Digitales Lernen Grundschule
Universität Potsdam

Deutsche Telekom Stiftung

Prof. Ulrich Kortenkamp, Peter Mahns, Heiko Etzold
März 2019, Überarbeitung einer erstmals im Mai 2017 erstellten
Version im Vorfeld eines Verbundtreffens im Projekt Digitales Lernen Grundschule

Überlegungen zur informatisch-algorithmischen Grundbildung in der Grundschule

Vorwort

Das Projekt Digitales Lernen Grundschule (Januar 2016 bis Dezember 2018) hatte unter anderem das Ziel, ein Curriculum zur handlungsorientierten Vermittlung informatisch-algorithmischer Kompetenzen zu erstellen. Aufgrund der individuellen Schwerpunktsetzung der einzelnen Projektstandorte (7 Hochschulen und Universitäten) sowie der verschiedenen Entwicklungen in den einzelnen Bundesländern schien es den Autoren nicht sinnvoll, eine Art Lehrplan als Sammlung von Inhalten oder Kompetenzen zu entwickeln. Vielmehr war den Autoren zufolge angedacht, dass prinzipielle Ideen, die Bestandteil einer informatisch-algorithmischen Grundbildung in der Grundschule sein müssen, die Entwicklung entsprechender Konzepte leiten. Angelehnt an Erkenntnisse aus der Didaktik der Informatik und angrenzender Wissenschaftsbereiche bieten sich hierzu sogenannte *fundamentale Ideen* an, die jedoch für informatische Inhalte in der Grundschule derzeit noch nicht ausgearbeitet wurden. Ziel dieses Dokumentes ist es daher, einen ersten Diskussionsanlass herzustellen und mit der fachdidaktischen und grundschulpädagogischen Community in Austausch zu kommen.

Notwendigkeit fundamentaler Ideen

Die fachdidaktische Sichtweise

Basierend auf Bruners *The Process of Education* (1960) sowie ausgehend von dem informatischen Vorgehensmodell des *Software Life Cycle* stellte Schwill (1993) einen Katalog von fundamentalen Ideen für die Informatik zusammen. Dieser besitzt eine hierarchische Struktur mit den drei Masterideen *Algorithmisierung*, *strukturierte Zerlegung* und *Sprache*, die als tragende Säulen für die Informatik aufgefasst werden können (vgl. Schubert & Schwill 2011, S. 68 ff.):

- Mit der fundamentalen Idee der *Algorithmisierung* verbindet sich die Vorstellung, „alle Probleme ließen sich durch maschinell nachvollziehbare Verfahren, deren Korrektheit jederzeit gesichert ist, effizient lösen“.
- Die *strukturierte Zerlegung* hat die Zielvorstellung einer „schrittweisen totalen Zerlegbarkeit jedes Systems in eine endliche Folge von Hierarchieebenen, die einen unterschiedlichen Abstraktionsgrad besitzen aber semantisch äquivalent sind und äquivalent ineinander überführt werden können.“
- Bei der Masteridee der *Sprache* gilt die folgende Assoziation: „Alle Sachverhalte [lassen] sich durch gewisse Zeichenfolgen beschreiben […], die nach einfachen Bildungsgesetzen aufgebaut sind und die untereinander auf effiziente algorithmische Weise semantikerhaltend transformierbar sind.“

Durch die Bedingung, dass jede Master- und Teilidee dem Horizontal-, Vertikal-, Ziel-, Zeit- und Sinnkriterium (Schwill 1993) genügen muss, gestaltet sich der Katalog relativ überschaubar und stabil, beispielsweise gegenüber der schnellen Entwicklung der Fachwissenschaft.

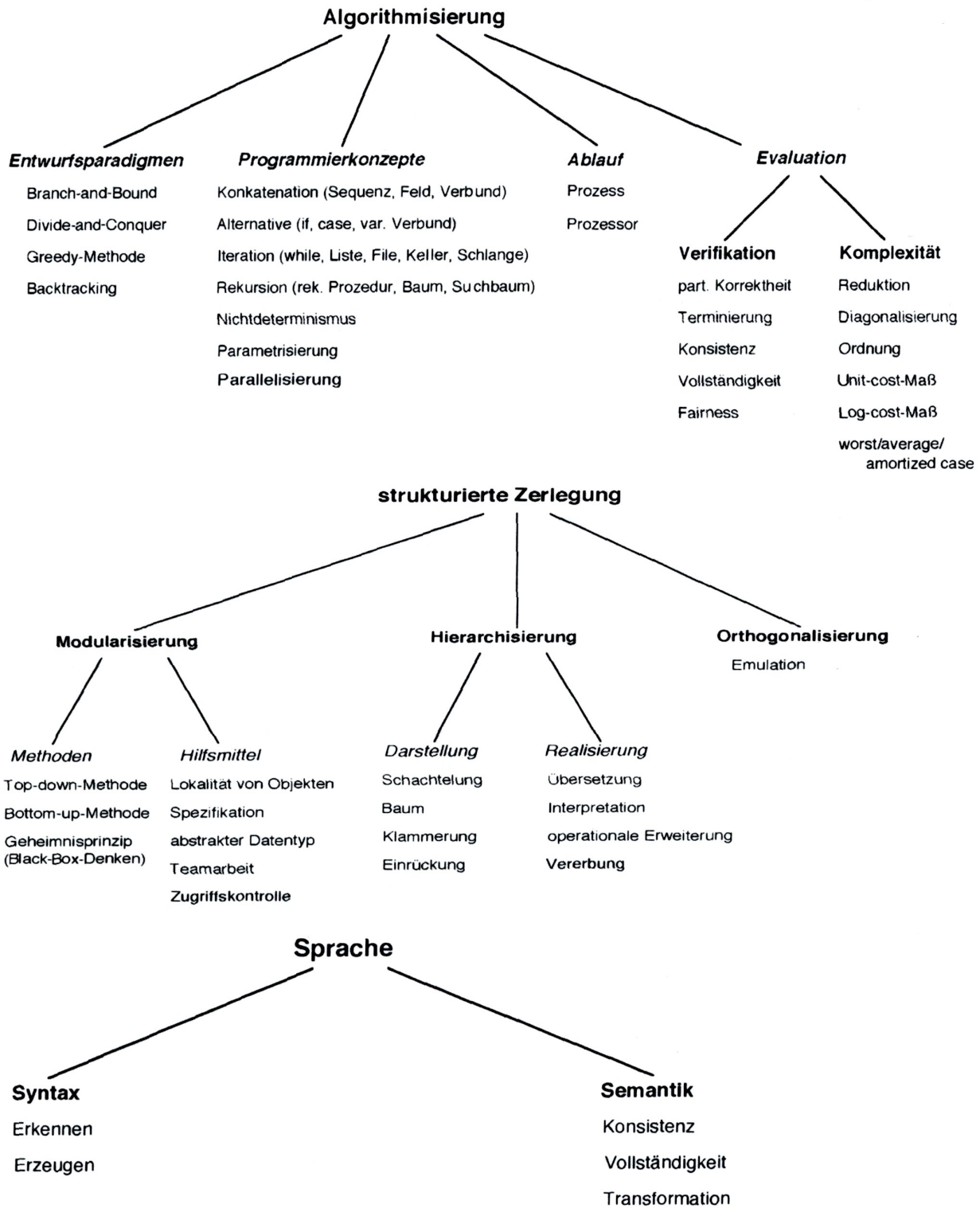

Abbildung 1: Fundamentale Ideen der Informatik (Schubert & Schwill 2011, S. 74)

Die curriculare Sichtweise

Viele Informatiklehrerinnen und -lehrer sind für einen Informatikunterricht nicht genügend fachwissenschaftlich und fachdidaktisch ausgebildet. Hinzu kommt, dass sie sich im Unterricht vorwiegend mit technischen Details der EDV beschäftigen und sich daher bezüglich anderer Themen hilflos fühlen. Aufgrund der Schnelllebigkeit der Fachwissenschaft gibt es für unausgebildete Fachkräfte nur wenige Konstanten im Unterricht. Gleichzeitig treten auch entsprechende Anforderungen durch die (digitale) Gesellschaft an die Lehrpersonen heran („Können wir nicht auch eine Webseite oder eine App programmieren?“).

Eine Orientierung an fundamentalen Ideen kann diese Anforderungsdiversität zumindest zum Teil auffangen und strukturelle Rahmenrichtlinien herstellen. Es ist möglich, die Grundlagen der Informatik zu strukturieren und insgesamt im Unterricht der Schule besser zu erfassen.

Ausgehend von Bruners Spiralprinzip muss ein an den fundamentalen Ideen ausgerichteter Unterricht drei Prinzipien erfüllen:

1. *Prinzip der Fortsetzbarkeit*: Die Darstellung von Begriffen und Konzepten erfolgt so, dass sie im weiteren Verlauf widerspruchsfrei fortgesetzt, formalisiert und abstrahiert werden kann. Dies kann auch einen frühzeitigen Gebrauch symbolischer Darstellungen bedeuten – gerade im Kontext der Grundschule natürlich stets in Bezug zu enaktiven und ikonischen Darstellungsmöglichkeiten.
2. *Prinzip der Präfiguration*: Es sollen nicht ständig völlig neue Themen und Inhalte in den Unterricht eingebracht werden, auch muss eine Vermittlung von Halbwahrheiten vermieden werden. Vielmehr müssen Inhalte so ausgewählt werden, dass ein geeignetes Arbeiten auf einem höheren Niveau möglich ist (z. B. durch komplexere Aufgaben, höheres Maß an Selbständigkeit)
3. *Prinzip des vorwegnehmenden Lernens*: Lerninhalte werden so früh wie möglich den Schülerinnen und Schülern auf deren Niveau vermittelt.

Zusammengefasst können damit die Vorteile des Brunerschen Spiralcurriculums erfüllt werden:

- Unterrichtsinhalte werden verständlicher, wenn man die dahinter liegenden Ideen versteht.[1]
- Fundamentale Ideen schaffen Beziehungsnetze.
- Fundamentale Ideen haben gegenüber reinem Faktenwissen (gerade in einer schnelllebigen Fachwissenschaft) eine längere Gültigkeit.

Im Projektverbund ergeben sich weitere strukturelle Vorteile:

- Die einzelnen Projektstandorte können konkret-inhaltlich individuell entsprechend ihrer Schwerpunkte arbeiten; dennoch geschieht dies auf einer gemeinsamen, verbindlichen Grundlage.
- Das im Projekt zu entwickelnde Curriculum kann exemplarisch erprobte Konzepte vorstellen, wobei die weitere Gestaltung unter Maßgabe der fundamentalen Ideen bewusst offen gehalten wird. Somit erhält die aus dem Projekt heraus entwickelte informatisch-algorithmische Grundbildung eine hohe Anschluss- und Erweiterungsfähigkeit, ohne dabei das „große Ganze“ aus den Augen zu verlieren.

[1] Bsp.: Wenn man verstanden hat, wie Formatvorlagen funktionieren, dann kann man diese bei MS Word, Pages, Open Office usw. anwenden.

Vorschlag für fundamentale Ideen zur informatisch-algorithmischen Grundbildung in der Grundschule

Die vorgeschlagenen fundamentalen Ideen werden zunächst in ihrer Struktur dargestellt und weiter unten genauer beschrieben. Es gibt die Masterideen *Algorithmisierung* und *Sprache* mit dem Bindeglied *Verlaufsbeschreibung*. Die Aufstellung ist an den fundamentalen Ideen von Schubert & Schwill (2011) orientiert und damit für die Sekundarstufe anschlussfähig. Dabei wurden jedoch nur solche fundamentale Ideen berücksichtigt, die tatsächlich in der Grundschule handhabbar sind.

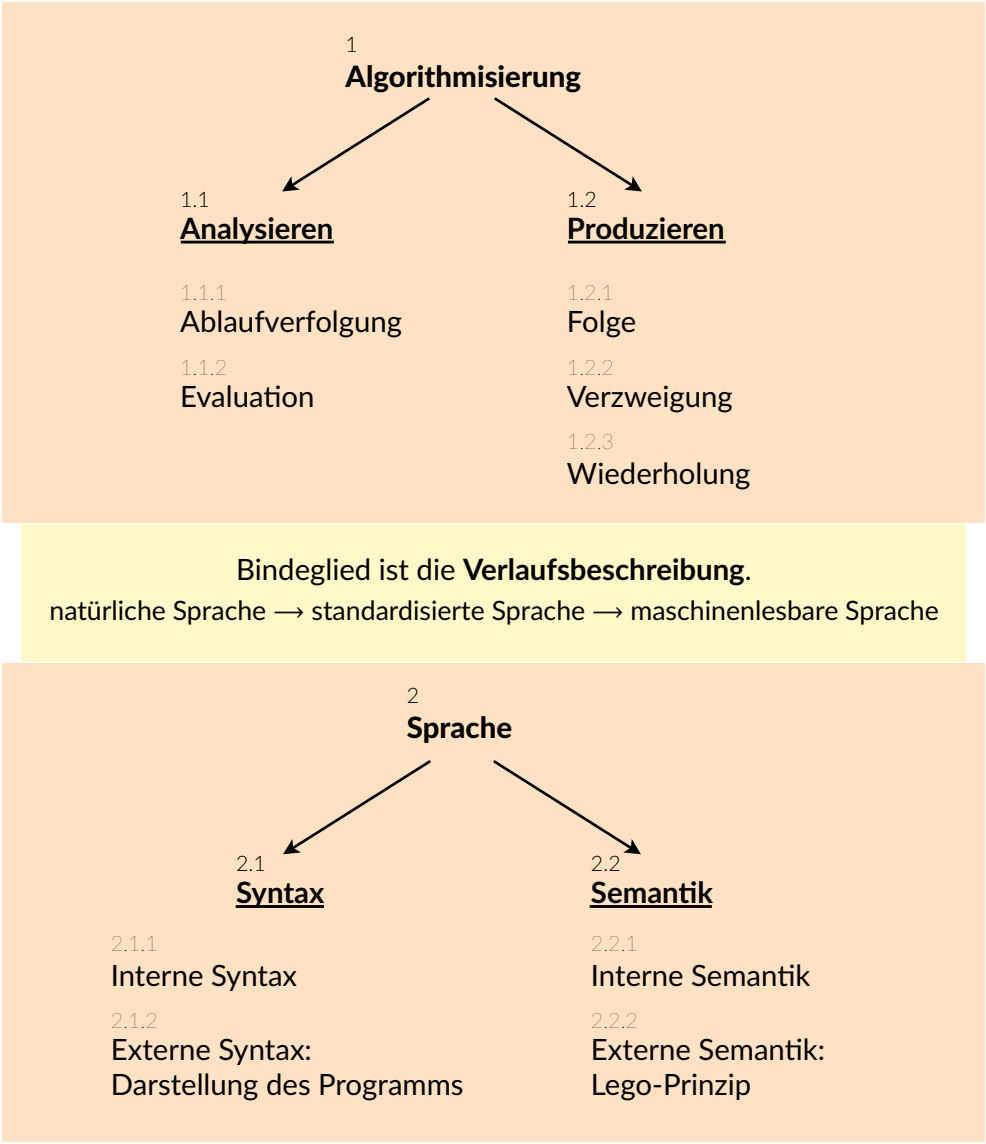

Abbildung 2: Vorschlag für fundamentale Ideen zur informatisch-algorithmischen Grundbildung in der Grundschule

Beschreibung der einzelnen Ideen

1. Algorithmisierung

Hinsichtlich der Formulierung, „alle Probleme ließen sich durch maschinell nachvollziehbare Verfahren, deren Korrektheit jederzeit gesichert ist, effizient lösen“ (Schubert & Schwill 2011, S. 69), sollte im Grundschulunterricht der Schwerpunkt präzisiert werden: Algorithmisches Denken ist nicht zwingend abhängig von einer digitalen Verarbeitung, sondern kann auch spielerisch mit

realen Materialien oder rein auf sprachlicher Ebene stattfinden (vgl. Döbeli Honeger 2016, S. 100). Das digitale Medium ist mithilfe der Programmierung nur *eine* Umsetzungmöglichkeit algorithmischen Denkens.

Der Schwerpunkt liegt also vielmehr darin, einen „Algorithmus [als] **endliche Folge** von **eindeutig bestimmten Elementaranweisungen**, die den Lösungsweg eines Problems **exakt und vollständig** beschreiben“ aufzufassen (Ziegenbalg u. a. 2010, S. 23, entnommen aus Kortenkamp & Lambert 2015, S. 5). Dies kann kann dann wiederum auf *verschiedenen* Ebenen realisiert werden.

Bezugnehmend auf die Kompetenzbereiche 3 und 6 der KMK-Digitalisierungsstrategie (2016, S. 16 ff.) wird die Algorithmisierung in die Teilideen *Algorithmen analysieren* und *Algorithmen produzieren* aufgeschlüsselt. Auch die Mediencurricula der einzelnen Bundesländer sehen das Analysieren und Produzieren von und mit Medien als wesentliche Kompetenzbereiche (vgl. z. B. BE/BB 2016, S. 14; BR 2012, S. 7), was hier auf den Umgang mit Algorithmen übertragen werden kann. Weiterhin beinhaltet die KMK-Digitalisierungsstrategie auch konkret die Kompetenzformulierung 5.5 *Algorithmen erkennen und formulieren*, in der spezifiziert wird, „eine strukturierte, algorithmische Sequenz zur Lösung eines Problems [zu] planen und [zu] verwenden“ (KMK 2016, S. 17).

1.1. Algorithmen analysieren

> „Es zeigen sich große Mängel bei Lernenden in der Fähigkeit Programme zu verstehen, sprich Code zu lesen und zu verfolgen. […] Unterricht sollte sich daher mehr auf das Lesen und Verstehen von Code konzentrieren.“ (Lobnig 2008, S. 9)

Diese auf Programmcode bezogene Kritik ist ohne weiteres auf allgemeine (nicht zwangsweise durch Programmierung erzeugte) Algorithmen übertragbar und ruft danach, das Analysieren und damit Nachvollziehen von Algorithmen zu einem Schwerpunkt des Unterrichts zu erheben. Auch die GI fordert (2008, S. 30): „Zum Verständnis von Algorithmen gehört auch, dass formale Darstellungen dieser Handlungsvorschriften gelesen, interpretiert und gedanklich nachvollzogen werden können“.

1.1.1. Ablaufverfolgung

Der Begriff der Ablaufverfolgung meint im eigentlichen Sinne die Untersuchung von Programmcode hinsichtlich Fehler, die nach der Ausführung des Übersetzers ausgegeben werden. Im Kontext der Grundschule geht es aber nicht primär um das Suchen und Beseitigen von Fehlern (sogenanntes *Debugging*), sondern um die schrittweise Ausführung bzw. Nachvollziehbarkeit einer gegebenen und vor allem korrekten Befehlsfolge eines Algorithmus. Die Ablaufverfolgung ist daher eher als eine Art *Verständnisstrategie* aufzufassen (vgl. Lobnig 2008, S. 9).

Eine Möglichkeit der Ablaufverfolgung ist beispielsweise die schrittweise Analyse der Algorithmenbefehle hinsichtlich des aktuellen Wertes bestimmter Variablen (sogenannte *konkrete Ablaufverfolgung*, siehe Lobnig 2008). Neben „herkömmlichen“ Variablen, die einen Zahlenwert annehmen, können solche Variablen aber auch die Geschwindigkeit eines Roboters, die Farbe einer LED oder ähnliches sein.

Insbesondere zum Verständnis der Grundbausteine *Verzweigung* und *Wiederholung* (siehe fundamentale Ideen 1.2.2 und 1.2.3) ist die Ablaufverfolgung von essenzieller Bedeutung (Mahns 2017).

1.1.2. Evaluation

Mit Evaluation ist die Auswertung von einer gegebenen oder selbst erstellten Befehlsfolge gemeint. Es sollen einzelne oder mehrere Algorithmen hinsichtlich bestimmter Kriterien bewertet bzw. verglichen werden. Diese Kriterien müssen allerdings sinnvoll für die Grundschule formuliert werden – übliche Leitkriterien der Informatik wie Effektivität oder Komplexität eines Programms spielen dabei eher eine untergeordnete Rolle. Vielmehr kommt „es bei der Formulierung von Algorithmen für unterrichtliche Zwecke […] auf gute Verständlichkeit (‚kognitive Effizienz') und nicht primär auf Laufzeit- oder Speicherplatzeffizienz" an (Ziegenbalg 2015, S. 311).

In der Grundschule können neben der *Verständlichkeit* beispielsweise auch *Schönheit* oder *Übersichtlichkeit* geeignete Kriterien sein, wobei im Klassenverbund stets eine Spezifizierung nötig ist, was damit nun genau gemeint ist.

Auch die prinzipielle Lauffähigkeit eines (selbst geschriebenen) Programms kann ein Kriterium zur Evaluation sein – selbst wenn beispielsweise noch logische Fehler enthalten sind. „Aber trotzdem ‚passiert' etwas, das System ‚funktioniert', wenn auch nicht zu 100%. Produktstolz unabhängig vom Urteil des Lehrers kann in diesem Zusammenhang nicht hoch genug bewertet werden" (Strecker, 2011).

1.2. Algorithmen produzieren

Eine mächtige und auch auf Grundschulebene auf entsprechendem Niveau durchführbare Handlung um Algorithmen zu produzieren ist das Programmieren. Von den bei Schubert & Schwill (2011) aufgezählten Programmierkonzepten muss eine für die Grundschule geeignete Auswahl getroffen werden. „Von Beginn an liegt der Fokus auf den algorithmischen Grundbausteinen Folge, Verzweigung, Wiederholung, die immer wieder benötigt werden. Sie sind sicher zu beherrschen, um diese in Programmen mit dem jeweils passenden Werkzeug zu implementieren", sagt die GI (2008, S. 30). Daher sollen diese drei Bestandteile zu den fundamentalen Ideen informatisch-algorithmischer Grundkompetenzen in der Grundschule gehören.

Die Grundbausteine können selbstverständlich auch isoliert voneinander und unabhängig von Algorithmen im Unterricht betrachtet werden und somit einer Vorbereitung auf algorithmisches Denken dienen.

1.2.1. Folge

Schülerinnen und Schüler erkennen, dass die Schritte eines Algorithmus nicht gleichzeitig, sondern nacheinander ausgeführt werden. Die Reihenfolge der Darstellung eines Algorithmus bestimmt die Reihenfolge dessen Verarbeitung.

1.2.2. Verzweigung

Mithilfe von Wenn-Dann-Befehlen und ähnlichen Konstrukten können Bedingungen überprüft und daraus die weiteren Algorithmenschritte beeinflusst werden. Schülerinnen und Schüler können Entscheidungsfragen auch unabhängig von der Programmierung auf unterschiedlichem Niveau bearbeiten.

1.2.3. Wiederholung

Schülerinnen und Schüler erkennen als einen elementareren Vorteil der Programmierung die schnelle Wiederholbarkeit von Programmabschitten gegenüber einer

manuellen Hintereinanderausführung. Solange eine Wiederholungsbedingung gültig ist, wird der entsprechende Programmabschnitt ausgeführt.

Verlaufsbeschreibung

Verlaufsbeschreibungen sind ein übliches Mittel im Sprachenunterricht, ob als Teil einer Geschichte bzw. Erzählung oder explizit als objektiver Bericht. Diese können nun genutzt werden, um daraus in einem Abstraktionsprozess von einer natürlichen Sprache über eine standardisierte zu einer formalen bzw. maschinenlesbaren Sprache zu kommen.

Während mündliche Kommunikation (*natürliche Sprache*) in der Regel von Mimik, Gestik und Betonung geprägt ist, ist das Schriftliche darauf ausgerichtet, dass Sender und Empfänger nicht am selben Ort sind. Dadurch muss die Ausdrucksfähigkeit expliziter werden und nonverbale Unterstützungen fehlen (Übergang zur *standardisierten Sprache*). Eine Programmiersprache ist dann sogar noch strenger als „normale“ geschriebene Sprache, da nun noch die Flexibilität des Empfängers fehlt (*formale* bzw. *maschinenlesbare Sprache*).

Eine mögliche Maßnahme auf dem Weg zum sprachlichen Abstrahieren ist das Identifizieren und Kategorisieren relevanter Merkmale einer Verlaufsbeschreibung (z. B. von Subjekten/Objekten und Prädikaten). Dies unterstützt nicht nur den Grammatikunterricht sondern schärft auch den Blick auf die notwendige Syntax einer standardisierten bzw. formalisierten (algorithmischen) Sprache und ermöglicht eine Reflexion zur Semantik der verwendeten Bestandteile.

2. Sprache

Die Sprache eines Algorithmus als Zeichen- oder Wortfolge gliedert sich in eine Strukturebene (Syntax) und eine Bedeutungsebene (Semantik). Beide Schwerpunkte können mit oder ohne Programmierung im Grundschulunterricht thematisiert werden.

2.1. Syntax

2.1.1. Interne Syntax: Regeln der Sprache

Hierunter ist zu verstehen, dass eine Programmiersprache strengen Regeln gehorchen muss, damit das Geschriebene vom Computer umgesetzt werden kann. Diese Regeln werden durch die jeweils verwendete Sprache bestimmt, wobei Ähnlichkeiten zwischen verschiedenen Sprachen bestehen können. Gerade an dieser Stelle ist ein Bezug zur Verlaufsbeschreibung herstellbar.

2.1.2. Externe Syntax: Darstellung des Programms

Die Darstellung des Programm strukturiert die interne Syntax nach außen. Dazu gehören Überlegungen, warum beispielsweise Text *geschachtelt*, *geklammert* oder *eingerückt* wird. Auch spielen hier *Farben* eine Rolle, die die Elemente der Programmiersprache verdeutlichen und strukturieren – dies ist beispielsweise auch bei *Bausteinen* im Rahmen der *visuellen Programmierung* von Bedeutung.

Schülerinnen und Schüler sollen verschiedene Darstellungsmöglichkeiten kennenlernen und deren jeweiligen Nutzen beurteilen können.

2.2. Semantik

2.2.1. Interne Semantik: Bedeutung des Dargestellten

Schülerinnen und Schüler sollen erkennen, dass den in einem Algorithmus, insbesondere in einem Programmcode oder einer visuellen Programmierumgebung, dargestell-

ten Objekten eine bestimmte Bedeutung zugeordnet werden muss. Über diesen Bedeutungen muss man sich im Klaren sein, um den Algorithmus zu verstehen oder erzeugen zu können.

So könnte bei der Bezeichnung bestimmter Befehle darauf geachtet werden, dass diese sprachlich an das Vorwissen der Schülerinnen und Schüler anknüpfen (z. B. „WIEDERHOLE-SOLANGE“ statt „REPEAT-WHILE“), um den Fokus auf das Erlernen des Programmierens zu legen und nicht durch zusätzlich nötige Übersetzungen gehindert zu werden.

2.2.2. Externe Semantik: Lego-Prinzip[2]

Diese Idee besagt, dass eine Programmiersprache auf *möglichst wenige Basiselemente* aufbaut, mithilfe derer dann alle anderen Elemente erzeugt werden. Die Basiselemente sind unabhängig voneinander, können also nicht selbst erzeugt werden. Damit strukturiert diese Idee die interne Syntax nach außen.

Im Kontext der Grundschule kann untersucht werden, welche „Bausteine“ zur Programmierung notwendig sind, wie sich andere Bausteine daraus erzeugen lassen oder ob manche Bausteine ggf. gar nicht notwendig sind, eben weil sie sich aus anderen erzeugen lassen (oder warum es vielleicht doch sinnvoll ist, eben jene aus Bausteinen erzeugbaren Bausteine zu nutzen). Damit kann auch wieder ein Anknüpfungspunkt zur Idee *Algorithmen produzieren* (1.2) gefunden werden.

Zusammenfassung

Die vorgeschlagenen fundamentalen Ideen für die Grundschule sind eine Teilmenge der in Schubert & Schwill (2011, S. 74) dargestellten fundamentalen Ideen, teils mit anderen Bezeichnungen, teils in leicht variierter Struktur. Somit ist eine Anschlussfähigkeit an die Sekundarstufe möglich und die Überlegungen können in der Fachdidaktik Informatik verankert werden.

Als Besonderheit ist in der Grundschule die *Verlaufsbeschreibung* als Bindeglied zwischen den fundamentalen Ideen *Algorithmisierung* und *Sprache* hervorzuheben. Die Verlaufsbeschreibung bietet kontextspezifische Anknüpfungspunkte für informatisch-algorithmische Grundbildung im Grundschulunterricht und ermöglicht zugleich fachübergreifende Konzeptentwicklungen zwischen Deutsch (Sprache), Mathematik (Algorithmen) und dem Fach des jeweiligen Kontextes, von dem aus informatisch-algorithmisches Denken heraus entwickelt werden soll.

Literatur

BE/BB (2016): Rahmenlehrplan Berlin/Brandenburg. Teil B: Fachübergreifende Kompetenzentwicklung.

BR (2012): Rahmenlehrplan Medienbildung Bremen.

Döbeli Honegger, B. (2016). *Mehr als 0 und 1 – Schule in einer digitalisierten Welt*. Bern: hep-Verlag

Gesellschaft für Informatik (GI) e.V. (2008). *Grundsätze und Standards für die Informatik in der Schule. Bildungsstandards Informatik für die Sekundarstufe I.* URL: http://informatikstandards.de/docs/bildungsstandards_2008.pdf (28.04.2017, 10:00 Uhr)

[2] Schubert & Schwill (2011) selbst nennen diese fundamentale Idee „Orthogonalität“, angelehnt an den Begriff aus der Linearen Algebra (Orthogonalbasis eines Vektorraums).

KMK (2016): *Bildung in der digitalen Welt. Strategie der Kultusministerkonferenz.* Berlin: Eigendruck

Lobnig, T. (2008): *Probleme von Programmieranfänger/inne/n*. Universität Klagenfurt. URL: https://www.ddi.edu.tum.de/fileadmin/tueds10/www/material/Lehre/Seminare/BestOf/08-KL-Lobnig-Programmieranfaenger.pdf (02.05.2017, 15:00)

Mahns, P. (2017): *Entwicklung einer Lernumgebung für einen Zugang zu Wiederholungsanweisungen im Mathematikunterricht*. Masterarbeit, Universität Potsdam

Schubert, S. & Schwill, A. (2011). *Didaktik der Informatik*. Heidelberg: Spektrum Akademischer Verlag

Schwill, A. (1993). Fundamentale Ideen der Informatik. Oldenburg. URL: https://pdfs.semanticscholar.org/9811/48ef2ba7b9534e917a31781e55c25fa4e925.pdf (24.04.2017, 10:00 Uhr)

Schwill, A. (2001). Ab wann kann man mit Kindern Informatik machen? In: Keil-Slawik, R. & Magenheim, J. (Hrsg.): *Informatikunterricht und Medienbildung*, *INFOS 2001*, *9. GI-Fachtagung Informatik und Schule*, *17.-20. September 2001 in Paderborn*, S. 13 – 30. Paderborn: GI-Edition

Strecker, K. (2011). Wie viel Programmierkompetenz braucht der Mensch? In: *LOGIN: Nr. 169-170*, S. 40 – 48. Berlin: LOG IN Verlag

Kortenkamp, U. & Lambert, A. (2015). Wenn …, dann … bis … – Algorithmisches Denken (nicht nur) im Mathematikunterricht. In: Kortenkamp, U. & Lambert. A. (Hrsg.): *mathematiklehren 188: Algorithmen*, S. 2 – 9. Seelze: Friedrich-Verlag

Ziegenbalg, J. (2015). Algorithmik. In: Bruder, R. & Hefendehl-Hebeker, L. & Schmidt-Thieme, B. & Weigand, H-G (Hrsg.): *Handbuch der Mathematikdidaktik*, S. 303 – 329. Berlin: Springer Spektrum

Autoren

Ulrich Kortenkamp, ulrich.kortenkamp@uni-potsdam.de, 0331 977-1470

Peter Mahns, peter.mahns@uni-potsdam.de, 0331 977-2494

Heiko Etzold, heiko.etzold@uni-potsdam.de, 0331 977-1068

Universität Potsdam
Institut für Mathematik
Karl-Liebknecht-Str. 24-25
14476 Potsdam

Weitere Informationen zum Projekt Digitales Lernen Grundschule in Potsdam
http://dlgs.uni-potsdam.de