DIPLOMA THESIS

Titel der Diplomarbeit / Title of the Diploma Thesis

„Computational Thinking im kompetenzorientierten Informatikunterricht (5. AHS)“

verfasst von / submitted by

Martha Ruhsam

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Magistra der Naturwissenschaften (Mag. rer. nat.)

Wien, 2018 / Vienna, 2018

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	A 190 299 884
Studienrichtung lt. Studienblatt / degree programme as it appears on the student record sheet:	Lehramtsstudium UF Psychologie und Philosophie UF Informatik und Informatikmanagement
Betreut von / Supervisor:	Ao. Univ.-Prof. Dipl.-Ing. Dr. Renate Motschnig

Eidesstattliche Erklärung

„Hiermit erkläre ich, dass ich diese Arbeit selbstständig verfasst habe. Ich habe alle verwendeten Quellen und Hilfsmittel entsprechend angegeben und alle nicht von mir stammenden Bilder oder Screenshots - aus dem Internet oder Büchern - unter Angabe der Quelle gekennzeichnet. Alle Stellen der Arbeit, die anderen Werken im Wortlaut oder dem Sinn nach entnommen wurden, sind als Entlehnung unter Angabe der Quelle kenntlich gemacht.“

11. Februar 2018

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation und Relevanz des gewählten Themas

Oft scheint das Finden von Lösungen komplexer algorithmischer Probleme im IT-Sektor eine unüberwindbare Hürde darzustellen, die nicht selten mit großen kognitiven Anstrengungen verbunden wird bzw. auch ist. Die Anwendung von Computational Thinking (= CT), u.a. einer Problemlösestrategie, fungiert dabei unterstützend: „Computational thinking is reformulating a seemingly difficult problem into one we know how to solve, perhaps by reduction, embedding, transformation, or simulation." (Wing, 2006: S. 33) D.h., das zu Beginn scheinbar schwere Problem wird modifiziert und beispielsweise in mehrere kleinere Probleme unterteilt, welche die Anwenderin/der Anwender ohne größere Anstrengungen lösen kann.

CT kann aber nicht nur im IT-Bereich, sondern auch in anderen, ganz grundlegenden Disziplinen, wie z.B. Lesen, Schreiben oder Mathematik lt. Jeannette M. Wing als eine Art der Problemlösung Verwendung finden. Sie betont zudem, „Computational thinking is a fundamental skill for everyone, not just for computer scientists." (Wing, 2006: S. 33)

Dieser sehr kleine Einblick in die Materie begründet bereits die Motivation einer tiefer gehenden Beschäftigung mit CT, welche im Rahmen vieler laufenden Projekte - vor allem in Schulen - in z.B. den Vereinigte Staaten, Mexico, dem Vereinigten Königreich, der Schweiz, Österreich, etc. seit einigen Jahren stattfindet.

Diese Projekte werden von vielen namhaften Unternehmen (u.a. BBC und Google) finanziell, aber auch mit Kursen für Lehrkräfte oder der Zurverfügungstellung von Unterrichtsmaterialien gefördert. Die dahinterstehende Motivation beschreibt folgender Satz von Alan Bundy und Jeremy Scott (2015: S. 37) sehr treffend: „It is important to create a community of tool*makers* rather than tool *users*." Daraus geht hervor, dass der schnelle technische Fortschritt von den nachkommenden Generationen nicht nur eine hervorragende Anpassungsfähigkeit, sondern auch das Können neue Innovationen zu entwickeln und diese zu warten bzw. zu reparieren abverlangt. Um also zukünftig im Beruf und im Alltag erfolgreich sein zu können, ist das Anwenden von CT zwingend notwendig.

Auch von Seiten vieler Organisationen und gemeinnütziger Vereine wird Unterstützung für die Vermittlung von CT angeboten. In Österreich beispielsweise setzt sich besonders die *Österreichische Computergesellschaft* (= OCG) für die Verbreitung in Schulen ein. Bereits

im April 2016 organisierte diese in Kooperation mit dem *Council of European Professional Informatics Societies* (= CEPIS) eine Mini-Conference mit dem Titel „Computational Thinking and Coding at Schools“. Themen waren u. a. neben den Herausforderungen CT in die Schulen zu bringen bzw. die PädagogInnen zu motivieren CT zu lehren, auch bereits verbuchte Erfolge, wie z.B. neue, angepasste Lehrpläne im Vereinigten Königreich oder die „Computer Science Education Week” in den Vereinigten Staaten, in der Schweiz und in Österreich. (OCG, 2016b)

Auch die erste Wiener Fortbildungsveranstaltung, speziell für Lehrkräfte, wurde von der OCG, diesmal in Kooperation mit IBM Wien, organisiert und fand im Februar 2017 statt: „Informatisches Denken und Programmieren“. Inhalte waren u.a. die Klärung des Begriffs CT und Beispiele für dessen spielerische Vermittlung. (OCG, 2017a)

1.2 Vorgangsweise und Aufbau der Arbeit

Die vorausgehende Idee dieser Diplomarbeit war es, die Inhalte des Begriffs „Computational Thinking“ zu verinnerlichen, um diese anschließend aktiv in eigene Unterrichtseinheiten einbauen zu können. Dies erforderte neben einer sehr tiefgreifenden Beschäftigung mit aktueller Literatur über CT auch den Besuch diverser Veranstaltungen das Thema betreffend (z.B. Mini-Conference des OCG (April 2016), Gödel Lecture (Vortragende: J. M. Wing; Juni 2016), Fortbildungsveranstaltung „Informatisches Denken und Programmierung“ (Februar 2017), etc.). Weiters bedurfte es der Beschäftigung mit Tools, die bei der Vermittlung von CT unterstützend wirken (können) und einer detaillierten Auseinandersetzung mit den aktuellen informatischen Lehrplänen in Österreich - im Speziellen mit jenem für die 5. Klasse der allgemeinen höheren Schule.

Der daraus resultierende Aufbau der Arbeit ist an diese Vorgangsweise angelehnt: Im Kapitel *2 Basiskonzepte des Computational Thinking und deren Vermittlung* ist neben einer Definition auch eine Gegenüberstellung der entwickelten Schlüsseltechniken bzw. Konzepte und Methoden inkl. Beschreibungen einer Auswahl dieser angeführt. Die Darstellung der zahlreichen Anwendungsbereiche bzw. -fächer erfolgt im Anschluss für Mathematik, Musik und das Erlernen von Sprachen unter Verwendung von passenden Beispielen. Auf Grund des besonderen Fokus (d.i. die Vermittlung von CT im Unterrichtsfach Informatik) ist auch eine Sammlung von ausgewählten (unterstützenden) Tools zu finden, welche jeweils eine kurze Beschreibung dieser beinhalten.

Um Unterrichtseinheiten für das Fach Informatik erarbeiten zu können, die neben der Ver-

mittlung von CT auch noch auf dem aktuellen, kompetenzorientierten Lehrplan basieren, ist die Erarbeitung einiger Begrifflichkeiten rund um das Thema „Kompetenzen“ und eine Analyse der Inhalte des gegebenen Lehrplans im darauffolgenden Kapitel *3 Kompetenzorientierter Lehrplan für das Unterrichtsfach Informatik in Österreich (5. AHS)* unabdingbar.

Der letzte Teil dieser Arbeit stellt eine Sammlung geplanter Unterrichtseinheiten dar, welche unter Anwendung der Lerntheorie „Konstruktionismus“ und der Verwendung von pädagogischen Patterns entwickelt wurden. Die gewählten Themenbereiche sind aus dem Lehrplan der 5. Klasse AHS entnommen und werden auf ihre tatsächliche Durchführbarkeit an einer Schule in Wien überprüft. Anschließend erfolgt eine schriftliche Reflexion des Erlebten und falls erforderlich eine Korrektur der geplanten Sequenzen auf Basis dessen. Alle vorbereiteten Materialien (inkl. Lösungen) und Beschreibungen, wie CT in den jeweiligen Situationen zum Einsatz kommt, sind dem Anhang zu entnehmen.

1.3 Verfolgte Ziele

Primäres Ziel bei der Verfassung dieser Arbeit ist neben der Darstellung der Wichtigkeit von CT auch die Entwicklung von Ressourcen unter Anwendung von CT, welche Lehrende des Unterrichtsfachs Informatik unterstützend zur Verfügung stehen.

Nach eingehender Recherche zum Begriff „Computational Thinking“ stellt auch das verständliche Darstellen der unterschiedlichen Auslegungen des Begriffs einen wichtigen Teil in dieser Arbeit dar. Ein weiteres großes Anliegen ist das Aufzeigen der unzähligen Tools, mit Hilfe deren die Vermittlung von CT erfolgen kann.

Ein erst etwas später (Herbst 2017) dazugestoßenes Ziel hat mit der derzeitigen Pilotierung der „Digitale Grundbildung“ zu tun. Diese soll voraussichtlich ab dem Schuljahr 2018/19 in Österreich in der Sekundarstufe 1 eingeführt werden. Dafür stehen jedoch keine zusätzlichen Einheiten zur Verfügung, d.h. die Vermittlung der Inhalte des Lehrplans erfolgt entweder anstelle bereits vorhandener Unterrichtsfächer oder integrativ in diesen, d.h. von den jeweiligen Fach-Lehrkräften. (*Anmerkung*: Die genaue Durchführung der „Digitalen Grundbildung“ ist schulintern zu regeln, d.h. die Vermittlung der Inhalte ist nicht auf ausgewählte Fächer begrenzt.) (BMB, 2017e) Da ein Teil des Lehrplans CT und dessen Vermittlung beinhaltet, ist es auch ein Ziel konkrete Beispiele zur Vermittlung von CT in anderen Unterrichtsfächern in dieser Diplomarbeit zu beschreiben und somit die Möglichkeit diese problemlos in nicht informatische Inhalte zu integrieren, aufzuzeigen.

2 Basiskonzepte des Computational Thinking und deren Vermittlung

Jeannette M. Wing legte in ihrem Artikel „Computational Thinking“, welcher in „Communications of the ACM“ im März 2006 erschienen ist, den eigentlichen Grundstein für Forschungen zum bzw. rund um das Thema.

Besonders betont sie, „Computational thinking is a fundamental skill for everyone, not just for computer scientists.“ (Wing, 2008: S. 33) Es sollte, genauso wie das Lesen, das Schreiben und das Rechnen, zu den grundlegenden analytischen Fähigkeiten jedes Kindes zählen. Wing begründet diese Aussage in etwa wie folgt: Es muss/soll die Möglichkeit bestehen, dass geistige Herausforderungen verstanden und gelöst werden können und die Beteiligung bei wissenschaftlichen Schwierigkeiten problemfrei funktioniert. Eine Begrenzung des Problem- bzw. Lösungsbereichs sollte nur durch unsere eigene Neugier und Kreativität stattfinden. (Wing, 2008: S. 35)

Weiters behaupten Punya Mishra und Aman Yadav (Voogt et al, 2013: S. 720), dass die Anwendung von CT einfache Konsumenten der Technologien zu kreativen Erstellern der Tools werden lässt, welche neue Formen der Ausdrücke designen und Kreativität fördern.

Auf Grund dessen hat es sich Jeannette M. Wing zum Ziel gemacht, CT bis Mitte des 21. Jahrhunderts als grundlegende analytische Fähigkeit in das Leben der Menschen zu integrieren. (Wing, 2014; TU Wien, 2016: 00:12:42 – 00:13:00)

2.1 Computational Thinking – Begriffsklärung und Erklärung

Im Artikel „Computational Thinking Benefits Society” (Wing, 2014) und im Vortrag an der Technischen Universität Wien im Juni 2016 bezeichnet Wing CT als den „[…] thought process involved in formulating a problem and expressing its solution(s) in such a way a computer – human or machine – can effectively carry on.”

CT beschreibt somit einen Denkprozess, welcher eine Formulierung des Problems inklusive einer rechnerischen Lösung als Ergebnis hat.[1] Dabei erfolgt die Annahme, dass der mentale Vorgang von einer Person und keiner Maschine durchgeführt wird. Das eigentliche Ergebnis berechnet jedoch der Computer.[2] (Wing, 2014; TU Wien, 2016: 00:17:10 – 00:18:00)

[1] Bei der Lösung eines Problems mittels CT wird dieses neu formuliert und unter Zuhilfenahme von Reduzierung, Einbettung, Transformation oder Simulation in eine Problematik umgewandelt, welche bereits bekannt und lösbar ist. (Wing, 2006: S. 33)

[2] Dieser Terminus muss allerdings nicht zwingend eine Maschine, sondern kann auch einen Menschen, darstellen. (Wing, 2014; TU Wien, 2016: 00:17:10 – 00:18:00)

Weiters kritisch zu hinterfragen sind die in dieser Definition von Wing verwendeten technischen Begriffe:

- Das Wort „expressing“ drückt eine sprachliche Repräsentation aus, welche für den Zweck einer kommunizierten Lösung zwischen Personen oder Maschinen besteht. Die Spezifikationen, die Verifikationen oder die Ausdruckskraft einer Sprache, beispielsweise einer Programmiersprache, begründen oft einen Unterschied zwischen einer eleganten bzw. einer nicht eleganten Lösung. (Wing, 2014; TU Wien, 2016: 00:18:10 – 00:18:19)
- „Effectively“ muss im Kontext des Modells der Turing Maschine[3] für Berechnungen betrachtet werden. Somit beschreibt Wing mit diesem Ausdruck also die Berechenbarkeit, die Entscheidbarkeit/die Bestimmbarkeit oder die Rekursivität. (Wing, 2014) In ihrem Vortrag (TU Wien, 2016: 00:18:00 – 00:18:10) erwähnt sie außerdem, dass das Wort „effectively“ in der CS nur dann Verwendung findet, wenn über tatsächlich effektiv lösbare Probleme gesprochen wird. Dies ist in diesem Kontext nicht zwingend der Fall.

Wing ordnet somit CT überwiegend dem Repertoire der Problemlösestrategien zu. Zudem beinhaltet CT auch das Designen von Systemen und das Verstehen von menschlichem Verhalten. (Wing, 2006: S. 33) Dabei wird immer auf die fundamentalen Konzepte von Computing[4] zurückgegriffen. (Wing, 2008: S. 3717)

Aus dieser Definition geht hervor, dass CT eine Art des analytischen Denkens darstellt und einige Teile bzw. Arbeitsweisen mit bereits davor existierenden Denkweisen teilt:

- **Mathematisches Denken**: Lösen eines Problems. (Wing, 2008: S. 3717)
  Die Mathematik, genauso wie CT, ist besonders auf den Prozess der Abstraktion fokussiert. Der bedeutende Unterschied stellt die Möglichkeit dar, dass bei CT den Abstraktionen „Leben eingehaucht“ werden kann. Das bedeutet, Maschinen operationalisieren diese Abstraktionen und in Folge dessen kann Verhalten simuliert und vorhergesagt werden. (TU Wien, 2016: 00:19:00 – 00:19:55)

[3] „Die Turingmaschine ist ein sehr einfaches abstraktes Modell eines Computers – das gleichwohl mächtig genug ist, alles zu berechnen, was berechenbar ist.“ (Lang, 2016)
[4] Unter Computing versteht Wing im weitesten Sinn jenen Bereich, in welchem Computer Science, Computer Engineering, Kommunikationen, Information Science und Informationstechnologie zugeordnet werden. (Wing, 2008: S. 3717) Weiters steht Computing in Verbindung zur Beantwortung folgender Frage: Wie bringe ich einen Computer (Mensch oder Maschine) dazu das gegebene Problem zu lösen? (Wing, 2008: S. 3719)

- **Technisches Denken**: Designen und evaluieren von großen, komplexen Systemen, welche mit den Beschränkungen der realen Welt interagieren. (Wing, 2008: S. 3717) D.h. CT baut auf der Macht und der Beschränktheit von Rechenprozessen auf, unabhängig davon ob diese von einem Menschen oder von einer Maschine ausgeführt werden. (Wing, 2006: S. 33) Jedoch sind die physikalischen Beschränkungen bei Computer, im Gegensatz zum Menschen, klar ersichtlich. (TU Wien, 2016: 00:20:24–00:21:57)
- **Wissenschaftliches Denken:** Verstehen von Berechenbarkeit, Intelligenz und das menschliche Verhalten. (Wing, 2008: S. 3717)

Zuzüglich zu diesen „geborgten" Teilbereichen der einzelnen Denkweisen beinhaltet CT eine spezielle Eigenschaft, welche keine andere technische Disziplin besitzt. Wing (TU Wien, 2016: 00:22:35 - 00:23:09) bezeichnet diese als den großen Unterschied, welcher zwischen der CS und den restlichen technischen Disziplinen besteht: Software.

Software erlaubt es virtuelle Welten zu erbauen, in welchen die physikalischen Gesetze so definiert werden können, wie dies gewünscht wird, d.h. mit Hilfe der Software kann alles gebaut werden. Eine Begrenzung der möglichen Bedingungen und Gegebenheiten ist nur durch die Kreativität des Computer Scientist bzw. des Entwicklers gegeben.[5] (TU Wien, 2016: 00:23:09 – 00:23:33)

2.1.1 Die Schlüsseltechniken bzw. Konzepte und Methoden von CT

Neben Jeannette M. Wing haben sich noch viele weitere WissenschaftlerInnen mit dem Thema *Computational Thinking* und dessen Erlernen auseinandergesetzt. Die sich dadurch entwickelte Vielfalt von Techniken bzw. Methoden soll mit folgender (chronologisch geordneten) Tabelle verdeutlicht werden. (Diese wurde von (Bocconi, S. & Chioccariello, A. & Dettori, G. & Ferrari, A. & Engelhardt, K. & Kampylis, P. & Punie, Y., 2016a: S. 17) entnommen und um Einträge ergänzt. Außerdem stellt die angeführte Tabelle keinen Anspruch auf Vollständigkeit.) Die grau hinterlegten Spalten kennzeichnen jene Konzepte, auf welche im Folgenden genauer eingegangen wird.

[5] Einige Lehrende der CS argumentieren in ihren Artikeln (Yadav et al., 2011: S. 466; Lu & Fletscher, 2009: S. 261), dass das Erlernen bzw. Beherrschen des Programmierens keine Rolle bei dem Lehren von CT spielt. Lu und Fletscher (2009: S. 261) erwägen sogar, dass ein Programmierschwerpunkt beim Erlernen von CT die Lernenden abschrecken könnte sich überhaupt für CS zu interessieren. (Voogt, Fisser, Good, Mishra & Yadav, 2013: S. 720)

Wing (2008)	**Barr & Stephenson (2011)**	**Lee et al. (2011)**	**Grover & Pea (2013)**	**Selby & Woollard (2013)**	**Barefoot (2014a)**	**BBC (2016a) Google (2016)**	**Angeli et al. (2016)**
Abstraction	Abstraction	Abstraction	Abstractions and pattern generalizations	Abstraction	Abstraction	Abstraction	Abstraction
	Algorithms & procedures		Algorithmic notions of flow of control	Algorithmic thinking	Algorithms	Algorithms	Algorithms (including Sequencing and Flow of control)
Automation	Automation	Automation					
		Analysis					
			Conditional logic		Logic		
Decomposition	Problem Decomposition		Structured problem decomposition (modularizing)	Decomposition	Decomposition	Decomposition	Decomposition
			Debugging and systematic error detention		Debugging		Debugging
			Efficiency and performance constraints	Evaluation	Evaluation		
				Generalizations			Generalization
			Iterative, recursive and parallel thinking				
	Parallelization				Pattern recognition	Pattern recognition	
	Simulation						
			Symbol systems and representations				
			Systematic processing of information				
					Tinkering		
					Creating		
					Perserving		
					Collaborating		

2.1.1.1 Jeannette M. Wing

Wing beschreibt in ihren wissenschaftlichen Artikeln drei Konzepte bzw. Methoden, welche für die Anwendung von CT unabdingbar seien – *abstraction, automation* und *debugging*. (Wing, 2006 & 2008)

- *Abstraction*: Die Abstraktionen, welche hier Verwendung finden, sind tendenziell komplexer und inhaltsreicher[6] als jene, die in Wissenschaften, wie beispielsweise der Mathematik oder der Physik, auftreten. Dies trifft zu, da die hier angewandten Abstraktionen notwendigerweise nicht aus reinen, eleganten oder einfach definierbaren algebraischen Eigenschaften bestehen, wie z.B. mathematische Abstraktionen (d.s. beispielsweise reale Zahlen oder die Zahlenmengen der physikalischen Welt). Weiters sind die zu benutzenden Abstraktionen bei CT so implementiert, dass diese innerhalb den Beschränkungen der physikalischen Welt arbeiten, d.h. Randfälle bzw. fehlgeschlagene Fälle müssen berücksichtigt werden.

  Im Abstraktionsprozess selbst wird entschieden welches Detail hervorgehoben werden muss bzw. welches Detail ignoriert werden kann. Dieser Vorgang unterliegt dem CT: Grundsätzlich wird immer gleichzeitig mit mindestens zwei Abstraktionsschichten, dem „layer of interest“ und dem „layer below“ bzw. dem „layer above“, gearbeitet. Um ein komplexes System (z.B. das Internetprotokoll (TU Wien, 2016: 00:20:03 - 00:20:20)) generieren zu können, ist ein gut definiertes Interface zwischen den einzelnen Schichten notwendig. Die Beziehungen, die zwischen jedem Schichtenpaar bereits bestehen, müssen beim Arbeiten mit den Schichten der Abstraktion immer berücksichtigt werden. Sie können mittels einer Abstraktionsfunktion, einer Simulationsbeziehung, einer Transformation oder einer generelleren Art der Zuordnung[7] festgehalten werden.

  Resümierend kann somit festgestellt werden, dass die praktischen Grundlagen die Abstraktionen von CT definieren. Diese arbeiten mit mehreren Abstraktionsschichten und verstehen die Beziehungen zwischen den verschiedenen Schichten. (Wing, 2008: S. 3718)

[6] Bei dem Arbeiten mit inhaltsreichen Abstraktion ist das Definieren der „richtigen“ Abstraktion kritisch. (Wing, 2008: S. 3718)

7 Zuordnungen können beispielsweise verwendet werden um die bemerkenswerte Äquivalenz zwischen einem abstrakten, endlichen Automaten und einer der möglichen Verfeinerungen zu zeigen. Dies geschieht durch den Beweis der Richtigkeit einer Implementation mit Achtung einer Spezifikation und einer Kompilierung eines Programms für effizienteren Maschinencode, welches in einer high-level Sprache geschrieben ist. (Wing, 2008: S. 3718)

- *Automation*: Die Abstraktion stellt eine Auswahl mentaler Fähigkeiten von Menschen dar, welche die Breite des Feldes Computing reflektieren und das Wesen von CT wiederspiegeln. (Wing, 2006: S. 33; Wing, 2008: S. 3718) Diese mentalen Fähigkeiten können durch technologische Hilfsmittel (Das ist nicht zwingend notwendig.) verstärkt werden, d.h. eine Automatisierung der Abstraktionen ist möglich. (Dies definiert Wing als Computing.) Weiters impliziert die Automatisierung auch die Möglichkeit der Abstraktionsinterpretation durch den Menschen bzw. den Computer. (Wing, 2008: S. 3718)

  Die Möglichkeit zur Durchführung der Automatisierung besteht jedoch nur dann, wenn die Abstraktionen, ihre Schichten und ihre Beziehungen (von den technologischen Hilfsmitteln) verstanden werden können. Sichergestellt wird dies mit Hilfe der Verwendung von detaillierten und genauen Modellen und Notationen. (Wing, 2008: S. 3718)
- *Decomposition***:** Wing beschreibt diese als eine der vorhandenen Abstraktionsschichten. (Siehe *Abstraction.*) (Wing, 2008: S. 3719-3720)

2.1.1.2 Irene Lee, Fred Martin, Jill Denner, Bob Coulter, Walter Allan, Jeri Erickson, Joyce Malyn-Smith und Linda Werner

Um neuartige Probleme mithilfe von CT lösen zu können sind *abstraction*, *automation* und *analysis* für Lee et al. (2011: S 33) Schlüsselbegriffe:

- Der Begriff *abstraction* beschreibt bei der Problemlösung unter Anwendung von CT den Prozess des Aufzeigens der grundlegenden Elemente eines Problems. Üblicherweise wird er auch angewandt um die häufigsten Charakteristika zusammenzufassen und zu beschreiben. Diese können anschließend verwendet werden um alle anderen Umstände zu repräsentieren. (Lee et al., 2011: S. 33)
- *Automation* ist ein Prozess, in welchem ein Computer (schneller als dies der Mensch je zu können vermag) einzelne, sich wiederholende Teilprobleme abarbeitet, d.h. es liegt eine Automatisierung der Abstraktion vor. (Lee et al., 2011: S. 33)
- Nach der erfolgreichen Anwendung der beiden vorgegangenen Schlüsselbegriffe wird in der *analysis* reflektiert, ob tatsächlich die Abstraktionen korrekt durchgeführt wurden. Mögliche Überlegungen können sein: Wurden die richtigen Annahmen getroffen um das Problem auf seine wichtigsten Charakteristiken zu minimieren? oder Ist auf wichtige Faktoren vergessen worden? (Lee et al., 2011: S. 33)

Lee et al. (2011: S. 33) erklären in ihrem Artikel „Computational Thinking for Youth in Practice“ außerdem anhand von Beispielen die Anwendung der beschriebenen Begriffe in drei Bereichen, d.s. Modeling & Simulation, Robotics, Game Design & Development.

2.1.1.3 BBC und Google

BBC (2016) und Google (2016) formulieren neben den Schlüsseltechniken von Wing (d.s. *abstraction, automation* und *decomposition*) noch eine weitere auf ihren Lernplattformen, d.i. *pattern recognition*. Die Beschreibung der Schlüsseltechniken wird von BBC und Google jedoch etwas anders vorgenommen als von Wing:

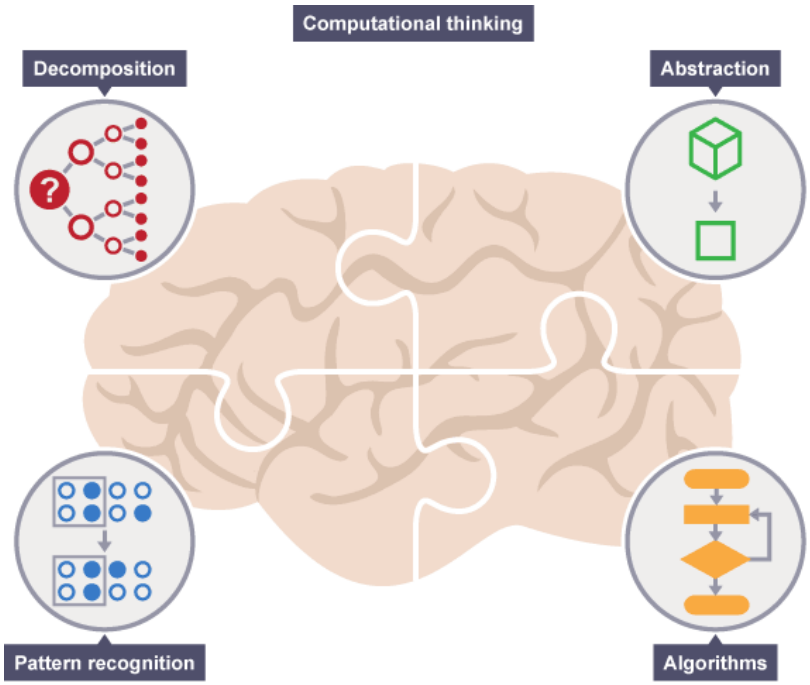


Abbildung 1: Die vier Schlüsseltechniken von CT (BBC, 2016a)

- Bei der *decomposition* werden komplexe Probleme, Systeme, Prozesse bzw. Daten in kleinere Bereiche aufgespalten, welche einfacher handhabbar sind. (BBC, 2016a; Google, 2016) Würde diese Schlüsseltechnik nicht angewendet und gleichzeitig an allen Bereichen eines Problems gearbeitet werden, so wäre dieses um vieles komplizierter und aufwendiger zu lösen. Auch das Eingehen auf kleinere Details wäre ohne der *decomposition* nur schwer möglich. (BBC, 2017a)
- Die *pattern recognition* beschreibt die Suche nach Ähnlichkeiten oder Mustern in den verschiedenen existierenden Bereichen der Probleme/Daten (BBC, 2016a; Google, 2016), welche unterstützend fungieren bei dem effizienteren Lösen von komplexeren Problemen. (BBC, 2017b)

- Bei der *abstraction* wird lt. BBC (2016; 2017) der Fokus auf wichtige Informationen gelegt, die Unwichtigen werden ignoriert. Diese Vorgehensweise erzeugt also „[…] a general idea of what the problem is and how to solve it." (BBC, 2017c) Google (2016) setzt den Schwerpunkt etwas anders fest. Hier liegt dieser auf der Identifizierung der generellen Richtlinien, welche das Muster erzeugen.
- *Algorithm/algorithms design* beschreibt die Entwicklung für das schrittweise Erstellen von Lösungen bzw. die Regeln für das Lösen eines Problems oder ähnlicher Probleme. (BBC, 2016a; Google, 2016)

Jede dieser vier Schlüsseltechniken ist im gleichen Maße wichtig. Fehlt eine, kann dies zu einem Ungleichgewicht und in weiterer Folge zu einem Scheitern bei der Lösungsfindung eines Problems mittels CT führen. (BBC, 2016a)

Die Anwendung der Schlüsseltechniken läuft lt. BBC bzw. Google wie folgt ab:

Theoretischer Ablauf (BBC, 2016a)	**Praktisches Beispiel**
1. Eine Person steht vor einem großen, komplexen Problem.	Julia benötigt einen neuen Rechner. Diesen möchte sie nicht bereits fertig kaufen, sondern selbst zusammenbauen.
2. *decomposition*: Dieses wird zuerst in viele kleine, einfacher überschaubare Probleme unterteilt.	Sie überlegt, welche Hardware-komponenten (d.s. Mainboard, CPU inkl. Kühler, RAM, SSD-/HDD-Festplatte, Netzteil, Grafikkarte, Lüfter, Gehäuse) sie für den Bau eines Rechners benötigt. Weiters denkt sie darüber nach, wo sie am besten jede einzelne Komponente der benötigten Hardware erstehen kann.
3. pattern recognition: Jedes dieser kleineren Probleme wird einzeln betrachtet und mit zuvor gelösten bzgl. möglicher Ähnlich-keiten verglichen.	Da sie bereits schon einen Rechner gebaut hat, kann Julia auf dieses Wissen (z.B. welcher Anbieter verkauft gute Qualität zu einem angemessenen Preis) aufbauen und ihre Suche individuell für jede Komponente eingrenzen.

4. *abstraction*: Weiters liegt der Fokus auf den relevanten und wichtigen Details des aufgeteilten Problems.	Julia informiert sich bzgl. dem derzeitigen Entwicklungsstandard jedes einzelnen Bestandteils der Hardware, ob diese zueinander kompatibel sind und ob der Anbieter X diese auch auf Lager hat.
5. *algorithms / algorithm design*: Einfache Schritte bzw. Regeln, welche jedes einzelne dieser kleineren Probleme lösen, können designed werden.	Sie legt nun für jede Komponente einen (gedanklichen) Plan an, wie sie bei der Auswahl (basierend auf den zuvor erhaltenen Fakten) vorgehen möchte. Weiters stellt Julia Regeln (z. B. wenn Komponente A, B, C und D von dem Anbieter X sind, dann wäre es weniger aufwendig die restlichen Bestandteile der Hardware auch bei Anbieter X zu bestellen) auf die sie im Laufe des Bestellvorgangs unterstützen sollen. Ziel ist es alle benötigten Hardware-komponenten möglichst bald und zeitgleich zu erhalten.
6. a) Abschließend können diese erstellten Schritte/Regeln verwendet werden um einen Computer zu programmieren, welcher - unter Anwendung dieser - das komplexe Problem in der besten Weise löst. (BBC, 2016a) b) Da lt. den Artikeln von Yadav et al. (2011: S. 466) und Lu & Fletscher (2009: S. 260) Programmieren keine Rolle beim Lehren von CT spielt	a) Die gesammelten Faken können in ein Computerprogramm eingegeben werden, welches auf Basis dessen die gewünschten Teile auswählt. (Da die Erstellung einer Software für dieses Problem sehr umfangreich und zeitaufwendig sein würde, wäre vermutlich Punkt b) effizienter.) b) Jede Hardwarekomponente wird von Julia selbst aus der gesamten Menge ausgewählt. Dabei beachtet sie jene Erkenntnisse, welche sie bereits zuvor

(Voogt et al., 2013: S. 720), muss der abschließende Punkt nicht gezwungener Weise eine Verbindung dazu herstellen. Das bedeutet, die erstellten Schritte/Regeln können auch von einem Menschen angewandt werden um die einzelnen Teil-Probleme zu lösen. Diese werden im Anschluss wieder zu dem ursprünglichen, ungeteilten Problem zusammengefasst. Die Betrachtung der Lösung kann nun erfolgen.	erarbeitet hat. Diese sind beispielsweise die Kompatibilität der Bauteile untereinander, der Preis und die Qualität der Komponenten eines Anbieters, die Möglichkeit einer schnellen Lieferung, etc. Nachdem sie eine Auswahl getroffen hat, kann Julia nun zum Abschluss des Bestellvorgangs der favorisierten Hardware übergehen.

2.1.1.4 Barefoot – Computing at School

Barefoot bezeichnet die von Wing (d.s. *abstraction, automation* und *decomposition*) bzw. BBC und Google definierten Schlüsseltechniken von CT (d.s. *decomposition*, *pattern recognition*, *abstraction* und *algorithm / algorithmic design*) als dessen Konzepte. Außerdem werden zwei weitere hinzugefügt: *logic* und *evaluation*. (Barefoot, 2014a) Neben diesen Konzepten formuliert Barefoot auch Methoden (= *approaches*), welche unterstützend beim Erlernen dieser fungieren: *tinkering, creating, debugging, perserving* und *collaborating*. (Barefoot, 2014a)

Abbildung 2: Konzepte und Methoden von CT (Barefoot, 2014a)

2.1.1.4.1 Konzepte von CT nach Barefoot

Logic bzw. *logic reasoning* steht bei der Vorgehensweise zur Problemlösung mittels CT an erster Stelle. Dieses Konzept soll erlernt werden um Vorhersagen bzgl. der Überlegung, ab welchem Zeitpunkt Software bei der Entwicklung der Fähigkeit des logischen Begründens und bei dem Ziehen von Schlussfolgerungen aus vorhandenen Informationen unterstützend fungiert, treffen zu können. (Miles, 2014) D.h. es unterstützt also bei dem Finden einer Erklärung, weshalb dieses oder jenes geschieht. (Barefoot, 2014a) Als Beispiel nennt Miles Barries (2014) den Computer. Dieser ist eine deterministische Maschine, d.h. das am PC laufende Programm befindet sich in einem Zustand X, welcher auf Grund eines bestimmten Inputs immer in den Zustand Y übergeht. Somit ist es mit einem Hintergrundwissen über ablaufende Algorithmen möglich, eine exakte Vorhersage bzgl. des weiteren Ablaufs zu tätigen.

Nach der Bewältigung aller Konzepte der Problemlösestrategie CT, d.s. *algorithms*, *decomposition*, *patterns* und *abstraction* (Siehe Kapitel 2.1.1.3 BBC und Google.), erfolgt die *Evaluation.* Dieser Prozess stellt sicher, dass die endgültige Lösung – ganz gleich ob es sich um einen Algorithmus, ein System oder einen Prozess handelt – passend für die jeweilige Problemstellung ist. Dafür müssen die Eigenschaften der gefundenen Lösungen der Bereiche eines Problems bzgl. der Korrektheit, der Geschwindigkeit, der ökonomischen Verwendung von Ressourcen, der (möglichst einfachen) Anwendung und ob die Förderung einer angebrachten Erfahrung vorliegt, evaluiert werden. Da es nur selten eine einzige ideale Lösung für alle Bereiche eines Problems gibt, müssen hier Kompromisse geschlossen werden. (Csizmadia & Curzon & Dorling & Humphreys & Ng & Selby & Woollard, 2015: S. 7)

2.1.1.4.2 Methoden zum Erlernen von CT nach Barefoot

- Bei *tinkering* steht das selbstständige Ausprobieren von neuen Dingen im Vordergrund. Kinder und Jugendliche werden in einem risikofreien Umfeld dazu angehalten gestellte Herausforderungen auf eine spielerische Art zu meistern und bzw. um dadurch Selbstbewusstsein und eine innere Einstellung, alles selbst ausprobieren zu wollen, zu erarbeiten. Außerdem wird explizit darauf hingewiesen, dass der Lernvorgang voller Spaß, Kreativität, Fragen und Überraschungen sein sollte. Die Erlernung dieser Fähigkeit ist heutzutage von enormer Wichtigkeit: Immer schneller entwickelt sich die Welt der Informatik weiter und dementsprechend größer bzw. anders werden die gestellten Anforderungen an unsere Gesellschaft. Um schnell

mit den Veränderungen klar kommen zu können oder diese sogar voranzutreiben, ist *tinkering* als Fähigkeit besonders wichtig. (Barefoot, 2014b)

- *Creating* wird (in diesem Fall) als Prozess verstanden, in dem die Planung, die Erstellung und die Evaluierung von Dingen, z.B. Programm-Code, etc. im Fokus stehen. Ziel dieser Fähigkeit ist es, originelle Ideen mit einem gewissen Wert (wenn möglich für mehrere Personen) hervorzubringen und diese anschließend umzusetzen. Als Begründung der Notwendigkeit wird auf der Homepage Barefoot (2014c) auf die Entwicklungsgeschichte von verschiedensten Systemen, z.B. Space-Shuttles, Smart Cities, Kunstinstallationen, uvm., zurückgegriffen. Ohne der hier beschriebenen Fähigkeit *creating* würde keines dieser Systeme existieren. (Barefoot, 2014c)
- *Debugging* bezeichnet den Prozess des Suchens und Findens von Fehlern (Diese können semantischer bzw. syntaktischer Art sein.) im geschriebenen Code und kann manchmal sogar länger dauern als das eigentliche Verfassen des Programms. Um die Zeit, die zur Fehlersuche aufgewendet wird möglichst kurz zu halten, wird folgender Vorgang vorgeschlagen:
  1. Vorhersage, was passieren sollte bei korrektem Code
  2. Recherche, was tatsächlich passiert
  3. Herausarbeiten der Stelle, die den Fehler verursacht
  4. Korrektur des Fehlers

  Eine zweite Möglichkeit Fehler zu finden wird *rubber duck debugging* genannt: Hierbei wird der verfasste Code Zeile für Zeile einer Person bzw. einem Gegenstand (Kinder können dafür beispielsweise ein Quietscheentchen verwenden.) erklärt. Dabei ist es durchaus wahrscheinlich, dass die erklärende Person selbst den Fehler findet. (Barefoot, 2014d)

- Die Methode *persevering* beschreibt Zielstrebigkeit, Hartnäckigkeit, Belastbarkeit und den Willen nicht aufzugeben. Diese Fähigkeit wird von Barefoot (2014e) besonders beim Programmieren als notwendig erachtet, da das Verfassen eines eleganten und effektiven Codes eine intellektuelle Herausforderung darstellt. Besonders das Verstehen der Algorithmen und die Anwendung der Sprache in welcher programmiert wird, aber auch die Bereitschaft komplizierte oder frustrierende Erlebnisse durchzustehen, muss von Seiten der Programmiererin/ des Programmierers gegeben sein.

Persevering kann jedoch neben dem Programmieren auch in vielen anderen Bereichen des Lebens zielführend eingesetzt werden: beim Ausprobieren von vielen verschiedenen Möglichkeiten um ein Problem zu lösen, beim Verändern des Denkens (von schnell und intuitiv zu etwas langsamerem und logischerem Denken), uvm. (Barefoot, 2014e)

- Die letzte angeführte Methode wird *collaborating* genannt. Diese sagt aus, dass die Zusammenarbeit mit Mitmenschen bessere Resultate liefert, da gemeinsam der Erfolg gefeiert bzw. der Misserfolg leichter verkraftet werden kann. Weiters sind Personen motivierter komplexe bzw. scheinbar unlösbare Aufgaben zu bewältigen, wenn die Arbeit in Teams bzw. Gruppen stattfindet. Die Möglichkeit Ideen mit jemandem zu teilen, ist dabei entscheidend.

  Barefoot (2014f) zeigt dies anhand des Beispiels „paarweise Programmieren" auf: Gemeinsam wird an einem Computer (mit nur einem Bildschirm, einer Maus und einer Tastatur) gearbeitet. Dabei sind die Rollen klar verteilt. Typischerweise beschäftigt sich eine Programmiererin/ein Programmierer mit den Details der Programmierung und die/der Zweite übernimmt die Rolle der Navigatorin/des Navigators, welche/r das Gesamtbild im Blick behält. Durch einen regelmäßigen Tausch der Rollen wird sichergestellt, dass beide Programmiererinnen/ Programmierer über jeden Bereich sehr gut informiert sind. (Barefoot, 2014f)

2.1.2 Auswahl einiger Anwendungsbereiche von CT

Computational Thinking kann nicht nur in der Informatik angewendet werden, sondern auch in vielen anderen Bereichen, z.B. in der Mathematik, in der Musik oder beim Erlernen von Sprachen.

2.1.2.1 Mathematik

James Lu und George Fletcher (2009) behandeln in ihrem Paper „Thinking About Computational Thinking" eine Möglichkeit die Konzepte der Mathematik[8] unter Zuhilfenahme von CT zu vermitteln: *computational thinking language* (CTL)

Die CTL beinhaltet Vokabeln und Symbole, welche vielseitig angewendet werden können. Beispielsweise zur Beschreibung und Erläuterung von Berechnungen und von

[8] Diese beinhalten beispielsweise die Verwendung von algebraischen Regeln um einfachere Formen herzuleiten, die Beschreibung des Suchradius von möglichen algebraischen Vereinfachungen unter Verwendung des Anfangszustandes, des Endzustandes und den möglichen anwendbaren Operationen, etc. (Lu & Fletcher, 2009: S. 261)

Abstraktionen, zum Vorschlagen von Informationen und von möglichen Varianten der Umsetzung oder auch zur Bereitstellung von Notationen, welche das semantische Verstehen von berechnenden Prozessen ausdrücken. Jedoch beschreibt die CTL in diesem Rahmen keine Programmiersprache. (Lu & Fletcher, 2009: S. 261-262)

Beispiel: Multiplikation (Lu & Fletcher, 2009: S. 262)
Die Multiplikation folgt zwei Regeln:

- Die Multiplikation ist eine sich wiederholende Addition.
- Das Ergebnis der Multiplikation ist immer gleich, unabhängig davon, welche Zahl zuerst geschrieben wird.

Die erste Regel eröffnet die Möglichkeit, dass die Iteration als rechnerisches Konzept vorgestellt werden kann: Jeder Zusatz des Symbols + bei einer Addition stellt eine Iteration dar. (2 + 2 => zwei Iterationen; 2 + 2 + 2 => drei Iterationen; usw.)

Ein zweites rechnerisches Konzept, die Effizienz, kann unter Anwendung der zweiten Regel auch erklärt werden: Unabhängig von der Reihenfolge der Zahlen, ist das Ergebnis einer Multiplikation immer gleich. Für die Anzahl der Iterationen trifft dies allerdings nicht (immer) zu.

2 * 7	=> 2 + 2 + 2 + 2 + 2 + 2 + 2	=> sieben Iterationen
7 * 2	=> 7 + 7	=> zwei Iterationen

Mögliche Aufgabenstellungen können folgende sein:

- Notiere jede Multiplikation als sich wiederholende Addition und berechne das Ergebnis. Halte auch jeweils die Anzahl der notwendigen Iterationen fest.
- Notiere eine Multiplikation. Vertausche die beiden Zahlen der Multiplikation und halte die Anzahl der notwendigen Iterationen pro Rechnung fest. Welche Rechnung ist effizienter?

Weitere Beispiele und Anregungen für mathematische Aufgaben können bei Lu und Fletcher (2009) nachgelesen werden.

Stephen Wolfram, Entwickler von Wolfram Alpha, verfolgt den Ansatz einer CTL zur einfacheren Vermittlung von Mathematik bereits schon seit über 30 Jahren. Mit Hilfe der Wolfram Language (WL) (In diesem Fall handelt es sich bei der CTL tatsächlich um eine

Programmiersprache.) können Kinder, Jugendliche und Erwachsene gratis und on- bzw. offline die Welt der Mathematik erforschen. Das zeitgleiche Erlernen von CT wird lt. Wolfram durch eine optimierte Benutzeroberfläche und durch die WL selbst gefördert. (Wolfram, 2016) Neben dem kostenfreien Buch „An Elementary Introduction to the Wolfram Language“ mit unzähligen Beispielen und Aufgaben (Wolfram, 2017a) steht auch ein Onlinekurs (auf Basis des Buchs) für die Erlernung der WL zur Verfügung.

Speziell für Kinder wurde das „Wolfram Programming Lab“ gegründet. Dabei stehen die Vermittlung von CT und der WL im Mittelpunkt. (Wolfram, 2017b)

Aufgaben und geplante Unterrichtseinheiten zu den einzelnen Themenbereichen der Mathematik werden auch kostenlos von der CT-STEM zur Verfügung gestellt. (CTSTEM, 2017) Dies ist ein Team von WissenschaftlerInnen und PädagogInnen der Northwestern University, welche die Vermittlung von CT im Unterrichtsfach Mathematik zum Ziel haben und damit sicher stellen wollen, dass die SchülerInnen die „computational future“ mitbestimmen können.

2.1.2.2 Musik

Auch im Unterrichtsfach Musik gibt es viele Möglichkeiten das Erlernen von CT einzubinden. Beispiele dafür sind:

- ***Analyse von Liedern:***

  pattern recognition: In fast allen Liedern werden einzelne Abschnitte (z.B. Melodien oder Teile der Bass-Stimme) einmal bzw. öfter wiederholt.
  Aufgabe: Die SchülerInnen sollen selbst einen Rhythmus bzw. eine Melodie erarbeiten, bei der sich einer oder mehrere Abschnitte wiederholen. (QuickStart Computing, 2017: S. 16)

  abstraction: Die Klavierpartitur eines Pop-Songs könnte als eine Abstraktion für dieses Musikstück gesehen werden. (QuickStart Computing, 2017: S. 15)
  Aufgabe: (Sehr begabte) SchülerInnen wählen einen Pop-Song aus und schreiben dazu die Klavierpartitur.
- ***Planung des Baus eines Musikinstruments:***

  Aufgabe: Die SchülerInnen bekommen ein Instrument vorgegeben bzw. können sich eines selbst aussuchen. Die/Der Lehrende weist darauf hin, dass dieses komplexe Problem in kleinere Teile untergliedert werden soll (*decomposition*), z.B. Planen des

Designs (*abstraction*: ein einfaches Design, welches die wichtigsten Elemente beinhaltet), Beschaffung von Materialien (*decomposition*: um die einzelnen Bestandteile zu identifizieren), Zusammenbauen des Materials (*algorithm*: schrittweise Anleitung), Evaluieren/Testen des Instruments. Nun soll jede bzw. jeder Überlegungen anstellen, wie der Bau am besten erfolgt. (QuickStart Computing, 2017: S. 8)

- ***Erstellen eines virtuellen Xylophons (mit Scratch):*** *abstraction*
  Aufgabe: Im Unterricht wurde das Konzept der diatonischen Tonleiter und jenes der Tonstufen erarbeitet. Die SchülerInnen sollen nun unter Verwendung von Scratch ein virtuelles Xylophon erstellen, welches die Tonleiter korrekt wiedergibt. Dabei werden die SchülerInnen feststellen, dass sich jeder Klangstab gleich verhält, jedoch die Tonhöhe variiert. (Barr & Harrison & Conery, 2011: S. 22)

2.1.2.3 Sprachunterricht

Unabhängig davon in welchen Sprachen das folgende Beispiel (Barr & Harrison & Conery, 2011: S. 22) Anwendung findet, kann dieses ab einem gewissen Vokabelschatz jederzeit durchgeführt werden:

Nach dem Lesen mehrerer Kurzgeschichten und bzw. oder als Abschluss eines erarbeiteten Themas können von den SchülerInnen Essays verfasst werden, welche die jeweiligen Inhalte behandeln. Dabei ist das Formulieren von klaren Thesen, welche mit mindestens drei stichhaltigen Begründungen gefestigt werden sollen, wichtig. Bei der Vorbereitung auf das Verfassen des Essays, d.h. vor dem Schreibprozess, erkunden die SchülerInnen auf diese Weise, welche Rollen bestimmte literarische Abschnitte in den einzelnen Geschichten spielen und wie diese sich auf deren Verlauf auswirken.

Während der Erarbeitung dieser Aufgabe treten lt. David Barr, John Harrison und Leslie Conery (2011: S. 22) folgende CT-Elemente auf:

- *Logisches Organisieren und Analysieren von Daten und Informationen:* Das Herausfiltern von den gewünschten Inhalten und dessen Belegen mittels Zitaten, welche aus gründlichen Textrecherchen hervorgehen, ist für dieses Element existenziell.
- *Abstraktion literarischer Elemente*: Die Repräsentation durch die Abstraktion literarischer Elemente, wie beispielsweise einer Handlungsstruktur, eines vorgegebenen Rahmens, einer bildhaften Sprache oder einer Sichtweise der Dinge, ist für eine Textanalyse mit klar voneinander abgegrenzten Thesen unbedingt erforderlich.

- *Kommunizieren und Erarbeiten von gemeinsamen Zielen oder Lösungen:* Diese Fähigkeit ermöglicht es den SchülerInnen aktiv an Klassengesprächen, insbesondere jenen die von einer vorgegebenen Fragestellung ausgehen, teilzunehmen.
- *Schaffen von themenübergreifenden, sprachlichen Fähigkeiten:* Bei dem Reflektieren der vorhergehenden, bearbeiteten Inhalte, dem bereits angeeigneten Wissen und ihrem sprachlichen Können stellen die SchülerInnen Verbindungen zwischen diesem und anderen Themenbereichen her.

2.2 Geschichtlicher Hintergrund

CT weist eine lange Geschichte in der Entwicklung von Computer Science auf. Bereits in den 1950er und 1960er Jahren war es – zwar unter anderem Namen und in etwas abgeänderten Form – als „algorithmic thinking“[9] bekannt. (Denning, 2009: S. 28)

Im Jahr 1980 beschäftigte sich Seymour Papert in seinem Buch „Mindstorms: Children, Computers, and Powerful Ideas“ mit der Überlegung, wie die bloße Anwesenheit von Computer zu einem mentalen Prozess von Personen beitragen bzw. das Denken von Personen durch einen Computer (beispielsweise auch wenn dieser nicht in direktem physischen Kontakt mit den Personen steht) beeinflusst werden kann. (Papert, 1980: S. 4) In diesem Kontext erwähnt Papert den Ausdruck „computational thinking“. (Papert, 1980: S. 182) Eine genaue Beschreibung des Begriffes unterlässt er jedoch in seinem Buch.

Zu Beginn der 1980er Jahre trat der Physiker und Nobelpreisträger Kenneth G. Wilson[10] einer Gruppe führender Wissenschaftler bei, welche in vielen verschiedenen Bereichen tätig waren. (Denning, 2009: S. 29) Diese, die „Renormalization Group“, vertrat die Meinung, dass die großen Herausforderungen der Wissenschaft mittels „computation“ (dt. Berechnung durch einen Computer) gelöst werden könnten. (Wilson, 1982: S. 103) Weiters

9 „Algorithmic thinking“ beinhaltet viele Fähigkeiten, welche in Verbindung miteinander das Konstruieren und Verstehen von Algorithmen ermöglichen, welche zum Lösen von Problemen dienen. (Denning, 2009: S. 28; Futschek, 2006: S. 160) Die hierbei benötigten Fähigkeiten sind lt. Gerald Futschek (2006: S. 160) die Fähigkeit gegebene Probleme zu analysieren, die Fähigkeit Probleme präzise zu spezifizieren, die Fähigkeit die Effizienz eines Algorithmus zu erhöhen, die Fähigkeit an alle möglichen Fälle (d.s. speziell bzw. normal) eines Problems zu denken, die Fähigkeit die grundlegenden Aktionen, welche adäquat zu dem gegebenen Problem sind, zu finden und die Fähigkeit zur Konstruktion eines korrekten Algorithmus zu einem Problem, welche die Grundaktionen verwendet.
Weiters ergänzt Futschek (2006: S. 160), dass bei “algorithmic thinking” die Kreativität eine große Rolle spielen würde. Diese fungiere bei der Lösung von Problemen und der damit eng verbundenen Konstruktion neuer Algorithmen unterstützend.

10 Wilsons Entdeckungen basieren auf Modellen, die “computational” sind. (Ismailova, 2014: S. 419)

verwendeten sie in ihren Überlegungen oft den Begriff „computational thinking“.[11] (Ismailova, 2014: S. 419)

Um den geplanten Prozess, dieser war der Ausbau von „computation“, schneller vorantreiben zu können, fragte die “Renormalization Group” bei der Regierung der Vereinigten Staaten an, ob diese mittels einer Erstellung eines Netzwerks von Supercomputer unterstützend fungieren könne. Als Begründung brachten sie folgendes vor: „Computation“ ist, neben den beiden traditionellen Säulen (d.s. die Theorie und das Experiment), als eine zusätzliche Säule der Wissenschaft zu sehen. (Ismailova, 2014: S. 419)

Daraufhin startete der Kongress der Vereinigten Staaten im Jahr 1991 die Förderinitiative HPC (= „high performance compuations“), welche die Wichtigkeit der Idee von „computation“ und der Idee von CT (Ismailova, 2014: S. 419) bzw. der Weiterentwicklung und Förderung der Wissenschaft, auch auf politischer Ebene, unterstreicht. Dieses Konzept wurde sogar in einem Bundesgesetz festgehalten. (Denning, 2009: S. 29)

1991 publizierte Seymour Papert, gemeinsam mit Idit Harel, das Buch „Constructionism: research reports and essays, 1985-1990“. In diesem sind Aufsätze einiger Studenten der Learning Research Group des MIT Media Laboratory zum Thema „Lernen“ gesammelt. Grundannahme jedes Aufsatzes ist, dass der Lernprozess ein aktives Handeln des Lernenden voraussetzt, d.h. Wissen wird durch die Lernenden selbst aufgebaut (durch das Bauen von Wissensstrukturen) und kann nicht durch Lehrende vermittelt werden. (Papert & Harel, 1991: S. 1) Lernende werden also nicht gelehrt, wie Wissenschaftlerinnen/Wissenschaftler forschen, sondern dürfen selbst (mit Unterstützung der Lehrenden) forschen. Die hier vertretene Lerntheorie ist der Konstruktionismus[12], welcher von Seymour Papert selbst begründet wurde und die zuvor beschriebenen Schlüsseltechniken bzw. Konzepte und Methoden von (Siehe Kapitel 2.1.1 Die Schlüsseltechniken bzw. Konzepte und Methoden von CT.) *Computational Thinking* fordert und fördert. (Papert & Harel, 1991: S. 10-11)

Fünfzehn Jahre danach, 2006, verfasste Jeannette M. Wing den drei seitigen wissenschaftlichen Artikel „Computational Thinking“ im Journal „Communication of the

[11] Auch hier wurde keine genaue Definition dieses Begriffes festgehalten.

[12] Als konstruktivistisches Vorbild diente der Philosoph Jean Piaget bereits schon in Paperts Buch „Mindstorms: Children, Computers, and Powerful Ideas“, welches 1980 erschienen war. (Papert, 1980) Für eine genauere Definition und Beschreibung des Konstruktionismus siehe Kapitel 3 Kompetenzorientierter Lehrplan für das Unterrichtsfach Informatik in Österreich (5. AHS).

ACM“. In diesem versucht sie CT und dessen Inhalte (zunächst nur vage) zu beschreiben: „Computational thinking involves solving problems, designing systems, and understanding human behaviour, by drawing on the concepts fundamental to computer science.” (Wing, 2006: S. 33) Wing fügt ergänzend hinzu, dass CT nicht nur für die Informatik von großer Wichtigkeit sein wird, sondern eine fundamentale Fähigkeit – gleichzusetzen mit dem Lesen, dem Schreiben und der Arithmetik – für jede Person sei. (Wing, 2006: S. 33) Mit ihrem Artikel trägt Wing einen großen Teil dazu bei den Begriff CT wieder in das Gedächtnis der Wissenschaften bzw. der Menschen zu rufen und „[the] steady decline in undergraduate enrollments in computer science […]“ (Wing, 2016) zu stoppen. (Wing, 2016)

Zwei weitere, viel detailliertere wissenschaftliche Arbeiten von Jeannette M. Wing folgen: „Computational thinking and thinking about computing“ (2008) und „Computational Thinking Benefits Society“ (2014). Als Ziel beschreibt Wing in diesen (Wing, 2006: S. 33; Wing, 2008: S. 3720) und in ihren Vorträgen[13] (TU Wien, 2016: 00:12:42 – 00:13:00) laufend, dass CT in der Mitte des 21. Jahrhundert eine Fähigkeit ist, die jede Person direkt oder indirekt berührt, weil es in jedem Bereich angewendet werden wird bzw. die jeder Mensch in jedem Bereich ihres/seines Lebens anwenden wird.

Neben Wing und weiteren Wissenschaftlern befassen sich auch Unternehmen, wie z.B. Microsoft, Google und BBC (Siehe Kapitel 2.1.1.3 BBC und Google.) mit CT und dessen Verbreitung:

- Google erarbeitete einen Online-Kurs für LehrerInnen, in welchem diese die Theorie, die Anwendung und das Lehren von CT erlernen können. (Siehe Kapitel 2.3.1.2.1 Google.) Weiters steht eine Community von über 6.400 Personen (Stand: 12. April 2017) für einen Austausch von Projekten, Ideen und Fragen zur Verfügung. (Google Community, 2017)
- BBC, Microsoft und einige andere Unternehmen finanzierten die Entwicklung und das Design von BBC micro:bit. Dies ist ein kleiner programmierbarer Minicomputer, welcher den Schülerinnen und Schülern in England, Wales, Schottland und Nordirland gratis zur Verfügung gestellt wird. (Siehe Kapitel 2.3.2.1 BBC micro:bit.)

[13] Bis heute hat sie mittlerweile in über 100 Colleges und Universitäten weltweit Workshops und Präsentationen gehalten um aufmerksam auf CT zu machen bzw. um CT zu vermitteln. (Wing, 2016)

Die dafür benötigte Programmiersprache, die dazugehörige Plattform und das Unterrichtsmaterial wurde von MSR Labs Touch Develop bereitgestellt. (Wing, 2016)

2.3 Neuere Entwicklungen (mit besonderem Fokus auf die Bildung)

Die neue industrielle Revolution, auch Industrie 4.0 genannt, hat in den letzten Jahren auch in Österreich Einzug gehalten und „[…] viele Branchen von Grund auf neu gestaltet […]." (Futschek, 2016: S. 20) Um den Anforderungen, die diese Veränderung mit sich bringt, gerecht zu werden, ist es notwendig auch die Ausrichtung der Bildung zu verändern, d.h. einen verstärkten Fokus auf die Fähigkeiten und die Fertigkeiten im Bereich Informatik zu legen. Eine dieser neuen Fähigkeiten betrifft die Art zu Denken: „Computational Thinking", oder wie es im Deutschen heißt „Informatisches Denken". (Futschek, 2016: S. 20)

Alan Bundy und Jeremy Scott spinnen diesen Gedanken in ihrem Artikel „Creating a New Generation of Computational Thinkers" (2015: S. 37-40) weiter und kommen zu dem Schluss, dass es für die heutige Gesellschaft wichtig sei, anstelle von „tool*users*" „tool*makers*" zu werden bzw. zu sein. Um dies erreichen zu können, sei es notwendig bereits in den Schulen, bei der Bildung von Kindern, Informatik und besonders das Programmieren in den Fokus zu stellen. Damit sei es aber laut Bundy und Scott noch nicht getan. Es ist außerdem erforderlich, dass die Schülerinnen und Schüler die neue Art zu Denken (d. i. CT) erlernen und diese auch für das Verstehen und das Lösen von Problemen einsetzen können. (Bundy & Scott, 2015: S. 37)

2.3.1 Internationale Förderung von CT

Die Motivation auch in der Zukunft noch mit der rasanten Entwicklung der (Informatik in der) Wirtschaft mithalten zu können, scheint ein triftiger Grund für die Förderung von CT durch Länder (z.B. Großbritannien, Schweiz, die Vereinigten Staaten, etc.) und durch Unternehmen (z.B. BBC, Google, Microsoft, etc.) zu sein. Aber auch die Europäische Union und länderübergreifende Verbände bzw. Programme (z.B. Bebras bzw. Bieber der Informatik, Computer Science Unplugged, etc.) legen großen Wert auf die Verbreitung von CT.

2.3.1.1 Europäische Union und ausgewählte Länder

2.3.1.1.1 Europäische Union (Stand: 22. Juli 2017)

Die *JRC-IPTS of the European Comission* mit Unterstützung von dem *Institute for Educational Technology of the Italian National Research Council* und dem *European*

Schoolnet arbeiteten gemeinsam an der Studie „CompuThink“, welche den pädagogischen Fortschritt der Vermittlung von CT an eine breitere Masse von Personen analysierte. (Bocconi et al., 2016b) Im Jahr 2016 veröffentlichten sie einen umfassenden Report: Neben einer ausführlichen Definition, der Beschreibung der Verbindung von CT mit der digitalen Kompetenz und mit dem Programmieren, beinhaltet dieser auch eine Begründung, Richtlinien, sowie Beispiele für das Eingliedern von CT in die Lehrpläne der EU-Mitgliedsstaaten. Weiters wird kurz eine mögliche Herangehensweise an das Lehren und Lernen inkl. einiger unterstützenden Tools beschrieben. (Bocconi et al., 2016b)

2.3.1.1.2 Österreich (Stand: 01. August 2017)

Die Problemlösestrategie *Computational Thinking* hat erst in den letzten Jahren in Österreich Einzug gehalten. Im schulischen Bereich erfolgt die Vermittlung im Rahmen des Informatikunterrichts. Dieser findet erst ab der Sekundarstufe 2[14] statt und ist abhängig davon, welchen Schultyp die Schülerin/der Schüler wählt:

- In der **AHS** erfolgt der Pflichtunterricht des Unterrichtsfachs Informatik nur in der 5. Klasse. Weiters kann das Wahlpflichtfach Informatik von den SchülerInnen von der 6. bis zur 8. Klasse belegt werden. Im Lehrplan (5. Klasse) ist dies wie folgt festgehalten: „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.“ (BMB, 2017a: S. 2) CT kann den „typischen Denk- und Arbeitsweisen“ zugeordnet werden. (Siehe auch Kapitel 3 Kompetenzorientierter Lehrplan für das Unterrichtsfach Informatik in Österreich (5. AHS).)
- In der **HAK** erfolgt die Vermittlung von CT im Unterrichtsfach Wirtschaftsinformatik und ist unter dem Punkt „Didaktische Grundsätze“ angeführt: „Praxisorientierte Aufgabenstellungen und kompetenzorientierter Unterricht sollen die Schülerinnen und Schüler zu logischem, kreativem und vernetztem Denken, zum genauen und ausdauernden Arbeiten, selbstständig und im Team, sowie zum

[14] In der Volksschule (BMB, 2017b) und in der AHS Unterstufe (BMB, 2017c) wird lt. Lehrplan kein Informatik bzw. CT unterrichtet. Auch in der Neuen Mittelschule (RIS, 2017) ist dies der Fall. Jedoch gibt es hier die Möglichkeit in ausgewählten Schulen Informatik als Freigegenstand oder als unverbindliche Übung mit zwei bis acht Wochenstunden zu belegen.

verantwortungsbewussten Entscheiden und Handeln führen.“ (Bundesgesetzblatt Teil 2, 2014: S. 67)

- Im Lehrplan der **HTL** (hier für den Zweig „Wirtschaftsingenieure – Betriebsinformatik“) ist die Vermittlung bzw. das Erlernen von CT nicht (detailliert) festgehalten, jedoch zwingend notwendig um beispielsweise folgende Anforderungen im Unterrichtsfach Softwareentwicklung und Projektmanagement“ erfüllen zu können: „Im Bereich **Strukturierte Programmierung** können die Absolventinnen und Absolventen Problemstellungen systematisch analysieren, algorithmische Lösungswege entwickeln und diese in einer höheren Programmiersprache strukturiert umsetzen.“ (RIS, 2015: S. 4; Hervorhebung im Original)

Demzufolge kommen Schülerinnen und Schüler, die eine Lehre bevorzugen und keine Sekundarstufe 2 besuchen, in ihrer Schulzeit nie in den Kontakt mit CT. Dem soll nun mit der verbindlichen Übung „Digitale Grundbildung“ in der Sekundarstufe 1 entgegengewirkt werden. Der Lehrplan dafür beinhaltet u.a. als Punkt „Computational Thinking“ inkl. den drei Untergliederungen „Mit Algorithmen arbeiten“, „Einfache Programme erstellen“ und „Kreative Nutzung von Programmiersprachen“. (BMB, 2017d: S. 8)

Die „Digitale Grundbildung“ wird im Schuljahr 2017/18 als Pilotprojekt an 169 Neuen Mittelschulen und AHS-Unterstufen im Umfang von zwei bis vier Wochenstunden durchgeführt, welche innerhalb von vier Jahren zu absolvieren sind. Im darauffolgenden Schuljahr soll eine „Verordnung des Lehrplans und die flächendeckende Umsetzung für alle Schulen der Sekundarstufe 1“ folgen. (BMB, 2017e)

Um die Vermittlung in der Schule durch die Lehrkräfte zu ermöglichen, werden seit September 2016 immer häufiger Vorträge und Workshops zu dem besagten Thema angeboten. Beispiele sind u.a. der „8. Informatiktag 2016“ (Donnerstag, 29. September 2016) (OCG, 2016a), welcher einen Workshop beinhaltete, oder die dreitägige Veranstaltung „Informatisches Denken und Programmieren“ (von Donnerstag, 16. Februar 2017 bis Samstag, 18. Februar 2017), welche von der OCG organisiert und von IBM Wien unterstützt wurde. (OCG, 2017a) Diese konnte nicht nur von Personen, die das Unterrichtsfach Informatik lehren, belegt werden, sondern war für alle Lehrkräfte frei zugänglich.

Die PH Wien errichtete zudem das „Lego Education Innovation Studio (LEIS)“ (Eröffnung: Donnerstag, 16. März 2017), welches die Vermittlung von CT, Coding,

Robotics und CS durch geschulte TrainerInnen des ZLI anhand von Lego WeDo ermöglicht. (ZLT, 2017 & PH Wien, 2016) Außerdem soll in weiterer Folge ein neuer Lehrveranstaltungsschwerpunkt ab dem Wintersemester 2017 in diesen Bereichen für die Primar- und Sekundarstufe und in der Berufsbildung aufgebaut werden. (ZLI, 2016) Ein Beispiel dafür ist die Lehrveranstaltung „Entdeckendes und problemlösendes Lernen mit Beebot und Ozobot Robotern“ (Mittwoch, 11. Oktober 2017 (Hübel-Fleischmann, 2017)).

Auch der „9. Informatiktag 2017” (Donnerstag, 28. September 2017) steht verstärkt unter dem Thema CT. Valentina Dagiene (Winner of Ada Lovelace Computing Excellence Award 2016) von der Vilnius University (Litauen) hält neben einem Vortrag auch einen Workshop ab. (OCG, 2017b)

In Österreich gibt es einige Projekte, die auf das Vermitteln von CT abzielen, beispielsweise „COOL“ von der Universität Klagenfurt oder der „Bieber der Informatik“.

2.3.1.1.3 Schweiz (Stand: 09. August 2017)

Das Schulwesen in der Schweiz ist ausgelegt auf eine Dauer von 11 Jahren und wird im Groben entweder in drei Gebiete, d.i. in den Deutsch, Französisch bzw. Italienisch sprechenden Teil, oder in die vorherrschenden Kantone unterteilt. (Grandl & Ebner, 2017: S. 2-3 & Kretschmar, 2016: S. 2)

Ein wichtiger Punkt der vorherrschenden Lehrpläne in der Schweiz ist laut Bocconi et al. (2016b: S. 26) die Vermittlung der Fähigkeit CT. Beispielsweise wird auf die Förderung von logischen Denkfähigkeiten, Problemlösefähigkeiten, anderen Schlüsselkompetenzen oder Programmierkenntnissen und -fähigkeiten Wert gelegt:

- In der Primärstufe und der Sekundarstufe 1 im **Deutsch sprechenden Teil** werden im Lehrplan Kompetenzen (u.a. programmieren) aufgelistet, die Teil von CT sind bzw. zum Erlernen dessen führen sollen. Das Unterrichtsfach, in dem CT gelehrt wird, ist außerdem nicht nur auf CS begrenzt. (Bocconi et al., 2016: S. 32)
- Im **Französisch sprechenden Teil** wird CT in einem bestimmten Teil des Lehrplans festgehalten: MITIC (= Média, image, technologie de l’information et de la communication). Dieser zielt darauf ab digitales Lernen generell zu entwickeln. (Bocconi et al., 2016: S. 32)

Eine weitere Möglichkeit der Unterteilung bzgl. der Lehrpläne erfolgt - wie zuvor erwähnt - durch die Kantone. Dies ist bei den Lehrplänen der Sekundarstufe 2 (Schweizer

Gymnasien) zurzeit überwiegend der Fall, d.h. das (vertiefende) Erlernen von CT ist von dem Ort des Schulbesuchs abhängig. (Kretzschmar, 2016: S. 2)

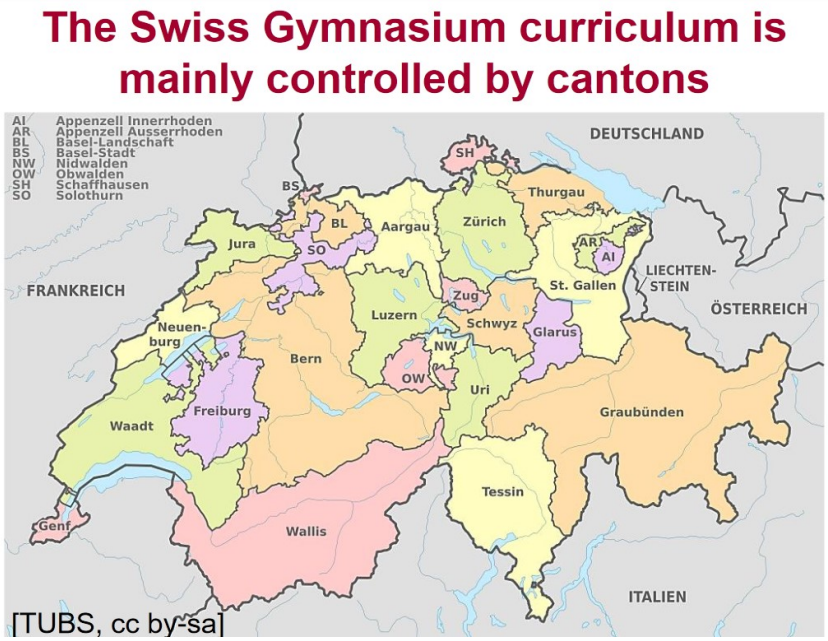


Abbildung 3: Lehrpläne der Schweizer Gymnasien – überwiegend kontrolliert durch Kantone (Kretzschmar, 2016: S. 2)

Um dieser Vielfältigkeit entgegen zu wirken, wurde für die 21 deutsch- und mehrsprachigen Kantone ein Kantonübergreifender Lehrplan entwickelt: **Lehrplan21**[15] (D-EDK, 2017) Der Begriff CT wird im Modul „Medien und Informatik" nicht explizit erwähnt. Jedoch weist eine der drei Zielsetzungen, d.i. „Grundkonzepte der Informatik verstehen und zur Problemlösung einsetzen", stark auf die gewünschte Vermittlung hin. (D-EDK, 2016: S. 4)

Viele Gruppen setzen sich in der Schweiz für die Einführung des Unterrichtsfaches Informatik als Pflichtgegenstand ein, z.B. die Arbeitsgruppe GFI@CH (= Grundlangenfach Informatik in der Schweiz), oder die Hasler-Stiftung zur informatischen Bildung. (Grandl & Ebner, 2017: S. 3) Leiter der Hasler-Stiftung Alexander

[15] Der Lehrplan21 wurde von der Deutschschweizer Erziehungsdirektoren-Konferenz (=D-EDK) erarbeitet, ist für die gesamte verpflichtende Schulzeit ausgelegt und „[…] soll zur Harmonisierung des Schulsystems […] beitragen". (Grandl & Ebner, 2017: S. 3) Neben Modulen wie z.B. „Grundlagen", „Sprachen" oder „Mathematik" beinhaltet er auch das Modul „Medien und Informatik".
Besonders großen Wert wird auf die Differenzierung zwischen dem Kompetenzbereich „Medien und Informatik" und den Anwendungskompetenzen gelegt. Letzteres wurde verstärkt in anderen Modulen, wie beispielsweise „Sprachen" oder „Mathematik" (mit dem Hinweis, dass diese explizit eingeführt werden müssen) integriert. (Grandl & Ebner, 2017: S. 3)
Bei der Anpassung des Lehrplans21 an die Kantone, müssen diese zunächst Zeitgefässe und Zuständigkeiten der Lehrpersonen definieren. Auf Basis dessen können die Schulen diese flexibel einsetzen. (D-EDK, 2016: S. 2) D.h. die Schulen können individuell bestimmen in welchem Ausmaß jedes einzelne Modul unterrichtet wird.
Für weitere Informationen bzgl. Lehrplan21 siehe https://www.lehrplan.ch/

Repenning veröffentlichte bereits 2014 in der Schriftenreihe der Hasler-Stiftung den Text „Computational Thinking in der Lehrerbildung“. In diesem versucht er „[…] den Begriff Computational Thinking in der Lehrerbildung in einer praxisorientierten Weise einzuführen.“ (Repenning, 2014: S. 3) Diese Einführung erfolgt für alle neuen Lehrkräfte ab dem Jahr 2017 im Rahmen eines verpflichtenden Trainings in Computer Science Education. (Repenning, 2016: S. 52) Grund dafür ist u.a. die Notwendigkeit die Anforderungen des Lehrplan21 erfüllen zu können.

Die Fachhochschule Nordwestschweiz unterstütz beispielsweise Lehrkräfte anhand von Unterlagen. Diese sind auf einer eigenen Plattform abzurufen: http://bit.ly/2uKA7sk

2.3.1.2 Ausgewählte Unternehmen

2.3.1.2.1 Google

Computational Thinking zählt für Google zu den fünf „core skills“[16]. Das Erlernen und erfolgreiche Anwenden von CT stellt also - besonders für bzw. in der Zukunft - eine große Notwendigkeit dar, da dies den Menschen erst zu einem produktiven und erfolgreichen Mitglied der Gesellschaft werden lässt. Demzufolge werden auch die zukünftig gestellten Aufgaben bzw. Anforderungen an die Arbeitnehmerin/den Arbeitnehmer ganz andere sein, als diese beispielsweise heute sind. (Johnson & Wocicki, 2017)

Um das Anwenden und Vermitteln von CT in der Schule zu fördern, erstellte Google den kostenlosen Onlinekurs „*Computational Thinking for Educators*“. (Google, 2016) Inhalt sind das Erforschen von Algorithmen, das Finden von Mustern, das Entwickeln von Algorithmen und ein abschließendes Projekt. Als Zielgruppe definiert Google im Besonderen Lehrende von Kindern, die den Kindergarten besuchen, bis hin zu Lehrenden, die Jugendlichen in der 12. Schulstufe unterrichten. Zusätzlich dazu schuf Google auch eine Plattform, auf welcher die TeilnehmerInnen untereinander und mit anderen Personen kommunizieren können. (Google Community, 2017)

Als Unterstützung für das Anwenden von CT finden die Lehrenden unter der Bezeichnung „*Exploring Computational Thinking*“ (= ECT) eine große Auswahl von Stundenplanungen (gegliedert in die jeweiligen Unterrichtsfächer und Altersstufen), Lernunterlagen, Videos, Programme (in Python oder Pencil Code), u.v.m. (Google for Education, 2017) Diese

[16] Diese beschreiben die wichtigsten Fähigkeiten einer Person, welche im Bereich des Lernens und des Arbeitens als notwendig gelten: Kommunikationsfähigkeit, rechnerische Fähigkeiten, Informations- und Kommunikationstechnik, Problemlösefähigkeit und Zusammenarbeit mit anderen. (SQA, 2017)

Materialien zielen alle auf die Förderung von jenem Konzept inkl. den dazugehörigen Schlüsseltechniken ab, an dessen Entwicklung Google maßgeblich beteiligt war. (Siehe Kapitel 2.1.1.3 BBC und Google.)

Neben dem Onlinekurs und dem ECT ist Google auch Förderer von vielen weiteren Projekten, die die Verbreitung von CT als Ziel haben, z.B. code.org (https://code.org/) oder Project Bloks (https://projectbloks.withgoogle.com/).

2.3.1.2.2 BBC

Ab dem Jahr 2020 werden lt. dem *US Department of Labor* voraussichtlich eine Million Arbeitsplätze im Bereich Computing unbesetzt sein. Grund dafür ist das Fehlen von qualifizierten Kandidaten, welche die Kunst des Programmierens und die Anwendung von CT beherrschen. (Harel, 2016)

Um diesem Problem entgegenzuwirken, hat BBC das Projekt micro:bit ins Leben gerufen. (Yuen, 2017) Gemeinsam mit 28 anderen Unternehmen wurde der kleine, programmierbare Computer entwickelt und die dafür notwendige Programmierumgebung geschaffen. (Siehe dazu Kapitel 2.3.2.1 BBC micro:bit.) Um die Verbreitung und im weiteren die Anwendung des micro:bits zu fördern, erhält jede Schülerin/jeder Schüler im Alter von 11 bzw. 12 Jahren ein Device kostenfrei zur Verfügung gestellt. Das Programmieren und korrekte Verwenden wird gemeinsam im Unterricht erlernt. (BBC, 2017d)

Unabhängig davon stellt BBC online eine Einführung für CT inkl. Beschreibungen der einzelnen Schlüsseltechniken (Siehe Kapitel 2.1.1.3 BBC und Google.) zur Verfügung. (BBC, 2017e) Weiters ist es pro Abschnitt möglich einen Test zu absolvieren, um das neu erworbene Wissen abzuprüfen. (BBC, 2017f)

2.3.1.2.3 Weitere Unternehmen

Neben BBC haben noch weitere Unternehmen den zuvor beschriebenen Grund (d.i. zukünftiges Fehlen von qualifizierten Kandidaten, die Programmieren und CT anwenden können) als Anlass genommen um in die ArbeitnehmerInnen von morgen zu investieren. (Smith, 2016)

- Apple: Ausbau von Möglichkeiten als Kind Programmieren zu erlernen durch weiterlaufende Investition in Workshops und durch die Weiterentwicklung des Lehrplans

- Cartoon Network: Investition in eine Kampagne ($ 30 Millionen) um junge Menschen zum kreativen Programmieren zu begeistern
- Facebook: Erreichen von Erwachsenen, z.B. Eltern oder Vormünder, um diesen Ressourcen bzgl. dem Programmieren zu Verfügung zu stellen
- Microsoft: Investition ($ 75 Millionen) in die CS Ausbildung von LehrerInnen und SchülerInnen und Einführung der Forderung „Make CS Count“ in den gesamten Vereinigten Staaten
- uvm.

2.3.1.3 Ausgewählte (länderübergreifende) Verbände bzw. Programme

Mittlerweile gibt es unzählige Verbände bzw. Programme, welche sich mit dem Lehren und der Vermittlung von Computational Thinking befassen. Um den Rahmen dieser Diplomarbeit nicht zu sprengen, wird hier auf eine detaillierte Beschreibung dieser verzichtet und begrenzt auf das (alphabetische) Festhalten einer Auswahl:

- Barefoot – Computing at School
- Bebras bzw. Bieber der Informatik
- Computer Science Education Week – Hour of Code
- Computer Science Teacher Association (= CSTA)
- Computer Science Unplugged
- International Society for Technologie in Education (= ISTE)
- Projekt „COOL” (=COoperative Open Learning) der Universität Klagenfurt
- Projekt „Informatik erLeben“ der Universität Klagenfurt
- Projekt „Learn to proGrAME“ der Forschungsgruppe CSLEARN-Educational Technologies der Universität Wien
- Projekt "MadeByKids" der Forschungsgruppe CSLEARN-Educational Technologies der Universität Wien in Kooperation mit dem DaVinciLab

2.3.2 Ausgewählte (unterstützende) Tools zur Vermittlung von CT

Die hier vorgestellten Tools sind alphabetisch geordnet und unterstützen u.a. beim Erlernen des Programmierens unter Verwendung von CT oder stellen Materialien für das Erlernen von CT zur Verfügung. Weitere sind beispielsweise Abenteuer Informatik (Gallenbacher, 2017), AgentSheet (www.agentsheets.com), AgentCubes online (www.agentcubesonline.com), code.org (https://code.org/curriculum/course3/1/Teacher), TigerJython (http://www.tigerjython.ch), etc.

2.3.2.1 BBC micro:bit

Der BBC micro:bit ist ein kleiner, programmierbarer Computer, welcher in Großbritannien von zurzeit 29 Unternehmen entwickelt und finanziert wird. (BBC micro:bit, 2015) Diese verfolgen das Ziel, Kinder und Jugendliche für digitale Kreativität zu begeistern und eine neue Generation von technischen Pionieren aufzuziehen. (BBC micro:bit, 2016) Jede Schülerin und jeder Schüler erhält deshalb in England und Wales im Year 7 (d.i. im Alter von 11 bis 12 Jahren), in Nord Irland im Year 8 (d.i. im Alter von 11 bis 12 Jahren) und in Schottland im S1 (d.i. im Alter von 11 bis 12 Jahren) gratis einen BBC micro:bit und erlernt im Rahmen des Unterrichts dessen Programmierung bzw. dessen Verwendung. Diese Aktion wurde erstmals im März 2016 durchgeführt (BBC, 2016b) und war ein voller Erfolg. 90 Prozent der TeilnehmerInnen gaben an, dass der BBC micro:bit verdeutlicht, dass tatsächlich jeder programmieren kann. (BBC, 2017d)

Das Erlernen des Programmierens unter Verwendung des BBC micro:bit kann entweder auf der Homepage http://microbitworld.me oder via App erfolgen. In beiden Fällen ist die Kommunikation (via Bluetooth bzw. Kabel) sehr einfach.

Der Minicomputer stellt drei Möglichkeiten zum Erlernen des Programmierens zur Verfügung: Python, Block Editor von Microsoft und JavaScript von CodeKingdoms. Für jede dieser Möglichkeiten gibt es Tutorials und vorgeschlagene Projekte mit einem Lösungsvorschlag, die umgesetzt werden können. Die (auf der Homepage) vorgegebenen Programmieroberflächen sind übersichtlich und logisch gegliedert:

- Programmieren mit Python: Es können vorgegebene Projekte geöffnet und adaptiert werden bzw. neue, eigene Ideen umgesetzt werden. Die selbst erstellten Scripts sind unter „My Scripts“ einsehbar.

Abbildung 4: BBC micro:bit - Programmieren mit Python

- Programmieren mit dem Block Editor von Microsoft: Die Gliederung der Programmoberfläche ist fast ident wie jene im Tool „Scratch“. (Siehe Kapitel 2.3.2.9 Scratch.) Auf der rechten Seite befindet sich die Entwicklungsoberfläche auf welcher die einzelnen Blöcke zusammengefügt werden können. Die Blöcke befinden sich gegliedert in Überpunkten im mittleren Drittel der Oberfläche. Ganz links ist ein BBC micro:bit abgebildet, auf welchem der verfasste Code ausgegeben werden kann. Möchte der User das Programm am Gerät selbst abspielen, so ist es dazu notwendig dieses herunterzuladen und auf dem Minicomputer zu speichern. Anschließend führt dieser das Programm automatisch aus.

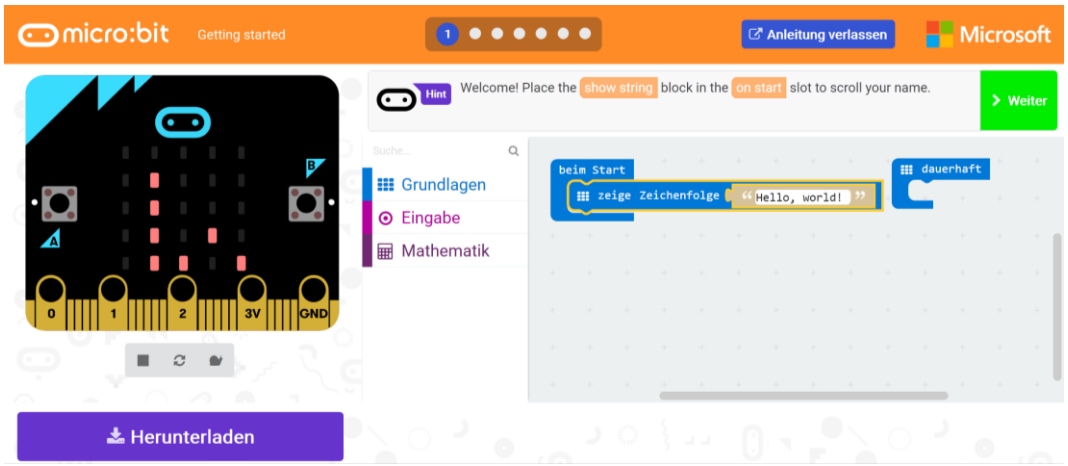


Abbildung 5: BBC micro:bit - Programmieren mit dem Block Editor

- Programmieren mit JavaScript von CodeKingdoms: Hier ist es möglich zwischen dem Block und dem Code Editor zu wechseln, d.h. Code, der im Block Editor verfasst wurde, kann im Code Editor angezeigt werden und umgekehrt.

Abbildung 6: BBC micro:bit - Programmieren mit JavaScript im Block Editor

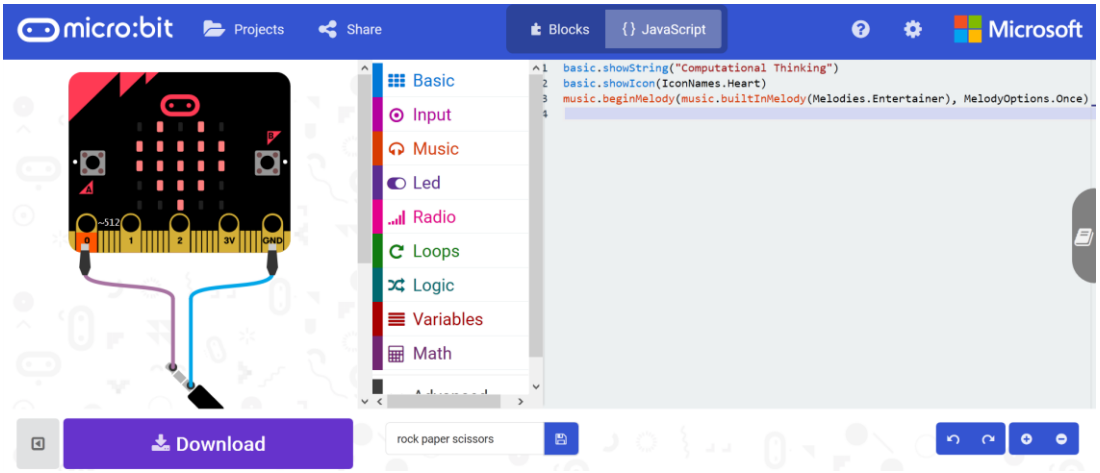


Abbildung 7: BBC micro:bit - Programmieren mit JavaScript im Code Editor

2.3.2.2 Bebras bzw. Biber der Informatik

Bebras bzw. Biber der Informatik ist ein Wettbewerb für SchülerInnen, der Wissen über Informatik vermittelt und die Anwendung von CT fördert. (Bebras, 2017) Durch die Präsentation der Fragen in Form von Rätseln werden die Informatikkonzepte leicht zugänglich gemacht. (OCG, 2017c)

Unabhängig vom Wettbewerb können die Fragestellungen (Diese werden jedes Jahr in einem Biber-Aufgabenheft gesammelt und beispielsweise von der OCG in Österreich gratis verteilt.) von Lehrenden als Einstieg für bestimmte Informatikthemen verwendet werden. (OCG, 2017c) Wollen die SchülerInnen gerne alleine üben, so können sie dies mit Hilfe der App „Bebras“ problemlos am Handy.

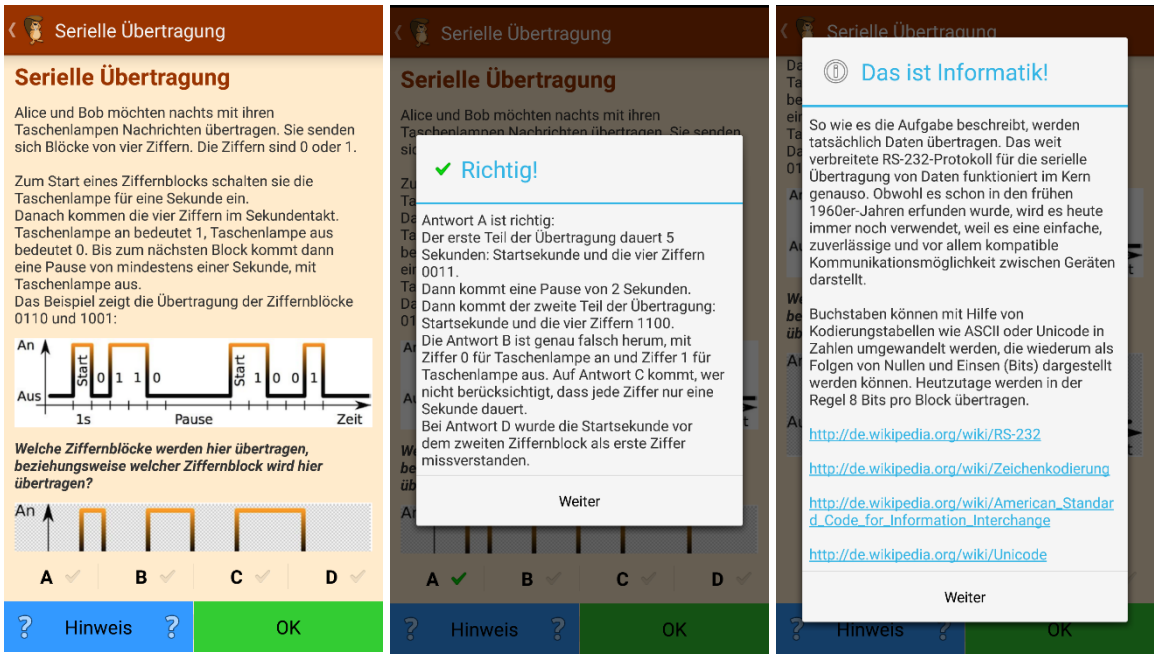


Abbildung 8: App „Bebras"

2.3.2.3 Bee-Bot/Blue-Bot bzw. Pro-Bot

Der Bee-Bot/Blue-Bot ist ein Roboter, welcher für jüngere Kinder entwickelt wurde. Er ist lt. den Entwicklern einfach zu bedienen und somit das perfekte Tool um die Bildung von Reihenfolgen, das Abschätzen von diversen Strecken und das Lösen von Problemen zu erlernen. (Terrapin Software, 2016)

Der Roboter kann sich in vier Richtungen bewegen (d.i. nach vor, nach links, nach rechts und zurück), eine Strecke von ca. 16 Zentimeter zurücklegen, sich um maximal 90 Grad drehen und bis zu 40 Befehle speichern. Nach jedem eingegebenen Befehl blinkt und piept das Gerät, um zu verdeutlichen, dass dieser soeben ausgeführt wurde. Ist das gesamte Programm durchgelaufen, wird dies wiederum durch Blinken und Piepen bekannt gegeben. (Terrapin Software, 2016)

Neben dem Bee-Bot/Blue-Bot gibt es auch noch den Pro-Bot. Dieser hat die Form eines Autos und kann mit Hilfe von Logo (Siehe Kapitel 2.3.2.7 FMSLogo.) unter Anwendung der integrierten Pfeil- und Zahlentasten am Bot selbst programmiert werden. Weiters sind Berührungs-, Ton- und Lichtsensoren im Gerät integriert und können verwendet werden. (Terrapin Software, 2016)

2.3.2.4 Hour of Code

Im Rahmen der „Computer Science Education Week" findet einmal pro Jahr die „Hour of Code" statt. Diese wird an über 1.300 Standorten, d.h. in über 180 Ländern, veranstaltet und zählt somit zu einer der größten Veranstaltung zum Thema CS und Programmierung in der heutigen Zeit. (Code.org, 2017) Die „Hour of Code" kann auch unabhängig von dieser Woche in kleinerem Rahmen abgehalten werden. (CSEd Week, 2017) Als Einstimmung darauf ist es möglich Inhalte einstündiger Tutorials (überwiegend auf Englisch) auf der Website https://csedweek.org/ zu erarbeiten.

Ziel der „Hour of Code" ist es, zu zeigen, dass jede und jeder die Grundlagen (des Programmierens) in kurzer Zeit erlernen kann. Unternehmen wie beispielsweise Microsoft, Apple oder Amazon unterstützen dies. (Code.org, 2017)

2.3.2.5 Computer Science Unpluggeld

CS Unplugged stellt auf der Homepage http://csunplugged.org/ Materialien und Aktivitätsvorschläge für Lehrende und Lernende, zu den verschiedensten Themen der Informatik, gratis zur Verfügung. Besonderer Fokus liegt neben dem computerfreien

Lernen auch auf der praktischen Vermittlung der Konzepte von CT. Diese sind besonders bei Themen wie „binary“, „numbers“ oder „algorithms“ integriert. (CS Unplugged, 2017a)

Die neue Homepage http://cs-unplugged.appspot.com/de/ wird ausschließlich für Lehrende gestaltet und ist darauf ausgelegt, dass die zuvor offline gelernten Inhalte nun auch in praktischer Verbindung mit dem Computer erarbeitet werden. Sie beinhaltet u.a. Stundenplanungen, Lehrvideos und Programmierübungen. (CS Unplugged, 2017b)

2.3.2.6 Lego Mindstorms

Lego bietet mit Mindstorms Kindern und Jugendlichen die Möglichkeit ihre Kreativität unter Verwendung von Lego in Kombination mit dem Programmieren auszuleben. Neben bereits vorgegebenen Skulpturen, wie z.B. ROBODO3R, TRACK3R, etc. können auch selbst welche entwickelt werden. Das Herzstück aller stellt der EV3 dar, ein programmierbarer Minicomputer, mit den dazugehörigen Sensoren (d.s. Farb-, Tast-, Infrarot- und Remote Infrarot-Sensor) und Motoren. (Mindstorms, 2017)

Der Lego Mindstorm kann auf zwei Arten programmiert werden: via EV3 bzw. via Computer. In beiden Fällen ist die Entwicklungsoberfläche sehr übersichtlich gehalten und basiert auf dem Programmieren mit Blöcken. Diese sind in Gruppen unterteilt, beispielsweise am Computer in „Flow“, „Action“, „Sensor“, „Data Operations“, „Advanced“ und „My Blocks“.

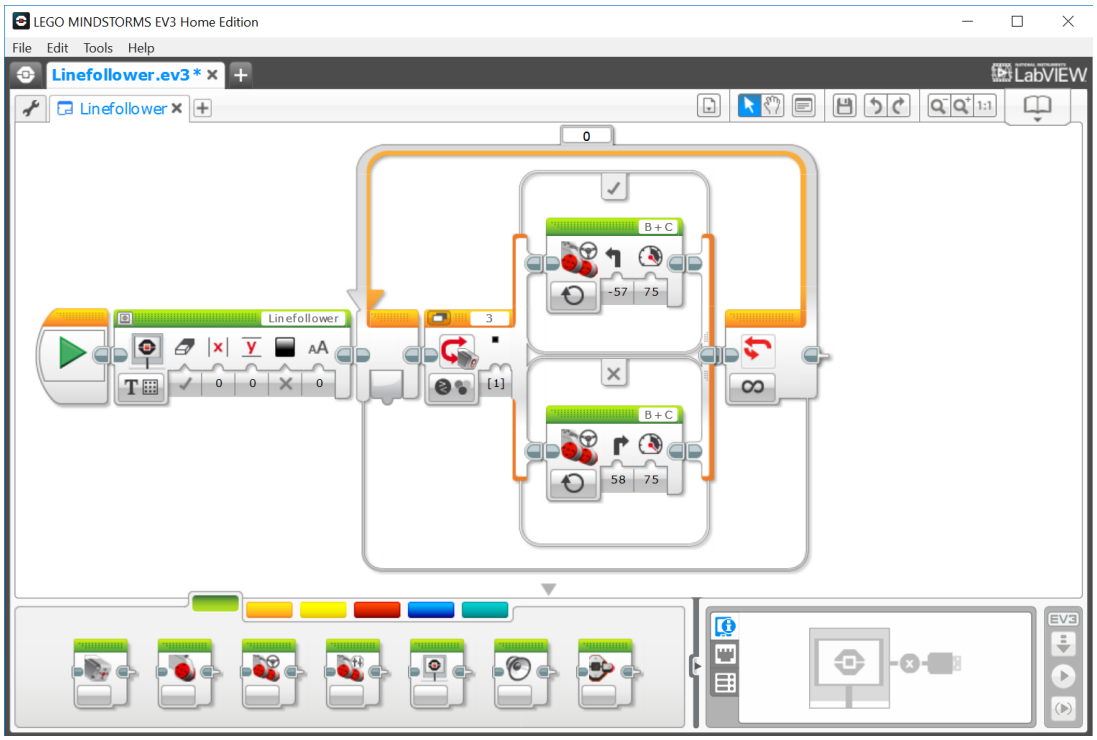


Abbildung 9: Lego Mindstorms - Programmieren mit dem Computer

Rund um Lego Mindstorms hat sich das Förderprogramm First Lego League (= FLL) gebildet. Dieses zielt im Rahmen eines Wettbewerbs darauf ab Kindern und Jugendlichen den „[…] Zugang zu naturwissenschaftlichen Fächern [zu erleichtern,] sowie ihre Motivation, einen Ingenieur- oder IT-Beruf zu erlernen […]“, zu fördern. (FLL, 2017a)

Kinder und Jugendliche (im Alter zwischen 9 und 16 Jahren) können in Teams von drei bis 10 Personen am Wettbewerb teilnehmen. Sie entwickeln gemeinsam einen vollautomatischen Roboter, welcher vorgegebene Missionen bestmöglich meistern soll. Der Wettbewerb steht jedes Jahr unter einem anderen Motto: 2017/18 ist dies „HYDRO DYNAMICS^{SM}, d.h. der Fokus liegt auf dem Bereich „Wasser – wie wir es finden, transportieren, nutzen oder es beseitigen“. (FLL, 2017b)

2.3.2.7 FMSLogo

FMSLogo ist eine Entwicklungsumgebung für die Programmiersprache Logo, mit welcher sich bereits Seymour Papert in seinem Buch „Mindstorms: Children, Computers, and Powerful Ideas“ im Jahr 1980 beschäftigte. (Papert, 1980: S. 126-151)

Die grafische Benutzeroberfläche des Programms ist einfach gehalten und soll zum Lernen anregen. (Sourceforge, 2017) Diese ist in drei Bereiche untergliedert: In den oberen zwei Drittel wird grafisch die Ausgabe des Befehls dargestellt, welchen die Benutzerin/der Benutzer im linken unteren Drittel zuvor eigegeben hat. Im rechten unteren Drittel stehen mehrere Auswahlmöglichkeiten zur Verfügung. Neben der Option das Protokoll anzuzeigen, die grafische Oberfläche zu leeren und vielen mehr, ist auch ein Editor vorhanden. Dieser kann für umfangreichere Implementationen verwendet werden.

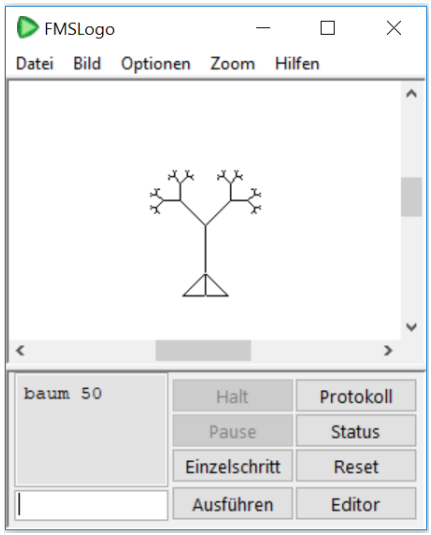


Abbildung 10: FMSLogo – Entwicklungsumgebung

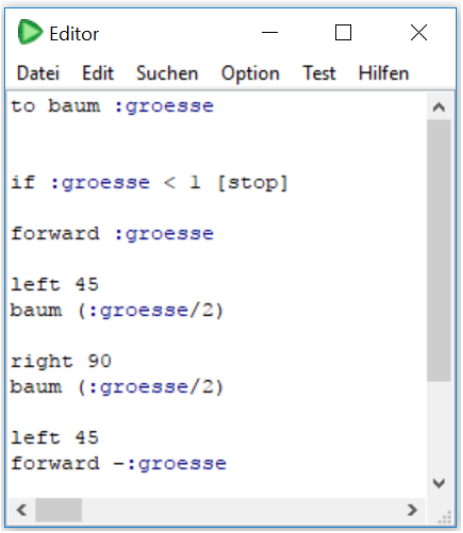


Abbildung 11: FMSLogo - Editor

2.3.2.8 Raspberry Pi

Der Raspberry Pi ist ein kleiner Computer (den es in mehreren Modellen gibt), mit welchem Programmieren gelernt und unzählige Projekte umgesetzt werden können. Neben einer großen Community gibt es auch Coderdojos, d.s. organisierte Treffen, wo sich Kinder und Jugendliche zwischen 7 und 17 Jahren monatlich treffen und gemeinsam Projekte entwickeln. (Raspberry Pi Foundation, 2017a) Auf der Homepage https://www.raspberrypi.org/ können außerdem verschiedene Betriebssysteme mit bzw. ohne grafischer Benutzeroberfläche (z.B. RASPBIAN, NOOBS) heruntergeladen und auf Ressourcen (gegliedert in „Teach“, „Learn“ und „Make“ (Raspberry Pi Fooundation, 2017b)) zugegriffen werden. Auch in den unzähligen Ausgaben des MagPi können Ideen für Projekte geholt werden. Dies ist eine Zeitschrift (auch online abrufbar), welche sich auf das Beschreiben und Umsetzen von Raspberry Pi Projekten spezialisiert hat.

Verwendet der User das Betriebssystem RASPBIAN mit grafischer Benutzeroberfläche, so kann dieser beispielsweise mit Python 2 bzw. 3, mit Scratch, etc. programmieren. Auch das Spielen von Minecraft Pi bzw. das Programmieren im Spiel Minecraft Pi ist möglich.

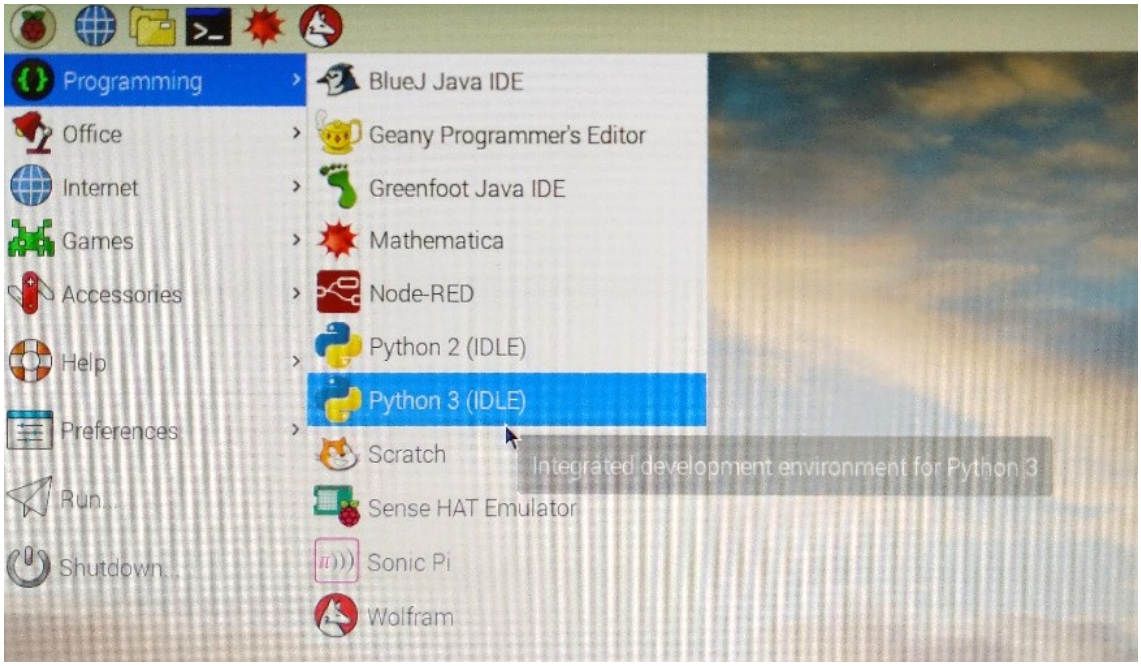


Abbildung 12: Raspberry Pi - Betriebssystem RASPBIAN

2.3.2.9 Scratch

Die EnwicklerInnen von Scratch verfolgten mit dessen Veröffentlichung im Mai 2007 das Ziel, Programmieren für jede und jeden – unabhängig von Alter und Hintergrund – spannend und interessant zu gestalten. Mittlerweile ist die Homepage

https://scratch.mit.edu Mittelpunkt einer großen online Community geworden, auf welcher Projekte (z.B. Videospiele, interaktive Newsletter, Wissenschaftssimulationen, animierte Tanzwettbewerbe, uvm.) geteilt, diskutiert und individuell verändert werden können. (Resnick & Maloney & Monroy-Hernández & Rusk & Eastmond & Bennan & Millner & Rosenbaum & Silver & Silverman & Kafai, 2009: S. 60)

Die Entwicklerumgebung von Scratch ist sehr einfach gehalten und in vier Teile gegliedert: Im rechten Drittel befindet sich der Bereich, in welchem unter Verwendung der Blöcke (auffindbar im mittleren Drittel) Ideen implementiert werden können. Die Blöcke sind in verschiedene Kategorien untergliedert, z.B. Bewegung, Aussehen, Klang, etc. Im oberen linken Drittel kann der programmierte Inhalt ausgeführt werden. Soll der Hintergrund bzw. das Sprite verändert werden, so kann dies im unteren, linken Drittel geschehen.

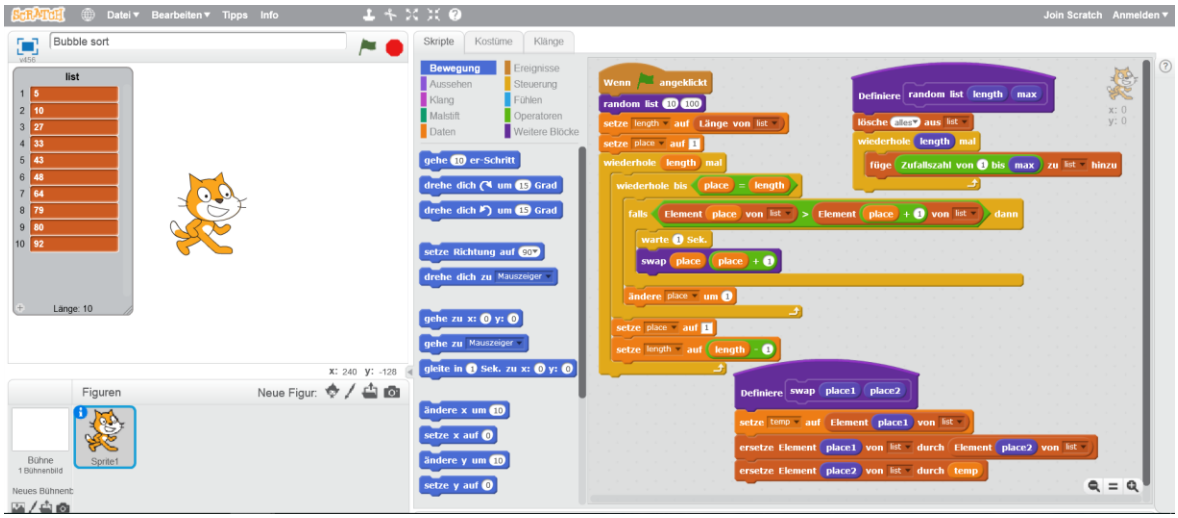


Abbildung 13: Scratch - Entwicklungsumgebung

Ist das Umsetzen des Projektes gelungen, so kann eine Analyse des fertigen Scratch Programms vorgenommen und dadurch Verbesserungsmöglichkeiten gefunden werden. Dazu können beispielsweise Tools wie Dr. Scratch (http://www.drscratch.org/), Scrape (http://happyanalyzing.com/downloads/) oder Hairball (https://github.com/ucsb-cs-education/hairball) verwendet werden.

Dr. Scratch analysiert verschiedene Bereiche, z.B. „Abstraction“, „Parallelism“, „Logic“, etc. welche eng mit einigen Schlüsseltechniken von CT in Verbindung stehen. Für diese Bereiche ist auf einer jeweils eigenen Seite genauestens definiert und beschrieben anhand welcher Kriterien die Punktevergabe erfolgt. Dadurch ist das Endresultat für die Testenden leichter nachvollziehbar und möglich durch Reflexion das eigene Programm zu verbessern.

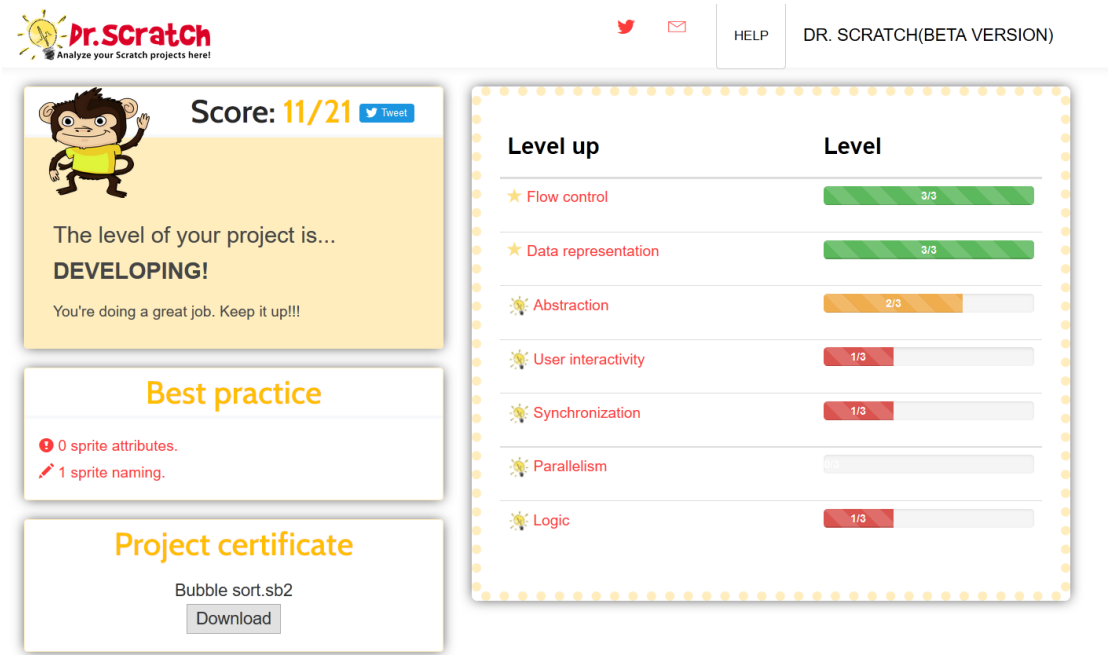


Abbildung 14: Dr. Scratch

3 Kompetenzorientierter Lehrplan für das Unterrichtsfach Informatik in Österreich (5. AHS)

In Österreich findet das Lehren von Computational Thinking in der Oberstufe ausschließlich im Rahmen des Unterrichtsfachs Informatik statt. In den allgemeinbildenden höheren Schulen ist dies also nur auf einen Jahrgang, d.i. die 5. Klasse, beschränkt. (Siehe Kapitel 2.3.1.1.2 Österreich (Stand: 01. August 2017).)

Der vorliegende AHS-Lehrplan des Unterrichtsfachs Informatik (gültig ab dem Schuljahr 2017/18) wird in diesem Kapitel bzgl. der zu erlernenden Kompetenzen und der vorgeschriebenen Inhalte analysiert. Weiters erfolgt die Klärung einiger wichtiger Begriffe, welche ausschließlich im Kontext der (Schul-) Bildung betrachtet werden.

3.1 Kompetenz

3.1.1 Begriffsklärung

In der Bildung werden Kompetenzen herangezogen, „[…] um die Bildungsziele, welche in Bildungssystemen erreicht werden sollen, zu charakterisieren." (Schweizer, 2006: S. 128) Weiters ist es anhand von einer Überprüfung möglich festzustellen, ob die festgesetzten Bildungsziele von den SchülerInnen tatsächlich erreicht wurden. (BMBF, 2007: S. 71)

Für eine etwas detailliertere Beschreibung wird in den österreichischen - und auch in den deutschen (Klieme, 2004: S. 11) - Bildungsstandards auf die Definition von Franz E. Weinert zurückgegriffen (BIFIE, 2017):

> „Dabei versteht man unter Kompetenzen die bei Individuen verfügbaren oder durch sie erlernbaren kognitiven[17] Fähigkeiten und Fertigkeiten, um bestimmte Probleme zu lösen, sowie die damit verbundenen motivationalen, volitionalen[18] und sozialen Bereitschaften und Fähigkeiten um die Problemlösungen in variablen Situationen erfolgreich und verantwortungsvoll nutzen zu können." (Weinert, 2001: S. 27-28)

Daraus ist zu schließen, dass Kompetenzen als Ergebnis von Lernprozessen gesehen werden können und diese neben Wissen und kognitive Fähigkeiten, „[…] das Vermögen der Selbstregulation sowie sozial-kommunikative und motivationale Elemente" (BIFIE,

[17] Kognitiv ist gleichzusetzen mit „das Denken, Verstehen oder Wissen betreffend" (Neues Wort, 2017), d.h. es bezieht sich auf mentalen Prozesse. Beispiele für kognitive Fertigkeiten und Fähigkeiten sind u.a. Konzentration, Lernen, Erinnerungsvermögen, etc. (Neuro Nation, 2017)

[18] Als Synonyme für den Begriff „Volition" wird oft die Willenskraft bzw. die Willensstärke herangezogen, d.h. es handelt sich hier um eine Fähigkeit „[…] Gedanken (Aufmerksamkeit), Emotionen und Aktionen auf das Wesentliche zu lenken […]." (Pelz, 2017)

2017) beinhalten. D.h. durch das Erlernen von Kompetenzen werden auch Verbindungen zwischen Können und Wissen hergestellt. (Klieme, 2004: S. 12) Dies erfolgt in der Regel kontextunabhängig und ermöglicht dadurch das Bewältigen verschiedenster Situationen – ob schulischer oder anderer Art. (BIFIE, 2017)

3.1.2 Verortung des Begriffes im AHS-Lehrplan

Den Einsatz von Kompetenzen im schulischen Bereich (in Österreich) begründet das Bundesinstitut für Bildungsforschung, Innovation und Entwicklung des Bildungswesens (= BIFIE) auf dessen Homepage wie folgt:

> „Kompetenzen ermöglichen es den Schülerinnen und Schülern, in variablen Situationen zu handeln und ihr Wissen zielgerichtet einsetzen zu können. Sie schaffen die Basis für den Erwerb und die Anwendung spezifischer Fähigkeiten und Fertigkeiten und verkörpern damit ein weitgehend stabiles Werkzeug, das zur Bewältigung wechselnder Herausforderungen befähigen soll.“ (BIFIE, 2017)

Die Formulierung der Kompetenzen im Lehrplan erfolgt auf zwei Ebenen (auch Kompetenzmodelle genannt): allgemein und pro Unterrichtsfach. (Dies ist wiederum abhängig vom jeweiligen Schuljahr.) (BIFIE, 2017)

Für das Unterrichtsfach Informatik in der AHS (Sekundärstufe 2) ist die allgemeine Formulierung im Lehrplan (RIS, 2017) zu finden, wohingegen die Kompetenzen an sich in digi.komp12 (PH OÖ, 2017b) ausführlicher beschrieben werden. (Siehe Kapitel 3.2 Kompetenzmodelle.)

3.2 Kompetenzmodelle

3.2.1 Begriffsklärung

Bei den Kompetenzmodellen ist streng genommen zwischen Kompetenzstrukturmodellen und Kompetenzniveaumodellen zu unterscheiden.

- Das Kompetenzstrukturmodell befasst sich mit den zu erfassenden Kompetenzdimensionen (Schweizer, 2006: S. 132) eines spezifischen Bereichs. (Klieme & Leutner, 2006: S. 883)
- Das Kompetenzniveaumodell beschreibt „[…] welche konkreten situativen Anforderungen Personen bei welcher Ausprägung einer Kompetenz bewältigen können.“ (Klieme & Leutner, 2006: S. 883)

Die Kombination dieser beiden Kompetenzmodelle ist zum einen die Grundlage für eine Operationalisierung von Bildungszielen, d.h. dadurch ist die Möglichkeit der Vergleichbarkeit von Leistungen mit Hilfe von empirischen Testverfahren im Bildungssystem gegeben. (Siehe Kapitel 3.3 Messung von Kompetenzen.) Zum anderen unterstützen Kompetenzmodelle auch die Unterrichtspraxis mit Anhaltspunkten, die neben der fachlichen Systematik von Lehrinhalten, auch Lernprozessen und Lernergebnissen im jeweiligen Lernbereich beinhalten. (BMBF, 2007: S. 71)

3.2.2 Verortung des Begriffes im AHS-Lehrplan

Jenen Kompetenzen, die pro Unterrichtsfach im dazugehörigen Lehrplan festgehalten sind, liegt immer ein „[…] abgeleitetes fachspezifisches Kompetenzmodell zugrunde, das wesentliche Kernbereiche eines Unterrichtsgegenstands umfasst und die Übersetzung abstrakter Bildungsziele in konkrete Aufgabenstellungen ermöglicht und unterstützt." (BIFIE, 2017) Auch im „zweigeteilten" Lehrplan für das Unterrichtsfach Informatik wird eine Trennung zwischen Kompetenzstruktur- bzw. Kompetenzniveaumodell vorgenommen. Dazu ein Beispiel:

- *Lehrplan (Kompetenzstrukturmodell):*
  Praktische Informatik (RIS, 2017)
  „Algorithmen erklären, entwerfen, darstellen und in einer Programmiersprache implementieren können"
- *digi.komp12 (Kompetenzniveaumodell):*
  Praktische Informatik: 4.2 Algorithmen, Datenstrukturen und Programmierung (PH OÖ, 2017b)
  - „Ich kann den Algorithmusbegriff erklären."
  - „Ich kann einfache Algorithmen nachvollziehen und erklären."
  - „Ich kann die Umsetzung von Algorithmen mit einem Computer erklären."
  - „Ich kann einfache Aufgaben mit Mitteln der Informatik modellieren."
  - „Ich kann einfache Algorithmen entwerfen, diese formal darstellen, implementieren und testen."
  - „Ich kann an Hand von einfachen Beispielen die Korrektheit von Programmen bewerten."

3.3 Messung von Kompetenzen

3.3.1 Begriffsklärung

Damit ein Leistungsvergleich auf Basis der zu erlernenden Kompetenzen sinnvoll bzw. korrekt durchgeführt werden kann, müssen die einzelnen Kompetenzen der formulierten Kompetenz(struktur- bzw. niveau-)modelle inhaltlich klar voneinander differenziert und korrekt beschrieben werden. (Schweizer, 2006: S. 133) Verschärfend dazu behauptet das BMBF (2007: S. 73) sogar: „Kompetenz kann nur leistungsbezogen erfasst und gemessen werden."

Aufbauend darauf werden Messmöglichkeiten dieser Kompetenzen erarbeitet. Bekannte Beispiele dafür sind PISA (= Programme for International Student Assessment), PIRLS (= Progress in International Reading Literacy Study) oder NAEP (= National Assessment of Educational Progress).

Die PISA-Testung findet alle drei Jahre statt, zielt auf unterschiedliche Kompetenzbereiche ab und lag beispielsweise im Jahr 2012 auf jenem der Mathematik. Im Jahr 2015 wurde speziell Wert auf die Naturwissenschaften gelegt. (TU München, 2017a) Die nächste Testung wird voraussichtlich im Jahr 2018 mit dem Schwerpunkt Lesen durchgeführt. (TU München, 2017b)

3.3.2 Verortung des Begriffes im AHS-Lehrplan

Um die erlernten Kompetenzen im Unterrichtsfach Informatik überprüfen zu können, wird zurzeit der *digi.check12* weiter entwickelt bzw. an die NOST (= Neue Oberstufe) Informatik angepasst. (PH OÖ, 2017a) Eine weitere, bereits existierende Option einen kleinen Teilbereich der zu erlernenden Kompetenzen zu überprüfen, stellt der *Biber der Informatik* dar. Bei diesem Test steht vor allem das Lösen von spannenden Rätselaufgaben zu Themenbereiche der Informatik - unter Anwendung von Computational Thinking - im Fokus. (OCG, 2016a)

3.4 Kompetenzorientierter Unterricht

3.4.1 Begriffsklärung

Mit der Einführung von Bildungsstandards[19] an österreichischen Schulen im Schuljahr 2008/09 (Wiesner & Schreiner & Breit & Pacher, 2017) erlangte die Kompetenz-

[19] Diese definieren „[…] konkret formulierte Lernergebnisse in Form von Könnensbeschreibungen (sogenannte „Can–Do–Statements") […]" (Wiesner & Schreiner & Breit & Pacher, 2017), welche aus den jeweiligen Lehrplänen abgeleitet werden.

orientierung immer mehr an Wichtigkeit. Im Fokus steht mittlerweile nicht mehr ausschließlich das überprüfbare Wissen, welches sich SchülerInnen im Laufe der Schulzeit aneignen, sondern „[…] auch andere Fähigkeiten und Fertigkeiten […]“. (Beer & Benischeck & Brock & Habringer & Herland & Mürwald-Scheifinger & Scherf & Staud & Weber & Werbowsky & Zöchlinger, 2011: S. 24) Das bedeutet, die Anwendung des Gelernten in unterschiedlichen Situationen und Aufgabenfeldern durch die Schülerin/den Schüler steht von nun an im Mittelpunkt. (Beer et al., 2011: S. 25) Demzufolge dient der Unterricht neben der Wissensvermittlung, „[…] auch der Erziehung und Vermittlung von Kompetenzen (z. B. Sozial- und Selbstkompetenz)“. (Beer et al., 2011: S.20)

Diese Wandlung von Schule beeinflusst auch den Beruf der Lehrerin/des Lehrers: Waren Lehrkräfte bisher überwiegend für das Lehren von fachbezogenem Wissen zuständig, so ist ihre zusätzliche und neue Aufgabe die Unterstützung der „[…] Entwicklung ihrer Schüler/innen beim Erwerb der vielfältigen Kompetenzen […]“. (Beer et al., 2011: S. 25)

3.4.2 Verortung des Begriffes im AHS-Lehrplan

Besonders für Lehrkräfte des Unterrichtsfachs Informatik lässt sich diese zusätzliche Aufgabe bei näherer Betrachtung nicht wirklich als neu einstufen. Dies begründet sich daraus, dass bereits vor der Einführung der Bildungsstandards die Wissensvermittlung oft bzw. meistens eng mit dem selbstständigen Anwenden des Erlernten durch SchülerInnen verknüpft war. Demzufolge wurden SchülerInnen auch früher schon darauf vorbereitet Gelerntes zur Bewältigung verschiedenster Situationen, welche mit Inhalte von Informatik zu tun hatten, einzusetzen. Als Beispiel kann der Themenbereich bzw. der Begriff „Algorithmus“ hergenommen werden. Eine Vermittlung der Theorie (d.i. die Begriffsdefinition) ist genauso wichtig, wie die Anwendung des gelernten Wissens (d.i. Implementation eines Algorithmus mit einer Programmiersprache).

4 Ressourcen

Die hier erarbeiteten Unterrichtsmaterialien bzw. -einheiten wurden überwiegend unter Verwendung der Lerntheorie des *Konstruktionismus*[20] und teilweise unter Anwendung von pädagogischen Patterns[21] für Unterrichtssequenzen von einer Dauer von 90 Minuten erstellt. Außerdem zielen diese auf die kompetenzorientierte Vermittlung von CT im Unterrichtsfach Informatik (5. AHS, Österreich) ab. Jeweils davor ist u.a. vermerkt, in welchen der vorgegebenen Unterrichtsinhalte des kompetenzorientierten Lehrplans die Materialien/die Sequenzen verortet werden können und in welchen anderen Schulstufen diese auch Anwendung finden können. Die für die Durchführung notwendigen Unterlagen inkl. der Beschreibung, wie CT[22] in den einzelnen Teilbereichen angewandt werden kann, befinden sich für jede geplante Sequenz im Anhang.

4.1 Sequenz zum Thema „Computational Thinking“

- Lehrplanverortung: *Praktische Informatik*
  - Begriffe und Konzepte der Informatik verstehen und Methoden und Arbeitsweisen anwenden können
  - Algorithmen erklären, entwerfen, darstellen

- Lehrziele:
  - Die SchülerInnen können das Konzept von Computational Thinking erklären und anhand von Beispielen ihrem Gegenüber begreiflich machen.
  - Die SchülerInnen können das Konzept von CT bei der Lösung konkreter Aufgaben erfolgreich anwenden und selbst Beispiele für das Anwenden von CT finden.
- Lehrbar in der Sekundarstufe 1 (3.-4. Klasse) und in der Sekundarstufe 2 (5.-6. Klasse)

[20] Der Konstruktionismus wurde von Seymour Papert entwickelt und basiert teilweise auf der von Jean Piagets erarbeiteten Lerntheorie des „Konstruktivismus“.
Beide Lerntheorien sagen aus, dass Lernende - unabhängig von den Umständen des Lernens - mentale Wissensmodelle konstruieren. Im Konstruktionismus geschieht dies besonders hervorragend, wenn die/der Lernende bewusst an einer Konstruktion eines öffentlichen Projekts beteiligt ist. (Papert, 1980: S. 1) D.h. für eine erfolgreiche Anwendung dieser Lerntheorie ist neben dem aktiven Handeln der/des Lernenden auch das eigenverantwortliche Mitarbeiten an Projekten notwendig. Daraus resultiert ein personenzentrierter Ansatz, welcher die Lernenden dazu animieren soll das bereits vorhandene Wissen als Grund für einen Ausbau dieses zu nehmen.

[21] Die grafische Darstellung der Zusammengehörigkeit der pädagogischen Patterns und eine kurze theoretische Beschreibung dieser (Standl, 2013) können dem Anhang (Siehe 9.4 Pädagogische Patterns.) entnommen werden.
Weiters ist anzumerken, dass in allen der gehaltenen Unterrichtssequenzen besonderen Wert auf aktives Zuhören durch die Lehrkraft, d.i. die Communication bzw. vertiefend die Face to Face Communication, gelegt wurde. Dies geschah vor allem bei den Erarbeitungen im Plenum bzw. bei Gesprächen mit SchülerInnen oder der Beantwortung von Fragen einzelner.

[22] Hier wurde auf die Schlüsseltechniken bzw. Konzepte und Methoden von BBC und Google zurückgegriffen, da diese überwiegend in etwas abgewandelter oder erweiterter Form auch in den weiteren Erarbeitungen (Siehe Kapitel 2.1.1 Die Schlüsseltechniken bzw. Konzepte und Methoden von CT.) vorhanden sind und somit die Kerninhalte von CT am treffendsten repräsentieren.

- Notwendige Vorkenntnisse von Seiten der SchülerInnen: Keine
- Pädagogische Patterns: Group Formation[23], Group Work[24], (Peer Check[25])
- Geplante Sequenz:

Geplante Dauer	*Inhalt*	*Arbeitsform*	*Konzepte und Methoden von CT*
ca. 10 min	Einführung in das Thema: • Frage an die SchülerInnen: Was stellt ihr euch unter dem Begriff CT vor? • kurze Beschreibung von CT • Weshalb sollte CT jede/r anwenden können? • Erläuterung der Anwendungsbereiche von CT • (evtl. eingehen auf die historische Entwicklung von CT)	Vortrag durch Lehrenden	Abstraction
ca. 10 min	Erarbeitung der notwendigen Vokabeln (d.s. Decomposition, Pattern Recognition, Abstraction, Algorithm/s) → Zuordnung der Begriffen zu den Erklärungen → gemeinsames Vergleichen der Lösung (Handout) und Klärung auftretender Fragen	Teamarbeit	Decomposition, Abstraction
ca. 20 min	Gemeinsame Erarbeitung eines Beispiels, damit die einzelnen Teilbereiche leichter nachvollziehbar sind → Römische in arabische Zahlen (mit System) umwandeln	Besprechung im Plenum	Decomposition, Pattern Recognition, Abstraction, Algorithm
ca. 25 min	Aufgabe zum Anwenden von CT: „Anleitungsloses Spiel“ →Präsentieren des erarbeiteten Spiel-Algorithmus	Team- bzw. Gruppen-arbeit (bis zu max. vier Personen)	Decomposition, Pattern Recognition, Abstraction, Algorithm
ca. 20 min	Überlegen, in welchen Bereichen des Alltags CT Verwendung finden kann und anschließendes Lösen eines selbst gewählten Beispiels anhand der gelernten Konzepte und Methoden von CT *Ziel*: selbstständig erarbeitete Lösung eines Problems anhand der Verwendung von Computational Thinking	Teamarbeit	Decomposition, Pattern Recognition, Abstraction, Algorithm

[23] Die hier angewandte Gruppenbildung geht bei den gemischten Gruppen bzw. der Mädchen-Gruppe von den SchülerInnen selbst aus. Bei der Buben-Gruppe bestimmt die Lehrkraft über die Zusammensetzung. Grund dafür ist die immer gleichbleibende Bildung der Gruppen bei vorhergehenden Gruppenaufgaben.
[24] Während der gesamten Zeit der Gruppenarbeit steht die Lehrkraft für die SchülerInnen als Unterstützung bereit. Diese wird jedoch von keiner der Gruppen benötigt.
[25] Der Peer Check kommt erst in der darauffolgenden Unterrichtssequenz bei der Wiederholung mittels Memory zum Einsatz. Hierbei kontrollieren die SchülerInnen selbst, ob die aufgedeckten Kärtchen tatsächlich zusammen passen.

	→Präsentieren der Ergebnisse		
Bonus	Lösen von zwei Aufgaben von dem *Biber der Informatik* inkl. Überlegung welche Konzepte und Methoden von CT verwendet wurden um zur Lösung zu gelangen	Einzelarbeit	Decomposition, Pattern Recognition, Abstraction, Algorithm
ca. 5 min	Reflexion und Wiederholung des Gelernten: • Angenommen, du kommst heute nach Hause und deine Großeltern fragen dich ganz interessiert, was du in Informatik gelernt hast. Wie erklärst du deinen Großeltern CT? • Du hast ein Problem, das nur sehr schwer zu lösen ist. Wie gehst du vor? • Was schlussfolgerst du daraus, wenn du viele Ähnlichkeiten zwischen verschiedenen Problemen entdeckst?	Besprechung im Plenum	

Analyse und Reflexion der durchgeführten Sequenz

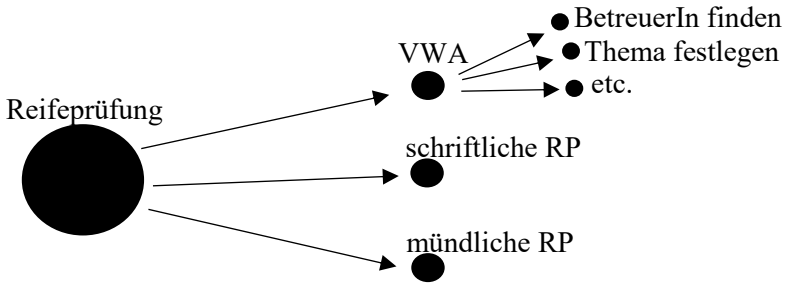
Die geplante Sequenz wurde in der 5. Klasse AHS mit einer reinen Mädchen-, einer reinen Buben- und mit zwei gemischten Gruppen erarbeitet.

Die Eingangsfrage, was sich die SchülerInnen unter dem Begriff „Computational Thinking" vorstellen, wurde in allen vier Gruppen ähnlich beantwortet. Die Kernaussage war, dass es wahrscheinlich etwas mit Programmieren zu tun haben könnte. In der Buben-Gruppe fiel sogar der Begriff „Algorithmus".

Aufbauend auf diese Antwort war es ein Leichtes die Inhalte von CT kurz zu beschreiben und auf die Frage einzugehen, weshalb ausgerechnet jede und jeder einzelne CT beherrschen sollte. Im Laufe der vorhergehenden Einheiten benötigen besonders die Mädchen bei Inhalten, die nicht mit Office in Zusammenhang stehen, eine Begründung, weshalb der betreffende Unterrichtsstoff von ihnen gelernt werden solle. Auch dieses Mal wirkte die Erklärung Wunder und – mit Ausnahmen von ein oder zwei SchülerInnen pro Gruppe – waren alle motiviert etwas über CT zu erfahren bzw. dessen Anwendung zu lernen.

Die Erarbeitung der notwendigen Vokabeln (d.s. decomposition, pattern recognition, abstraction und algorithm), also der Konzepte und Methoden, erfolgte zunächst gemeinsam und nur oberflächlich. D.h. nach dem Notieren eines englischen Begriffs, erfolgte die Frage, was dieser wohl auf Deutsch bedeuten könne. Die Übersetzung der Begriffe

„decomposition“ und „algorithm“ gelang nur in einer Gruppe. Die anderen beiden Begriffe stellte für keine der Gruppen ein Problem dar. Nachdem die Übersetzung geschafft war, stellte sich nun die Frage was die jeweiligen Begriffe in Bezug auf CT bedeuten würden: Die Begriffe „decomposition“ und „pattern recognition“ konnte keine der Gruppen erklären. Jedoch fand eine gemischte Gruppe ein tolles Beispiel für die Erklärung von „decomposition“:



Die Buben schafften es als einzige Gruppe die Begriffe „abstraction“ und „algorithm“ teilweise zu erklären.

Um die Inhalte der jeweiligen Begriffe noch zu vertiefen, erhielten die SchülerInnen die Aufgabe die vorbereiteten Kärtchen korrekt zuzuordnen. Dies erfolgte mit ein paar wenigen Ausnahmen ohne Probleme. Im Anschluss daran bekamen die SchülerInnen ein Handout, welches die erarbeiteten Begriffe beinhaltete.

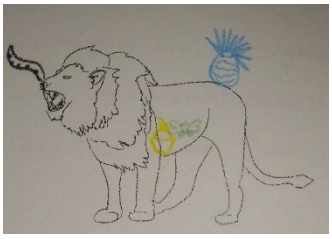
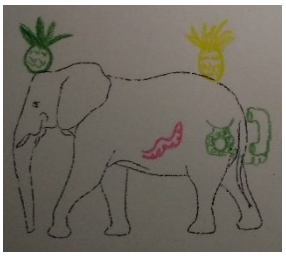
Um die einzelnen Konzepte und Methoden von CT im Zusammenspiel miteinander darzustellen, erfolgte die gemeinsame Erarbeitung eines Beispiels: Umwandeln von römische in arabische Zahlen. In der Mädchen- und in der Buben-Gruppe kam gleich zu Beginn der Hinweis, dass die Subtraktionsregel befolgt werden müsse. Auch in den anderen beiden Gruppen erfolgte dieser Hinweis spätestens bei dem Punkt, wo das Addieren der Werte, also die „abstraction“, durchgeführt werden sollte.

Bei einem erneuten Halten dieser Sequenz wäre ein Austausch des hier verwendeten Beispiels förderlich, da dieses auf Grund des Wissens bzgl. der Subtraktionsregel für die überwiegende Mehrheit der SchülerInnen eher demotivierend wirkte. Auch die Dauer, die das Erarbeiten des Beispiels in Anspruch nahm, war nicht optimal gewählt, d.h. zu lange.

Im darauf anschließenden „Anleitungslosen Spiel“ hingegen waren alle SchülerInnen wieder motiviert und freuten sich sichtlich darüber, selbst tätig zu werden. Im Gegensatz zu den Buben hielten sich die Mädchen überwiegend an den vorgeschriebenen Ablauf. Dies hatte auch zur Folge, dass diese das Spiel sehr viel länger spielen konnten. Die

ausgearbeiteten Spielanleitungen der einzelnen Gruppen unterschieden sich im Kern nicht voneinander, jedoch wurden die gegebenen Umstände unterschiedlich interpretiert. Beispielsweise fasste es eine Gruppe so auf, dass die Tiere, auf diese die einzelnen Gegenstände gezeichnet werden müssen, davor selbst auf einen Zettel gemalt werden sollten. Eine andere Gruppe ordnete ein bestimmtes Tier einer bestimmten Person zu, d.h. diese Person durfte den gewürfelten Gegenstand nur auf dem gewürfelten Körperteil des zuvor ausgewählten Tieres zeichnen. Im Verlauf des Spiels durfte also das Tier nicht geändert werden.

Ein künstlerischer Ausschnitt einer besonders fleißigen Gruppe:

(*Anmerkung*: Die AutorInnen dieser Zeichnungen haben ihr Einverständnis für die Weiterverwendung in diesem Rahmen gestattet.)

Nach dem erfolgreichen Spielen, wurden die erarbeiteten Algorithmen des Spiels kurz von einer Gruppe präsentiert. Die anderen Gruppen erklärten anschließend in einigen Sätzen, welche Unterschiede zu ihrem Algorithmus bestünden.

Um den SchülerInnen nochmals zu verdeutlichen, dass sie Computational Thinking auch für die Lösung eines Problems aus ihrem Alltag anwenden können, erhielten diese die Aufgabe sich in der Gruppe auf eines zu einigen und mittels der Anwendung von CT eine Lösung dafür zu finden. Eine kleine Auswahl der gewählten Probleme:

- Lösen von schweren Mathematik-Aufgaben
- Erfolgreiches bestehen einer Schularbeit
- Klamottensuche in der Früh
- Mangelnde Zeit für Hausübungen (Wie verbessere ich mein Zeitmanagement?)

In einigen Gruppen stellte sich die Anwendung von „pattern recognition“ und von „abstraction“ als Herausforderung dar. Andere verwendeten die vier erarbeiteten Begriffe völlig korrekt. Lediglich bei einer Gruppe war es anhand der von ihnen vorgenommen Problemlösung klar ersichtlich, dass die Materie nicht korrekt angewandt und somit nicht vollständig verstanden worden war.

Da in der Doppelstunde keine Zeit mehr für die Präsentation der bearbeiteten Problemstellungen vorhanden war, wurde dies in der darauffolgenden Sequenz nachgeholt.

Bei der abschließenden Reflexion und Wiederholung des Gelernten konnten die SchülerInnen die gestellten Fragen zufriedenstellend beantworten.

Überprüfung des gelernten Wissens in der darauffolgenden Unterrichtssequenz

Um in der darauffolgenden Sequenz feststellen zu können, ob das erarbeitete Wissen tatsächlich von den SchülerInnen verinnerlicht wurde, erfolgten drei unterschiedliche Vorgangsweisen:

- *Mündliche Stundenwiederholung:* Bei dieser wurden gemeinsam die vier besprochenen Konzepte und Methoden und deren Inhalte wiederholt, die Vokabeln erneut auf die Tafel geschrieben und die Symboldarstellungen aufgezeichnet. Die SchülerInnen konnten sich an alle grafischen Darstellungen inkl. deren Erklärungen erinnern. Jedoch stellte der englische Begriff „decomposition" eine Herausforderung dar. Die deutsche Übersetzung fiel den SchülerInnen ohne Probleme ein.
- *Schriftliche Stundenwiederholung:* Um eine differenzierte Rückmeldung bzgl. der gelernten Inhalte von jedem/r SchülerIn zu erhalten, wurde die Wiederholung in einer Gruppe schriftlich durchgeführt. Die Resultate waren durchgehend sehr zufriedenstellend. Jedoch war auch hier die Schwierigkeit der Benennung der einzelnen Konzepte und Methoden mit den englischen Begriffen teilweise präsent. Die Zuordnung der einzelnen Vokabeln zu den Aussagen hingegen stellte kein Problem dar.
- *Spielen eines Memorys:* Dieses besteht aus den Kärtchen der vorhergehenden Sequenz, wo eine Zuordnung der Begriffe zu den jeweiligen Erklärungen erfolgt. Das Memory wird zu zweit gespielt und zielt darauf ab, dass möglichst viele korrekte Paarbildungen von jeder/m erzielt werden.

Im direkten Vergleich der drei beschriebenen Stundenwiederholungen wurde klar ersichtlich, dass spielerisches Erinnern mittels Memory für die SchülerInnen am motivierendsten und begeisterndsten war. Jedoch konnte dadurch nicht differenziert festgestellt werden, welche SchülerInnen die Inhalte der vorhergehenden Sequenz sehr gut bzw. nur begrenzt verstanden und verinnerlicht haben. Auch eine Beurteilung, ob CT nun aktiv angewandt werden kann, stellte sich als nicht wirklich möglich dar. Um die soeben

beschriebenen Mankos zu umgehen, ist die Durchführung einer schriftlichen Stundenwiederholung eine Möglichkeit. Diese wiederum wurde von den SchülerInnen weniger positiv aufgefasst und nur mit großem Wiederstreben ausgefüllt. Die mündliche Stundenwiederholung stellt in dieser Gegenüberstellung einen Mittelweg dar. D.h. eine Überprüfung, ob die vermittelten Inhalte tatsächlich gelernt und verstanden wurden, ist nicht bei allen, jedoch bei einigen SchülerInnen, möglich. Ein Eingriff von Seiten der Lehrkraft bei nicht korrekt wiedergegebenem Stoff kann unmittelbar passieren und im Anschluss daran können sich alle, auch die vergesslicheren SchülerInnen, wieder an die Inhalte erinnern.

4.2 Sequenz zum Thema „Algorithmen“

- Lehrplanverortung: *Praktische Informatik*
  - Algorithmen erklären, entwerfen, darstellen und in einer Programmiersprache implementieren können
- Lehrziele:
  - Die SchülerInnen können den Begriff „Algorithmus“ erklären.
  - Die SchülerInnen können einfache Algorithmen nachvollziehen und erklären.
  - Die SchülerInnen können die Umsetzung von Algorithmen mit einem Computer erklären.
  - Die SchülerInnen können einfache Algorithmen entwerfen, diese formal darstellen, implementieren und testen.
- Lehrbar in der Sekundarstufe 1 (3.-4. Klasse) und in der Sekundarstufe 2 (5. Klasse)
- Notwendige Vorkenntnisse von Seiten der SchülerInnen: Keine
- Pädagogische Patterns: Individual Work[26]
- Geplante Sequenz:

Geplante Dauer	*Inhalt*	*Arbeitsform*	*Konzepte und Methoden von CT*
ca. 15 min	Notieren der heutigen Morgenroutine mit Nummerierung, z.B.: 1. Wecker klingelt 2. Augen öffnen 3. … Der notierte Ablauf kann von den MitschülerInnen verwendet werden um diesen nachzumachen. = Algorithmus für Morgenroutine	Einzelarbeit Besprechung im Plenum	Decomposition, Abstraction, Algorithm

[26] Die SchülerInnen dürfen die vorgegebenen Aufgaben und damit verbundenen Probleme in Scratch eigenständig lösen. Dies setzte bei den meisten einen intensiven Denkprozess in Gang. Dabei wurde neues Wissen erarbeitet und kleine Erfolge den MitschülerInnen begeistert präsentiert.

ca. 15 min	Algorithmus: • gemeinsame Überlegung einer Definition • gemeinsame Überlegung, welche Eigenschaften ein Algorithmus erfüllen muss Herstellung eines Bezugs zum Programmieren • Beim Programmieren ist eine detaillierte Handlungsanweisung für den Computer wichtig, da er sonst nichts mit dem geschriebenen Code anfangen kann. → Fehlermeldung	Besprechung im Plenum	Decomposition, Pattern Recognition, Abstraction, Algorithm
ca. 15 min	Programmieren mit Scratch • Erklärung des Aufbaus der Programmierumgebung • Erklärung der wichtigsten Grundlagen: o Sprites erstellen/ändern/löschen o Hintergrund ändern o der erste Block, damit ein Algorithmus funktionieren kann →  o Einfügen weiterer Blocks o Starten des Programms	Vortrag durch Lehrenden	Abstraction
ca. 45 min	Programmieraufgaben mit anschließender Abgabe	Einzelarbeit	Decomposition, Pattern Recognition, Abstraction, Algorithm

Analyse und Reflexion der durchgeführten Sequenz

Diese Unterrichtssequenz wurde in einer gemischten, sehr interessierten und diskussionsliebenden Informatikgruppe gehalten.

Nach dem Geben der ersten Arbeitsaufgabe, d.i. das schrittweise Notieren der heutigen Morgenroutine, folgten zunächst fragende Blicke der SchülerInnen. Erst nach einer kurzen Begründung weshalb die Aufgabe für den weiteren Verlauf der Unterrichtssequenz wichtig sei, wurde mit dem Lösen dieser begonnen. Während der Ausarbeitung stellte sich das detaillierte Festhalten der einzelnen Schritte als Herausforderung dar. Anstelle beispielsweise einer genauen Beschreibung, wie das Frühstück zubereitet wurde, notierten einige SchülerInnen lediglich den Begriff „frühstücken“. Erst nach dem Hinweis, dass tatsächlich jeder einzelne Handgriff notiert werden solle, wurde präziser gearbeitet.

(Weitere Möglichkeiten aus dem Alltag für das Festhalten eines Algorithmus wären z.B. das Kochen eines Gerichts oder das Einkaufen von Lebensmittel.) Nach Ablauf der vorgegebenen Zeit, unabhängig davon, wie weit die SchülerInnen bereits mit dem Beschreiben gekommen waren, erfolgte die gemeinsame Erarbeitung des Begriffs „Algorithmus“. Das Ergebnis des Definitionsversuchs war folgendes: Ein Algorithmus ist eine schrittweise Abfolge von Handlungen, die (nur) in dieser Reihenfolge abgearbeitet werden können. D.h. er beschreibt einen vorgegebenen Ablauf, mit dem das vorhandene Problem gelöst werden soll.

Auf Basis dessen war es einfach die notwendigen Eigenschaften eines Algorithmus mit den SchülerInnen zu erarbeiten. Ohne Unterstützung beschrieben diese die Inhalte der Begriffe „Determinismus“ und „Verständlichkeit“. Lediglich die Fachtermini mussten noch ergänzt werden. Bei dem Begriff „Effektivität“ benötigten die SchülerInnen ein wenig Unterstützung um eine korrekte Beschreibung dessen zu finden. Der letzte Begriff „Endlichkeit“ stellte sich als Herausforderung dar. Auch nach dem Hinweis, dass die notiere Morgenroutine in Form eines Algorithmus irgendwann endet, kamen nur unpassende Vermutungen.

Um nun einen Übergang zwischen dem Erarbeiten des Begriffs inkl. dessen Eigenschaften und dem Programmieren herzustellen, erfolgte die Frage, wie beides zusammenhängen könnte. Die Beantwortung fiel den SchülerInnen sichtlich leicht, da bereits die erste gegebene Antwort korrekt war. Weiters folgte die Frage, ob wir nun endlich zu programmieren beginnen würden, da sie sich schon seit Schulbeginn darauf freuen würden. Besondere Begeisterung kam von Seiten der SchülerInnen auf, als diese Frage bejaht wurde und Scratch geöffnet werden durfte.

Die geplante Einführung gestaltete sich sehr produktiv, d.h. nach den einzelnen Teilen der Erklärung stellten die SchülerInnen sehr viele Fragen betreffend weiterführender Inhalte. Diese Begeisterung und Motivation Neues zu erlernen war großartig. Um diese nicht zu Beginn zu schmälern, wurde nicht bereits nach den geplanten 15 Minuten abgebrochen. Erst nach Klärung vieler zusätzlicher Fragen und weiteren 10 Minuten begannen die SchülerInnen mit dem Erarbeiten der Programmieraufgaben.

Eine besondere Schwierigkeit bei Aufgabe 1 war das Tanzen der Figuren. Die Überlegung, in welche Richtung und wieviele Schritte die Sprites jeweils gehen sollten damit die

Bewegungen wie tanzen aussieht, stellte sich für einzelne SchülerInnen als Herausforderung dar.

Auch großartige Leistungen wurden von den SchülerInnen vollbracht. So schaffte es beispielsweise ein Schüler, dass er eines seiner Sprites unter Verwendung von Schleifen in eine beliebige Richtung gleiten ließ. Wenn die Figur am Rand des Feldes angekommen war, so drehte sich diese (d.h. die Figur wurde gespiegelt) und bewegte sich in eine andere Richtung weiter.

Unabhängig davon, ob kleinere oder größere Fortschritte bei der Lösung der Aufgabe erzielt wurden, zeigten die SchülerInnen ihren KollegInnen gerne und oft unaufgefordert die neuesten.

Alle SchülerInnen schafften in der verbleibenden Zeit das Lösen der gestellten Aufgabe (ohne Bonus). Die besonders Schnellen erarbeiteten auch den Bonus. Mit der Ausarbeitung der Aufgabe 2 begann keine Schülerin/kein Schüler. Dafür war nicht mehr genug Zeit vorhanden.

Überprüfung des gelernten Wissens in der darauffolgenden Unterrichtssequenz

Den gemeinsam erarbeiteten Begriff „Algorithmus“ inkl. dessen zu erfüllenden Eigenschaften wurde in Form einer schriftlichen Stundenwiederholung abgeprüft. Um sicherzustellen, dass sich jeder an die Inhalte der letzten Unterrichtssequenz erinnern kann, wurde nach dem Absammeln der schriftlichen Arbeiten noch kurz mündlich die Lösung besprochen.

Das Wiederholen von den mittels Scratch erarbeiteten Inhalten erfolgte in mündlicher Form. Das bedeutete, jede/r SchülerIn erhielt eine Frage gestellt zu welcher sie/er mittels Lehrer-PC via Beamer die Lösung inkl. verbaler Erklärung präsentieren durfte.

Die gestellten Fragen bzw. Aufgaben waren in etwa folgende:

- Erkläre kurz den Aufbau der Oberfläche von Scratch.
- Wie kann ich ein zweites Sprite hinzufügen?
- Wo sehe ich, ob es mehrere Kostüme für ein Sprite gibt? Welche Kostüme gibt es beispielsweise für das soeben eingefügte Sprite?
- Wie kann ich zwischen den einzelnen Kostümen während dem Ablauf meines Programmes wechseln?

- Wie kann ich das Bühnenbild ändern? Kann ich auch eines selber zeichnen bzw. vom Internet einfügen? Wie würde das funktionieren?
- Welcher ist der erste Baustein, der eingefügt werden soll damit das Programm startet, wenn auf die grüne Fahne geklickt wird?
- Das Sprite möchte dich gerne nach deinem Namen fragen. Welchen Befehl musst du dafür verwenden?
- Wenn das Sprite nach Inhalte fragt und der User diese eingibt, wie können diese vom Sprite in einer Sprechblase wiedergegeben werden?
- Das Sprite möchte 10 Schritte nach rechts und anschließend 10 Schritte nach links gehen. Implementiere das bitte!
- Wenn diese Aktion nun nicht nur einmal sondern beispielsweise 10 Mal hintereinander ablaufen soll, wie kann ich das ohne großem zusätzlichen Programmieraufwand umsetzen?
- Wie kann ich sicherstellen, dass sich mein Sprite immer zu Beginn an der gleichen Stelle befindet?
- Wie kann ich Musik einbinden in mein Programm? Ab welchem Zeitpunkt ist dies sinnvoll?

4.3 Sequenz zum Thema „Algorithmen – Sortierverfahren – Teil 1“

- Lehrplanverortung: *Praktische Informatik*
  - Algorithmen erklären, entwerfen, darstellen und in einer Programmiersprache implementieren können
- Lehrziele:
  - Die SchülerInnen können einfache Algorithmen nachvollziehen und erklären.
  - Die SchülerInnen können die Umsetzung von Algorithmen mit einem Computer erklären.
  - Die SchülerInnen können einfache Algorithmen entwerfen, diese formal darstellen, implementieren und testen.
- Lehrbar in der Sekundarstufe 2 (5.-8. Klasse)
- Notwendige Vorkenntnisse von Seiten der SchülerInnen: Algorithmusbegriff, Computational Thinking, Scratch

- Pädagogische Patterns: Group Formation[27], Group Work[28], Presentation[29]

- Geplante Sequenz:

Geplante Dauer	*Inhalt*	*Arbeitsform*	*Konzepte und Methoden von CT*
ca. 15-20 min	Die SchülerInnen erhalten kleine, undurchsichtige Döschen mit verschieden schwerem Inhalt. *Aufgabe*: aufsteigende Reihung der Döschen anhand des Gewichts → Dafür soll ein Algorithmus entwickelt werden, der auf jedes einzelne Döschen anwendbar ist. →Der Vorgang der Entwicklung inkl. des fertigen Algorithmus soll im Anschluss daran notiert werden.	Team- bzw. Gruppen-arbeit (bis max. vier Personen)	Decomposition, Pattern Recognition, Abstraction, Algorithm
ca. 10-15 min	Präsentieren eines ausgewählten, erarbeiteten Algorithmus (oder max. zwei Algorithmen) →Dies erfolgt auf der Tafel mit Kärtchen, die Zahlen beinhalten. →Bezug nehmen auf andere Gruppen: Unterschiede in der Vorgangsweise	Präsentation eines Teams bzw. einer Gruppe	
ca. 10 min	Überleitung zu einem tatsächlich bestehenden Sortierverfahren →Hier wird auf Ähnlichkeiten zwischen dem präsentierten und dem bereits bestehenden Algorithmus hingewiesen. *Anmerkung*: Die SchülerInnen erarbeiten oft einen Algorithmus, der ähnlich ist zu Selection Sort bzw. Bubble Sort.	Vortrag durch Lehrenden	Pattern Recognition, Abstraction
ca. 15 min	Übungsblatt: Selection Sort bzw. Übungsblatt: Bubble Sort →Wenn ein/e SchülerIn besonders schnell ist, kann diese/r Überlegungen anstellen, wo Sortieralgorithmen im Alltag Verwendung finden.	Einzelarbeit	Pattern Recognition, Abstraction, Algorithm
ca. 5 min	Vergleich des Übungsblatts	Plenum	
ca. 10-15 min	Überlegung, wo Sortieralgorithmen Verwendung finden im Alltag Hinweis darauf, dass es nicht nur ein Sortierverfahren gibt	Partnerarbeit mit anschließender Besprechung im Plenum	Pattern Recognition, Abstraction

[27] Das Bilden der Gruppen wurde den SchülerInnen überlassen.
[28] Das Arbeiten in den Gruppen funktionierte hervorragend, da das Sortieren mit den unterschiedlich schweren Döschen eine nette Abwechslung zu den theoretischen Inhalten darstellte.
[29] Die anschließende Präsentation der erarbeiteten Ergebnisse erfolgte unter Verwendung von Kärtchen, die Zahlen beinhalteten. Mit Bezugnahme auf die gefundenen Lösungen wurde der darauffolgende Unterricht fortgeführt. D.h. bei der Erklärung des Bubble Sorts bzw. des Selection Sorts erwähnte die Lehrkraft die ähnlichen, von den SchülerInnen selbst erarbeiteten Sortierverfahren und zeigte die bestehenden Unterschiede auf.

	→Begründung, weshalb mehrere wichtig sind Vergleichsvideo (https://www.youtube.com/watch?v=kPRA0W1kECg) →Selection Sort: gleich zu Beginn →Bubble Sort: ab Minute 4:00	Vortrag durch Lehrenden	
ca. 15 min	Übungsblatt: Selection Sort bzw. Übungsblatt: Bubble Sort →Hier soll das noch nicht bearbeitete Übungsblatt von den SchülerInnen ausgefüllt werden.	Einzelarbeit	Decomposition, Pattern Recognition, Abstraction, Algorithm
ca. 5 min	Vergleich des Übungsblattes	Plenum	

Analyse und Reflexion der durchgeführten Sequenz

Um die Tauglichkeit dieser geplanten Sequenz für ältere SchülerInnen feststellen zu können, wurde diese im Wahlpflichtfach der sechsten Klasse durchgeführt.

Als Einstieg erhielt jede Gruppe á drei SchülerInnen kleine, undurchsichtige und unterschiedlich schwere Döschen. Diese sollten nach Gewicht und unter Anwendung eines selbst erarbeiteten Algorithmus aufsteigend sortiert werden. Um die Korrektheit der vorgenommenen Sortierung schnell und einfach überprüfen zu können, wurden auf jedes Döschen jeweils ein Buchstabe geklebt. Korrekt sortiert ergaben diese entweder den Begriff „Informatik“, „Algorithmus“ oder „Bubble Sort“. Auf diese Überprüfungsmöglichkeit wurde bei dem Erläutern des Arbeitsauftrages dezidiert hingewiesen. Auch erfolgte die Anweisung ausschließlich nach dem Gewicht zu sortieren. Daran hielten sich die SchülerInnen in etwa so lange bis sie die jeweiligen Wörter errieten. Um dies in Zukunft zu umgehen, aber trotzdem eine einfache Möglichkeit zur Überprüfung der gereihten Döschen zur Verfügung zu haben, könnte anstelle der Buchstaben eine bestimmte Farbreihenfolge aufgeklebt und diese zur Kontrolle verwendet werden.

Bei dem anfänglichen Sortieren nach dem Gewicht der Döschen entpuppte sich das Wiegen per Hand doch als sehr ungenau. Aus diesem Grund riet ich zwei der drei Gruppen dazu eine Linealwippe zu bauen und diese zu verwenden. Die dritte Gruppe verwendete eine Handy-App, die die Funktion einer Waage übernahm.

Das Erarbeiten eines Algorithmus stellte sich für eine Gruppe als kleine Herausforderung dar, da es den SchülerInnen nicht bewusst war, dass der entwickelte Algorithmus für jedes Döschen erneut durchgegangen werden sollte und sich hinter dem Begriff „Algorithmus“

u.a. ein logischer, sich innerhalb des Problems wiederholender und auf alle Döschen anwendbarer Handlungsablauf verbirgt. Im Anschluss auf diese Erklärung, war es auch dieser Gruppe möglich einen adäquaten Algorithmus für das Sortieren der Döschen zu entwickeln.

Eine Gruppe erklärte sich dazu bereit den entwickelten Algorithmus an der Tafel mit den dafür vorbereiteten Kärtchen vorzuführen. Dieser hatte große Ähnlichkeit zu dem bereits bestehenden Sortierverfahren „Selection Sort".

Das Gespräch mit den SchülerInnen ergab, dass eine andere Gruppe eine ähnliche Lösung zu dem Sortierverfahren „Bubble Sort" gefunden hatte: Es wurde in der Döschenmenge nach dem leichtesten Döschen gesucht und dieses an den Anfang der Menge gesetzt, d.h. alle anderen, welche vor dem Platz des entnommenen Döschen lagen, rückten um einen Platz weiter nach rechts. So wurde sichergestellt, dass keine Stelle leer blieb.

Die dritte Gruppe entschied sich dafür die erhaltenen Döschen in drei Gruppen einzuteilen: leichtere, mittelschwere und schwerere Döschen. Anschließend verglichen sie die Döschen in den jeweiligen Gewichtsklassen miteinander und legten diese in der korrekten Reihenfolge auf. Für diesen Schritt entwickelten die SchülerInnen keinen zu befolgenden Algorithmus, sondern führten ihn immer unterschiedlich aus. (*Anmerkung*: Dieser entwickelte Algorithmus beinhaltet Grundzüge des Sortierverfahrens „Quicksort".)

Um eine Überleitung zu einem tatsächlich bestehenden Sortierverfahren herzustellen, wurde nochmals auf das erarbeitete Ergebnis, welches große Ähnlichkeiten mit Bubble Sort hatte, hingewiesen und im Anschluss dessen die kleinen Unterschiede zwischen den beiden aufgezeigt. Die SchülerInnen der jeweiligen Gruppe waren erstaunt, dass es ihnen fast gelungen war ein bereits bestehendes Sortierverfahren zu erarbeiten. Daraufhin folgte am Übungsblatt das gemeinsame Festhalten der Funktionsweise von Bubble Sort. Dies zielte darauf ab, dass jede/r SchülerIn eine korrekte Beschreibung in ihren/seinen Unterlagen vorfindet und bei Bedarf diese nachlesen kann. Das Ausfüllen des zweiten Teils des Übungsblattes stellte keine Probleme dar und war sehr schnell erledigt. Zwei der SchülerInnen arbeiteten besonders zügig und konnten noch kurz gemeinsam Überlegungen anstellen, wo Sortierverfahren im Alltag sinnvoll erscheinen und möglicherweise Anwendung finden könnten.

Das Vergleichen des Lösungsweges der vorgegebenen Zahlen am Übungsblatt erfolgte mit den zuvor verwendeten Kärtchen auf der Tafel. Eine Schülerin meldete sich freiwillig und zeigte diesen fehlerfrei vor.

Anschließend wurden die SchülerInnen dazu aufgefordert fünf Minuten lange zu zweit Überlegungen darüber anzustellen, in welchen Bereichen Sortierverfahren benötigt werden und auch Beispiele für deren Anwendung im Alltag zu finden. Diese Aufgabe fiel den meisten schwerer als gedacht. Ein Schüler erinnerte sich an das vorhergehende Jahr, wo im Unterricht Excel-Tabellen sortiert wurden, um ein schnelleres Finden der Daten für den Menschen zu ermöglichen. Weitere Beispiele bzgl. der Listenerzeugung (für menschliche Leser) kamen erst nach einigen Hilfestellungen:

- Wenn ihr euch in Deutsch bei einer Schularbeit nicht sicher seid wie ein Wort geschrieben wird, wo seht ihr dann nach?
- Wenn ihr jemanden anrufen möchtet und dessen Telefonnummer nicht auswendig wisst, wo findet ihr diese? Erfolgt in diesem Fall eine Sortierung?
- Klassenlisten sind auch geordnet. Weshalb meint ihr ist das so?
- Wenn ihr wissen möchtet, welche Apps bzw. Hintergrundprozesse auf eurem Computer derzeit den meisten Arbeitsspeicher benötigen, müsst ihr da die gesamte Liste händisch durchsuchen?

Auf den Gedanken, dass Sortierverfahren auch hilfreich bei einer Duplikatermittlung sind oder für das Rendern eines 3D Objektes unterstützend fungieren, kamen die SchülerInnen nicht. Jedoch begeisterte sie letzterer Punkt sehr.

Um den SchülerInnen zu verdeutlichen, dass diese Beispiele nicht alle nur mittels Bubble Sort gelöst werden können, sondern auch andere Sortierverfahren dafür notwendig sind, wurde nochmals auf die selbstständig erarbeiteten Algorithmen zu Beginn der Sequenz hingewiesen. Diese unterschieden sich in ihren Grundzügen stark voneinander. Zur Verdeutlichung dieser Verschiedenheit und auch der variierenden Laufzeit der existierenden Sortierverfahren kamen zwei kurze Ausschnitte des vorbereiteten YouTube-Videos zum Einsatz: Bubble Sort und Selection Sort.

Die Reaktion der SchülerInnen darauf fielen unterschiedlich aus: Ein paar meinten, dass diese Sortierverfahren doch sehr langsam seien und fragten, ob es da auch noch etwas Schnelleres gäbe. Andere überlegten wie viele Sortierverfahren existieren würden. Und

wieder andere fragten, ob es spezielle Anwendungsbereiche für die unterschiedlichen Sortierverfahren gäbe.

Im Anschluss an die Klärung dieser Fragen wurde gemeinsam die Funktionsweise des Selection Sort auf der Tafel erarbeitet. Auf Grund des YouTube-Videos schlossen die SchülerInnen, dass beim Selection Sort aus der vorhandenen Menge an Zahlen die kleinste gesucht und an die erste Stelle geordnet wird. Danach wird die zweit kleinste Zahl gesucht und an die Stelle rechts daneben gegeben. Was allerdings mit den Zahlen passiert, die sich an der jeweiligen Stelle befinden, wo die jeweilig kleinste Zahl eingefügt wird, konnten die SchülerInnen auch nach mehrmaligem wiederholen des Videos nicht daraus ablesen. Jedoch stellten sie zwei Vermutungen auf: Es könnte sein, dass, so wie in dem bereits zu Beginn erarbeiteten Algorithmus, alle Zahlen um eine Stelle nach rechts verschoben werden um Platz für die kleinste unsortiertste Zahl zu schaffen. Die zweite und korrekte Annahme war, dass die kleinste unsortiertste Zahl mit der Zahl an der ersten Stelle der unsortierten Menge der Zahlen vertauscht wird.

Das Sortierverfahren „Selection Sort“ wurde von einem Schüler auf der Tafel für eine Zahlenreihe durchgespielt und hinterher die Funktionsweise auf dem zweiten Übungsblatt gemeinsam notiert.

Da die Unterrichtssequenz bereits zu Ende ging, bekamen die SchülerInnen die Aufgabe das Übungsblatt bis zum nächsten Mal zu Hause fertig zu stellen.

Überprüfung des gelernten Wissens in der darauffolgenden Unterrichtssequenz

Zu dem Thema „Algorithmen – Sortierverfahren“ wurden zwei Sequenzen geplant. Die Beschreibung der Überprüfung des von den SchülerInnen angeeigneten Wissens erfolgt deshalb zu Beginn des zweiten Teils.

4.4 Sequenz zum Thema „Algorithmen – Sortierverfahren – Teil 2“

- Lehrplanverortung: *Praktische Informatik*
  - Algorithmen erklären, entwerfen, darstellen und in einer Programmiersprache implementieren können
- Lehrziele:
  - Die SchülerInnen können einfache Algorithmen nachvollziehen und erklären.
  - Die SchülerInnen können die Umsetzung von Algorithmen mit einem Computer erklären.
  - Die SchülerInnen können einfache Algorithmen entwerfen, diese formal darstellen, implementieren und testen.

- Lehrbar in der Sekundarstufe 2 (5.-8. Klasse)
- Notwendige Vorkenntnisse von Seiten der SchülerInnen: Algorithmusbegriff, Computational Thinking, Scratch, Sortierverfahren „Bubble Sort“/“Selection Sort“ bzw. Sequenz zum Thema „Algorithmen – Sortierverfahren - Teil 1“
- Pädagogische Patterns: Individual Work[30]
- Geplante Sequenz:

Geplante Dauer	*Inhalt*	*Arbeitsform*	*Konzepte und Methoden von CT*
ca. 5 min	Schriftliche Wiederholung: *Bubble Sort*	Einzelarbeit	
ca. 15 min	Mündliche Wiederholung: *Selection Sort* →Funktionsweise →Vergleich der Hausübung →Sortieren der Zahlen aus der schriftlichen Wiederholung	Besprechung/ Erarbeitung im Plenum	Decomposition, Pattern Recognition, Abstraction, Algorithm
ca. 30 min	Notieren eines Pseudocodes für Bubble Sort auf der Tafel. →Ziel ist es, den Algorithmus mit Scratch zu implementieren.	Erarbeitung im Plenum	Decomposition, Pattern Recognition, Abstraction, Algorithm
ca. 45 – 50 min	Implementieren des Sortierverfahrens mit Scratch	Einzelarbeit	Decomposition, Pattern Recognition, Abstraction, Algorithm

Analyse und Reflexion der durchgeführten Sequenz

Zu Beginn der Unterrichtssequenz, welche erneut in einer sechsten Klasse durchgeführt wurde und an die vorhergehende geplante Unterrichtssequenz anknüpfte (Siehe Kapitel 4.3 Sequenz zum Thema „Algorithmen – Sortierverfahren – Teil 1“.), erfolgte eine kurze schriftliche Stundenwiederholung. Aufgabe war es lediglich das Sortierverfahren „Bubble Sort“ korrekt auf eine vorgegebene Zahlenreihe anzuwenden. Dies gelang jedoch nicht allen SchülerInnen, da einige den gewünschten Algorithmus mit jenem von der Hausübung, dem Selection Sort, verwechselten. Auf Grund dessen wurde der Bubble Sort nochmals gemeinsam auf der Tafel wiederholt. Dies stellte sicher, dass sich nun wieder alle SchülerInnen an die Funktionsweise erinnern und das Sortierverfahren korrekt anwenden konnten.

[30] Gemeinsam mit der Lehrkraft erarbeiteten die SchülerInnen den Pseudocode, welchen sie anschließend selbständig implementierten. Dafür wurde Wissen bzgl. Scratch vorausgesetzt.

Im Anschluss daran erfolgte auch die mündliche Wiederholung der Funktionsweise des Selection Sorts und das gemeinsame Vergleichen der Hausübung. Dies fiel den SchülerInnen Großteils sichtlich leichter, da sich die meisten zu Hause mit der gestellten Aufgabe auseinandergesetzt hatten und diese somit an der Tafel präsentieren konnten. Jene beiden, die auf das Machen der Hausübung vergessen bzw. verzichtet hatten, konnten so von dem Wissen ihrer MitschülerInnen profitieren und auch das Übungsblatt fertig ausfüllen. Um die Vorgangsweise des Sortierens beim Selection Sort noch ein wenig zu festigen, durften die SchülerInnen die Zahlen aus der schriftlichen Wiederholung hernehmen und diese auch anhand des Selction Sort sortieren. Diese Aufgabe wurde im Anschluss abgesammelt und korrigiert, da die Zeit schon etwas weiter fortgeschritten war, als geplant. Zudem wurde dadurch auch klarer ersichtlich, wer das dahinterstehende System tatsächlich durchschaut hatte und welche/r SchülerIn noch Hilfe benötigte.

Der darauffolgende Punkt „Gemeinsames Erarbeiten eines Pseudocodes für Bubble Sort auf der Tafel“ erforderte nun abermals das Wissen bzgl. der Funktionsweise des Bubble Sorts. Dieses erneute Wechseln zwischen den beiden Sortierverfahren verwirrte die SchülerInnen etwas.

Eine bessere Abfolge der einzelnen geplanten Teile wäre in diesem Fall folgende gewesen:

Geplante Dauer	*Inhalt*	*Arbeitsform*	*Konzepte und Methoden von CT*
ca. 15 min	Mündliche Wiederholung: *Selection Sort* →Funktionsweise →Vergleich der Hausübung →Sortieren der Zahlen aus der schriftlichen Wiederholung	Besprechung/ Erarbeitung im Plenum	Decomposition, Pattern Recognition, Abstraction, Algorithm
ca. 5 min	Schriftliche Wiederholung: *Bubble Sort*	Einzelarbeit	
…	…	…	…

Um der Verwirrung der SchülerInnen entgegenzuwirken, wurde nochmals kurz ein Durchlauf für eine Reihe von Zahlen auf der Tafel durchgespielt. Dies reichte aus, damit sich alle an die Funktionsweise von Bubble Sort erinnerten.

Das Erarbeiten des Pseudocodes selbst stellte sich als sehr langwierig und herausfordernd heraus. Es bedurfte sehr viel Geduld und Unterstützung bis dieser korrekt an der Tafel stand. Dabei waren die zuvor notierten Gedanken inkl. möglicher Fragestellungen eine große Hilfestellung. Auch der vorher festgehaltene Pseudocode war eine tolle Stütze. Für die SchülerInnen besonders schwer nachzuvollziehen waren die beiden verschachtelten

Schleifen. Erst nach einer langen Erklärung inkl. einem erneuten Durchbesprechen der Funktionsweise und dem wiederholten grafischen Darstellen der Abläufe innerhalb des Sortierverfahrens konnten die SchülerInnen nachvollziehen weshalb diese erforderlich sind.

Die Motivation und Ausdauer, die die SchülerInnen dabei an den Tag legten, war erstaunlich. Immer wieder hinterfragten diese die notierten Inhalte um die Hintergründe des gemeinsam erarbeiteten Algorithmus besser zu verstehen und nachvollziehen zu können. Dadurch entstand eine richtig angenehme Arbeitsatmosphäre, die alle Beteiligten sichtlich antrieb ihr Bestes zu geben. Die zuvor befürchtete Langeweile, weil das Erarbeiten des Pseudocodes doch sehr viel Zeit in Anspruch nahm, kam trotz der fast 40 Minuten benötigten Zeit nicht auf.

Da das Erarbeiten des Pseudocodes auch von den SchülerInnen als sehr anstrengend und herausfordernd empfunden wurde, äußerten diese den Wunsch nach frischer Luft und einigen Auflockerungsübungen. Dieser Bitte wurde gerne nachgekommen.

Das anschließende Implementieren des Sortierverfahren „Bubble Sort“ mittels Scratch stellte für die SchülerInnen keine großen Schwierigkeiten mehr dar. Das Übertragen des zu programmierenden Codes von der Tafel erschien fast trivial im Vergleich zur Erarbeitung dessen. Lediglich ein Schüler benötigte eine kurze Erklärung wie Variablen mit Scratch erstellt werden können. Weiters kamen die Fragen auf, wie viele Zahlen in der zu sortierenden Menge vorhanden sein sollten und in welchem Bereich die Zufallszahlen in etwa liegen müssen.

Die etwas schnelleren SchülerInnen schafften die Implementierung eines funktionsfähigen Bubble Sorts bis zum Ende der Unterrichtssequenz. Den anderen fehlte meist nur mehr der Vorgang „Vertauschen der Elemente“. Damit in der darauffolgenden Doppeleinheit alle SchülerInnen am gleichen Stand sein würden, war das Fertigstellen des implementierten Sortierverfahrens Hausübung.

Überprüfung des gelernten Wissens in der darauffolgenden Unterrichtssequenz

Es erfolgte eine mündliche Stundenwiederholung. Hauptinhalt dieser war das Sortierverfahren „Bubble Sort“, der erarbeitete Pseudocode und der implementierte Code. Gemeinsam wurde nochmals der Algorithmus besprochen und im Anschluss daran Zeile

für Zeile des Pseudocodes begründet, weshalb diese jeweils für den funktionsfähigen Algorithmus notwendig ist.

Die gestellten Fragen bzgl. der Funktionsweise konnten von den SchülerInnen problemfrei beantwortet werden. Jene betreffend des Pseudocodes stellten sich teilwiese als etwas herausfordernder dar: Nur in etwa zwei Drittel der SchülerInnen zeigten bei der Frage auf, weshalb zwei Schleifen für einen funktionsfähigen Algorithmus notwendig sind und welche Aufgaben jede einzelne von ihnen übernimmt. Bei der gewünschten Erklärung, weshalb eine dritte Variable bei dem Vertauschen der Elemente notwendig sei, meldeten sich hingegen alle SchülerInnen.

4.5 Sequenz zum Thema „Hardware – Teil 1“

- Lehrplanverortung: *Informatiksysteme*
  - Den Aufbau von digitalen Endgeräten beschreiben und erklären können.
- Lehrziele:
  - Die SchülerInnen können ein Computersystem samt Peripheriegeräten sachgerecht nutzen.
  - Die SchülerInnen können die Komponenten eines Computersystems nennen und die Aufgaben dieser beschreiben.
- Lehrbar in der Sekundarstufe 1 (4. Klasse) und in der Sekundarstufe 2 (5.-8. Klasse)
- Notwendige Vorkenntnisse von Seiten der SchülerInnen: Keine
- Pädagogische Patterns: Individual Work[31], Presentation[32]
- Geplante Sequenz:

Geplante Dauer	*Inhalt*	*Arbeitsform*	*Konzepte und Methoden von CT*
ca. 15-20 min	Die SchülerInnen erhalten jeweils ein Bauteil eines Rechners. Zu diesem sollen sie folgende Informationen in einem Word-Dokument festhalten: • Bezeichnung/Name des Teils • Aufgaben in einem Computer • Baujahr • Preis • Bei Speichermedien: Speichervolumen →Abgabe der erarbeiteten Inhalte	Einzelarbeit	Decomposition, Pattern Recognition, Abstraction

[31] Die SchülerInnen waren dazu angehalten, selbständig Informationen über die jeweils zugeteilte Hardwarekomponente herauszufinden. Welche genau erforderlich waren, wurden auf der Tafel festgehalten.
[32] Die gesammelten Inhalte des individuellen Arbeitsprozesses wurden im Anschluss daran vor der gesamten Gruppe präsentiert. Dadurch erhielten alle SchülerInnen einen Überblick über die Hardware eines Rechners.

ca. 35 min	Präsentation der recherchierten Inhalte vor der Klasse	Präsentation	
ca. 10-15 min	Klären der Frage „Wie gehören die Bestandteile korrekt zusammengefügt?“ →gemeinsame Besprechung anhand eines geöffneten Rechners →Handout	Besprechung im Plenum	Decomposition, Pattern Recognition
ca. 10 min	Anschlüsse eines Rechners →Ausfüllen eines Arbeitsblattes →Vergleich und Besprechung im Plenum	Teamarbeit; Besprechung im Plenum	Pattern Recognition
ca. 15-20 min	Rechner vs. Laptop vs. Smartphone: →gemeinsame Überlegungen, welche Unterschiede zwischen den Hardwarekomponenten sein könnten →Erklärung mit Hilfe von geöffneten Geräten	Besprechung im Plenum	Pattern Recognition, Abstraction

Analyse und Reflexion der durchgeführten Sequenz

Diese vorbereitete Sequenz wurde in einer reinen Mädchen-, in einer reinen Buben- und in einer gemischten Gruppe in der 5. Klasse gehalten.

Nach der zu Beginn gestellten Frage, wer sich denn schon näher mit der Hardware eines Computers auseinandergesetzt hatte, wurden die einzelnen Komponenten bzgl. ihres Schwierigkeitsgrades an die SchülerInnen verteilt. (Der Schwierigkeitsgrad hing von dem Alter der betreffenden Teile ab, d.h. je älter, desto unwahrscheinlicher ist bei den SchülerInnen Wissen darüber vorhanden. Beispiele dafür sind u.a. das Diskettenlaufwerk oder die Netzwerkkarte.)

Einige Mädchen aus der gemischten Gruppe konnten mit der gestellten Aufgabe nur sehr wenig anfangen. Sie waren der Meinung, dass Wissen über die einzelnen Hardwarekomponenten völlig nutzlos und sinnfrei sei, da die Geräte auch zusammengebaut zu erwerben seien und es für eine eventuell notwendige Reparatur Spezialisten gäbe. Baff von dieser Aussage und der daraus zu schließenden Bildungsresistenz folgte keine Zurechtweisung und Begründung, weshalb dies dennoch wissenswert sei. Meine Diplomarbeitsbetreuerin erläuterte im darauf anschließenden Gespräch eine mögliche Reaktion auf diese Aussage der Schülerinnen: Wissen über das Gerät, mit welchem gearbeitet wird, ist nicht nur in der Informatik wichtig. Beispielsweise bei dem Ablegen der praktischen Führerscheinprüfung ist auch das Kennen des Autos ein wichtiger Bestandteil um auf mögliche, zukünftige Schwierigkeiten oder Probleme besser vorbereitet zu sein.

Die Ausarbeitungen der übrigen SchülerInnen dieser Gruppe waren Großteils in Ordnung, aber wurden überwiegend knapp gehalten. Im Gegensatz dazu arbeitete die reine Buben-Gruppe besonders genau und detailliert. Neben einer hervorragenden Beschreibung der zu erfüllenden Aufgaben der jeweiligen Hardwarekomponenten, fanden ausnahmslos alle die genaue Bezeichnung (z.B. Intel Core i5-8600K Prozessor) und den Kaufpreis für das betreffende Teil heraus. Auch für Komponenten, die auf Grund des Alters nicht mehr erworben werden können, recherchierten die Schüler, wieviel dieses früher im Einkauf gekostet hatte. Bei der darauf anschließenden Präsentation erklärte ein Schüler sogar fehlerfrei die Funktionsweise eines DVD-Laufwerks und den Speichervorgang auf einer DVD.

Die reine Mädchen-Gruppe recherchierte zwar nicht so ausführlich, stellte sich jedoch auch sehr gut an. Nach einigen wenigen Hilfestellungen war es allen möglich die erforderlichen Informationen über die einzelnen Hardwarekomponenten herauszufinden um im Anschluss daran diese präsentieren zu können.

Nachdem die jeweiligen Aufgaben der Komponenten geklärt waren, stellte sich besonders in der gemischten Gruppe die Frage wie diese korrekt zusammengefügt gehörten. Für die beiden anderen Gruppen erschien dies oft nur eine gute Wiederholung des bereits Gewussten zu sein.

Die Erklärung der korrekten Zusammensetzung erfolgte anhand eines geöffneten Rechners um diese so realitätsbezogen wie möglich zu gestalten. Außerdem wurden dabei auch gleich die soeben präsentierten Inhalte abgefragt. Dies passierte wie folgt: Nach dem Zeigen auf eine Komponente wurde nach der allgemeinen Bezeichnung und den betreffenden Aufgaben gefragt. Damit auch der Aufbau eines Computers im Heft festgehalten wurde, erfolgte das alleinige Ausfüllen des Handouts mit dem darauffolgenden Vergleich dessen.

Das Aufgabenblatt für die Anschlüsse eines Rechners durften die SchülerInnen ohne vorhergehenden Input ausfüllen. Von den beiden abgebildeten Anschlüssen PS/2 Keyboard und PS/2 Maus waren den meisten SchülerInnen die Bezeichnungen nicht bekannt. Auch die Unterscheidung zwischen USB 2.0 und USB 3.0 konnten einige nicht vornehmen. Eine besonders amüsante Beschreibung des Netzwerkanschlusses lieferte eine Schülerin: Diese meinte, das sei der Anschluss, wo das Internet angesteckt werden könne.

Neben den Bezeichnungen der einzelnen Anschlüsse wurde auch besprochen, welche Peripheriegeräte an diese angeschlossen werden können. Auch eine Erklärung des Begriffes „Peripheriegeräte“ erfolgte.

Besonders für die technikbegeisterte Buben-Gruppe entpuppte sich die nachfolgende Gegenüberstellung von Rechner, Laptop und Smartphone bzgl. der jeweiligen Hardwarekomponenten als sehr spannend. Bei der gemeinsamen Erarbeitung im Plenum kamen von Seiten der Schüler sehr viele Ideen und Vermutungen, die überwiegend zutrafen. Bei der Mädchen-Gruppe und der gemischten Gruppe kam nur wenig Begeisterung für solcherlei Details auf. Nach knappen fünf Minuten wurde die Besprechung beendet und in der verbleibenden Zeit nochmals anhand eines Rechners die Bezeichnungen der vorhandenen Anschlüsse, der Peripheriegeräte, die jeweils angeschlossen werden können und die Begriffsdefinition der Peripheriegeräte wiederholt.

Überprüfung des gelernten Wissens in der darauffolgenden Unterrichtssequenz

Da eine darauf anschließende Unterrichtssequenz geplant und diese (teilweise in der Mädchen-Gruppe und der gemischten Gruppe) bzw. vollständig (in der Buben-Gruppe) durchgeführt wurde, erfolgt die Beschreibung und Reflexion der Stundenwiederholung zu Beginn dieser. (Siehe Kapitel 4.6 Sequenz zum Thema „Hardware – Teil 2“.)

4.6 Sequenz zum Thema „Hardware – Teil 2“

- Lehrplanverortung: *Informatiksysteme*
  - Den Aufbau von digitalen Endgeräten beschreiben und erklären können.

- Lehrziele:
  - Die SchülerInnen können ein Computersystem samt Peripheriegeräten sachgerecht nutzen.
  - Die SchülerInnen können die Komponenten eines Computersystems nennen und die Aufgaben dieser beschreiben.
  - Die SchülerInnen können die Komponenten eines Computersystems so zusammenfügen, dass dieses im Anschluss funktionsbereit ist.
  - Die SchülerInnen können einfache Fehler diagnostizieren und beheben.

- Lehrbar in der Sekundarstufe 2 (5.-8. Klasse)

- Notwendige Vorkenntnisse von Seiten der SchülerInnen: Sequenz zum Thema „Hardware – Teil 1“

- Pädagogische Patterns: Teacher Check[33], Paired Work[34]
- Geplante Sequenz:

Geplante Dauer	*Inhalt*	*Arbeitsform*	*Konzepte und Methoden von CT*
ca. 10 min	Wiederholung des Gelernten mittels Kahoot!	Überprüfung des Gelernten	
ca. 2-3 min	Die SchülerInnen erhalten ein Handout mit den gesammelten Inhalten der Recherche-Aufgabe der einzelnen Hardwarekomponenten aus der vorhergehenden Sequenz. Dieses wirkt unterstützend für die darauffolgende Aufgabe.		
ca. 80 min	Die Gruppen sollen aus den vorhandenen Teilen jeweils einen funktionsfähigen Rechner zusammenbauen. →Davor müssen die Gruppen für sich einen Ablauf festlegen, den sie dabei befolgen. →Die Lehrkraft fungiert bei dieser Aufgabe unterstützend.	Teamarbeit	Decomposition, Abstraction, Algorithm

Analyse und Reflexion der durchgeführten Sequenz

Der Beginn in der gemischten Gruppe bestand daraus, besonders den Schülerinnen, die in der letzten Sequenz gemeint hatten, dass das Wissen über Hardware für sie nicht wissenswert genug bzw. notwendig sei, zu erklären weshalb dies doch der Fall wäre. Eine Schülerin, die diesen Zwischenfall letztes Mal anscheinend nicht mitbekommen hatte, schaltete sich ein und meinte, dass wir bereits in der ersten Einheit besprochen hatten, in welchen Berufen ein bestimmtes Grundwissen der Informatik verlangt wird bzw. notwendig ist. Sie erinnerte zudem mit Nachdruck an das gemeinsam erarbeitete Ergebnis: Wenn auch, was heutzutage schon sehr selten ist, im Beruf kein Bezug zur Informatik (im weitesten Sinn) vorhanden ist, so findet dieser zumindest im Privatleben statt.

Diese doch sehr energische Äußerung erstaunte nicht nur die restlichen SchülerInnen. Weiters schien sie in der Gruppe einen bleibenden Eindruck hinterlassen zu haben, da viele

[33] Mittels Kahoot! wurde das Erlernte Wissen der SchülerInnen überprüft. Auch eine gemeinsame Analyse, weshalb die übrigen Antworten nicht stimmen konnten, wurde durchgeführt. Ein positiver Aspekt dieser Wiederholung war das sofortige Erhalten der erzielten Resultate.

[34] Das Zusammensetzen der Rechner passierte in Teams. Dies wäre in etwas größeren Gruppen nicht möglich gewesen, da hierbei das eigenständige Arbeiten am Rechner mitunter als Ziel definiert wurde. Auch ein individuelles Arbeiten hätte sich als nicht sinnvoll herausgestellt. Dies hätte den sich gegenseitig beeinflussenden Lernprozess der SchülerInnen nicht in diesem Ausmaß gefördert.

sehr nachdenklich wirkten und sich augenscheinlich die allererste Informatikeinheit in Erinnerung riefen. Ein Schüler fügte hinzu, dass wir auch die rasche Zunahme der Verwendung der Informatik in den letzten 60 Jahren besprochen hatten. Und wenn sich diese so weiter entwickeln würde, wir alle noch sehr viel in diesem Bereich dazu lernen müssten.

Nach diesem doch sehr emotional geladenen Gespräch wurde mit der Stundenwiederholung fortgefahren. Die beiden anderen Gruppen begannen mit dieser den Informatikunterricht.

Die Durchführung erfolgte unter Verwendung der Homepage kahoot.it. Dies ist eine spielbasierte Lernplattform, welche es Lehrenden ermöglicht selbst digitale Fragebögen zu ausgewählten Themenbereichen zu erstellen. Zu jeder gestellten Frage können zwei bis vier Antwortmöglichkeiten angegeben und von den SchülerInnen im Spielverlauf jeweils eine gewählt werden. Für jede korrekte Antwort erhalten diese Punkte. Auf Basis dessen steht am Ende des durchgeführten Kahoots ein Gewinner fest.

Gruppenunabhängig erweckte diese Art der Stundenwiederholung bei den SchülerInnen einen regelrechten Wettkampfmodus mit Wissen und Geschwindigkeit als Mittelpunkt. Nach jeder absolvierten Frage folgten erstaunte, begeisterte und enttäuschte Zwischenrufe. Um den Inhalt der gestellten Fragen dennoch im Fokus zu behalten, wurde nach dem Geben der Antwort kurz alle Antwortmöglichkeiten durchbesprochen und gemeinsam begründet, weshalb diese korrekt bzw. falsch waren.

Eine der Gruppen war so davon begeistert, dass sie dieses Kahoot! unbedingt noch ein zweites Mal durchspielen wollten. Begründet wurde dies damit, dass sich einige bei manchen Antworten verklickt hatten und nun fest der Überzeugung waren, bei einem erneuten Durchgang alle Fragen richtig beantworten zu können. Diese Aussage bestätigte sich definitiv nicht.

In der vorhergehenden Unterrichtssequenz mussten die SchülerInnen ihre erarbeiteten Inhalte bzgl. der Hardwarekomponenten abgeben. Diese erhielten sie im Anschluss an die durchgeführte Stundenwiederholung in gesammelter und korrigierter Form als Handout.

Da für den letzten geplanten Punkt sehr viel Disziplin, Begeisterung und Motivation von Seiten der SchülerInnen benötigt wurde und die Anzahl der zur Verfügung stehenden Komponenten begrenzt war, durfte bzw. konnte nur eine der drei Gruppen diesen durchführen: die Buben-Gruppe. Die anderen beiden Gruppen begannen mit der

Erarbeitung der softwarebezogenen Funktionsweise eines Computers. (Da eine Beschreibung dessen den hier vorgegebenen Rahmen sprengen würde, musste darauf leider verzichtet werden.)

Die Buben-Gruppe, die die Aufgabe erhielten in Teamarbeit jeweils einen funktionsfähigen Rechner zusammen zu bauen, bemühte sich sehr. Zunächst wurde nochmals gemeinsam in Erinnerung gerufen welche Hardwarekomponenten unbedingt notwendig seien für den erfolgreichen Bau eines Rechners. Weiters erfolgte der Hinweis, dass die erhaltenen Handouts sehr wohl verwendet und bei auftretenden Fragen die Mitschüler oder die Lehrkraft als Unterstützung hinzugezogen werden sollten. Auch wurde ein schriftlich festgehaltener Ablaufplan vor Beginn des Zusammenbauens gefordert. Erst nach Fertigstellung und Durchsicht durch die Lehrkraft durfte begonnen werden.

Begeistert fingen die Schüler mit der gestellten Aufgabe an. Nach bereits knappen zehn Minuten war das erste Team mit der Ablaufbeschreibung fertig und starteten mit dem Zusammensuchen und -bauen der einzelnen Hardwarekomponenten. Ein Team bemühte sich besonders bei der Erstellung des Plans und wurde nach ca. 20 Minuten dabei unterbrochen. Nach einem Hinweis auf die bereits benötigte Zeit dafür, entgegneten die beiden Schüler, dass sie alles so gut und detailliert wie möglich beschreiben möchten um im Anschluss keinen Fehler beim Zusammenbau zu machen. Nach weiteren fünf Minuten waren auch sie fertig mit der geforderten Ablaufbeschreibung.

Das Zusammenfügen bei den anderen Teams lief derzeit schon auf Hochtouren. Alle hatten die notwendigen Komponenten geholt und neben sich hingelegt. Fleißig wurde bereits das erste Teil in das Gehäuse geschraubt. Viele der Schüler begannen mit dem Mainboard oder dem Netzteil. Auch das korrekte Montieren der restlichen Hardwarekomponenten verlief ohne Fragen von Seiten der Schüler. Erst als die Stecker für die Stromverteilung überall angesteckt werden musste, wurde Unterstützung benötigt. Nach einer kurzen Umfrage, ob bei diesem Schritt voraussichtlich alle Teams Hilfe brauchen würden und dies von fast allen bejaht wurde, erfolgte anhand eines bereits fertig zusammengesetzten Rechners die Erklärung für alle Schüler. Dennoch gab es darauf anschließend beim selbstständigen Anstecken noch Probleme und Bedarf an Unterstützung, weil manche beispielsweise vergaßen die Festplatte oder andere Komponenten anzustecken.

Bei einem erneuten Durchführen dieser Unterrichtssequenz wäre in diesem Fall ein anderes Vorgehen sehr viel einfacher und Zeitsparender: Nachdem alle Teile korrekt im Gehäuse

angebracht wurden, sollten die etwas schnelleren Teams den etwas Langsameren unterstützend zur Seite stehen oder zu einem vorgegebenen Thema Recherchen anstellen, z.B. Aufsetzen eines Computers oder Betriebssysteme. Wenn wieder alle auf dem gleichen Arbeitsstand sind, kann von der Lehrkraft vorgezeigt werden wo jeweils ein spezielles Kabel angesteckt werden soll. Damit wird sichergestellt, dass die SchülerInnen diese Aufgabe überwiegend korrekt ausführen und im Anschluss daran funktionierende Rechner als Endprodukte erhalten. Weiters kann dadurch ein gleichzeitiges Fertigwerden garantiert werden.

Trotz dieser zuvor beschriebenen Problematik, waren die Schüler die gesamte Arbeitszeit über begeistert bei der Sache. Gerne wurde auch den Mitschülern geholfen, wenn diese Unterstützung bei der Verlegung der Kabel benötigten. Diese Aufgabe förderte sichtlich den Zusammenhalt innerhalb der Gruppe und brachte ein sehr angenehmes Arbeitsklima mit sich.

Da nach dem Zusammensetzen der Rechner und dem Aufräumen der Arbeitsplätze noch in etwa zehn Minuten Zeit vorhanden waren, wurde im Plenum noch über die soeben gesammelten Eindrücke jedes Schülers gesprochen. Auch ein Vergleich zwischen den zuvor gehabten Vorstellungen und der darauffolgenden Realität wurde gezogen, da viele der Schüler zum ersten Mal einen Rechner zusammenbauten.

Im weiteren Verlauf des Schuljahres wird das Wahlpflichtfach der 7. Klasse die zusammengebauten Rechner aufsetzen und ausprobieren dürfen.

Überprüfung des gelernten Wissens in der darauffolgenden Unterrichtssequenz

Da in dieser Unterrichtssequenz die Anwendung des zuvor vermittelten Wissens im Fokus lag und daraus klar ersichtlich wurde, wer von den Schülern die Inhalte gelernt und verstanden hatte, erfolgte in der darauffolgenden Doppeleinheit keine tiefergehende, fachliche Stundenwiederholung, sondern lediglich ein kurzes in Erinnerung rufen des Inhaltes der Sequenz.

4.7 Sequenz zum Thema „Datenbanken“

- Lehrplanverortung: *Praktische Informatik*
  - Datenbanken benutzen und einfache Datenmodelle entwerfen können

- Lehrziele:
  - Die SchülerInnen können Datenbankmodelle, Tabellen und ihre Beziehungsmuster sowie weitere Datenbankobjekte erklären.
  - Die SchülerInnen können strukturiert (in Tabellen) erfassen, abfragen und auswerten.
- Lehrbar in der Sekundarstufe 2 (5.-8. Klasse)
- Notwendige Vorkenntnisse von Seiten der SchülerInnen: Theorie zu ER-Modellen inkl. Chen-Notation, relationales Datenbankschema
- Pädagogische Patterns: Teacher Check[35], Paired Work[36]
- Geplante Sequenz:

Geplante Dauer	*Inhalt*	*Arbeitsform*	*Konzepte und Methoden von CT*
ca. 10 min	Wiederholung der Theorie zu ER-Modellen inkl. Chen-Notation mittels Kahoot!	Überprüfung des Gelernten	
ca. 20 min	Gemeinsames Erarbeiten eines ER-Modells inkl. Chen-Notation	Erarbeitung im Plenum	Decomposition, Pattern Recognition, Abstraction, Algorithm
ca. 15 - 20 min	Überführen des ER-Modells in die 3. Normalform →Das relationale Datenbankschema fungiert dabei unterstützend.	Erarbeitung im Plenum	Decomposition, Pattern Recognition, Abstraction, Algorithm
ca. 10 min	Überführen des relationalen Datenbankschemas in Tabellen und Einfügen von möglichen Datensätzen	Erarbeitung im Plenum	Pattern Recognition, Algorithm
ca. 30 min	Selbstständiges Erarbeiten einer Aufgabe →ER-Diagramm →3. Normalform →Relationales Datenbankschema Abgabe der erarbeiteten Ergebnisse	Teamarbeit	Decomposition, Pattern Recognition, Abstraction, Algorithm

Analyse und Reflexion der durchgeführten Sequenz

Die Durchführung im Wahlpflichtfach der 7. Klasse erfolgte erneut mit der Motivation, die geplante Sequenz auf ihre Funktionsfähigkeit in höheren Schulstufen zu überprüfen.

[35] Die durchgeführte Stundenwiederholung mittels Kahoot! hatte zum Vorteil, dass das von den SchülerInnen erzielte Resultat sofort nach Beendigung dieser abrufbar ist.
[36] Gemeinsam, d.h. in Teams zu zweit, sollten die SchülerInnen einen Aufgabentext bearbeiten und neben dem ER-Modell, die 3. NF und das relationale Datenbankschema notieren. Das Profitieren vom Wissen der Mitschülerin/des Mitschülers stand hier im Fokus.

Mittels Kahoot! wurde die in der letzten Doppeleinheit erarbeitete Theorie zu den Entity Relationship-Modellen und der Chen-Notation überprüft. Nicht so sehr begeistert, wie die SchülerInnen der 5. Klassen startete die Stundenwiederholung. Dies änderte sich im Verlauf des Quiz zunehmend. Nach und nach drang auch bei diesen SchülerInnen der Wettbewerbsgeist und die Motivation, bessere Leistung als die anderen zu bringen, durch.

Da bis zu diesem Zeitpunkt im Unterricht nur die Theorie behandelt wurde, erfolgte nun eine gemeinsame Erarbeitung eines ER-Modells zu einem vorgegebenen Beispiel. Um dies wie geplant durchführen zu können, erhielten die SchülerInnen ein Handout mit dem zu bearbeitenden Text. Das selbständige Ermitteln der Entitäten und der Attribute fiel ihnen trotz Anleitung (1. Unterstreichen aller Nomen; 2. Treffen der Unterscheidung zwischen brauchbaren und vernachlässigbaren Nomen; 3. Herausfinden der dazugehörigen Attribute) nicht leicht. Nach mehrmaligem Fragen der SchülerInnen wurde aus diesem Grund gemeinsam der Text gelesen und die Lösung Schritt für Schritt erarbeitet.

Als die erforderlichen Entitäten „Buch“ und „Person“ und die dazu passenden Attribute gefunden waren, war es für die SchülerInnen ein leichtes dies korrekt grafisch mittels ER-Modell darzustellen. Das Finden der Relation stellte auch keine Probleme dar. Das darauf anschließende Bestimmen der Kardinalitäten mittels Chen-Notation entpuppte sich wieder als etwas herausfordernder. Nach einer kurzen Wiederholung, welche Möglichkeiten vorhanden sind, und einer gemeinsamen Überlegung, was denn nun auf dieses Beispiel zutreffen würde, erfolgte eine korrekte Zuordnung der Kardinalitäten.

Um den SchülerInnen einen vollständigen Ablauf der Erarbeitung einer Datenbank näher zu bringen, folgte die gemeinsame Überführung des ER-Modells in das relationale Datenbankschema unter der vorhergehenden Anwendung der ersten drei Normalformen (= NF).

Die Anwendung der Normalformen fiel den SchülerInnen nicht sehr schwer, da sie auf dem Handout alle dazu notwendigen Informationen vorfanden. Das damit eng verbundene relationale Datenbankschema hingegen stellte einige SchülerInnen trotz Erklärung bzgl. der Auflösung einer 1:n-Beziehung auf der Tafel vor Problemen. Vier Mal wurde die Entität „Buch“ in drei unterschiedlichen Versionen notiert:

1. Buch (<u>Inventarnummer</u>, Titel, Erscheinungsjahr, Vorname, Nachname, Verlag)
2. Buch (<u>Inventarnummer</u>, Titel, Erscheinungsjahr, Vorname, Nachname, Verlag, Beginn, Ende)

3. Buch (Inventarnummer, E-Mail, Titel, Erscheinungsjahr, Vorname, Nachname, Verlag, Beginn, Ende)

Die übrigen SchülerInnen überführten das ER-Modell unter Berücksichtigung der 3. NF völlig korrekt in das relationale Datenbankschema.

Anschließend wurden gemeinsam die beiden Tabellen händisch skizziert und für das bessere Verständnis dieses Vorgangs jeweils zwei bis drei Datensätze eingefügt. Weiters stellte eine Schülerin die Frage, was jetzt mit diesem erarbeiteten Ergebnis gemacht werden könne und wofür dies nun sinnvoll sein solle. Die darauffolgende Erklärung löste bei einigen SchülerInnen einen Aha-Moment aus, welcher auch von einigen verbalisiert wurde. Weiters löste dies die Frage aus, ob nun mit der Übertragung der erarbeiteten Mini-Datenbank in ein Datenbankprogramm begonnen werden würde.

Da die vorhergehende Erarbeitung doch für einige SchülerInnen eine Herausforderung darstellte und noch sichtlich Übung in Form einer weiteren Aufgabe notwendig erschien, wurde diese Frage verneint. Auch erfolgte der Hinweis darauf, dass dies Thema der darauffolgenden Unterrichtssequenz sein werde.

Die darauffolgende Aufgabe, welche erneut das Erstellen eines ER-Modells, die Erarbeitung der 3. NF und dem Übertragen in das relationale Datenbankschema beinhaltete, fiel einigen SchülerInnen trotz Teamarbeit abschnittsweise sehr schwer. Besonders viele Fragen wurden zur korrekten Zuordnung der Kardinalitäten und zum Übertragen in das relationale Datenbankschema gestellt. Die Überführung in die 3. Normalform jedoch wurde von allen SchülerInnen problemlos und fehlerfrei angewendet.

Die aufgetretenen Schwierigkeiten während der selbstständigen Erarbeitung der gestellten Aufgabe zeigten deutlich auf, dass unbedingt noch weitere Aufgaben in der kommenden Unterrichtssequenz für die Festigung der vermittelten Inhalte notwendig waren. Dieser Eindruck bestätigte sich bei der Korrektur der abgegebenen Aufgabe.

Überprüfung des gelernten Wissens in der darauffolgenden Unterrichtssequenz

Da von Seiten der SchülerInnen noch einige Defizite vorhanden waren, die unbedingt einer Erklärung bedurften, wurden diese vor dem Abhalten einer Stundenwiederholung zu Beginn der Unterrichtssequenz durchgeführt.

Die darauf anschließende Überprüfung des gelernten Wissens erfolgte anhand eines kurzen Beispiels, welches von den SchülerInnen an der Tafel erarbeitet wurde. Dabei wurde wie folgt vorgegangen: Die Angabe wurde mittels Beamer an die Wand projiziert. Jeweils ein/e SchülerIn bekam eine Frage gestellt, welche sie/er lösen sollte. War dies nicht möglich, erhielt jene/r SchülerIn die Möglichkeit zur korrekten Beantwortung, welche die/der Nächste in der Sitzplatzreihenfolge war. Dieser Ablauf stellte sicher, dass alle in gleichem Ausmaß befragt wurden und somit keine Bevorzugung bzw. Benachteiligung einzelner SchülerInnen passierte.

Neben einem Aufzeichnen des ER-Modells, dem Überführen dieses in die 3. NF und der Darstellung mittels des relationalen Datenbankschemas, war auch eine Skizzierung der erarbeiteten Tabellen Inhalt der Stundenwiederholung. Die SchülerInnen bemühten sich sehr jede der gestellten Fragen zu beantworten, jedoch fiel ihnen dies besonders bei der Bestimmung der Kardinalitäten und dem Erstellen des relationalen Datenbankschemas etwas schwerer.

5 Zusammenfassung und Ausblick

Wissen über Computational Thinking und die Anwendung dessen Schlüsseltechniken bzw. dessen Konzepte und Methoden im Beruf oder im Alltag erleichtert das erfolgreiche Meistern dieser beiden Lebensbereiche enorm. Dieser Hintergrund und mein Beruf als Lehrerin für das Unterrichtsfach Informatik motivierten insbesondere für das Verfassen der vorliegenden Diplomarbeit, welche die Vermittlung von CT und dessen Erlernen durch SchülerInnen im kompetenzorientierten Informatikunterricht (5. AHS, Österreich) als Schwerpunkt setzt.

Die anfänglich durchgeführte Recherche zum Begriff „Computational Thinking“, die Gegenüberstellung einzelner Konzepte und das Bedürfnis neben einem historischen Einblick auch über die neuesten Entwicklungen in Bezug auf CT Bescheid zu wissen, begründet die Inhalte des zweiten Kapitels „Basiskonzepte des Computational Thinking und deren Vermittlung“. Weiters erfolgt die Bestärkung der von Wing getätigten Aussage, CT könne in vielen Bereichen als Problemlösestrategie angewandt werden (Wing, 2006: S. 33), mittels Darlegung von Beispielen aus den Bereichen der Mathematik, der Musik und dem Erlernen von Sprachen.

Um zum eigentlichen Schwerpunkt, d.i. das Erstellen von Ressourcen für den kompetenzorientieren Informatikunterricht und dessen funktionale Überprüfung in der Realität, zu gelangen, setzte dies eine tiefergehende Auseinandersetzung mit den Inhalten des derzeit vorliegenden Lehrplans (gültig ab dem Schuljahr 2017/18) und mit jenen rund um den Begriff „Kompetenzorientierung“ im dritten Kapitel „Kompetenzorientierter Lehrplan für das Unterrichtsfach Informatik in Österreich (5. AHS)“ voraus.

Als Herausforderung stellte sich das Herausfiltern einiger spannender und zugleich für die Vermittlung von CT passender Unterrichtsthemen aus dem Lehrplan dar, da nicht alle zu vermittelten Inhalte in gleichen Maßen für diesen Verwendungszweck geeignet sind. Das darauffolgende Planen und Durchführen der Einheiten unter Berücksichtigung bzw. Anwendung der Lerntheorie des „Konstruktionismus“, welche Seymour Papert für den selbständigen Aufbau von (u.a. informatischen) Wissen entwickelte (Papert, 1980: S. 1), wurde mit großer Motivation und Einsatzbereitschaft bewerkstelligt. Bei auftretenden Fragen bzgl. fachlicher oder fachdidaktischer Inhalte fungierten Kollegen unterstützend, die bereits Erfahrung mit den jeweiligen Themen im Unterricht gesammelt hatten. Die erhaltenen Tipps und gemeinsam entwickelten Ideen entpuppten sich als besonders

großartig bei dem Erarbeiten und Halten der zweiten Unterrichtssequenz zum Thema Sortierverfahren. Diese hatte eine sehr interessante Doppeleinheit mit wunderbar motivierten SchülerInnen zur Folge.

Während der Arbeit an dieser Diplomarbeit waren auch immer wieder Besuche von diversen themenbezogenen Veranstaltungen wichtige Motivatoren. Die dort vorgetragenen Inhalte und die im Anschluss geführten Gespräche mit den anderen TeilnehmerInnen erweckten den Eindruck, dass die Vermittlung von CT – begrenzt auf Wien – trotz der umfangreichen Bemühungen von mehreren Seiten (z.B. der OCG, Google, etc.) noch in den Kinderschuhen stecke. Besonders das Verbreiten von Wissen über und das Anwenden von Computational Thinking in den Schulen scheint noch nicht sehr stark verbreitet zu sein. Auch die Anzahl jener Lehrkräfte, die sich mit den Inhalten der betreffenden Materie beschäftigen, kommt mir relativ gering vor.

Die soeben beschriebenen Beobachtungen könnten als Anlass genommen werden um weiterführende Forschung zu betreiben. Neben einer Analyse, ob die bereits investierten Bemühungen CT erfolgreich in das Schulbildungssystem einzugliedern gefruchtet haben, erscheint im weiteren Verlauf auch das Erarbeiten von zusätzlichen Maßnahmen, welche dies (auch weiterhin) garantieren bzw. unterstützend für die Vermittlung durch Lehrende fungieren, erforderlich.

Bei vorhandenen Defiziten von Seiten der Lehrenden würde beispielsweise eine (verpflichtende) Online-Fortbildung eine einfache und kostengünstige Möglichkeit darstellen um sich dieses fehlende Wissen anzueignen. In diese Richtung gehend werden bereits ab dem Sommersemester 2018 Fortbildungsveranstaltungen, die das Anwenden von CT implizieren, an der Pädagogischen Hochschule Wien angeboten. Allerdings fehlen Lehrveranstaltungen, die nur die Inhalte von CT vermitteln, noch völlig.

Auch präventiv, d.h. bereits in der Ausbildung von Lehrenden an den Universitäten, wären (verpflichtende) Lehrveranstaltungen zum Thema CT sinnvoll. Dies wird beispielsweise seit dem Wintersemester 2017 an der Universität Wien bzw. an deren Fakultät für Informatik mit dem wählbaren *Erweiterungscurriculum Computational Thinking* im Ausmaß von 15 ECTS ermöglicht. Dieses Angebot wurde jedoch auf maximal 50 Studierende, die kein Informatikstudium betreiben, beschränkt. (Fakultät für Informatik, 2017)

Die hier beschriebenen Anfänge zeigen auf, dass noch viel Arbeit vor uns liegt um das von Jeannette M. Wing (2014) angepeilte Ziel in Österreich zu erreichen: „My grand vision is that computational thinking will be a fundamental skill – just like reading, writing, and arithmetic – used by everyone by the middle of the 21st Century."

6 Literaturverzeichnis

Computational Thinking

Angeli, C. & Voogt, J. & Fuck, A. & Webb, M. & Cox, M. & Malyn-Smith, J. & Zahami, J.: *A K-Computational Thinking Curriculum Framework: Implications for Teacher Knowledge*, in: Educational Technology & Society (2016: Vol. 19, No. 3), S. 47-57.

Barefoot. (2014a). *Computational Thinking*. Zugriff am 03. Februar 2017 unter http://barefootcas.org.uk/barefoot-primary-computing-resources/concepts/computational-thinking/

Barefoot. (2014b). *Tinkering.* Zugriff am 01. Juni 2017 unter https://barefootcas.org.uk/barefoot-primary-computing-resources/computational-thinking-approaches/tinkering/

Barefoot. (2014c). *Creating.* Zugriff am 01. Juni 2017 unter https://barefootcas.org.uk/barefoot-primary-computing-resources/computational-thinking-approaches/creating/

Barefoot. (2014d). *Debugging*. Zugriff am 01. Juni 2017 unter https://barefootcas.org.uk/barefoot-primary-computing-resources/computational-thinking-approaches/debugging/

Barefoot. (2014e). *Persevering.* Zugriff am 01. Juni 2017 unter https://barefootcas.org.uk/barefoot-primary-computing-resources/computational-thinking-approaches/perserving/

Barefoot. (2014f). *Collaborating.* Zugriff am 01. Juni 2017 unter https://barefootcas.org.uk/barefoot-primary-computing-resources/computational-thinking-approaches/collaborating/

Barr, D. & Harrison, J. & Conery, L.: *Computational Thinking: A Digital Age*, in: Learning & Leading with Technology (März-April 2011: Vol. 38, No. 6), S. 20-23.

Barr, V. & Stephenson, Ch.: *Bringing Computational Thinkung to K-12: What is Involved and What is the Role of the Computer Science Education Community?*, in: ACM Inroads (März 2011: Vol. 2, No. 1), S. 48-54.

BBC. (2016a). *Bitesize – Introduction to Computational Thinking*. Zugriff am 13. Juli 2016 unter http://www.bbc.co.uk/education/guides/zp92mp3/revision

BBC. (22. März 2016b). *BBC micro:bit launches to a generation of UK students*. Zugriff am 10. August 2017 unter http://www.bbc.co.uk/mediacentre/latestnews/2016/bbc-micro-bit-schools-launch

BBC. (2017a). *Decomposition*. Zugriff am 03. Februar 2017 unter http://www.bbc.co.uk/education/guides/zqqfyrd/revision

BBC. (2017b). *Pattern recognition*. Zugriff am 03. Februar 2017 unter http://www.bbc.co.uk/education/guides/zxxbgk7/revision

BBC. (2017c). *Abstraction*. Zugriff am 03. Februar 2017 unter http://www.bbc.co.uk/education/guides/zttrcdm/revision

BBC. (07. Juli 2017d). *BBC micro:bit celebrates huge impact in first year, with 90% of students saying it helped show that anyone can code.* Zugriff am 10. August 2017 unter http://www.bbc.co.uk/mediacentre/latestnews/2017/microbit-first-year

BBC. (2017e). *Introduction to computational thinking*. *Revise*. Zugriff am 26. September 2017 unter http://www.bbc.co.uk/education/guides/zp92mp3/revision/1

BBC. (2017f). *Introduction to computational thinking*. Test. Zugriff am 26. September 2017 unter http://www.bbc.co.uk/education/guides/zp92mp3/test

BBC micro:bit. (2015). *Partners*. Zugriff am 21. September 2017 unter https://www.microbit.co.uk/partners

BBC micro:bit. (2016). *BBC micro:bit*. Zugriff am 10. August 2017 unter http://microbitworld.me

Bebras. (2017). *Bebras. International Challenge on Informatics and Computational Thinking.* Zugriff am 14. August 2017 unter http://www.bebras.org/

Bocconi, S. & Chioccariello, A. & Dettori, G. & Ferrari, A. & Engelhardt, K. & Kampylis, P. & Punie, Y. (4. – 6. Juli 2016a). *Exploring the Field of COMPUTATIONAL THINKING as a 21st Century Skill*. Zugriff am 22. Juli 2017 unter https://www.slideshare.net/CompuThink/exploring-the-field-of-computational-thinking-as-a-21st-century-skill

Bocconi, S. & Chioccariello, A. & Dettori, G. & Ferrari, A. & Engelhardt, K. & Kampylis, P. & Punie, Y. (2016b). *Developing Computational Thinking in Compulsory Education. Implications for policy and practice.* Zugriff am 22. Juli 2017 unter https://ec.europa.eu/jrc/en/publication/eur-scientific-and-technical-research-reports/developing-computational-thinking-compulsory-education-implications-policy-and-practice

Bundesgesetzblatt Teil 2 (27. August 2014). *Lehrplan der Handelsakademie*. Zugriff am 01. August 2017 unter https://www.hak.cc/files/syllabus/Lehrplan_HAK_2014.pdf

Bundeskanzleramt Rechtsinformationssystem (RIS). (2017). *Bundesrecht konsolidiert: Gesamte Rechtsvorschrift für Lehrpläne - Neue Mittelschule*. Zugriff am 01. August 2017 unter https://www.ris.bka.gv.at/GeltendeFassung.wxe?Abfrage=Bundesnormen&Gesetzesnummer=20007850

Bundeskanzleramt Rechtsinformationssystem (RIS). (17. September 2015). *Lehrplan der höheren Lehranstalt für Wirtschaftsingenieure – Betriebsinformatik*. Zugriff am 01. August 2017 unter http://www.htl.at/fileadmin//content/Lehrplan/HTL_VO_262_2015/BGBl_II_Nr_262_2015_Anlage_1.24.pdf

Bundesministerium für Bildung (BMB). (2017a). *Informatik*. Zugriff am 01. August 2017 unter https://www.bmb.gv.at/schulen/unterricht/lp/lp_neu_ahs_14_11866.pdf

Bundesministerium für Bildung (BMB). (2017b). *Lehrplan der Volksschule*. Zugriff am 01. August 2017 unter https://www.bmb.gv.at/schulen/unterricht/lp/lp_vs_gesamt_14055.pdf?4dzgm2

Bundesministerium für Bildung (BMB). (2017c). *Verordnung der Bundesministerin für Bildung, Wissenschaft und Kultur, mit der die Verordnung über die Lehrpläne der allgemein bildenden höheren Schulen geändert wird; Bekanntmachung der Lehrpläne für den Religionsunterricht.* Zugriff am 01. August 2017 unter https://www.bmb.gv.at/schulen/unterricht/lp/11668_11668.pdf?61ebzh

Bundesministerium für Bildung (BMB). (20. April 2017d). *Verbindliche Übung „Digitale Grundbildung" in Sekundarstufe 1. Inhalte für Pilotierung im Schuljahr 2017/18.* Zugriff am 09. August 2017 unter http://zli.phwien.ac.at/wp-content/uploads/2017/05/Beilage_Digitale_Grundbildung_Inhalte_Pilotierung.pdf

Bundesministerium für Bildung (BMB). (25. Oktober 2017e). *Digitale Grundbildung*. Zugriff am 03. November 2017 unter https://www.bmb.gv.at/schulen/schule40/dgb/index.html

Bundy, A. & Scott, J.: *Creating a New Generation of Computational Thinkers,* in: Communications of the ACM (Dezember 2015: Vol. 58, No. 12), S. 37-40.

Code.org (2017). *Hour of Code*. Zugriff am 14. August 2017 unter https://hourofcode.com/de/en

Computer Science Education Week (CSEd Week). (2017). *How to Help*. Zugriff am 14. August 2017 unter https://csedweek.org/help

Computer Science Unplugged (CS Unplugged). (2017a). *Free activities for classroom or home*. Zugriff am 14. August 2017 unter http://csunplugged.org/

Computer Science Unplugged (CS Unplugged). (2017b). *Computer Science without a computer*. Zugriff am 14. August 2017 unter http://cs-unplugged.appspot.com/de/

Csizmadia, A. & Curzon, P. & Dorling, M. & Humphreys, S. & Ng, T. & Selby, C. & Woollard, J. (12. November 2015) *CAS computational thinking - A Guide for teachers.* Zugriff am 03. Februar 2017 unter http://community.computingatschool.org.uk/resources/2324

CTSTEM. (2017). *Computational Thinking in Science and Math*. Zugriff am 15. August 2017 unter http://ct-stem.northwestern.edu/

Denning, P.: *The Profession of IT. Beyond Computational Thinking*, in: COMMUNICATIONS OF THE ACM (Juni 2009: Vol. 52, No. 6), S. 28-30.

Deutschschweizer Erziehungsdirektoren-Konferenz (D-EDK). (2017). *Willkommen beim Lehrplan 21.* Zugriff am 06. August 2017 unter https://www.lehrplan.ch/

Deutschschweizer Erziehungsdirektoren-Konferenz (D-EDK). (29. Februar 2016). *Medien und Informatik*. Zugriff am 06. August 2017 unter http://v-ef.lehrplan.ch/container/V_EF_DE_Modul_MI.pdf

Fakultät für Informatik. (2017). *Erweiterungscurriculum Computational Thinking*. Zugriff am 04. Jänner 2018 unter http://informatik.univie.ac.at/studierende/der-weg-durchs-studium/erweiterungscurriculum/ec-computational-thinking/

First Lego League (FLL). (2017a). *Was ist FLL?* Zugriff am 10. August 2017 unter https://www.first-lego-league.org/de/allgemeines/was-ist-fll.html

First Lego League (FLL). (2017b). *Hydro Dynamics[SM].* Zugriff am 10. August 2017 unter https://www.first-lego-league.org/de/2017/hydrodynamics.html

Futschek, G.: *Algorithmic Thinking: The Key for Understanding Computer Science*, in: Informatics Education – The Bridge between Using and Understanding Computers (November 2006), S. 159-168.

Futschek, G.: *Bildung 4.0: Informatisches Denken ist Schlüsselkompetenz,* in: OCG Journal (2016: Vol. 41, No. 2), S. 20-21.

Gallenbacher, J. (2017, 4. Auflage). *Abenteuer Informatik. IT zum Anfassen für alle von 9 bis 99 – vom Navi bis Social Media.* Berlin: Springer.

Google. (2016). *Computational Thinking for Educators*. Zugriff am 13. Juli 2016 unter https://computationalthinkingcourse.withgoogle.com/unit?lesson=8&unit=1

Google Community. (2017). *Computational Thinking Course.* Zugriff am 21. September 2017 unter https://plus.google.com/communities/104843475244210707629

Google for Education. (2017). *CT Materials*. Zugriff am 21. September 2017 unter https://edu.google.com/resources/programs/exploring-computational-thinking/#!ct-materials

Grandl, M. & Ebner, M. (Februar 2017). *Informatische Grundbildung – ein Ländervergleich.* Zugriff am 09. August 2017 unter http://www.medienimpulse.at/pdf/Medienimpulse_Informatische_Grundbildung_____ein_Laendervergleich_Grandl_20170514.pdf

Grover, S. & Pea, R.: *Computational Thinking in K-12: A Review of the State of the Field*, in: Educational Researcher (2013: Vol. 42, No. 1), S. 38-43.

Harel, I. (25. Mai 2016). *American schools are teaching our kids how to code all wrong*. Zugriff am 26. September 2017 unter https://qz.com/691614/american-schools-are-teaching-our-kids-how-to-code-all-wrong/

Hübel-Fleischmann, B. (2017). *Entdeckendes und problemlösendes Lernen mit Beebot und Ozobot Robotern.* Zugriff am 01. August 2017 unter https://www.ph-online.ac.at/ph-wien/wbLv.wbShowLVDetail?pStpSpNr=231129

Ismailova, L. Y.: *Criteria for Computational Thinking in Information and Computational Technologies,* in: Life Science Journal (2014: Vol. 11, No. 9s), S. 415-420.

Johnson, M. & Wocicki, E. (08. Februar 2017). *Education for 2017: Learning Computational Thinking is the Key to Jobs.* Zugriff am 21. September 2017 unter http://www.huffingtonpost.com/entry/to-all-students-learning-computational-thinking-will_us_57a15f7ee4b004301c522b7c

Kretzschmar, R. (2016). *Informatics at Swiss Gymnasium schools*. Zugriff am 06. August 2017 unter https://www.ocg.at/sites/ocg.at/files/medien/pdfs/160418_Kretzschmar.pdf

Lang, H. W. (13. Mai 2016). *Turingmaschine*. Zugriff am 03. November 2017 unter http://www.iti.fh-flensburg.de/lang/theor/turing-maschine.htm

Lee, I. & Martin, F. & Denner, J. & Coulter, B. & Allan, W. & Erickson, J. & Malyn-Smith, J. & Werner, L.: *Computational Thinking for Youth in Practice*, in: ACM Inroads (März 2011: Vol. 2, No. 1), S. 32-37.

Lu, J. J. & Fletcher, G. H. L.: *Thinking About Computational Thinking*, in: Proceedings of the 40th ACM technical symposium on Computer science education (04. März 2009), S. 260-264

Miles, B. (24. März 2014). *Computational Thinking in Primary Schools*. Zugriff am 03. Februar 2017 unter http://milesberry.net/2014/03/computational-thinking-in-primary-schools/

Mindstorms. (2017). *31313 Mindstorms EV3*. Zugriff am 10. August 2017 unter https://www.lego.com/en-us/mindstorms/products/mindstorms-ev3-31313

Österreichische Computer Gesellschaft (OCG). (2016a). *8. Informatiktag 2016.* Zugriff am 01. August 2017 unter https://www.ocg.at/de/informatiktag16

Österreichische Computer Gesellschaft (OCG). (2016b). *Computational Thinking & Coding @ Schools*. Zugriff am 02. November 2017 unter http://www.ocg.at/ct-c-at-schools

Österreichische Computer Gesellschaft (OCG). (2017a). *Informatisches Denken und Programmieren*. Zugriff am 01. August 2017 unter http://www.ocg.at/de/Informatisches_Denken_2017

Österreichische Computer Gesellschaft (OCG). (2017b). *9. Informatiktag 2017*. Zugriff am 01. August 2017 unter https://www.ocg.at/de/informatiktag-2017

Österreichische Computer Gesellschaft (OCG). (2017c). *Biber der Informatik*. Zugriff am 14. August 2017 unter http://www.ocg.at/de/biber

Papert, S. (1980). *Mindstorms. Children, Computers, and Powerful Ideas.* New York: Basic Books, Inc., Publishers.

Pädagogische Hochschule Wien (PH Wien). (2016). *PH Wien bekommt Lego Labor*. Zugriff am 01. August 2017 unter https://www.phwien.ac.at/86-paedagogische-hochschule-wien/nachlese/2183-ph-wien-bekommt-legolabor

QuickStart Computing. (2017). *Computational Thinking*. Zugriff am 15. August 2017 unter http://primary.quickstartcomputing.org/resources/pdf/comp_thinking.pdf

Raspberry Pi Foundation. (2017a). *Coderdojo*. Zugriff am 14. August 2017 unter https://www.raspberrypi.org/education/coderdojo/

Raspberry Pi Foundation. (2017b). *Resources*. Zugriff am 14. August 2017 unter https://www.raspberrypi.org/resources/

Repenning, A. (Dezember 2014). *Computational Thinking in der Lehrerinnenbildung*. Zugriff am 06. August 2017 unter https://www.haslerstiftung.ch/documents/d/fit_schriftenreihe/haslerstiftung_schriften04_de_v02.pdf

Repenning, A. (2016). *Scalable Game Design. A Systemic Strategy for Introducing Computational Thinking in Schools.* Zugriff am 06. August 2017 unter https://www.ocg.at/sites/ocg.at/files/medien/pdfs/160418_Repenning.pdf

Resnick, M. & Maloney, J. & Monroy-Hernández, A. & Rusk, N. & Eastmond, E. & Bennan, K. & Millner, A. & Rosenbaum, E. & Silver, J. &Silverman, B. & Kafai, Y.: *Scratch: Programming for All*, in: Communications of the ACM (November 2009: Vol. 52, No. 11), S. 60-67.

Selby, C. C. & Woollard, J. (2013). *Computational Thinking: The Developing Definition*. Zugriff am 20. Juli 2017 unter https://eprints.soton.ac.uk/356481/1/Selby_Woollard_bg_soton_eprints.pdf

Smith, M. (30. Jänner 2016). *Computer Science For All*. Zugriff am 26. September 2017 unter https://obamawhitehouse.archives.gov/blog/2016/01/30/computer-science-all

Sourceforge. (2017). *Welcome to the world of FMSLogo!* Zugriff am 11. August 2017 unter http://fmslogo.sourceforge.net/

SQA. (2017). *About Core Skills*. Zugriff am 21. September 2017 unter https://www.sqa.org.uk/sqa/37801.7292.html

Terrapin Software. (2016). *Bee-Bot*. Zugriff am 10. August 2017 unter https://www.bee-bot.us/

TU Wien. (2016). *Jeannette Wing: Computational Thinking.* Veröffentlicht am 04. Juli 2016. Zugriff am 04. Juli 2016 unter https://www.youtube.com/watch?v=YVEUOHw3Qb8&index=5&list=PLQku6m__XAxGEb1-ZIWSYnLxINppXHbXF

Wilson, K. (08. Dezember 1982). *Kenneth G. Wilson - Nobel Lecture. The Renormalization Group And Critical Phenomena.* Zugriff am 08. Februar 2017 unter https://www.nobelprize.org/nobel_prizes/physics/laureates/1982/wilson-lecture.html

Wing, J. M.: *Computational Thinking*, in: COMMUNICATIONS OF THE ACM (März 2006: Vol. 49, No. 3), S. 33-35.

Wing, J. M. (2016). *Computational thinking, 10 years later*. Zugriff am 12. April 2017 unter https://www.microsoft.com/en-us/research/blog/computational-thinking-10-years-later/

Wing, J. M.: *Computational thinking and thinking about computing*, in: Philosophical Transactions of the Royal Society (28. Oktober 2008). Zugriff unter http://rsta.royalsocietypublishing.org/content/366/1881/3717

Wing, J. M. (10. Jänner 2014). *Computational Thinking Benefits Society*. Zugriff am 08. Juli 2016 unter http://socialissues.cs.toronto.edu/index.html%3Fp=279.html

Wolfram, St. (2017a, 2. Auflage). *An Elementary Introduction to the Wolfram Language*. Zugriff am 15. August 2017 unter http://www.wolfram.com/language/elementary-introduction/2nd-ed/preface.html

Wolfram. (2017b). *Wolfram Programming Lab. Berechnungsorientiertes Wissen beginnt hier.* Zugriff am 15. August 2017 unter https://www.wolfram.com/programming-lab/

Wolfram, St. (7. September 2016). *How to Teach Computational Thinking*. Zugriff am 15. August 2017 unter http://blog.stephenwolfram.com/2016/09/how-to-teach-computational-thinking/

Voogt, J. & Fisser, P. & Good, J. & Mishra, P. & Yadav, A.: *Computational thinking in compulsory education: Towards an agenda for research and practice*, in: Education and Information Technologies (Dezember 2015: Vol. 20, No. 4), S. 715-728.

Yadav, A. & Zhou, N. & Mayfield, C. & Hanbrusch, S. & Korb, J. T.: *Introducing computational thinking in education courses*, in: Proceedings of the 42nd ACM technical symposium Computer science education (09. März 2011), S. 465-470

Yuen, L. (13. Jänner 2017). *Coding and Computational Thinking In Schools*. Zugriff am 26. September 2017 unter http://bolt.athabascau.ca/index.php/2017/01/13/coding-and-computational-thinking-in-schools/

Zentrum für Lernechnologie und Innovation (ZLT). (13. Februar 2017). *Eröffnung des Lego Education Innovation Studios beim eBazar am 16.03.2017.* Zugriff am 01. August 2017 unter http://zli.phwien.ac.at/eroeffnung-des-lego-education-innovation-studios-beim-ebazar-am-16-03-2017/

Zentrum für Lernechnologie und Innovation (ZLT). (01. November 2016). *PH Wien bekommt Lego Labor*. Zugriff am 01. August 2017 unter http://zli.phwien.ac.at/ph-wien-bekommt-lego-labor/

Kompetenzorientierter Lehrplan

Beer, R. & Benischek, I. & Brock, R. & Habringer, G. & Herland, G. & Mürwald-Scheifinger, E. & Scherf, S. & Staud, H. & Weber, W. & Werbowsky, I. & Zöchlinger, B. (2011, Hrsg. BIFIE). *Kompetenzorientierter Unterricht in Theorie und Praxis*. Graz: Leykam. Zugriff am 27. Oktober 2017 unter http://www.bifie.at/wp-content/uploads/2017/06/bist_vs_sek1_kompetenzorientierter_unterricht_2011-03-23.pdf

Bildungsforschung, Innovation & Entwicklung des österreichischen Schulwesens (BIFIE). (2017). *Kompetenzen und Modelle*. Zugriff am 22. August 2017 unter https://www.bifie.at/node/49

Bundeskanzleramt. (09. August 2016). BUNDESGESETZBLATT FÜR DIE REPUBLIK ÖSTERREICH. Zugriff am 24. August 2017 unter https://www.ris.bka.gv.at/Dokumente/BgblAuth/BGBLA_2016_II_219/BGBLA_2016_II_219.html

Bundeskanzleramt Rechtsinformationssystem (RIS). (01. September 2017). *Bundesrecht konsolidiert: Gesamte Rechtsvorschrift für Lehrpläne – allgemeinbildende höhere Schulen, Fassung vom 01.09.2017*. Zugriff am 27. Oktober 2017 unter https://www.ris.bka.gv.at/GeltendeFassung.wxe?Abfrage=Bundesnormen&Gesetzesnummer=10008568&FassungVom=2017-09-01

Bundesministerium für Bildung und Forschung (BMBF). (2007, Band 1). *Zur Entwicklung nationaler Bildungsstandards. Expertise*. Zugriff am 23. August 2017 unter https://www.bmbf.de/pub/Bildungsforschung_Band_1.pdf

Klieme, E.: *Was sind Kompetenzen und wie lassen sie sich messen?*, in: Pädagogik (Weinheim) (2004: Vol. 56, No. 6), S. 10-13.

Klieme, E. & Leutner, D.: *Kompetenzmodelle zur Erfassung individueller Lernergebnisse und zur Bilanzierung von Bildungsprozessen. Beschreibung eines neu eingerichteten Schwerpunktprogramms der DFG*., in: Pädagogik (2006: Vol. 52, No. 6), S. 876-903.

Neues Wort. (2017). *kognitiv*. Zugriff am 22. August 2017 unter https://neueswort.de/kognitiv/

Neuro Nation. (2017). *Kognitive Fähigkeiten*. Zugriff am 22. August unter https://www.neuronation.de/kognitives-training/kognitive-faehigkeiten

Österreichische Computer Gesellschaft (OCG). (11. November 2016). *10 Jahre „Biber der Informatik" in Österreich.* Zugriff am 27. Oktober 2017 unter https://www.ots.at/presseaussendung/OTS_20161027_OTS0172/10-jahre-biber-der-informatik-in-oesterreich

Pädagogische Hochschule Oberösterreich (PH OÖ). (2017a). *digi.check12 – Onlinetest.* Zugriff am 27. Oktober 2017 unter https://eeducation.at/index.php?id=563&L=0

Pädagogische Hochschule Oberösterreich (PH OÖ). (2017b). *digi.komp12 – Das Kompetenzmodell Informatik 5. Klasse.* Zugriff am 27. Oktober 2017 unter https://eeducation.at/index.php?id=224&L=0

Pelz, W. (2017). *Volitionskompetenz (Volition als Kompetenz).* Zugriff am 22. August 2017 unter http://www.volitionskompetenz.de/

Schweizer, K. (2006, Hrsg.). *Leistung und Leistungsdiagnostik*. Heidelberg: Springer Medizin Verlag.

Standl, B. (2013). *Conceptual Modeling and Innovative Implementation of Person-centered Computer Science Education at Secondary School Level.* Dissertation an der Universität Wien.

Technische Universität München (TU München). (2017a). *Kompetenzbereiche*. Zugriff am 20. September 2017 unter https://www.pisa.tum.de/kompetenzbereiche/

Technische Universität München (TU München). (2017b). *PISA 2018*. Zugriff am 20. September 2017 unter https://www.pisa.tum.de/pisa-2018/

Weinert, F. E. (S. 17-31): Vergleichende Leistungsmessung in Schulen – eine umstrittene Selbstverständlichkeit., in: Weinert, F.E. (2002, 2. Auflage, Hrsg.). (2002). *Leistungsmessungen in Schulen*. Weinheim und Basel: Beltz Verlag.

Wiesner, C. & Schreiner, C. & Breit, S. & Pacher, K. (2017). *Bildungs-standards und kompetenz-orientierter Unterricht*. Zugriff am 28. Oktober 2017 unter https://www.bifie.at/bildungsstandards-und-kompetenzorientierter-unterricht/

7 Abkürzungsverzeichnis

AHS	Allgemeine bildende höhere Schule
BIFIE	Bildungsforschung, Innovation & Entwicklung des österreichischen Schulwesens
BMBF	Bundesministerium für Bildung und Forschung
bzw.	beziehungsweise
CEPIS	Council of European Professional Informatics Societies
CS	Computer Science
CSEd Week	Computer Science Education Week
CT	Computational Thinking
CTL	Computational Thinking Language
d.h.	das heißt
d.i.	das ist
dt.	deutsch
ECT	Exploring Computational Thinking
etc.	et cetera
FLL	First Lego League
GFI@CH	Grundlangenfach Informatik in der Schweiz
HAK	Handelsakademie
HPC	high performance computations
HTL	Höhere Technische Lehranstalt
IBM	International Business Machines Corporation
IT	Information Technology
LEIS	Lego Education Innovation Studio
lt.	laut
NF	Normalform
NOST	Neue Oberstufe
OCG	Österreichischen Computergesellschaft
PH OÖ	Pädagogische Hochschule Oberösterreich
u.a.	unter anderem
uvm.	und vieles mehr
WL	Wolfram Language
ZLI	Zentrum für Lerntechnologie und Innovation der PH Wien

8 Abbildungsverzeichnis

9 Anhang

9.1 Zusammenfassung

Computational Thinking (= CT) repräsentiert eine große Menge an Methoden und Modellen, die unterstützend bei der Bearbeitung komplexer Probleme und Aufgabenstellungen fungieren und ohne deren ein Hervorbringen von Lösungen nicht möglich wäre. (Wing, 2006: S. 33)

Diese Diplomarbeit beschäftigt sich im ersten Teil mit der Beschreibung des Begriffs „Computational Thinking“ und der Erläuterung einiger vorhandener Methoden und Modelle. Neben einer Analyse welche Fortschritte in der Schweiz und EU-weit, mit besonderem Fokus auf Österreich, bei der Einführung von CT in das jeweilige schulische Bildungssystem bereits erzielt wurden, erfolgt auch eine Darstellung von Unternehmen und Tools, die dabei unterstützend wirken.

Dies und die im darauffolgenden Teil festgehaltene Beschreibung der Begriffe zum Thema „Kompetenzorientierung“ und die hergestellte Verbindung zum aktuellen Informatik-Lehrplan der AHS (gültig ab dem Schuljahr 2017/18) legen den Grundstein für den letzten Teil: das Erarbeiten, Durchführen und Reflektierten von Unterrichtssequenzen inkl. der jeweils dazugehörigen Materialien.

9.2 Abstract

Computational Thinking (= CT) represents a large number of methods and models which facilitate the handling of complex problems and tasks. Without this necessary support devising corresponding solutions would be impossible. (Wing, 2006: p. 33)

The first main chapter of this thesis introduces the term “Computational Thinking” and explains a variety of existing methods and models. In addition, recent progress on the adoption of CT in educational systems is identified, considering Switzerland and the European Union focusing particularly on Austria. Furthermore, several tools and companies supporting the development of CT are presented.

The discussion regarding the terms related to “Kompetenzorientierung” as well as the description of the requirements for “Kompetenzorientierung” in Austria’s computer science curriculum lay the foundation for the last part of this thesis: a collection of compiled teaching units including a critical reflection of the conducted lessons and the corresponding materials.

9.3 Lehrplan des Unterrichtsfachs Informatik – 5. AHS (Bundeskanzleramt, 2016)

Bildungs- und Lehraufgabe (5. Klasse):

Bildungsziele und Bildungsinhalte sind immer ein Spiegelbild des gesellschaftlichen, politischen und ökonomischen Umfeldes. Gegenwärtig bildet die Informatik den Wesenskern des digitalen Zeitalters und damit auch das Fundament moderner Informations- und Kommunikationstechnologien.

Ihre Inhalte sind daher allgemeinbildend und dienen sowohl einem fundierten Weltverständnis als auch der fachlichen Basis für zukünftige Berufsbilder. Der Informatik kommt als Wissenschaft und als schulisches Fachgebiet eine Schlüsselrolle zu, da sie die automatische Datenverarbeitung und digitale Informationsrepräsentation zum Gegenstand hat und diese mit Hilfe von Informatiksystemen nutzbar macht.

Das Fach Informatik eröffnet allen Schülerinnen und Schülern einen gleichberechtigten Zugang zu informatischen Denk- und Arbeitsweisen als Voraussetzung für den produktiven Umgang mit digitalen Informations- und Kommunikations-technologien.

Beiträge zu den Bildungsbereichen

Sprache und Kommunikation

Konstruktiver Informatikunterricht ist auch Sprachunterricht. Der Mensch-Maschine-Kommunikation liegt im Gegensatz zu natürlichen Sprachen eine abstrakte formale Sprache zugrunde.

Informatiksysteme tragen wesentlich zu Veränderungen der Kommunikationskultur bei. Unterschiedliche digitale Repräsentationsformen von Information ergänzen die traditionelle Verständigung und erfordern neue technologische und methodische Kompetenzen.

Die vielfältigen Möglichkeiten der elektronischen Kommunikation ermöglichen einen Austausch über Grenzen hinweg und erleichtern die virtuelle Begegnung mit anderen Kulturen. Die davon ausgehende Motivation, Fremdsprachenkenntnisse zu erwerben, wird durch die Verfügbarkeit aktueller und authentischer fremdsprachlicher Informationen und das Fachvokabular verstärkt.

Mensch und Gesellschaft

Arbeitswelt und privates Umfeld der Menschen verändern sich durch den Einfluss der Informationstechnologien permanent. Durch die Beschäftigung mit diesen Technologien lernen Schülerinnen und Schüler deren Auswirkungen, Möglichkeiten, Grenzen und Gefahren kennen.

Die Schülerinnen und Schüler erkennen das Potenzial ihrer eigenen Fähigkeiten als denkende, handelnde, fühlende und sich entwickelnde Menschen im Unterschied zu einer lernenden Maschine. Dies erfordert einen verantwortungsvollen Umgang mit Informationstechnologien.

Natur und Technik

Durch Modellbildung, Formalisierung und Abstraktion leistet die Informatik einen wesentlichen Beitrag zur Auseinandersetzung mit Natur und Technik und führt zu einer besseren Entscheidungs- und Handlungskompetenz.

Kreativität und Gestaltung

Der Umgang mit Informationstechnologie gibt den Schülerinnen und Schülern Gelegenheit, selbst kreativ tätig zu sein und Gestaltungserfahrungen zu machen.

Gesundheit und Bewegung

Die Verantwortung für den eigenen Körper erfordert als Ausgleich zur Arbeit am Computer gezielte Bewegung. Den Schülerinnen und Schülern soll die Bedeutung eines ergonomisch gestalteten Arbeitsplatzes bewusst werden.

Der Einsatz von Informationstechnologien zur Erfassung und Analyse von Daten im Sport- und Gesundheitsbereich bietet die Möglichkeit zur kritischen Reflexion.

Didaktische Grundsätze (5. Klasse):

Der Lehrplan bietet den Lehrerinnen und Lehrern Freiräume für die eigenständige und verantwortliche Unterrichtsgestaltung, in der eine ausgewogene Abdeckung aller Kompetenzbereiche anzustreben ist. Dabei sind die Themen und Inhalte so auszuwählen und zu organisieren, dass sie die Vorkenntnisse und Vorerfahrungen der Schülerinnen und Schüler berücksichtigen und daran anknüpfen. Die Themen sind dabei so auszuwählen, dass sie vielfältige Bezüge zur Lebens- und Begriffswelt der Jugendlichen herstellen. Im Informatikunterricht besondere fachdidaktische Überlegungen anzustellen um Defizite aus den vorangehenden Schulstufen auszugleichen und individuelle Stärken einzubinden und zu fördern.

Die Unterrichtsplanung hat sich an für Schülerinnen und Schüler transparenten Lehrzielen zu orientieren. Variierende Arbeitsformen wie Einzelarbeit, Gruppenarbeit und Teamarbeit geben Schülerinnen und Schülern Gelegenheit, Neues zu erforschen und bereits Gelerntes in verschiedenen kommunikativen und inhaltlichen Kontexten anzuwenden. Selbsttätigkeit und Eigenverantwortung sind zu fördern und Möglichkeiten zur persönlichen Lernzielkontrolle anzubieten.

Gemeinschaftliches Problemlösen in einem projektorientierten Unterricht soll gefördert werden. Dabei ist kooperativen Entscheidungsstrukturen entsprechender Platz einzuräumen. Methodische Überlegungen sollen sich an den spezifischen Anforderungen von Einstieg, Entwicklung und Abschluss von Unterrichtsphasen orientieren. Explorative, systematische und exemplarische Vorgehensweisen sollen zur Vertiefung von Wissen und Erweiterung von Kompetenzen in der Informatik führen.

Schülerinnen und Schülern ist Gelegenheit zu geben, durch Transfer und Analogiebildung den Lernertrag zu sichern. Der Informatikunterricht soll beispielhaft für den sinnvollen Einsatz verfügbarer Technologien sein. Dem Erwerb einer wissenschaftlichen Arbeits- und Dokumentationsweise ist die Erstellung eines Portfolios dienlich.

Der Erwerb informatischer Kompetenzen erfordert passende Formen der Wissensdarstellung und -verarbeitung. Grundlegende Strukturen und Prozesse in Gesellschaft, Natur und Technik werden aus Sicht der Informatik veranschaulicht. Dazu sind Methoden der Visualisierung und der Abstraktion zu verwenden. Die zyklische Vorgangsweise des Sammelns, Auswählens, Strukturierens, Abstrahierens, Auswertens und Interpretierens von Daten ist beim Problemlösen zu berücksichtigen.

Zur Motivation und zur Sicherung des Unterrichtsertrags sind den Schülerinnen und Schülern im Rahmen des Informatikunterrichts vielfältige Möglichkeiten anzubieten, ihr Wissen zu präsentieren, sich der Kritik anderer zu stellen und ihre Arbeit zu argumentieren.

Die Gestaltung eines angenehmen und erfolgreichen Lernklimas beruht auf Vertrauen, auf der Förderung der individuellen Stärken und des kreativen Potenzials. Auf die unterschiedlichen Interessen sowohl der Schülerinnen als auch der Schüler ist durch Auswahl entsprechender Inhalte und Aufgabenstellungen einzugehen.

Exkursionen und Einladungen von Expertinnen und Experten zu Vorträgen und zur Diskussion sollen den Erfahrungshorizont erweitern.

Bildungs- und Lehraufgabe, Lehrstoff:

Informatische Bildung ist das Ergebnis von Lernprozessen, in denen fachliche Grundlagen verdeutlicht und Anwendungskompetenzen durch planvolle Arbeitsweisen systematisch erworben werden. Sie befähigt Schülerinnen und Schüler, die gesellschaftliche und wirtschaftliche Dimension digitaler Informations- und Kommunikationstechnologien zu erfassen. Aufgabe des Informatikunterrichts ist es, die Schülerinnen und Schüler zum Erwerb informatischer und informationstechnischer Kompetenzen hinzuführen, um sie zu befähigen, diese zur Lösung verschiedener Problemstellungen einzusetzen.

Durch die Analyse realer Probleme vor allem aus ihrer Erfahrungswelt sollen sie Strukturen und Zusammenhänge erkennen und die Notwendigkeit von Abstraktion und Reduktion bei der Modellbildung von einfachen realen Systemen erfahren und diese Modelle auf empirische Daten anwenden lernen. Sie sollen kooperative und kommunikative Arbeitsweisen unter Einsatz von Kommunikationstechnologien anwenden lernen. In allen Bildungsbereichen stehen dabei Erweiterung und Festigung von Sach-, Selbst-und Sozialkompetenz im Mittelpunkt.

Die Schülerinnen und Schüler sollen erkennen, dass die Informatik einer wissenschaftlichen Systematik unterliegt und Interesse und Wertschätzung verdient. Der Informatikunterricht fasst die vorhandenen Fähigkeiten von Schülerinnen und Schülern in der Informatik durch Beschäftigung mit Entwurf, Gestaltung und Anwendung von Informationssystemen zusammen und baut sie aus. Bei der kritischen Auseinandersetzung mit den dabei ablaufenden Prozessen und deren Ergebnissen sollen die Schülerinnen und Schüler ihr kognitives, emotionales und kreatives Potenzial nützen. Dies soll die Jugendlichen bei der Entwicklung und Festigung einer persönlichen Werthaltung und Weltsicht unterstützen und einen tieferen Einblick in die gesellschaftlichen Zusammenhänge und Auswirkungen moderner Informationstechnologie ermöglichen.

5. Klasse (1. und 2. Semester)

Informatik, Mensch und Gesellschaft

- Die Bedeutung von Informatik in der Gesellschaft beschreiben, die Auswirkungen auf die Einzelnen und die Gesellschaft einschätzen und Vor- und Nachteile an konkreten Beispielen abwägen können
- Maßnahmen und rechtliche Grundlagen im Zusammenhang mit Datensicherheit, Datenschutz und Urheberrecht kennen und anwenden können
- Die Entwicklung der Informatik beschreiben und bewerten können

- Informatikberufe und Einsatzmöglichkeiten der Informatik in verschiedenen Berufsfeldern benennen und einschätzen können

Informatiksysteme
- Den Aufbau von digitalen Endgeräten beschreiben und erklären können
- Die Funktionsweise von Informatiksystemen erklären können
- Grundlagen von Betriebssystemen erklären, eine graphische Oberfläche und Dienstprogramme bedienen können
- Grundlagen der Vernetzung von Computern beschreiben und lokale und globale Computernetzwerke nutzen können

Angewandte Informatik
- Standardsoftware zur Kommunikation und Dokumentation sowie zur Erstellung, Publikation und multimedialen Präsentation eigener Arbeiten einsetzen können
- Standardsoftware für Kalkulationen und zum Visualisieren anwenden können
- Informationsquellen erschließen, Inhalte systematisieren, strukturieren, bewerten, verarbeiten und unterschiedliche Informationsdarstellungen verwenden können
- Digitale Systeme zum Informationsaustausch, zur Unterstützung der Unterrichtsorganisation und zum Lernen auch in kommunikativen und kooperativen Formen verwenden können

Praktische Informatik
- Begriffe und Konzepte der Informatik verstehen und Methoden und Arbeitsweisen anwenden können
- Algorithmen erklären, entwerfen, darstellen und in einer Programmiersprache implementieren können
- Grundprinzipien von Automaten, Algorithmen, Datenstrukturen und Programmen erklären können
- Datenbanken benutzen und einfache Datenmodelle entwerfen können

9.4 Pädagogische Patterns

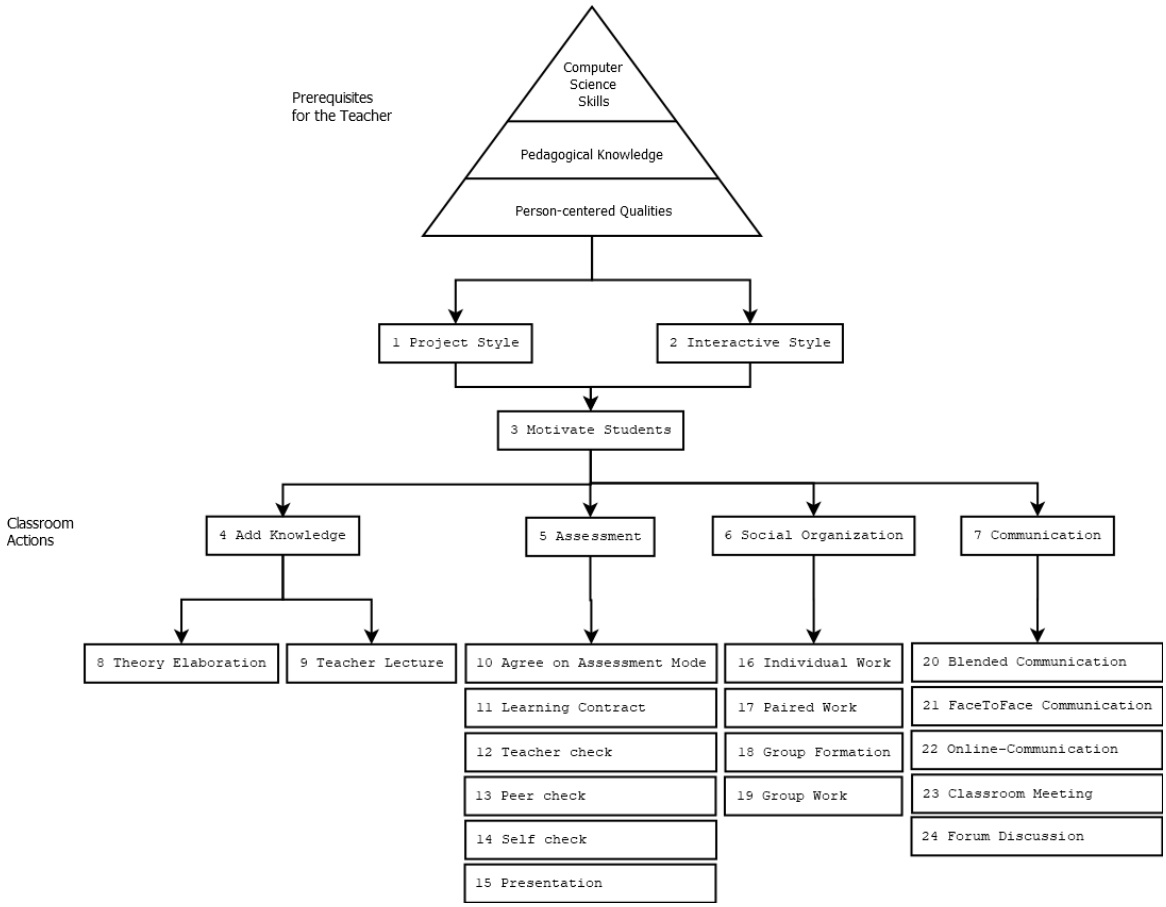


Abbildung 15: Übersicht des gesamten Pattern-Netzwerkes (Standl, 2013: S. 89)

Diese Übersicht gliedert sich in zwei große Bereiche:

- „*Prerequisites for the Teacher*“: Hier erfolgt die Angabe, welche Fähigkeiten eine Lehrkraft beherrschen muss, um die erwähnten Patterns überhaupt anwenden zu können. (Standl, 2013: S. 90)
- Unter dem Begriff „*Classroom Actions*“ werden 24 Patterns verstanden, die von Bernhard Standl im Rahmen seiner Dissertation im Jahr 2013 erarbeitet wurden. (Standl, 2013: S. 90) Neben einer Beschreibung dieser, erfolgt auch die Erklärung der Abhängigkeiten der einzelnen Patterns untereinander, eine Darstellung des ursprünglichen Problems inkl. der Stärken der Patterns und der tatsächlichen Anwendung im Schulalltag. Der gesamte Ablauf eines Patterns erfolgt auch grafisch. Um den Rahmen dieser Arbeit nicht zu sprengen, erfolgt hier nur eine kurze Zusammenfassung der Inhalte der jeweiligen pädagogischen Patterns:
  1. Der ***Project Style*** beinhaltet einen vorgegebenen Ablauf, welcher über einen längeren Zeitrahmen den Fokus auf selbstorganisiertes Lernen legt. Dabei ist die

Lehrkraft für das zur Verfügung stellen von Unterstützung verantwortlich. Dies soll sicherstellen, dass die SchülerInnen die jeweiligen Projekte erfolgreich abschließen. (Standl, 2013: S. 91-93)

2. Der ***Interactive Style*** beschreibt Abschnitte von Unterrichtseinheiten, welche interaktiv geplant bzw. gehalten werden um neue Inhalte zu vermitteln. Hintergrundgedanke dabei ist es, mit möglichst vielen unterschiedlichen Lernmethoden die SchülerInnen immer aufs Neue zu begeistern. (Standl, 2013: S. 94-95)
3. ***Motivate Students****:* Begeisterte SchülerInnen sind besonders für eine positive Lernerfahrung wichtig. Können diese nicht begeistert werden neue Inhalte eines Themas zu erarbeiten, so bleibt die Motivation teilweise oder vollständig aus. Um das Interesse zu wecken, ist oft eine Begründung, weshalb dieses wichtig bzw. spannend sei, notwendig. (Standl, 2013: S. 96-97)
4. ***Add Knowledge***: Neues Wissen wird von SchülerInnen schneller gelernt, wenn diese einen einleuchtenden Grund für das Können dieses erklärt bekommen. Dieses Pattern wird mit zwei kleineren (d.s. Theory Elaboration und Teacher Lecturer) kombiniert, die den Transfer des Wissens beinhalten. (Standl, 2013: S. 98-99)
5. ***Assessment*****:** Ein wichtiger Teil der Bildung ist das Messen von Leistungen. Dies erfolgt lt. Standl meist personenzentriert. Als mögliches Problem wird die nicht ausschließliche objektive Beurteilung beschrieben. (Standl, 2013: S. 100-101)
6. ***Social Organization***: Auch soziale Fähigkeiten sind für das Erlernen von Inhalten wichtig. Da der Klassenverband ein sehr komplexes System darstellt, ist das Bereitstellen einer Umgebung, in dem sich alle Beteiligten wohl fühlen, sehr wichtig. (Standl, 2013: S. 102-103)
7. ***Communication*** zwischen den einzelnen Personen ist ebenfalls ein bedeutender Part. Diese sollte vertrauensvoll, respektvoll und authentisch zwischen allen Parteien möglich sein und fördert konstruktive Kommunikation. Eine laufende Verbesserung dieser Fähigkeiten der SchülerInnen ist wünschenswert. (Standl, 2013: S. 104-105)
8. ***Theory Elaboration***: Eine selbständige Erarbeitung neuer Inhalte (Als Quelle werden hier neben vorgegebenen Websites, auch Methoden bereitgestellt, die die SchülerInnen bei dem selbständigen Finden von Quellen unterstützen.) steht bei diesem Pattern im Fokus und kann auch mit einer *Teacher Lecture* kombiniert werden. (Standl, 2013: S. 106-108)

9. Die ***Teacher Lecture*** ist eine weitere Möglichkeit SchülerInnen mit Wissen zu versorgen. Dies geschieht nicht im Rahmen des traditionellen Frontalunterrichts, sondern mittels interaktiver Vermittlung der Inhalte. Die Beobachtung, wie die SchülerInnen diese aufnehmen und im Weiteren auch das Eingehen auf Fragen sind unbedingt erforderlich für das Gelingen. (Standl, 2013: S. 109-111)
10. ***Agree on Assessment Mode***: SchülerInnen erhalten bei der Anwendung dieses Patterns die Möglichkeit den Entscheidungsprozess bzgl. der Beurteilung mitzubestimmen. Dies kann nur für ein Projekt, aber auch für ein gesamtes Schuljahr, erfolgen. (Standl, 2013: S. 112-113)
11. Der ***Learning Contract*** legt zu erlernende Schlüsselfähigkeiten, Inhalte und das Beurteilungsschema fest. Dabei ist folgendes zu beachten: Je größer die Mitbestimmung durch die SchülerInnen erfolgt, desto wichtiger ist es, dass beide Parteien sich bzgl. der Struktur des Lernprozesses einig sind. (Standl, 2013: S. 114-115)
12. Eine Überprüfung der gelernten Inhalte ist auch mittels ***Teacher check*** möglich. Schwerpunkt hierbei ist das Wiederholen von Wissen. Weiters können auch andere Kompetenzen, beispielsweise das Analysieren oder das Kombinieren von Wissen, abgeprüft werden. Als positiv empfinden die SchülerInnen lt. Stand eine digitale Prüfung mit schnellem (d.h. automatisiertem) und elektronischem Feedback. (Standl, 2013: S. 116-117) Dies wird beispielsweise von der Website Kahoot! ermöglicht.
13. Die Lehrkraft gibt bei dem ***Peer check*** lediglich den Rahmen vor, in welchem dieser stattfinden soll, d.h. die SchülerInnen überprüfen sich gegenseitig. Da dieses Vorgehen sehr zeitaufwendig sein kann, sollte es nur nach einer Projektphase erfolgen. (Standl, 2013: S. 118-119)
14. ***Self check***: SchülerInnen reflektieren ihre erarbeiteten Ergebnisse selbstkritisch. Für die erfolgreiche Durchführung dieses Patterns ist eine detaillierte Beschreibung der im Fokus stehenden Bereiche wichtig. (Standl, 2013: S. 120-121)
15. ***Presentation***: Erarbeiten SchülerInnen Projekte, etc., ist es für sie sehr wichtig Anerkennung in Form von einer Präsentation, zu erhalten. Eine im Anschluss erfolgende Veröffentlichung auf einer Klassen- bzw. Schulhomepage oder das darauffolgende Aufhängen des Posters im Klassenraum beschreibt eine zusätzliche Würdigung der geleisteten Arbeit. (Standl, 2013: S. 122-124)

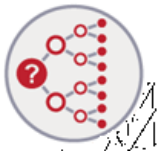
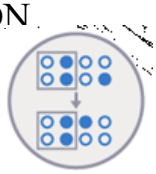
16. ***Individual Work*** beschreibt das alleinige Arbeiten einer Schülerin/eines Schülers und fokussiert sich auf die individuelle Persönlichkeit und den gegebenen Fähigkeiten. Intensive Denkprozesse oder persönliche Reflektionen werden unbedingt benötigt um zufriedenstellende Resultate zu erhalten. (Standl, 2013: S. 125-126)
17. ***Paired Work***, d.h. das Arbeiten zu zweit, ermöglicht eine flexiblere Organisation als *Group Work*. Außerdem ist eine dynamischere soziale Struktur als bei *Individual Work* vorhanden. Eine positive Beeinflussung der sozialen Interaktionen zwischen SchülerInnen untereinander ist möglich. (Standl, 2013: S. 127-128)
18. ***Group Formation***: Es gibt viele Möglichkeiten Gruppen zu formen: anhand von Lernmethoden, zufällig, nach Interesse des Themas oder durch die Wahl der SchülerInnen oder der Lehrkraft. Zu beachten gilt es jedoch, dass Gruppen nicht mehr als vier Personen beinhalten und immer wieder durchgemischt werden sollten um das Gleichbleiben dieser zu verhindern. (Standl, 2013: S. 129-130)
19. ***Group Work*** setzt Managementkompetenzen der Lehrkraft voraus, da diese viele unterschiedliche Prozesse gleichzeitig bewältigen muss. Auch eine gute Vorbereitung und eine entsprechende Förderung der einzelnen Gruppen während der Durchführung dieses Patterns sind unumgänglich. (Standl, 2013: S. 131-132)
20. ***Blended Communication*** beschreibt die Kombination von *Face to Face Communication* und *Online-Communication* zwischen SchülerInnen und der Lehrkraft. Begonnen werden sollte lt. Standl unbedingt mit einem persönlichen Gespräch während der Unterrichtseinheit, da dies von den SchülerInnen als ernster empfunden wird. Nach Abschluss der virtuellen Kommunikation sollte erneut eine persönliche Phase folgen. Diese wird als Abschluss der *Blended Communication* sehr empfohlen. (Standl, 2013: S. 133-134)
21. Für die ***Face to Face Communication*** ist ein gutes Kommunikationsklima inkl. personenzentrierter Fähigkeiten wichtig. Die Bereitstellung dieser ist Aufgabe der Lehrkraft. Anhand aktivem Zuhören kann diese die Förderungsbedürfnisse der SchülerInnen herausfiltern und so bereits während dem Gespräch unterstützend fungieren. Eine Verbesserung des Könnens der SchülerInnen ist somit die Folge. (Standl, 2013: S. 135-136)
22. ***Online-Communication***: Hier ist besondere Aufmerksamkeit von Nöten, da die konstruktive Kommunikation zahlreichen Beschränkungen unterliegt. Für einen

positiven Ablauf dieser ist es erforderlich, dass die Lehrkraft präsent ist und die SchülerInnen relativ rasch Antworten erhalten. (Standl, 2013: S. 137-138)

23. Bei einem ***Classroom Meeting*** steht eine Art personenzentriere Begegnungsgruppe und das Ziel ein vertrauensvolles Klima für Gespräche in der Gruppe zu schaffen, im Fokus. Um dies zu erreichen, ist es lt. Standl wichtig bei den ersten Treffen genug Raum für (fachliche bzw. nicht fachliche) Dialoge zur Verfügung zu stellen. Nach Erreichen eines bestimmten Niveaus werden die Gespräche bedeutender und offener geführt. (Standl, 2013: S. 139-141)
24. Die ***Forum Discussion*** beschreibt, wie Diskussionen in einer virtuellen Lernumgebung vonstattengehen können. Besonders passende Umgebungen sind dafür asynchrone Tools, wie z.B. Onlineforen. Um eine Diskussion zu starten, können von der Lehrkraft Inputs und Postings zur Verfügung gestellt werden. Auch das Beantworten von SchülerInnenkommentaren trägt zum Gelingen der *Forum Dicussion* bei. (Standl, 2013: S. 142-144)

9.5 Materialien für die Sequenz zum Thema „Computational Thinking“

Vokabel inkl. Erklärung: Kärtchen zum Ausschneiden und Folieren

DECOMPOSITION (= Zerlegung) 	ABSTRACTION (= Abstraktion) 
PATTERN RECOGNITION (= Mustererkennung) 	ALGORITHM/S (= Algorithmus/ Algorithmen) 
Komplexe Probleme, Systeme, Prozesse bzw. Daten werden in kleinere Bereiche aufgespalten.	Es wird nach Ähnlichkeiten oder Mustern in den Teilbereichen eines Problems gesucht.
… ermöglicht das Eingehen auf kleinere Details.	Der Kern des Problems wird herausgefiltert.
Nach dem erfolgreichen Anwenden dieser Methode habe ich eine generelle Idee bzgl. meines Problems bzw. meiner Aufgabe.	Zusammenhänge zwischen bereits gelösten Bereichen von Problemen und den ungelösten Bereichen des neuen Problems werden hergestellt.
Es erfolgt das schrittweise Erstellen einer Lösung.	Unwichtige Informationen werden ignoriert.
Es werden Regeln für das Lösen eines Problems bzw. einer Aufgabe festgelegt.	

Quellen: (BBC, 2016a) (BBC, 2016) (BBC, 2017) (BBC, 2017a/b/c) (Google, 2016)

Vokabel inkl. Erklärung: Lösung

DECOMPOSITION (= Zerlegung)	Komplexe Probleme, Systeme, Prozesse bzw. Daten werden in kleinere Bereiche aufgespalten.
	… ermöglicht das Eingehen auf kleinere Details.
	Ohne dieser Methode wäre es viel komplizierter und aufwendiger Probleme bzw. Aufgaben zu lösen.
PATTERN RECOGNITION (= Mustererkennung)	Es wird nach Ähnlichkeiten oder Mustern in den Teilbereichen eines Problems gesucht.
	Zusammenhänge zwischen bereits gelösten Bereichen von Problemen und den ungelösten Bereichen des neuen Problems werden hergestellt.
ABSTRACTION (= Abstraktion)	Der Kern des Problems wird herausgefiltert.
	Unwichtige Informationen werden ignoriert.
	Nach dem erfolgreichen Anwenden dieser Methode habe ich eine generelle Idee bzgl. meines Problems bzw. meiner Aufgabe.
ALGORITHM/S (= Algorithmus/Algorithmen)	Es erfolgt das schrittweise Erstellen einer Lösung.
	Es werden Regeln für das Lösen eines Problems bzw. Aufgabe festgelegt.

Quellen: (BBC, 2016a) (BBC, 2016) (BBC, 2017) (BBC, 2017a/b/c) (Google, 2016)

Computational Thinking

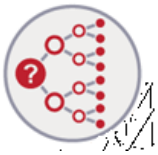
„… is the thought process involved in formulating a problem and expressing its solution(s) in such a way a computer – human or machine – can effectively carry on.“ (Wing, 2014)

CT beschreibt also einen Denkprozess, welcher eine Formulierung des Problems inklusive einer Lösung als Ergebnis hat. Dabei erfolgt die Annahme, dass der mentale Vorgang von einer Person und keiner Maschine durchgeführt wird. Weiters ist zu beachten, dass ein Computer nicht zwingend eine Maschine, sondern auch einen Menschen darstellen kann. (Wing, 2014)

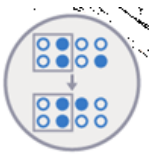
Die vier häufigsten Schlüsseltechniken bzw. Konzepte von CT:

(BBC, 2017a/b/c); (Google, 2016)

- *Decomposition (= Zerlegung)*
  - Komplexe Probleme, Systeme, Prozesse bzw. Daten werden in kleinere Bereiche aufgespalten.
  - … ermöglicht das Eingehen auf kleinere Details.
  - Ohne dieser Methode wäre es viel komplizierter und aufwendiger Probleme bzw. Aufgaben zu lösen.



- *Pattern Recognition (= Mustererkennung)*
  - Es wird nach Ähnlichkeiten oder Mustern in den Teilbereichen eines Problems gesucht.
  - Zusammenhänge zwischen bereits gelösten Bereichen von Problemen und den ungelösten Bereichen des neuen Problems werden hergestellt.



- *Abstraction (= Abstraktion)*
  - Der Kern des Problems wird herausgefiltert.
  - Unwichtige Informationen werden ignoriert.
  - Nach dem erfolgreichen Anwenden dieser Methode habe ich eine generelle Idee bzgl. meines Problems bzw. meiner Aufgabe.

- *Algorithm/s (= Algorithmus/Algorithmen)*
  - Es erfolgt das schrittweise Erstellen einer Lösung.
  - Es werden Regeln für das Lösen eines Problems bzw. einer Aufgabe festgelegt.

Literaturverzeichnis

BBC. (2017a) *Decomposition*. Zugriff am 03. Februar 2017 unter http://www.bbc.co.uk/education/guides/zqqfyrd/revision

BBC. (2017b) *Pattern recognition*. Zugriff am 03. Februar 2017 unter http://www.bbc.co.uk/education/guides/zxxbgk7/revision

BBC. (2017c) *Abstraction*. Zugriff am 03. Februar 2017 unter http://www.bbc.co.uk/education/guides/zttrcdm/revision

Google. (2016) *Computational Thinking for Educators*. Zugriff am 13. Juli 2016 unter https://computationalthinkingcourse.withgoogle.com/unit?lesson=8&unit=1

Wing, J. M. (10. Jänner 2014). *Computational Thinking Benefits Society*. Zugriff am 08. Juli 2016 unter http://socialissues.cs.toronto.edu/index.html%3Fp=279.html

Beispiel für die gemeinsame Erörterung der Teilbereiche:
Umrechnen von römischen in arabische Zahlen

Aufgabe: *Die römische Zahl „DLVII" soll in eine arabische Zahl umgewandelt werden.*

1. *Decomposition: Aufspalten des komplexen Problems*
   Jedes Symbol einer römischen Zahl hat einen bestimmten Wert zugewiesen:
   I = 1; V = 5; X = 10; L = 50; C = 100; D = 500; M = 1000; etc.
   Mögliche Fragen an die SchülerInnen:
   - Wie könnte diese Aufgabe zielführend angegangen werden?
   - Was ist die erste Überlegung (noch bevor wir etwas berechnen), wenn ihr diese Zahl seht?

2. *Pattern Recognition: Suche nach Ähnlichkeiten*
   Hier wird die zu umrechnende römische Zahl hergenommen und überprüft welchen Wert die jeweiligen Symbole in den arabischen Zahlen darstellen.
   D = 500; L = 50; V = 5; I = 1
   Mögliche Überleitung/Fragen an die SchülerInnen:
   - Legen wir nochmals den Blick auf unsere Aufstellung, wo wir festgehalten haben welchen Wert eine römische Zahl im arabischen Zahlensystem hat. Was wäre nun unser nächster Schritt?

3. *Abstraction: der Fokus wird auf die wirklich wichtigen Dinge gelegt*
   Um das endgültige Ergebnis zu erhalten, müssen nun die „übersetzten" Symbole miteinander addiert werden.
   500 + 50 + 5 + 1 + 1 = 557
   Mögliche Überleitung/Fragen an die SchülerInnen:
   - Wir haben nun die jeweiligen „übersetzten" Werte der römischen Zahlen aufgeschrieben. Welcher Schritt fehlt uns noch um an das Ergebnis zu gelangen?

4. *Algorithm/s: schrittweises Erstellen von Lösungen/Regeln für das Lösen der Aufgabe*
   Ein möglicher Vorgang eine römische in eine arabische Zahl umzurechnen ist wie folgt:
   a. Notieren der römischen Symbole inkl. deren arabischen Wert.
   b. Hernehmen der zu umrechnenden römischen Zahl und (erneutes) Zuweisen der arabischen Zahlen zu den Symbolen der römischen Zahl.
   c. Addieren der arabischen Werte.

d. Notieren des endgültigen Ergebnisses.

Mögliche Überleitung/Fragen an die SchülerInnen:

- Als letzten Schritt halten wir schrittweise den Ablauf fest, wie wir zum Ergebnis gekommen sind. Was war unser erster Schritt, den wir gemeinsam gemacht haben?
- Der nächste Schritt war welcher?

Erweiterung des Beispiels: *Die römische Zahl „XL" soll in eine arabische Zahl umgerechnet werden.*

Mögliche Überleitung/Fragen an die SchülerInnen:

- Kann der soeben entwickelte Algorithmus auf alle Zahlen angewendet werden?
- Erhalten wir bei der Zahl „XL" die korrekte Lösung?
- Welche Schritte können wir ohne Probleme durchlaufen?
- Ab welchem Schritt ist dies nicht mehr möglich? Weshalb?
- Welche Konzepte von CT müssen wir nochmals durchlaufen um unsere Lösung zu korrigieren?

1. *Decomposition* und
2. *Pattern Recognition* ist ident mit der bereits zuvor notierten Erläuterung und kann somit - nach einem kurzen Hinweis auf diese Tatsache - übersprungen werden.
3. *Abstraction: der Fokus wird auf die wirklich wichtigen Dinge gelegt*
   Hier muss die Subtraktionsregel für einige „Sonderfälle" eingeführt werden:
   „I", „X" bzw. „C" darf nur vor den beiden jeweils größeren Symbolen Wertvermindernd wirken, d.h. „I" darf nur vor „V" und „X" stehen (***1. Regel***), „X" darf nur vor „L" und „C" stehen (***2. Regel***) und „C" darf nur vor „D" und „M" stehen (***3. Regel***).
   Erst anschließend können die „übersetzten" Symbole miteinander addiert werden.
   Mögliche Überleitung/Fragen an die SchülerInnen:
   - Wie kann die Zahl 9 mit der bisherigen Regel dargestellt werden? → VIIII
   - Wie kann die Zahl 9 noch dargestellt werden? → IX
   - Kann die Zahl 49 mit der Zeichenkette „IL" dargestellt werden? → Nein.

- Wie würde die Zahlt 49 korrekt dargestellt aussehen? → XLIX
- Darf das Symbol „I“ auch vor dem Symbol „L“ oder „C“ stehen?
- Welche Regel kann daraus abgeleitet werden? → ***1. Regel***
- Darf das Symbol „X“ auch vor dem Symbol „D“ oder „M“ stehen?
- Ausschließlich wovor kann das Symbol „C“ stehen, wenn es als Verminderung eines anderen Symbols fungieren soll?

Weitere Beispiele für das Erarbeiten der Regeln:

99	= LXXXXVIIII	= XCIX	→ falsch wäre IC	→ 1. Regel
490	= CCCCLXXXX	= CDXL	→ falsch wäre XD	→ 2. Regel
990	= DCCCCLXXXX	= DCDXC	→ falsch wäre XM	→ 2. Regel
1904	= MDCCCCIIII	= MCMLIV		→ 3. Regel

4. *Algorithm/s: schrittweises Erstellen von Lösungen/Regeln für das Lösen der Aufgabe*
   Ein möglicher Vorgang eine römische in eine arabische Zahl umzurechnen ist wie folgt:
   a. Notieren der römischen Symbole inkl. deren arabischen Wert.
   b. Hernehmen der zu umrechnenden römischen Zahl und (erneutes) Zuweisen der arabischen Zahlen zu den Symbolen der römischen Zahl.
   c. NEU: Überprüfen, ob für die gegebene Zahl eine Regel der Subtraktion zutrifft, d.h. es muss überprüft werden, ob eine kleinere vor einer größeren Zahl steht.
   d. NEU: Wenn der vorherige Schritt zutrifft, dann soll die betreffende Subtraktionsregel angewandt werden. Ansonsten kann gleich mit Schritt e. fortgefahren werden.
   e. Addieren der arabischen Werte.
   f. Notieren des endgültigen Ergebnisses.

   Mögliche Überleitung/Fragen an die SchülerInnen:
   - Gehen wir nochmals zu unserem bereits erarbeiteten Ablauf zurück. Wenn ihr den Schritt für Schritt durchgeht und überlegt, welche Schritte fehlen noch?
   - Wo genau müssen diese noch eingebaut werden?

Anleitungsloses Spiel – Ablauf

code.org (2017). *Computational Thinking*. Zugriff am 28. Oktober 2017 unter https://code.org/curriculum/course3/1/Teacher

1. Seht euch die Unterlagen (vor allem die „***Phrasen von SpielerInnen***“) an und überlegt, wie andere dieses Spiel gespielt haben könnten.
   Verwendet für Punkt 2 bis Punkt 4 die „***Phrasen von SpielerInnen***“.
2. *Pattern Recognition*: Kreist nun alle gleichen Satzteile ein, die bei jedem Spieler vorhanden sind.
3. *Abstraction*: Unterstreicht die Satzteile, die sich bei den einzelnen Spielern unterscheiden.
4. *Pattern Recognition/Abstraction*: Notiert die eingeringelten Satzteile und lasst anstelle der unterstrichenen Satzteile eine etwas größere leere Stelle.
5. *Algorithm*: Haltet nun eine Liste mit Anweisungen fest, wie dieses Spiel gespielt werden soll. Diese basiert auf Basis der Erfahrungen, die ihr soeben gemacht habt.
   Gratuliere! Ihr habt nun gemeinsam einen Ablauf für das Spiel entwickelt.
6. Gemeinsam sollen nun unter Verwendung des selbst entwickelten Algorithmus das Spiel gespielt werden.
   Sollte dafür eine Anpassung des Algorithmus aus Punkt 5 notwendig sein, haltet dies inkl. der zusätzlich notwendigen Punkte fest.

Phrasen von SpielerInnen

SpielerIn 1:

Ich wähle den Löwen und habe eine sechs gewürfelt, anschließend eine vier und dann eine zwei. Das bedeutet, ich muss einen schwarzen Cupcake auf den Po meines Löwen zeichnen.

SpielerIn 2:

Ich wähle den Affen und habe eine drei gewürfelt, anschließend eine zwei und dann eine eins. Das bedeutet, ich muss eine gelbe Ananas auf den Kopf meines Affens zeichnen.

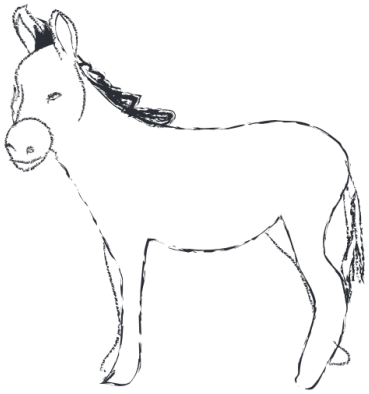
SpielerIn 3:

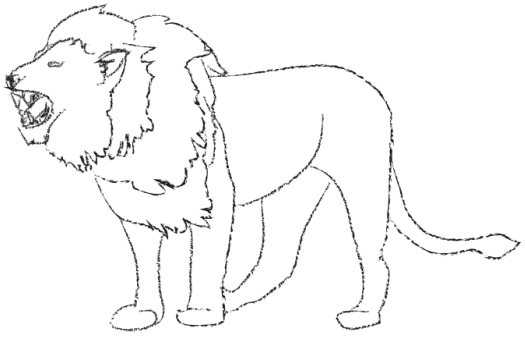
Ich wähle den Hund und habe eine fünf gewürfelt, anschließend eine drei und dann eine fünf. Das bedeutet, ich muss einen pinken Fisch auf die Nase meines Hundes zeichnen.

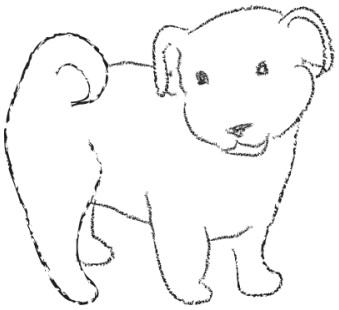
Zu Punkt 4:

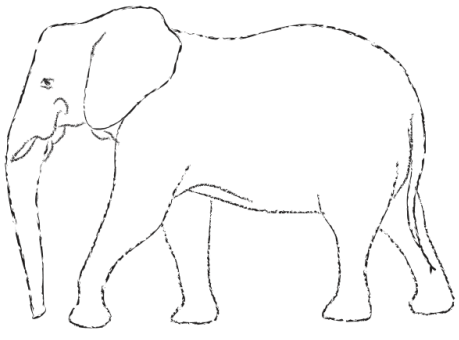
Ich wähle

Farbe	**Gegenstand**	**Körperteil**
1. rot	1. Handy	1. Kopf
2. blau	2. Ananas	2. Po
3. gelb	3. Fisch	3. Hand/Fuß/Pfote/Huf
4. grün	4. Cupcake	4. Bauch
5. pink	5. Tentakel	5. Nase
6. schwarz	6. Schleife	6. Rücken



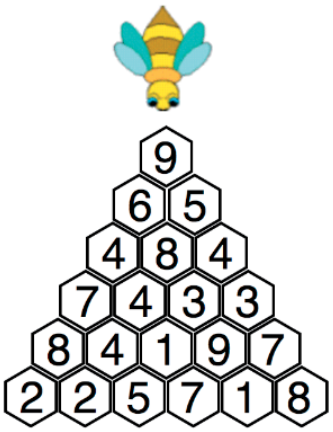






Aufgaben – Biber der Informatik

Effiziente Biene



Die Biene Summ fliegt über einen interessanten Bibergarten: Die Blumenbeete sind sechseckig. Sie grenzen aneinander und sind insgesamt in einem Dreieck angeordnet.

Für jedes Sechseck kann Summ sehen, wie viele Milligramm Nektar dort zu holen sind. Sie beginnt an der Spitze des Dreiecks, wo sie heute 9 Milligramm sammeln kann. Summ ist in Eile und will deshalb von jedem Beet nur zu einem der zwei in Flugrichtung angrenzenden Beete weiterfliegen:

Wie viele Milligramm Nektar kann Summ unter dieser Einschränkung heute höchstens einsammeln? Begründe deine Lösung!

Welche Konzepte und Methoden von CT musstest du für die Lösung der Aufgabe anwenden? Begründe!

Verbindung zur Informatik:
Viele Computerprogramme suchen nach der optimalen Lösung eines Problems. Dazu suchen sie, wie Summ im Bibergarten, unter verschiedenen Möglichkeiten die beste aus. Solches Suchen effizient, also mit möglichst wenig Aufwand an Prozessorzeit und Speicherplatz durchzuführen, ist ein wichtiges Thema in der Informatik.

Literaturverzeichnis

Pohl, W. & Hein, H.-W. & Bastisch, M. (2009, S. 18). *Informatik-Bieber* 2009. Zugriff am 28. Oktober 2017 unter http://www.ocg.at/sites/ocg.at/files/medien/pdfs/BiberAufgaben2009.pdf

Codierung von Buchstaben

Der Biber möchte Buchstaben nur mit den zwei Ziffern “0” und “1” darstellen. Folgende Buchstaben hat er schon „übersetzt“:

- R = “1”
- S = “011”
- T = “010”

Der Code „01011011“ steht beispielsweise für die Zeichenkette „TRRS“.

Nun möchte der Biber seinem System einen weiteren Buchstaben “U” hinzufügen. Dafür braucht er eine Codierung, die keine Mehrdeutigkeiten zulässt. D.h. er kann z.B. nicht „11“ nehmen, da sonst „RR“ und „U“ mit dem gleichen Code dargestellt würden.

Auf welche Weise kann Biber den Buchstaben “U” eindeutig codieren?

A) U = “101”

B) U = “110”

C) U = “01110”

D) U = “00”

Begründe deine Lösung!

_____

_____

Welche Konzepte und Methoden von CT musstest du für die Lösung der Aufgabe anwenden? Begründe!

_____

_____

_____

_____

Verbindung zur Informatik:
Information kann mehrdeutig sein. So ist das Leben. Beim Codieren und Decodieren ist Mehrdeutigkeit aber eine höchst unerwünschte Eigenschaft. Sie sicher auszuschließen, ist eines der Themen im Informatik-Bereich Codierungstheorie.

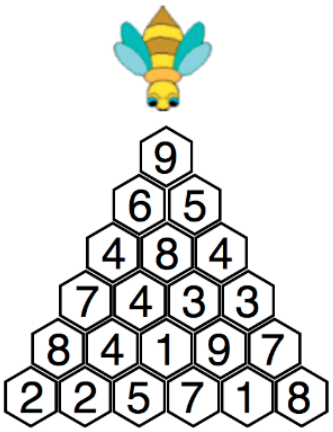
Literaturverzeichnis

Goethe Gymnasium Astgasse. (2016). *Biber der Informatik Logo*. Zugriff am 28. Oktober unter https://www.astgasse.net/cms1/images/stories/1415/Logo-Informatik-Biber.jpg

Pohl, W. & Hein, H.-W. & Bastisch, M. (2009, S. 13). *Informatik-Bieber* 2009. Zugriff am 28. Oktober 2017 unter http://www.ocg.at/sites/ocg.at/files/medien/pdfs/BiberAufgaben2009.pdf

*Aufgaben - Biber der Informatik - **Lösung***

Effiziente Biene

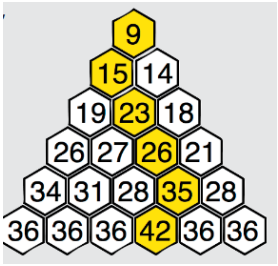


Die Biene Summ fliegt über einen interessanten Bibergarten:
Die Blumenbeete sind sechseckig. Sie grenzen aneinander und sind insgesamt in einem Dreieck angeordnet.

Für jedes Sechseck kann Summ sehen, wie viele Milligramm Nektar dort zu holen sind. Sie beginnt an der Spitze des Dreiecks, wo sie heute 9 Milligramm sammeln kann. Summ ist in Eile und will deshalb von jedem Beet nur zu einem der zwei in Flugrichtung angrenzenden Beete weiterfliegen:

Wie viele Milligramm Nektar kann Summ unter dieser Einschränkung heute höchstens einsammeln? Begründe deine Lösung!

42 ist die richtige Antwort.
Wenn man von oben nach unten Zeile für Zeile ausrechnet, wie viel Honig man maximal bis dahin sammeln kann, dann ergibt sich dieses Bild. Summ sammelt so 42 Milligramm Honig ein.



Verbindung zur Informatik:
Viele Computerprogramme suchen nach der optimalen Lösung eines Problems. Dazu suchen sie, wie Summ im Bibergarten, unter verschiedenen Möglichkeiten die beste aus. Solches Suchen effizient, also mit möglichst wenig Aufwand an Prozessorzeit und Speicherplatz durchzuführen, ist ein wichtiges Thema in der Informatik.

Literaturverzeichnis

Pohl, W. & Hein, H.-W. & Bastisch, M. (2009, S. 18). *Informatik-Bieber* 2009. Zugriff am 28. Oktober 2017 unter http://www.ocg.at/sites/ocg.at/files/medien/pdfs/BiberAufgaben2009.pdf

Codierung von Buchstaben

Der Biber möchte Buchstaben nur mit den zwei Ziffern “0” und “1” darstellen. Folgende Buchstaben hat er schon „übersetzt“:

- R = “1”
- S = “011”
- T = “010”

Der Code „01011011“ steht beispielsweise für die Zeichenkette „TRRS“.

Nun möchte der Biber seinem System einen weiteren Buchstaben “U” hinzufügen. Dafür braucht er eine Codierung, die keine Mehrdeutigkeiten zulässt. D.h. er kann z.B. nicht „11“ nehmen, da sonst „RR“ und „U“ mit dem gleichen Code dargestellt würden.

Auf welche Weise kann Biber den Buchstaben “U” eindeutig codieren?

A) U = “101”

B) U = “110”

C) U = “01110”

D) U = “00”

Antwort D ist richtig:

- A geht nicht, weil z. B. „1011“ dann „UR“ oder „RS“ bedeuten könnte.
- B geht nicht, weil z. B. „11011“ dann „URR“ oder „RRS“ bedeuten könnte.
- C geht nicht, weil z. B. „0111011“ sowohl „URR“, als auch „SRS“ bedeuten könnte.

Verbindung zur Informatik

Information kann mehrdeutig sein. So ist das Leben. Beim Codieren und Decodieren ist Mehrdeutigkeit aber eine höchst unerwünschte Eigenschaft. Sie sicher auszuschließen, ist eines der Themen im Informatik-Bereich Codierungstheorie.

Literaturverzeichnis

Goethe Gymnasium Astgasse. (2016). *Biber der Informatik Logo*. Zugriff am 28. Oktober unter https://www.astgasse.net/cms1/images/stories/1415/Logo-Informatik-Biber.jpg

Pohl, W. & Hein, H.-W. & Bastisch, M. (2009, S. 13). *Informatik-Bieber* 2009. Zugriff am 28. Oktober 2017 unter http://www.ocg.at/sites/ocg.at/files/medien/pdfs/BiberAufgaben2009.pdf

Name: _____ Klasse:_____

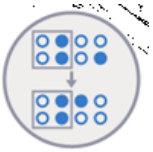
Schriftliche Stundenwiederholung

Fülle die fehlenden Inhalte aus:

_____ Thinking

_____ = Zerlegung

_____ = _____



Abstraction = Abstraktion

_____ = Algorithmus

Ordne folgende Aussagen den Schlüsseltechniken zu:

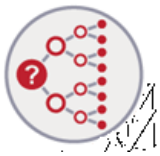
- _____: Es wird nach Ähnlichkeiten oder Mustern in den Teilbereichen eines Problems gesucht.
- _____: Es erfolgt das schrittweise Erstellen einer Lösung.
- _____: Unwichtige Informationen werden ignoriert.
- _____: Komplexe Probleme, Systeme, Prozesse bzw. Daten werden in kleinere Bereiche aufgespalten.

Schriftliche Stundenwiederholung - Lösung

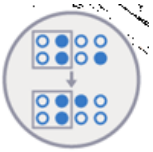
Fülle die fehlenden Inhalte aus:

Computational Thinking

Decomposition
= Zerlegung



Pattern Recognition
= Mustererkennung



Abstraction
= Abstraktion

Algorithmus
= Algorithmus

Ordne folgende Aussagen den Schlüsseltechniken zu:

- Pattern Recognition: Es wird nach Ähnlichkeiten oder Mustern in den Teilbereichen eines Problems gesucht.
- Algorithm: Es erfolgt das schrittweise Erstellen einer Lösung.
- Abstraction: Unwichtige Informationen werden ignoriert.
- Decomposition: Komplexe Probleme, Systeme, Prozesse bzw. Daten werden in kleinere Bereiche aufgespalten.

9.6 Materialien für die Sequenz zum Thema „Algorithmus“

Notieren der heutigen Morgenroutine

- *Decomposition*: Die SchülerInnen sollen jede größere Handlung in ihre einzelnen Teile aufspalten, z.B. frühstücken: auf den Sessel beim Tisch setzen, Schüssel nehmen, Milch nehmen, Milch in die Schüssel gießen, Cornflakes nehmen, Cornflakes öffnen, Cornflakes in die Schüssel mit Milch geben, …
- *Abstraction*: Bei dem Notieren soll der Fokus auf die essentiellen Dinge gelegt werden. Beispielsweise ist es für die Erstellung eines Ablaufs nicht wichtig, was am Frühstückstisch mit den Familienmitgliedern gesprochen wird.
- *Algorithm*: Das schrittweise Notieren der heutigen Morgenroutine ergibt einen Algorithmus, welcher in dieser Form auch von anderen Personen durchgeführt werden kann.

Theorie „Algorithmus“

Ein Algorithmus besteht aus einer Reihe von detaillierten Anweisungen, die nacheinander ausgeführt werden müssen. Das dabei verfolgte Ziel ist das Lösen einer Aufgabe bzw. eines Problems. (Schanze, 2017)

Eigenschaften eines Algorithmus: (Piskernik, 2016: S. 1-2)

- *Determinismus*: Ein Algorithmus besteht aus einzelnen Schritten, die nacheinander ausgeführt werden. Es ist immer klar, welcher Schritt der nächste in der Ausführung ist.
- *Effektivität*: Jeder Schritt besteht aus einer möglichst einfachen und offensichtlichen Grundaktion.
- *Verständlichkeit*: Die Schritte müssen genau genug angegeben werden, um sie präzise ausführen zu können.
- Fast immer Voraussetzung – *Endlichkeit*:
  Ein Algorithmus muss nach endlich vielen Schritten terminieren, d.h. sich beenden. Ein Algorithmus, der potenziell unendlich lange läuft ist das Betriebssystem.

Gemeinsame Überlegung: Definition und Eigenschaften eines Algorithmus - Fragen an die SchülerInnen:

- *Abstraction*: Was ist euch bei dem Notieren eurer heutigen Morgenroutine aufgefallen?
- *Decomposition*: Wie wurden die Anweisungen formuliert? Mit Überbegriffen (z.B. frühstücken) oder eher Schritt für Schritt (z.B. Schüssel nehmen, Milch nehmen, …).
- *Abstraction*: Können die notierten Anweisungen auch in einer anderen Reihenfolge stattfinden?
- *Pattern Recognition*: Gibt es vielleicht Ähnlichkeiten zu der Morgenroutine von gestern?
- *Abstraction*: Welches Ziel habt ihr verfolgt?
- *Algorithm*: Welche Eigenschaften eines Algorithmus können wir aus den bisherigen Überlegungen ableiten?

Beispiele für einen Algorithmus im Alltag können sein:
Geld beheben bei einem Bankomat, Kochen eines Gerichts, Einkaufen in einem Supermarkt, Autofahren, Führen eines Telefongesprächs, Sortieren von Zahlen, Wurzelziehen, …

Literaturverzeichnis

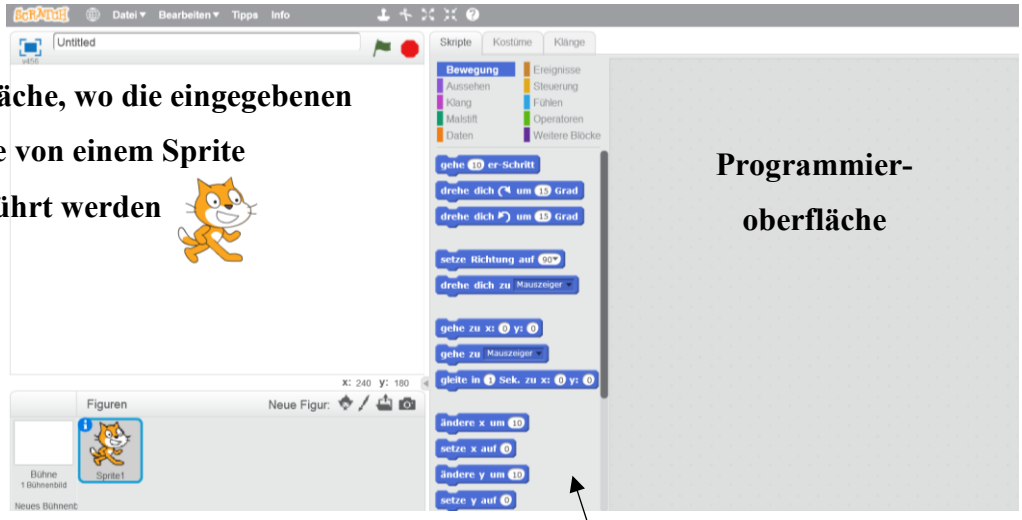
Piskernik, M. (WS 2016). *Programmierpraktikum*. Zugriff am 31. Oktober 2017 unter https://moodle.univie.ac.at/pluginfile.php/2376047/mod_resource/content/17/Skriptum%20Python.pdf

Schanze, R. (07. Juni 2017). *Was ist ein Algorithmus? – Einfach erklärt*. Zugriff am 31. Oktober 2017 unter http://www.giga.de/extra/ratgeber/specials/was-ist-ein-algorithmus-einfach-erklaert/

Programmieren mit Scratch

Auf der Homepage https://scratch.mit.edu/ kannst du in der Menüleiste auf „***Entwickeln***“ klicken und so zu deiner Programmieroberfläche kommen.

Oberfläche, wo die eingegebenen Befehle von einem Sprite ausgeführt werden



Programmier-oberfläche

Auswahl an Befehlen mit Gliederung in versch. Bereiche

Aufgabe 1: Dance Party

Wähle zwei Sprites aus, die jeweils mehrere Kostüme besitzen und ändere den Hintergrund so, dass dieser zu einer Dance Party passt.

Ablauf:

- Die Sprites sollen sich zuerst gegenseitig begrüßen.
  Beachte: Die Kommunikation soll so realistisch wie möglich ablaufen, d.h. gleichzeitiges Sprechen soll vermieden werden.
- Nun lasse deine Figuren tanzen: Die zwei gewählten Sprites sollen sich ein wenig nach rechts und wieder ein wenig nach links bewegen und dabei ihr „Kostüm“ wechseln.
  Hinweis: Dies soll ca. 10 Mal ablaufen.
- Speichere dein Ergebnis ab: Aufgabe1_Vorname

Bonus:

- Tanzen ohne Musik ist nicht lustig. Starte nach der Begrüßung die Musik.
- Die Sprites sollen sich immer an der gleichen Stelle begrüßen. Dies muss von dir voreingestellt werden in deinem Programm.

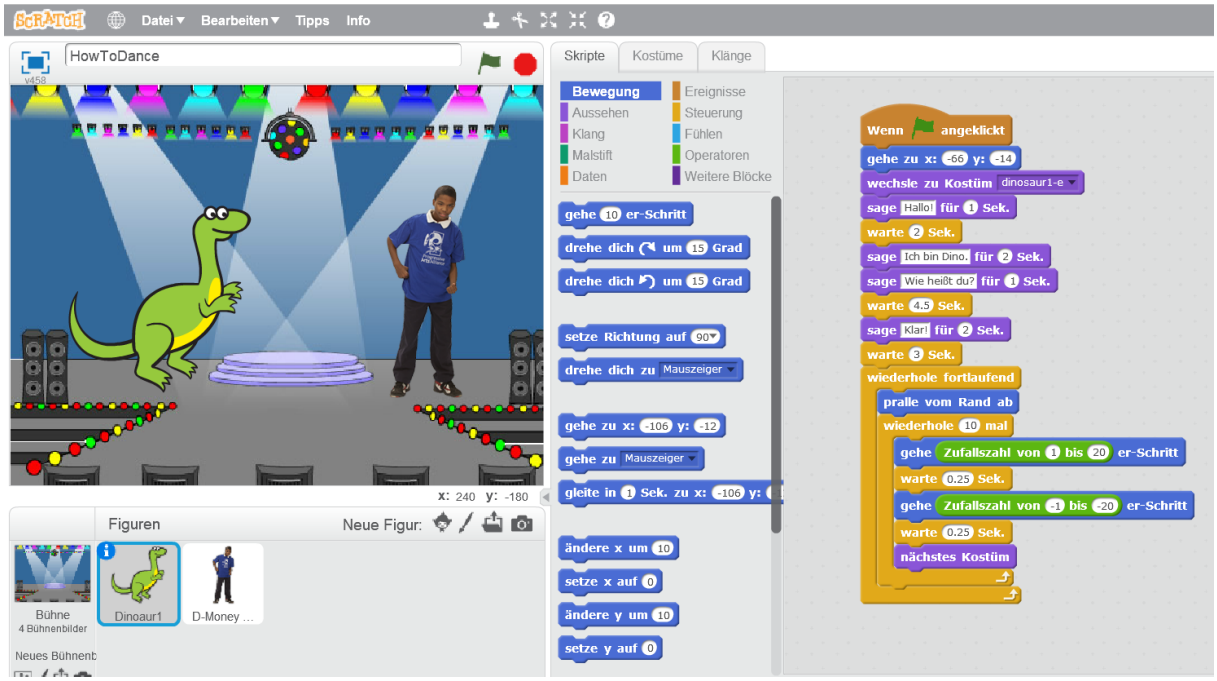
Aufgabe 2:

Überlege dir selbst ein Spiel, dass du mit deinen gelernten Fähigkeiten programmieren kannst. Notiere hier kurz deine Idee und mache auch eine Skizze, wie deine Spieloberfläche aussehen soll.

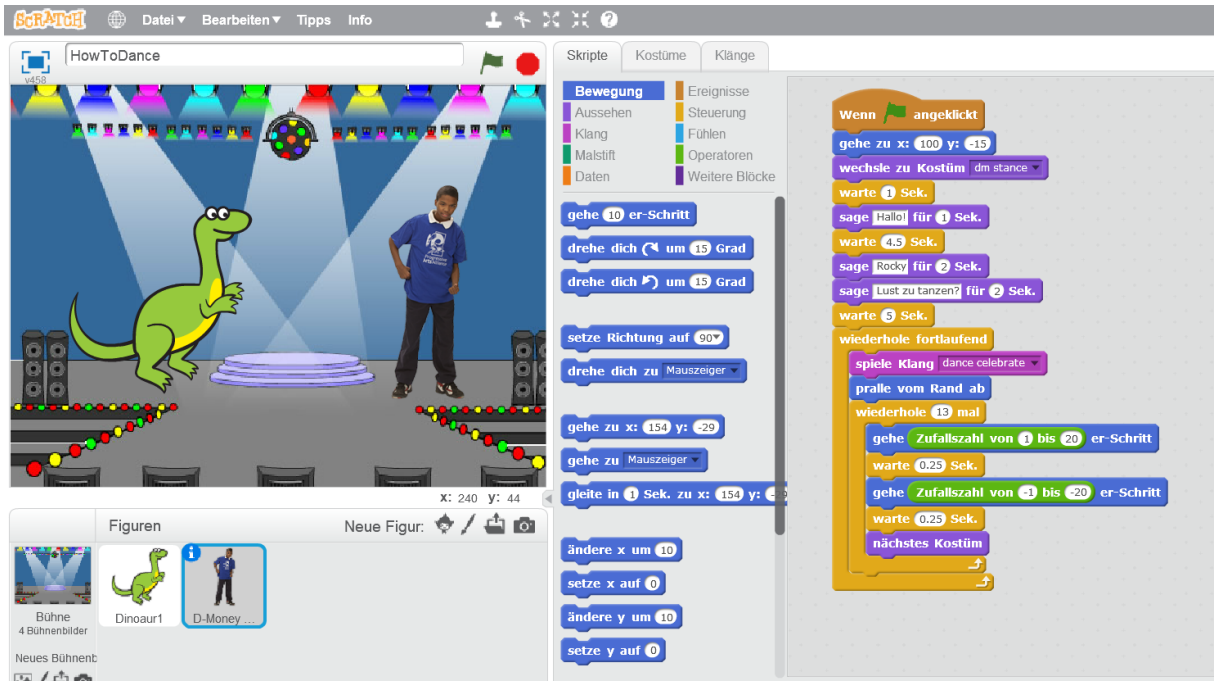
Speichere dein Ergebnis ab: Aufgabe2_Vorname

Lösung – Aufgabe 1:

Sprite: Dinosaur1



Sprite: D-Money Hip-Hop



Informatik Name:_____ Klasse:_____

Schriftliche Stundenwiederholung

Definition „Algorithmus“:

Beschreibe zwei der vier Eigenschaften, die ein Algorithmus erfüllen muss!
Notiere auch die jeweiligen Fachtermini!

Schriftliche Stundenwiederholung - Lösung

Definition „Algorithmus“:
Ein Algorithmus besteht aus einer Reihe von detaillierten Anweisungen, die nacheinander ausgeführt werden müssen. Das dabei verfolgte Ziel ist das Lösen einer Aufgabe bzw. eines Problems. (Schanze, 2017)

Beschreibe zwei der vier Eigenschaften, die ein Algorithmus erfüllen muss!
Notiere auch die jeweiligen Fachtermini! (Piskernik, 2016: S. 1-2)

- *Determinismus*: Ein Algorithmus besteht aus einzelnen Schritten, die nacheinander ausgeführt werden. Es ist immer klar, welcher Schritt der nächste in der Ausführung ist.
- *Effektivität*: Jeder Schritt besteht aus einer möglichst einfachen und offensichtlichen Grundaktion.
- *Verständlichkeit*: Die Schritte müssen genau genug angegeben werden, um sie präzise ausführen zu können.
- Fast immer Voraussetzung – *Endlichkeit*: Ein Algorithmus muss nach endlich vielen Schritten terminieren, d.h. sich beenden.

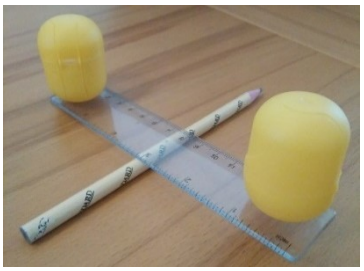
Literaturverzeichnis

Piskernik, M. (WS 2016). *Programmierpraktikum*. Zugriff am 31. Oktober 2017 unter https://moodle.univie.ac.at/pluginfile.php/2376047/mod_resource/content/17/Skriptum%20Python.pdf

Schanze, R. (07. Juni 2017). *Was ist ein Algorithmus? – Einfach erklärt*. Zugriff am 31. Oktober 2017 unter http://www.giga.de/extra/ratgeber/specials/was-ist-ein-algorithmus-einfach-erklaert/

9.7 Materialien für die Sequenz zum Thema „Algorithmen – Sortierverfahren – Teil 1“

Entwicklung eines Algorithmus zum Sortieren der unterschiedlich schweren Döschen



- *Decomposition*: Nach dem Erhalt der zu sortierenden Döschen müssen sich die SchülerInnen zuerst einig werden über den Vorgang, wie sie feststellen können, ob ein Döschen leichter ist als ein anderes. Dies kann beispielsweise mit der Linealwippe erfolgen.
- *Pattern Recognition*: Bei der Entwicklung der Regeln für die Erstellung des Algorithmus muss immer wieder überprüft werden, ob bereits erarbeitete Regeln erneut verwendet werden können.
- *Abstraction*: Der Fokus wird neben dem Vergleichen der Döschen bzgl. des Gewichts, auch auf das Entwickeln eines Algorithmus gelegt.
- *Algorithm*: Die SchülerInnen erarbeiten Regeln, die in einer bestimmten Anzahl hintereinander angewendet werden müssen um das Sortieren der Döschen zu gewähren.

Sortierverfahren „Selection Sort“

Beschreibe die Vorgangsweise beim Selection Sort:

Löse folgende Beispiele vollständig. Verwende zur deutlicheren Veranschaulichung mind. zwei verschiedene Farben:

3	2	7	5	1	4	6

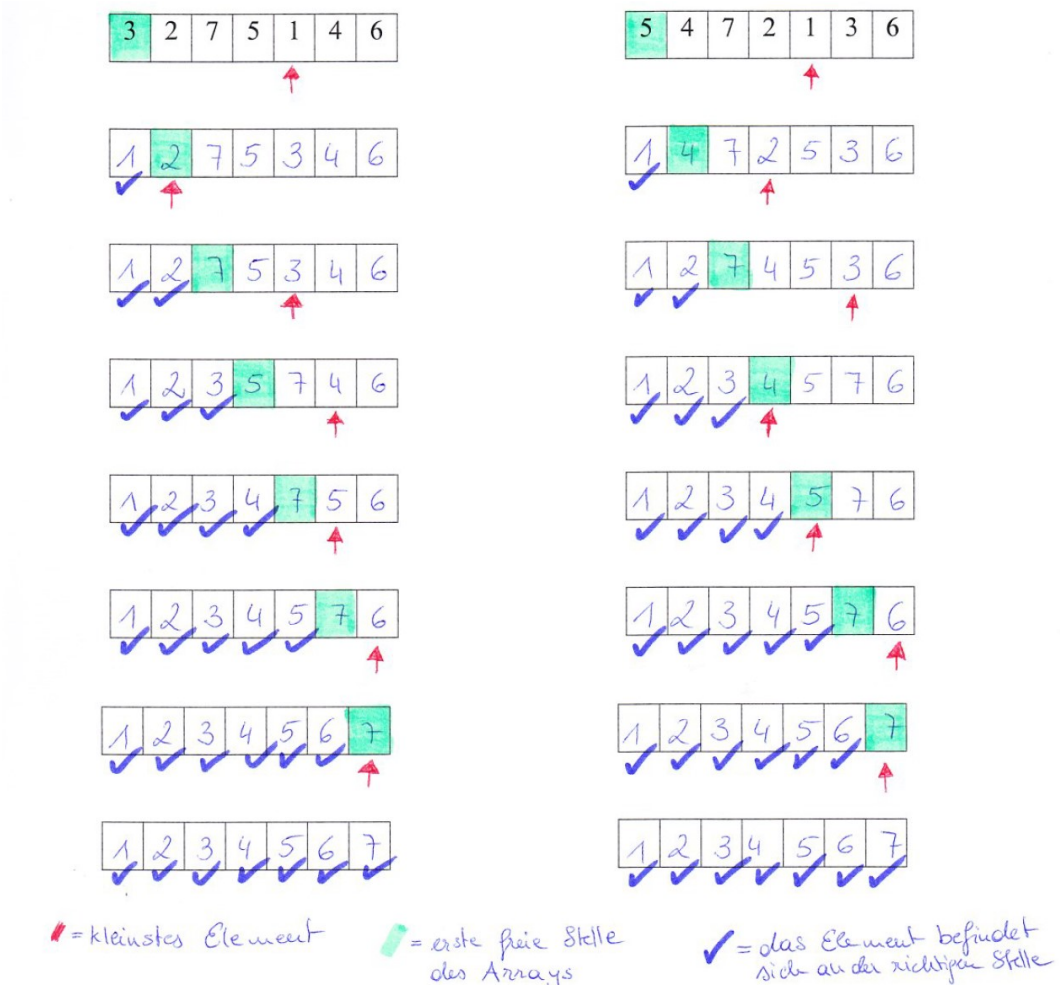
5	4	7	2	1	3	6

Sortierverfahren „Selection Sort“ - Lösung

Beschreibe die Vorgangsweise beim Selection Sort:

Er folgt dem Leitsatz „Sortieren durch Auswählen“. D.h. das kleinste Element der unsortierten Menge wird mit dem Element an der ersten freien Stelle getauscht. Dies erfolgt so lange, bis keine unsortierte Menge mehr vorhanden ist. (Helmich, 2017)

Löse folgende Beispiele vollständig. Verwende zur deutlicheren Veranschaulichung mind. zwei verschiedene Farben:



Literaturverzeichnis

Helmich, U. (26. Februar 2017). *8.4 Der Selection Sort*. Zugriff am 02. November 2017 unter http://www.u-helmich.de/inf/BlueJ/kurs11/Folge08/folge08-4.html

Sortierverfahren „Bubble Sort“

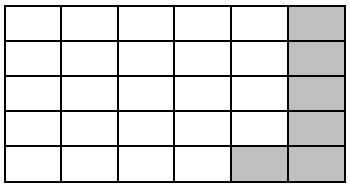
Beschreibe die Vorgangsweise beim Bubble Sort:

Löse folgendes Beispiel vollständig. Markiere jeweils die zu vergleichenden Elemente:

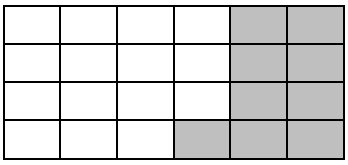
1. Durchlauf:

5	2	1	7	6	3

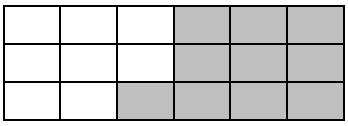
2. Durchlauf:



3. Durchlauf:



4. Durchlauf:



5. Durchlauf:

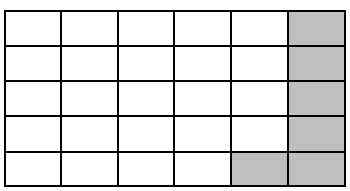
6. Durchlauf:

Wie würde die Lösung für dieses Beispiel aussehen?

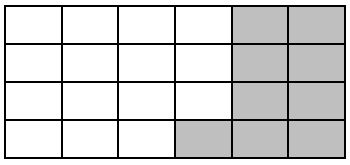
1. Durchlauf:

5	2	3	2	1	4

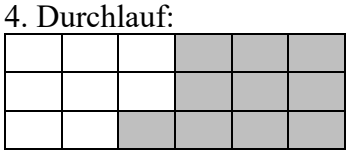
2. Durchlauf:



3. Durchlauf:

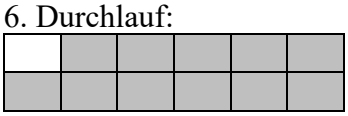


4. Durchlauf:



5. Durchlauf:

6. Durchlauf:

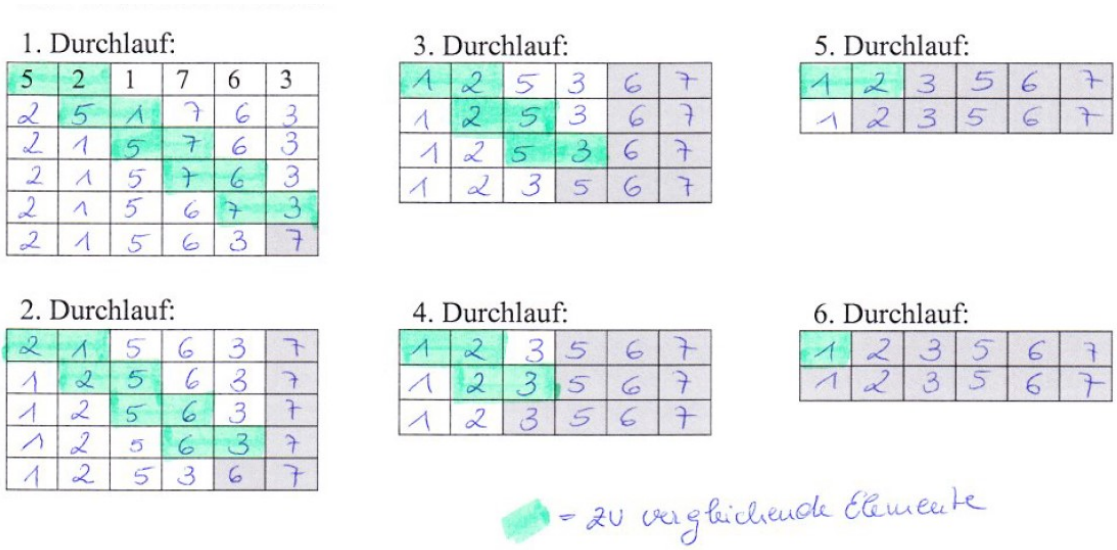


Sortierverfahren „Bubble Sort“ - Lösung

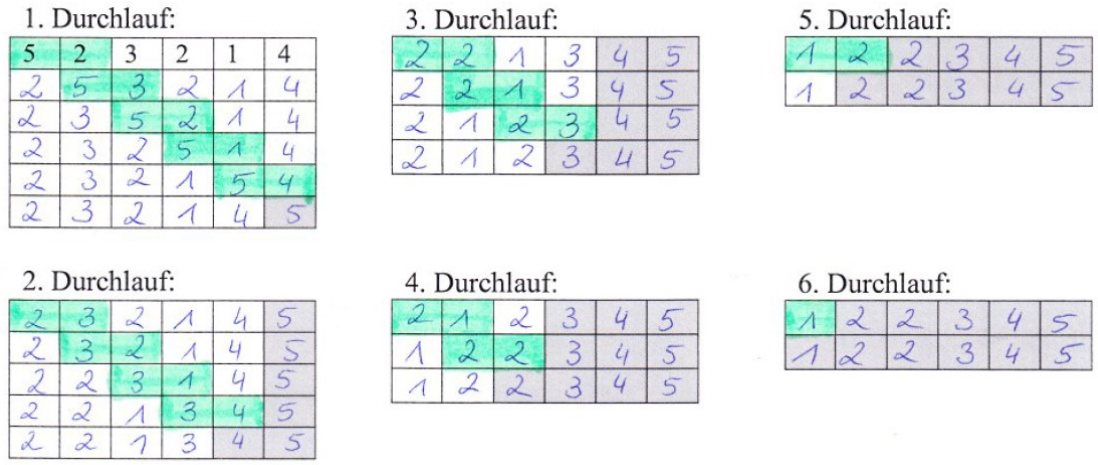
Beschreibe die Vorgangsweise beim Bubble Sort:

Er folgt dem Leitsatz „Sortieren durch Austauschen“. D.h. es werden immer die jeweils aufeinander folgenden Elemente verglichen. Das kleinere Element wird an die linke Stelle gesetzt und das größere Element an die rechte Stelle. Nach dem Abschlusses eines Durchgangs, erfolgt x-Mal (x ist die Anzahl der gegebenen Elemente.) ein erneuter Durchgang. (Helmich, 2017)

Löse folgendes Beispiel vollständig. Markiere jeweils die zu vergleichenden Elemente:



Wie würde die Lösung für dieses Beispiel aussehen?



Literaturverzeichnis

Helmich, U. (26. Februar 2017). *8.3 Der Bubble Sort*. Zugriff am 02. November 2017 unter http://www.u-helmich.de/inf/BlueJ/kurs11/Folge08/folge08-4.html

Überlegung, wo Sortieralgorithmen Anwendung finden im Alltag (Pomberger & Dobler, 2008: S. 301)

- *Listenerzeugung (für menschliche Leser):* Das Finden eines bestimmten Eintrags in einem Datenbestand ist in der Regel schneller, wenn dieser sortiert ist. Beispiele: Telefonbuch, Wörterbuch, Tabellenkalkulationsprogramme: Sortierfunktion um die Tabellen nach unterschiedlichen Kriterien neu anzuordnen und um die Suche effizient zu gestalten.
- *Duplikatermittlung*: Die Suche nach gleichen Elementen in mehreren Datenbeständen kann effizienter durchgeführt werden, wenn diese sortiert sind. Da Duplikate unmittelbar hintereinander liegen, können diese schnell identifiziert werden.
- *Computergrafik*: Sollen im Rahmen der Erstellung einer Computergrafik dreidimensionale, grafische Objekte auf einem zweidimensionalen Ausgabegerät dargestellt werden, so müssen diese gerendert werden. Um dies durchführen zu können ist eine Sortierung der Tiefenwerte des 3D Objektes hilfreich.

Literaturverzeichnis

Pomberger, G. & Dobler, H. (2008). *Algorithmen und Datenstrukturen: eine systematische Einführung in die Programmierung.* Deutschland: Pearson GmbH

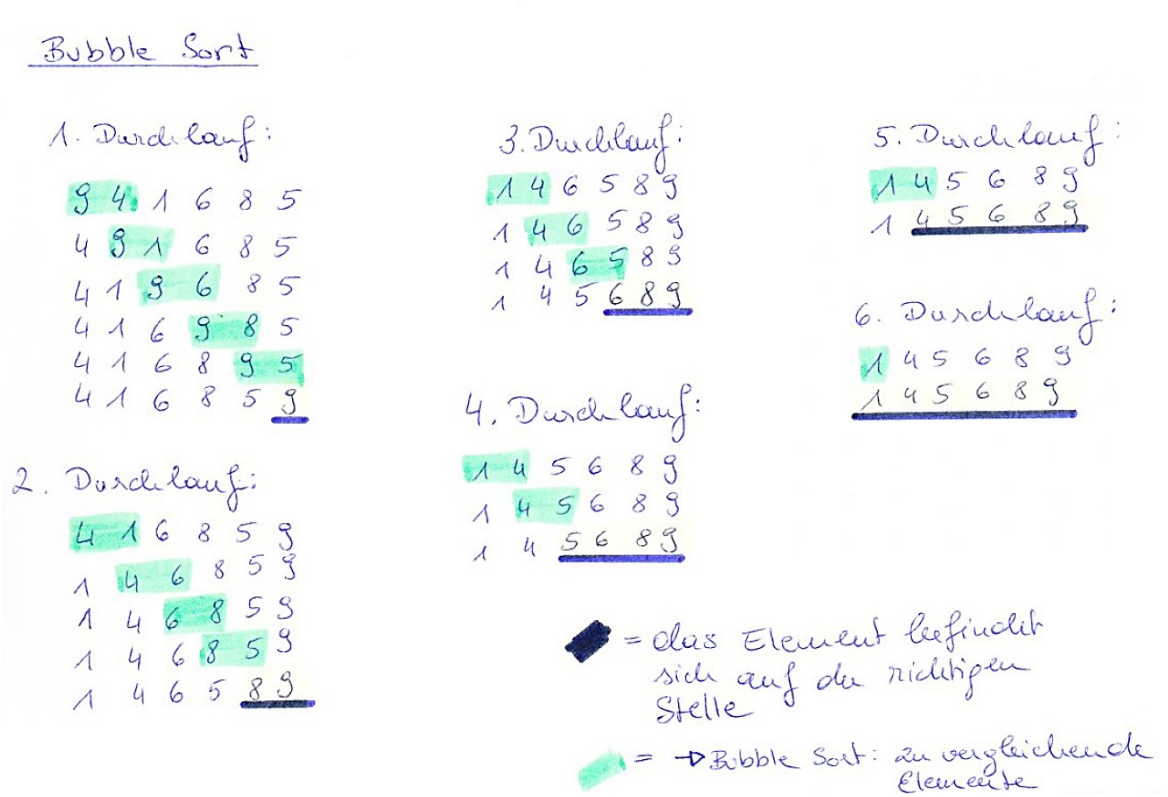
9.8 Materialien für die Sequenz zum Thema „Algorithmen – Sortierverfahren – Teil 2“

Wiederholungsaufgabe

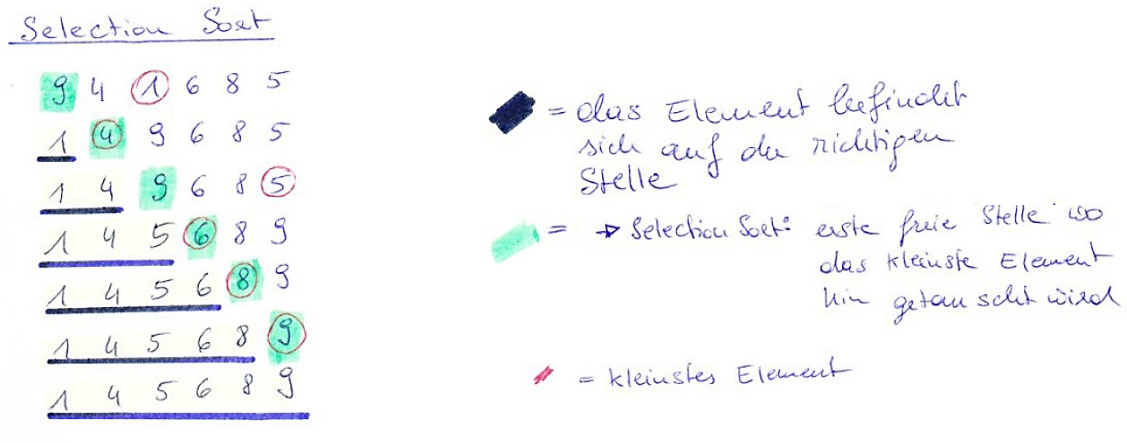
Sortiere die folgenden Zahlen mit Bubble Sort: 9 4 1 6 8 5 2 7 3

Wiederholungsaufgabe – Lösung:

Sortiere die folgenden Zahlen mit Bubble Sort: 9 4 1 6 8 5



Selection Sort: Gemeinsames Sortieren der Zahlen aus der schriftlichen Wiederholung



Pseudocode für das Sortierverfahren „Bubble Sort“

Die kursiv geschriebenen Worte werden anschließend bei der Umsetzung zu Variablen.

Sortierverfahren

Liste erzeugen

unsortierte Elemente = Länge der Liste

Platz = 1

wiederhole *Länge* der Liste mal

    wiederhole bis *Platz* = *unsortierte Elemente*

        falls Element *Platz* > Element (*Platz* + 1)

            tauschen der Elemente

        Platz + 1

Platz = 1

unsortierte Elemente - 1

Erzeugen einer Liste mit Zufallszahlen

Löschen aller Inhalte der Liste

wiederhole *Länge* mal

    füge eine Zufallszahl hinzu

Vertauschen der Elemente

Speichern des Inhalts von *Element x* in *Temporary*

Ersetzen des Inhalts von *Element x* durch den Inhalt des *Elements y*

Ersetzen des Inhalts von *Element y* durch den Inhalt von *Temporary*

Gemeinsame Erarbeitung des Pseudocodes: ***Sortierverfahren***

1. *Decomposition*: Vor der Erarbeitung des Pseudocodes sollte nochmals gemeinsam die genaue Funktionsweise des Sortierverfahrens besprochen und der Ablauf durchgegangen werden.

2. *Abstraction/Algorithm*: Der Fokus wird auf die Kernaufgabe des Sortierverfahrens gelegt, d.i. das Vergleichen und das Vertauschen der einzelnen Elemente miteinander.

   Mögliche Fragestellungen, die Unterstützend wirken:

   - Was ist die Kernaufgabe des Bubble Sorts?
   - Welche Elemente werden miteinander verglichen?
   - Gibt es Vorschläge, wie dies notiert werden kann? Ist die Definition einer Bedingung notwendig, in welchem Fall die miteinander verglichenen Elemente vertauscht werden?
   - Wie können die beiden Elemente, die miteinander verglichen werden, nur mit einer Variable ausgedrückt werden?

3. *Abstraction/Algorithm*: Nun wird die Aufmerksamkeit der SchülerInnen daraufgelegt, wie oft das Vergleichen von Elementpaaren für einen Durchlauf durchgeführt werden muss.

   Mögliche Fragestellungen, die Unterstützend wirken:

   - Wie viele Vergleiche von Elementen sind pro Durchlauf erforderlich? Wovon ist dies abhängig?
   - Wie könnte dies hier notiert werden?
   - Wir weisen dem ersten Element der Liste die Variable *Platz* zu und dem zweiten Element die Variable (*Platz* + 1*).* Wie können wir nun sicherstellen, dass nicht nur die beiden Elemente miteinander verglichen werden, sondern auch Element zwei mit Element drei?

4. *Abstraction/Algorithm*: Alle Durchläufe des Sortierverfahren sollen absolviert werden.

   Mögliche Fragestellungen, die Unterstützend wirken:

   - Was fällt uns auf, wenn wir uns den bereits notierten Pseudocode ansehen? Wird die vorhandene Liste schon vollständig sortiert?

- Was brauchen wir, damit wir sicherstellen können, dass nicht nur ein Durchlauf absolviert wird, sondern alle?
- Wie viele Durchläufe sind beispielsweise für eine Liste mit fünf Elementen notwendig? Und wenn die Liste 13 Elemente hat. Wie viele Durchläufe benötigen wir dann?
- Gibt es einen Zusammenhang zwischen der Anzahl der Durchläufe und der Anzahl der vorhandenen Elemente einer Liste?
- Wie kann das im Pseudocode notiert werden?
- Nun wird unsere Schleife *unsortierte Elemente*-mal wiederholt, d.h. zurzeit werden immer alle Elemente unserer Liste miteinander verglichen. Ist das notwendig? Wenn nein, wie könnten wir das unterbinden?
- Wenn wir diesen Code jetzt genau so umsetzen würden, würde dies funktionieren? Warum nicht?
- Jedes Mal, wenn mit der äußersten Schleife begonnen wird, soll wieder mit dem Vergleichen der Elemente an der ganz linken Stelle begonnen werden. Wie können wir das sicherstellen?

5. *Pattern Recognition/Algorithm*: Im Pseudocode können nun die einzelnen Variablen färbig markiert werden. Auf Basis dessen wird ersichtlich, welche Variablen benötigt werden für die Implementierung. Weiters ist daraus deutlich erkennbar, dass die Variablen zu Beginn des Sortierverfahrens einen Wert zugewiesen bekommen müssen.

   Mögliche Fragestellungen, die Unterstützend wirken:

   - Welche der notierten Begriffe könnten Variablen sein? Und warum?
   - Welchen Wert müssen wird der Variable *Platz* zuweisen und weshalb?
   - Welcher Wert soll für die Variable *unsortierte Elemente* verwendet werden? Wie könnte dies erfolgen?

6. *Abstraction/Algorithm*: Um überhaupt eine Liste mit Elementen sortieren zu können, muss diese zuerst erstellt werden.

   Mögliche Fragestellungen, die Unterstützend wirken:

   - Was wird benötigt, damit wir überhaupt mit dem Sortieren einer Liste beginnen können?

Gemeinsame Erarbeitung des Pseudocodes: ***Erzeugen einer Liste mit Zufallszahlen***

1. *Abstraction/Algorithm*: Der Fokus wir auf das wiederholte hinzufügen einer Zufallszahl gelegt.

   Mögliche Fragestellungen, die Unterstützend wirken:

   - Wie oft soll eine Zufallszahl hinzugefügt werden?

2. *Abstraction/Algorithm*: Um immer die gewünschte Länge der Liste zu erhalten, ist es notwendig vor dem erneuten Erstellen einer Lite diese vollständig zu leeren.

   Mögliche Fragestellungen, die Unterstützend wirken:

   - Was geschieht, wenn ich mein Programm ein zweites Mal starte? Haben wir immer die gewünschte Anzahl von Zufallszahlen in der Liste?
   - Wie können wir dies sicherstellen?

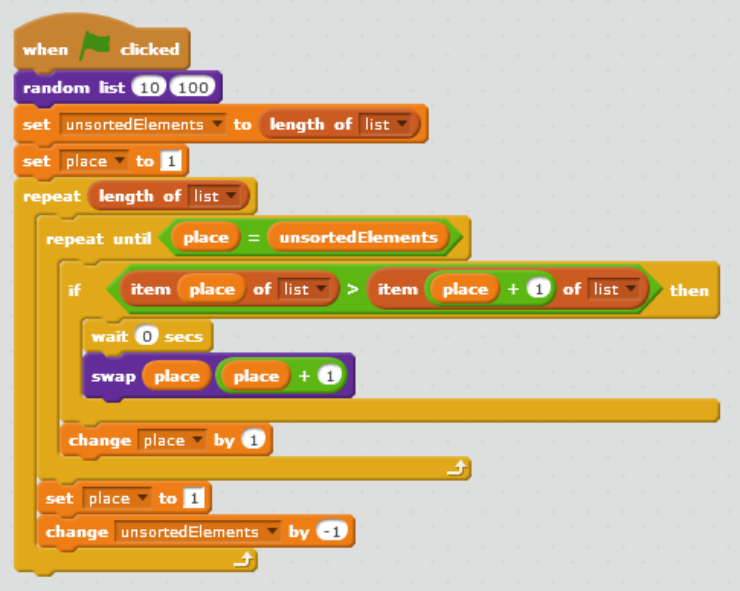
Gemeinsame Erarbeitung des Pseudocodes: ***Vertauschen der Elemente***

1. *Decomposition/Algorithm*: Das Tauschen der Inhalte zweier Elemente kann nur mittels der Verwendung von drei Variablen erfolgen.

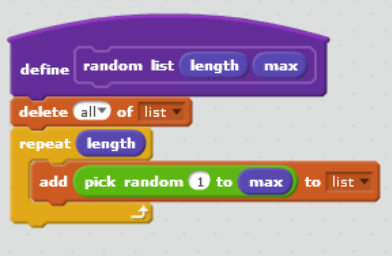
   Mögliche Fragestellungen, die Unterstützend wirken:

   - Wie können wir das Vertauschen der beiden Elemente umsetzen? Müssen wir dabei etwas beachten?
   - Weshalb kann dies nicht nur mit zwei Variablen erfolgen?
   - Wie kann das im Pseudocode notiert werden?

Sortierverfahren



Erzeugen einer Liste mit Zufallszahlen



Vertauschen der Elemente

9.9 Materialien für die Sequenz zum Thema „Hardware – Teil 1“

Rechercheauftrag zu einem Bauteil eines Rechners:

- *Decomposition*: Die SchülerInnen müssen zu der jeweiligen Hardwarekomponente, die sie erhalten haben, die gewünschten Inhalte recherchieren.
- *Pattern Recognition/Abstraction*: Die erhaltene Komponente stellt lediglich eine Ausführung eines Produzenten dar. D.h. die SchülerInnen müssen im Weiteren nicht nur die Bezeichnung, welche durch den Produzenten erfolgt ist (z.B. Interne Festplatte 8.9 cm (3.5 Zoll) 1 TB Western Digital Blue™ Bulk WD10EZEX SATA III), sondern auch den Namen des Bauteils (z.B. HDD) herausfinden.

Korrektes zusammenfügen der Bauteile zu einem Rechner:

- *Decomposition:* Zuerst muss festgelegt werden, welche Hardwarekomponenten für den Bau eines Rechners unbedingt erforderlich sind.
- *Pattern Recognition*: Möglicherweise haben einige SchülerInnen bereits einen geöffneten Rechner in Videos oder zu Hause betrachtet und wissen deshalb wie das Zusammenfügen korrekt funktioniert.

Rechner vs. Laptop vs. Smartphone:

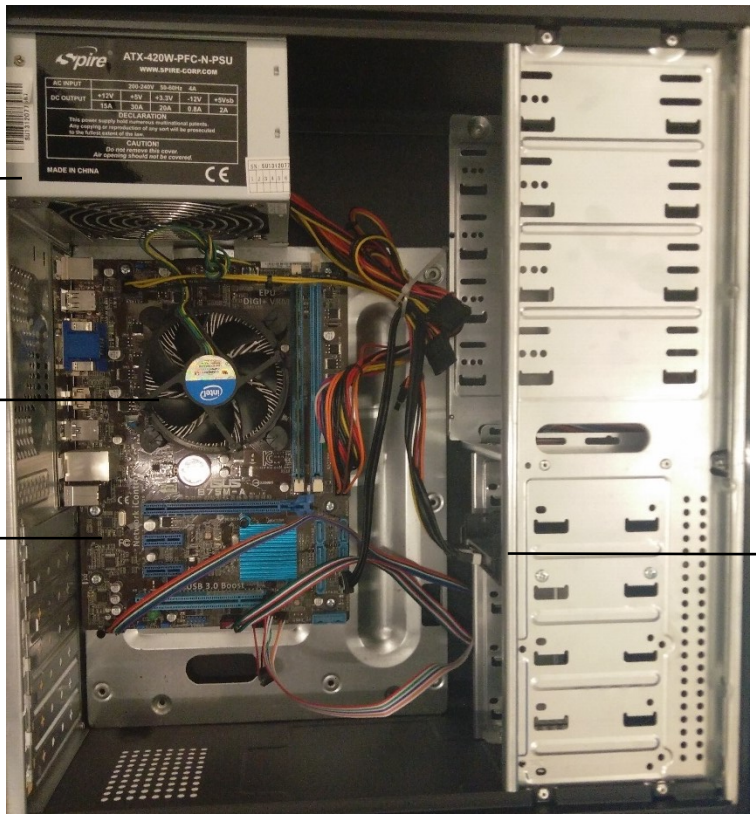
- *Abstraction*: Der Fokus der SchülerInnen wird auf die einzelnen Komponenten der Geräte gelenkt.
- *Pattern Recognition*: Anhand der geöffneten Geräte können die SchülerInnen Vergleiche bzgl. der einzelnen Komponenten durchführen und so überprüfen, ob tatsächlich in jedem Gerät z.B. eine Festplatte integriert ist.

Anschlüsse eines Rechners:

- *Pattern Recognition*: Hier gilt erneut, dass die SchülerInnen bereits im Alltag mit einigen Anschlüssen in Kontakt gekommen sein könnten, z. B. USB 2.0 bzw. 3.0 Anschluss, Netzwerkanschluss, etc.

Hardware eines Computers

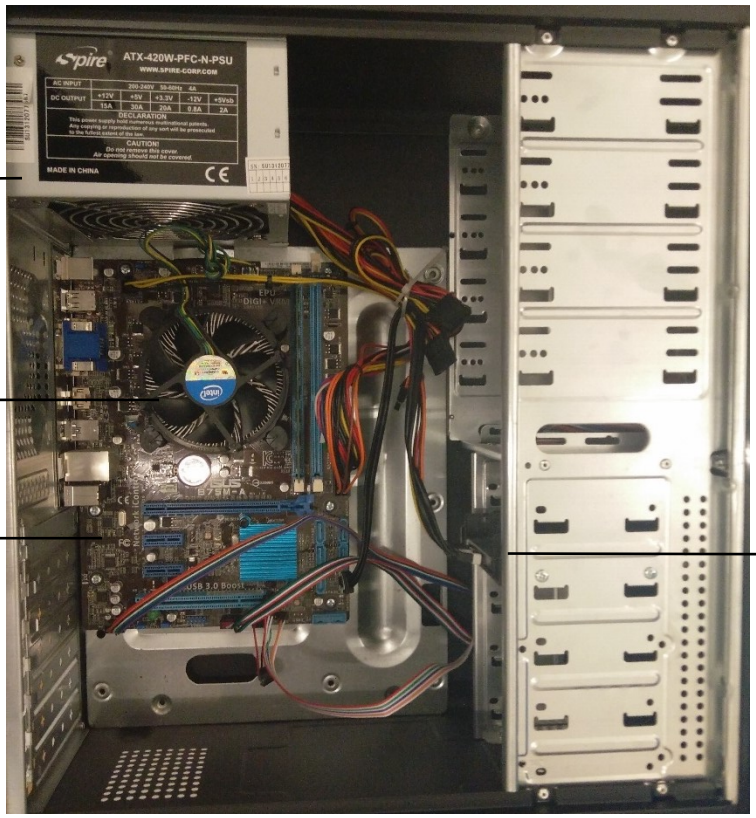
Beschrifte die einzelnen Bestandteile eines Rechners:



Welche Komponenten könnten noch in einem Rechner eingebaut sein?

Hardware eines Computers - Lösung

Beschrifte die einzelnen Bestandteile eines Rechners:



Welche Komponenten könnten noch in einem Rechner eingebaut sein?

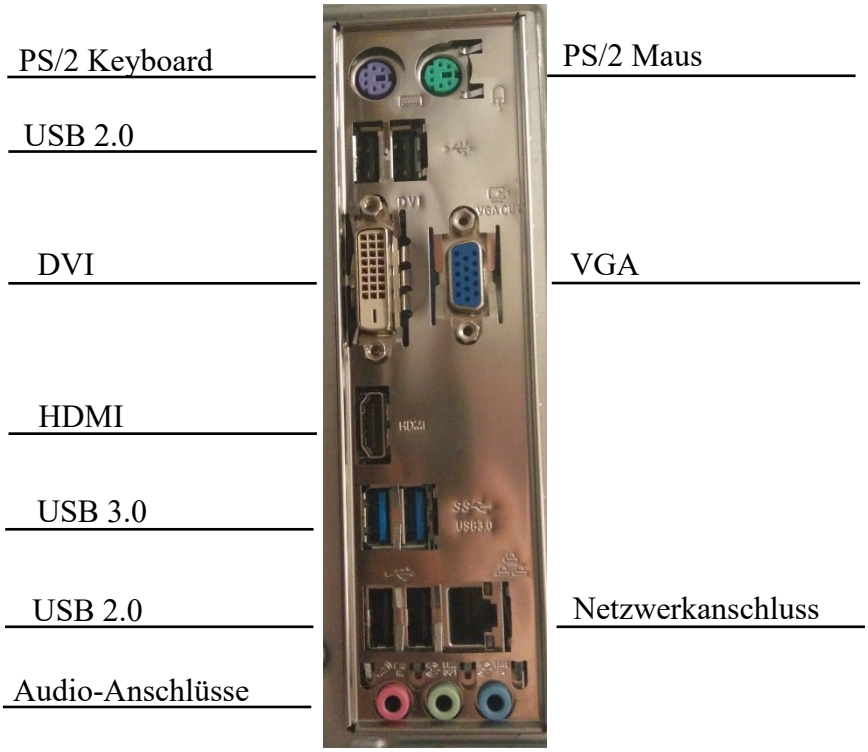
HDD Festplatte, Grafikkarte, Soundkarte

Anschlüsse eines Rechners

Beschrifte die einzelnen Anschlüsse eines Rechners:

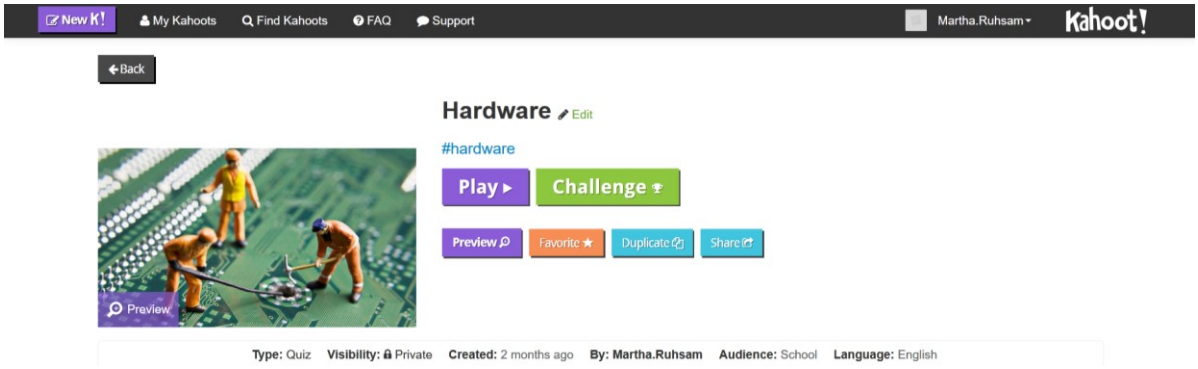
Anschlüsse eines Rechners - Lösung

Beschrifte die einzelnen Anschlüsse eines Rechners:



9.10 Materialien für die Sequenz zum Thema „Hardware – Teil 2“

Wiederholung des Gelernten mittels Kahoot!

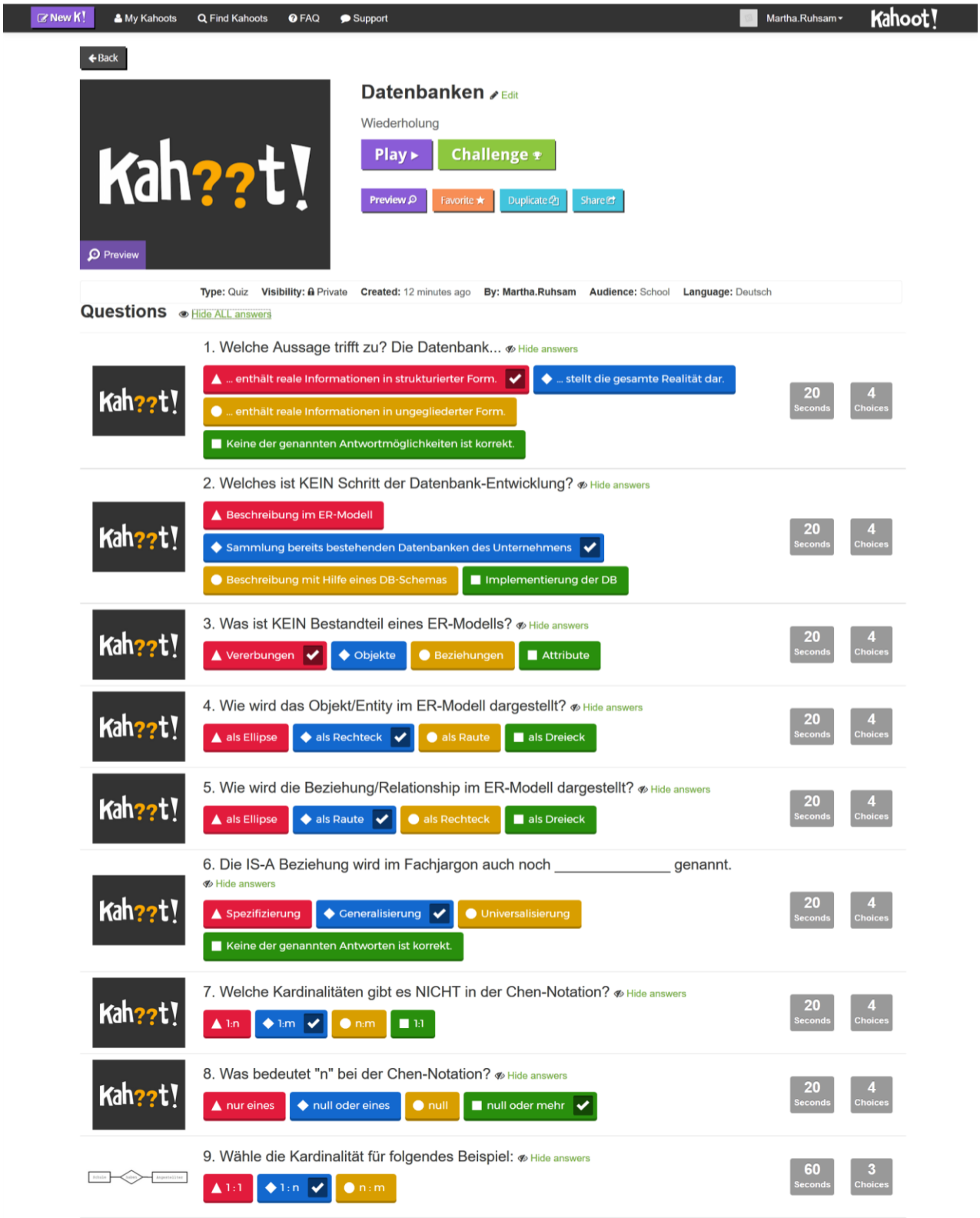


Bau eines funktionsfähigen Rechners:

- *Algorithm*: Am Beginn des Arbeitsauftrags sollen die SchülerInnen einen schrittweisen Plan erstellen, wie sie beim Bau eines Rechners vorgehen wollen.
- *Decomposition*: Die SchülerInnen überlegen welche Komponenten für den Bau eines Rechners benötigt werden und welche Komponenten notfalls auch weggelassen werden können.
- *Abstraction*: Bei dem Zusammenstecken der einzelnen Komponenten soll der Fokus immer auf das jeweilige Bauteil gerichtet werden. Weiters zu beachten ist, ob die vorhandenen Bauteile evtl. defekt sind.
  Sind bereits alle Komponenten zusammengesteckt, kann der Rechner gestartet werden. Sollte dieser nicht funktionieren, wird der Fokus nun auf mögliche Ursachen gerichtet.

9.11 Materialien für die Sequenz zum Thema „Datenbanken“

Wiederholung des Gelernten mittels Kahoot!



Normalformen (= NF)

Erste Normalform

Die Attribute einer Tabelle müssen unteilbar sein, d.h. sie können nicht in weitere Einzelinformationen zerlegt werden.

Schüler (KatNr., Name) → Schüler (KatNr., Vorname, Nachname)

Hier ist die Aufteilung des Feldes „Name“ in „Vor-„ und „Nachname“ erforderlich.

Zweite Normalform

Es muss eine Abhängigkeit vom gleichen Primärschlüssel zwischen der 1. Normalform und den Attributen bestehen.

Schüler (KatNr., Vorname, Nachname) → Schüler (Klasse, KatNr., Vorname, Nachname)

Da die Katalognummer eines Schülers nicht eine ausreichende Identifizierung garantiert (In jeder Klasse gibt es z.B. SchülerInnen mit der KatNr. 1.), ist das zusätzliche Vermerken der Klasse notwendig.

Dritte Normalform

Hierfür muss die zweite Normalform gegeben sein und keine transitiven Abhängigkeiten auftreten.

Schüler (Klasse, KatNr., Vorname, Nachname, Klassenvorstand)

→ Schüler (Klasse, KatNr., Vorname, Nachname)

→ Klassenvorstand (Klasse, LehrerNr.)

Das Feld „Klassenvorstand“ darf in diesem Fall nicht der Relation „Schüler“ zugeordnet werden, da so im Falle eines Tausches des Klassenvorstands dieser Inhalt bei jedem Schüler aktualisiert werden müsste. Wird der Klassenvorstand aber einer Klasse zugeordnet, genügt es in der Tabelle „Klassenvorstand“ den Namen des neuen Lehrers einzutragen.

Literaturverzeichnis

Wirtschaftsinformatik Expert. *Datenbankmodellierung*. (S. 109) Zugriff am 02. Jänner 2018 unter http://www.breitenfellner.info/klassenbereich/downloak/damo.pdf

Gemeinsames Erarbeiten eines ER-Modells inkl. Kardinalitäten

Angabe

Die Schulbibliothek möchte eine Datenbank erstellen. Ziel ist es anhand dieser eine Liste mit den zurzeit entlehnten Büchern erstellen zu können, um einen Überblick zu behalten. Neben einem Autor, einem Titel und dem Erscheinungsjahr soll auch eine eindeutige Inventarnummer für jedes Buch gespeichert werden. Auch das Beginn- und Enddatum der Ausleihe muss festgehalten werden.

Um Bücher an Personen verleihen zu können bzw. um Mahnungen ausstellen zu können, wenn ein Buch zu lange verborgt ist, sollen auch personenbezogene Daten gespeichert werden: Name, Adresse, E-Mail und Telefonnummer sind dafür jeweils notwendig.

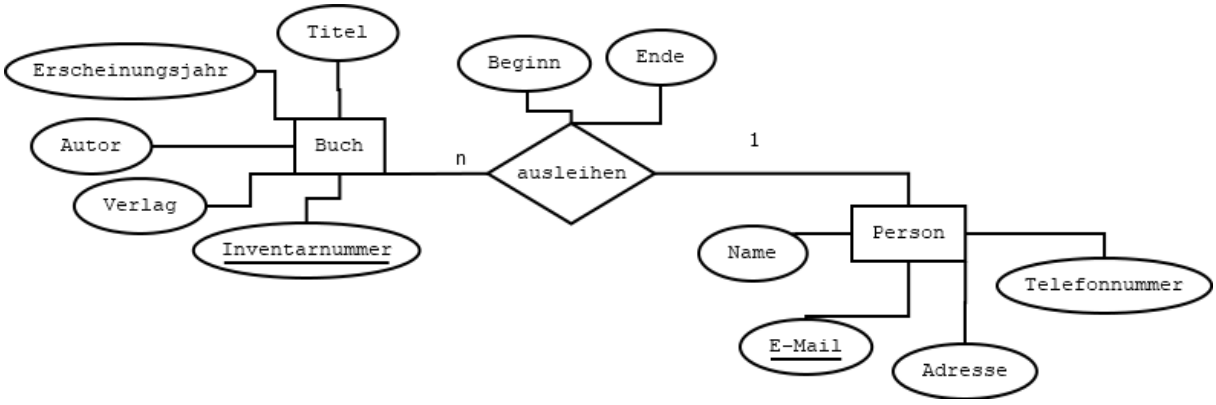
Mögliche Anleitung der SchülerInnen durch die Lehrkraft:

- *Decomposition*: Die SchülerInnen sollen zu Beginn alle im Text vorkommenden Nomen unterstreichen, da diese im weiteren Verlauf der Erstellung eines ER-Modells Entitäten bzw. Attribute darstellen können.
- *Pattern Recognition*: Zwischen den einzelnen unterstrichenen Nomen soll nun nach Ähnlichkeiten bzw. Mustern gesucht werden. So kann von den SchülerInnen herausgefunden werden, welche Nomen Entitäten (im Text blau markiert) oder Attribute (im Text grün markiert) darstellen.
- *Abstraction*: Es wird somit auch erarbeitet, welche Nomen (im Text rot markiert) nicht relevant für das Erstellen des betreffenden ER-Modells sind, z.B. Nationalbibliothek, Wien, Liste, etc.

Die Schulbibliothek möchte eine Datenbank erstellen. Ziel ist es anhand dieser eine Liste mit den zurzeit entlehnten Büchern erstellen zu können um einen Überblick zu behalten. Neben einem Autor, einem Titel und dem Erscheinungsjahr soll auch eine eindeutige Inventarnummer für jedes Buch gespeichert werden. Auch das Beginn- und Enddatum der Ausleihe muss festgehalten werden.

Um Bücher an Personen verleihen zu können bzw. um Mahnungen ausstellen zu können, wenn ein Buch zu lange verborgt ist, sollen auch personenbezogene Daten gespeichert werden: Name, Adresse, E-Mail und Telefonnummer sind dafür jeweils notwendig.

- *Algorithm*: Nun kann systematisch begonnen werden die Entitäten inkl. deren Attribute grafisch darzustellen. Im Anschluss erfolgt noch das Einfügen einer Relation inkl. der betreffenden Kardinalität (Chen-Notation).



Überführen des ER-Modells in die 3. Normalform

- *Decomposition* mit darauf anschließender *Abstraction*: Um feststellen zu können, ob die dritte Normalform gegeben ist, wird zuerst das Vorhandensein der ersten und der zweiten überprüft. D.h. es erfolgt eine Aufspaltung des Problems in drei Teile:
  1. NF: Alle Attribute werden auf ihre Unteilbarkeit hin überprüft. Dies ist beispielsweise bei den Attributen „Autor“, „Name“ und „Adresse“ nicht gegeben.
  2. NF: Da alle Attribute einer Entität von einem Primärschlüssel abhängen, ist diese bereits gegeben.
  3. NF: Wird diese Datenbank nur auf diese eine Bibliothek bezogen, gibt es keine transitiven Abhängigkeiten. Ist dies nicht der Fall, müsste bei der Entität „Buch“ auch noch das Attribut „Bibliotheksbezeichnung“ hinzugefügt werden.

Erarbeiten des relationalen Datenbankschemas

- *Pattern Recognition:* Auf die bereits zuvor festgestellte Ähnlichkeit zwischen den Entitäten bzw. zwischen den Attributen kann hier wieder Bezug genommen werden. Ein weiterer Verweis kann auf die gleiche Darstellungsart von den Entitäten Buch und Person (Rechteck) bzw. den Attributen (Ellipse) erfolgen.

- *Abstraction:* Bei der Erarbeitung des relationalen Datenbankschemas muss vor allem besonderer Fokus auf die gesetzten Kardinalitäten gelegt werden, da diese das endgültige Resultat stark beeinflussen.
- *Algorithm:* Die SchülerInnen können anhand der erarbeiteten Inhalte nun ein relationales Datenbankschema in der 3. Normalform erstellen.

Buch (Inventarnummer, E-Mail, Titel, Erscheinungsjahr, Vorname des Autors, Nachname des Autors, Verlag, Ausleihbeginn, Ausleihende)
PK: Inventarnummer
FK: E-Mail ◊ Person

Person (E-Mail, Vorname, Nachname, Straße, Hausnummer, Türnummer, PLZ, Ort, Telefonnummer)
PK: E-Mail

Überführen des relationalen Datenbankschemas in Tabellen

- *Pattern Recognition:* Wieder kann auf die Ähnlichkeit zwischen den Relationen Buch und Person (inkl. den jeweiligen Attributen) und deren gleiche Darstellungsart hingewiesen werden.
- *Algorithm:* Nun können die SchülerInnen die beiden Tabellen erstellen und diese mit möglichen Datensätzen befüllen.

Überführen des erarbeiteten Ergebnisses in Tabellen

Buch								
Inventar-nummer	*Titel*	*EJ*	*Vorname des Autors*	*Nach-name des Autors*	*Verlag*	*Ausleih-beginn*	*Ausleih-ende*	*E-Mail*
00235	Geschichte der Zauberei	1981	Bathilda	Bagshot	Zauber AG	04.11.17	04.12.17	granger@ hogwarts.com
00532	Dunkle Kräfte	1999	Quirin	Sumo	Zauber AG	27.10.17	27.11.17	potter@ hogwarts.com
03456	Verwandlungen für Anfänger	1960	Emeric	Wendel	Zauber AG	01.09.17	01.10.17	r.weasley@ hogwarts.com
15385	Zaubertränke und Zauberbräue	1927	Arsenius	Bunsen	Zauber AG	06.10.17	06.11.17	dumbledore@ hogwarts.com
…	…	…	…		…	…	…	

Person								
Vorname	*Nachname*	*E-Mail*	*Straße*	*Haus-nummer*	*Tür-nummer*	*PLZ*	*Ort*	*Telefonnummer*
Harry	Potter	h.potter@ hogwarts.com	Privet Drive	4	-	WH3 2JD	Little Whinging	0049650123456
Hermine	Granger	h.granger@ hogwarts.com	Westover Rd	17	-	BH1 2BU	Bournemouth	0049412034856
Ronald	Weasley	r.weasley@ hogwarts.com	Wilson Rd	69	-	PO2 8LE	Portsmouth	0049536728934
Albus	Dumbledore	a.dumbledore@ hogwarts.com	Oxford Rd	13	1	OX1 2JD	Oxford	0049648390345
…	…	…	…	…	…	…	…	…

Angabe

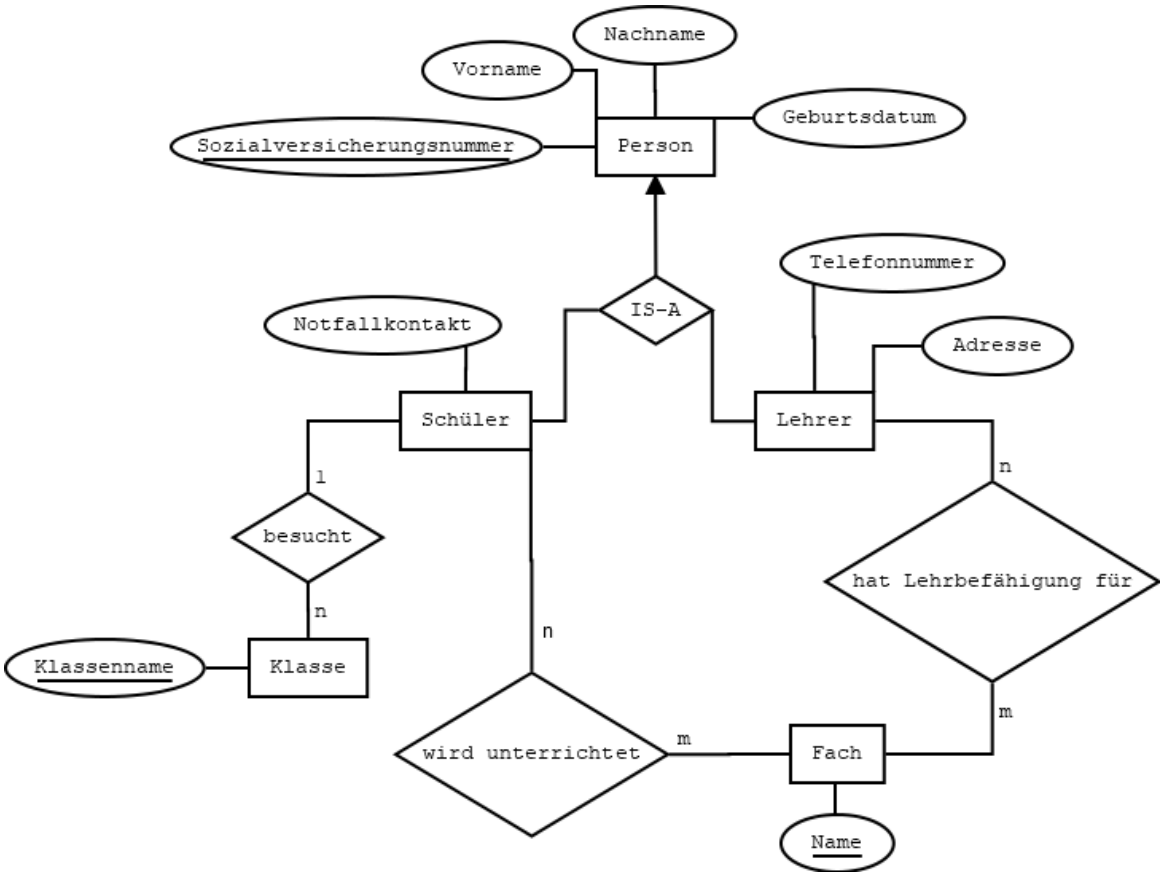
In einer Schule sind viele Personen, u.a. Lehrer und Schüler, beschäftigt. Um in der Administration einen Überblick zu behalten, werden in einer internen Datenbank Informationen wie, z.B. Vorname, Nachname, Geburtsdatum und Sozialversicherungsnummer, zu allen Personen gespeichert.

Zusätzlich wird bei Schülern ein Notfallkontakt vermerkt und bei den Lehrkräften die Telefonnummer inkl. Adresse.

Ein Schüler besucht genau eine Klasse. Diese ist durch den zugewiesenen Namen (z.B. 7A, 7B, …) eindeutig identifizierbar. Weiters erhält der Schüler in diversen Fächern Unterricht. (Diese werden auch durch den jeweiligen Namen eindeutig bezeichnet.)

Die Lehrkraft hat eine Lehrbefähigung in jeweils mindestens einem Fach (meistens in zwei bzw. in seltenen Fällen auch in drei). Jedoch kann es durchaus mehrere Lehrkräfte geben, die das gleiche Fach unterrichten.

ER-Modell



Überführung des ER-Modells in die 3. Normalform inkl. Anwendung des relationalen Datenbankschemas

Person (Sozialversicherungsnummer, Vorname, Nachname, Geburtsdatum)
PK: Sozialversicherungsnummer

Schüler (Sozialversicherungsnummer, Vorname des Notfallkontakts, Nachname des Notfallkontakts, Telefonnummer des Notfallkontakts)
PK: Sozialversicherungsnummer ◊ Person

Lehrer (Sozialversicherungsnummer, Telefonnummer, Straße, Hausnummer, Türnummer, PLZ, Ort)
PK: Sozialversicherungsnummer ◊ Person

Klasse (Klassenname, Sozialversicherungsnummer)
PK: Klassenname
FK: Sozialversicherungsnummer ◊ Schüler

Fach (FName)
PK: FName

Wird unterrichtet (Sozialversicherungsnummer, FName)
PK + FK: Sozialversicherungsnummer ◊ Schüler
PK + FK: Name ◊ Fach

Hat Lehrbefähigung für (Sozialversicherungsnummer, FName)
PK + FK: Sozialversicherungsnummer ◊ Lehrer
PK + FK: FName ◊ Fach

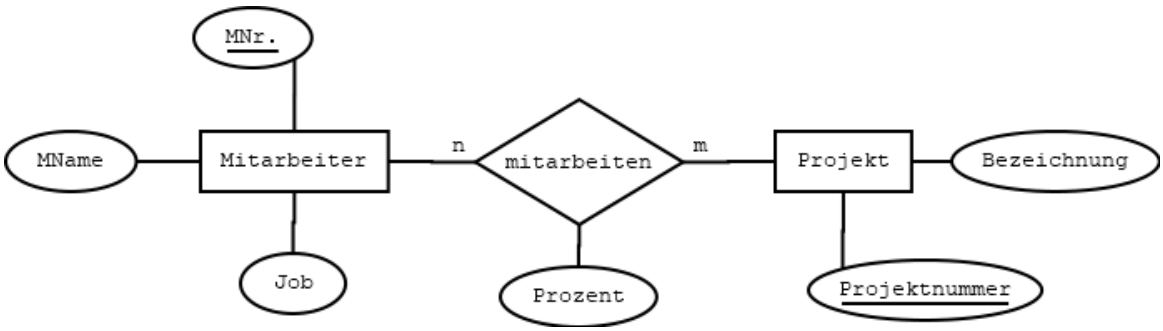
Stundenwiederholung

Angabe

Die Mitarbeiter eines Unternehmens arbeiten alle gemeinsam an Projekten. Um sie alle eindeutig identifizieren zu können wird Firmenintern eine Mitarbeiternummer vergeben. Auch der Name des Mitarbeiters und die genaue Jobbezeichnung wird festgehalten.

Um gegenüber Kunden genau darlegen zu können welche Personen in welchem Ausmaß an dem jeweiligen Projekt beteiligt waren, wird dies in Prozent vermerkt. Auch die Vergabe einer projektbezogenen Nummer und der genauen Bezeichnung des Projekts wird in der Datenbank des Unternehmens notiert.

ER-Modell



Überführung des ER-Modells in die 3. Normalform inkl. Anwendung des relationalen Datenbankschemas

Mitarbeiter (MNr., Vorname des Mitarbeiters, Nachname des Mitarbeiters, Job)
PK: MNr.

Projekt (Projektnummer, Bezeichnung)
PK: Projektnummer

Mitarbeit (MNr., Projektnummer, Prozent der geleisteten Mitarbeit)
PK & FK: MNr. ◊ Mitarbeiter
PK & FK: Projektnummer ◊ Projekt

Überführen des erarbeiteten Ergebnisses in Tabellen

Mitarbeiter			
MNr.	*Vorname*	*Nachname*	*Job*
123456	Bertie	Higgs	Projektmanagement
123457	Lyall	Lupin	Software Developer
…	…	…	…

Projekt	
Projektnummer	*Bezeichnung*
170091	Erstellung einer Homepage für das Ministerium für Hexerei und Zauberei
170092	Erstellung einer Homepage für Hogwarts, Schule für Hexerei und Zauberei
…	…

Mitarbeit		
MNr.	*Projektnummer*	*Bisher geleisteten Mitarbeit (in Prozent)*
123456	170091	17,23
123457	170091	10,59
…	…	…