Eckpunktepapier

PROGRAMMIEREN LERNEN ALS EINE WICHTIGE VORAUSSETZUNG UM DIGITALE KOMPETENZEN ZU ERLANGEN

PROGRAMMIEREN LERNEN IN DER SCHULE – ECKPUNKTE FÜR EIN PRAXISKONZEPT

von Veikko Krypczyk

Welche Lehrinhalte gehören in den informationstechnischen Unterricht in der Schule? Diese Frage wird auf den unterschiedlichsten Ebenen diskutiert. In diesem Papier geht es um die Frage, ob die Vermittlung von Programmierkenntnissen in der Schule stattfinden sollte. Mit Hilfe der Skizze eines Praxiskonzeptes wird dabei sowohl das Dilemma aus Zeitlosigkeit und Aktualität betrachtet. Es wird aufgezeigt, dass Basiskonzepte der Programmierung durchaus die Voraussetzungen erfüllen, auch langfristig in die Lehrpläne der Schulen aufgenommen zu werden.

Informationstechnischer Unterricht

Die Frage, welche Inhalte Gegenstand des modernen Informatikunterrichts an Schulen sein sollten, lässt sich nicht einfach beantworten. Grundsätzlich soll Informatikunterricht dazu befähigen, den Schülerinnen und Schülern die Grundlagen der Informationstechnologie, der elektronischen Datenverarbeitung und den sachgemäßen Umgang mit digitalen Medien zu vermitteln. Einig ist man sich insofern, dass eine grundlegende informationstechnische Ausbildung notwendig ist und in die modernen Lehrpläne der Schule gehört. Man kann jedoch unterschiedliche Standpunkte ausmachen, welche konkreten Inhalte nun auf theoretischer Ebene und auch in praktischer Hinsicht vermittelt werden sollten. Ein weiterer Diskussionspunkt ist die Frage danach, ob die Vermittlung von Wissen aus dem Bereich der Programmierung mit zum Lernstoff der Schule gehört. Diese Frage wird in diesem Papier aus unterschiedlicher Sichtweise erörtert und Eckpunkte für ein Praxiskonzept, basierend auf Erfahrungen abgeleitet.

Programmieren Lernen in der Schule – Ein Diskurs

Dieser Diskurs wird nicht nur in der didaktischen Fachwelt zur Schulbildung geführt, sondern er trägt auch in die Breite der Gesellschaft. Der CEO von Apple Tim Cook vertritt hierzu eine eindeutige Meinung: „Programmieren ist für Kinder wichtiger, als eine zweite Sprache zu lernen“. Doch es geht ihm weniger um den Programmcode an sich. Vielmehr argumentiert er: „Programmieren bringt Menschen bei, Dinge kritisch zu hinterfragen, es beruht zu einem großen Maß auf Kreativität. Es ist eine tolle Mischung aus Problemlösung, der Fähigkeit, kritisch zu denken, und Kreativität, die alle zu einem Gesamtpaket vereint werden.“ [CO21] Man findet auch andere Meinungen zum Thema. Der in der digitalen Welt gut vernetzte und häufig zitierte Kolumnist Sascha Lobo antwortet auf die Frage, ob Kinder in der Schule Programmieren lernen sollten, eher abweisend. Er führt als Begründung auf, dass die Fähigkeiten des Programmierens nicht essenziell für eine Grundlagenbildung sind. Konkret heißt

es: „Wer programmieren kann, kann programmieren – was aber dringend und immer schmerzlicher fehlt, ist ein Verständnis der Zusammenhänge einer digital vernetzten Welt und nicht ihrer kleinsten Bausteine. Es lässt sich grob mit dem Kenntnisunterschied zwischen einer Stadtplanerin und einem Maurer vergleichen, wenn man das Ziel hat, eine Stadt zu verstehen“. [LO17]

Zusätzlich wird als Argument angeführt, dass der schnelle Fortschritt in diesem Bereich – insbesondere getrieben durch den vermehrten Einsatz von künstlicher Intelligenz – keine ausreichende Stabilität der relevanten Lehrinhalte bieten würde. Vielmehr meint Lobo „Bildungspolitik muss auch in Jahrzehnten gedacht werden“ und kommt zu dem Schluss: „Die Chance ist nicht gering, dass im klassischen Takt deutscher Bildungspolitik im Spätsommer 2032 ein Rahmenplan "Programmieren in der Schule" feststünde, der eher digitalarchäologischen Kriterien genügt, als ein digitales Grundverständnis zu vermitteln.“ Ebenso wird als Argument angeführt, dass man die im Falle einer Etablierung der Vermittlung von Programmierfähigkeiten in der Schule „sich die Schulbildung weniger nach den Bedürfnissen der Schüler als viel mehr nach denen der Wirtschaft ausrichte“.

Diese beiden diametralen Pole zum Thema enthalten nach Meinung des Autors dieses Papiers sowohl richtige als auch weniger nachvollziehbare Argumente. In den folgenden Textabschnitten wird versucht, einen konzeptionellen Vorschlag für die Etablierung von Informatikunterricht mit der Spezialisierung „Programmieren lernen“ zu begründen und dabei eine mögliche Ausgestaltung in der Praxis zu skizzieren. Es geht also um eine Antwort auf die Frage, welche Inhalte in einen solchen Kurs gehören sollten und wie man damit das Dilemma aus „Aktualität und Zeitlosigkeit“ ggf. auflösen könnte. Die Praxisempfehlungen beruhen dabei auf persönlichen Lehrerfahrungen des Autors bei der Wissensvermittlung von grundlegenden Programmierkenntnissen. Weitere Argumente, welche „für“ oder „gegen“ das Erlernen des Programmierens in der Schule sprechen, findet man beispielsweise in einem Vortrag von Henry Herper der Fakultät für Informatik der Otto von Guericke Universität Magdeburg. [HE19]

Zum Aspekt der Zeitlosigkeit

Das Argument, dass die zu vermittelnden Fähigkeiten zum Programmieren nicht über eine längere Zeit konstant sind, gilt weitestgehend lediglich für die jeweils angewendete Technologie, wie Bibliotheken, Frameworks, konkrete Methoden und Vorgehensweisen usw. Hier ist in der Tat viel „Bewegung“ und eine Vermittlung dieses Wissens würde meist deutlich zu speziell und bereits am Ende der Schulzeit der Lernenden wieder veraltet sein. Etwas anderes gilt, wenn man sich auf die Grundkonzepte der Programmierung (Basics) bezieht (Abbildung 1).

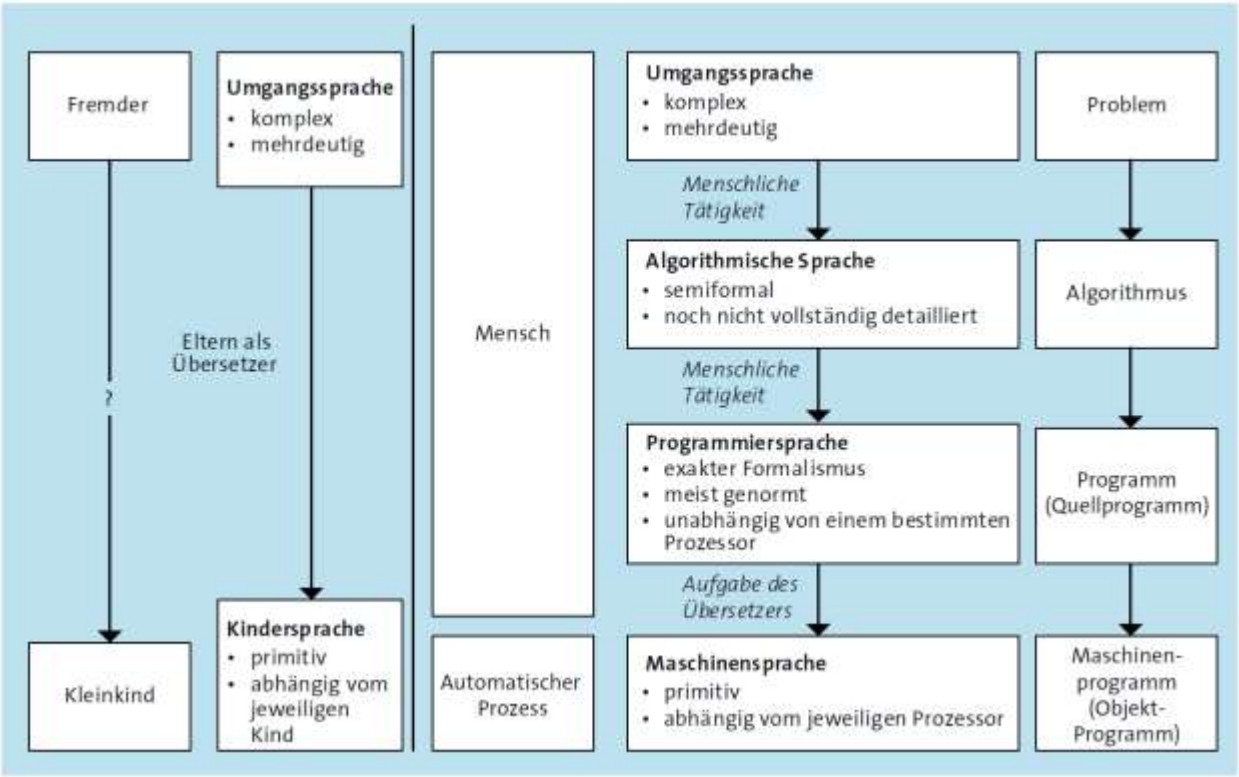

Abbildung 1: Mensch-Maschine-Kommunikation über die Programmiersprache. [BA05]

Schon seit den Anfängen der IT werden Probleme aus der realen Welt in Algorithmen transformiert, und damit in eine Lösung überführt. Die Basiselemente eines solchen Algorithmus sind seit den Anfängen der IT nahezu identisch und werden bestenfalls variiert, erweitert oder in einen anderen Kontext verwendet. Zu diesen Basiskonzepten gehören zweifelsfrei Konstrukte wie Schleifen, Bedingungen, Verzweigungen, die Arbeit mit Variablen, die Definition und Verwendung von Datentypen usw. Gleichgültig, welche Art von Anwendungen man entwickelt, welche Programmiersprache man verwendet oder welches Werkzeug man einsetzt, diese Basiskonzepte der Programmierung sind nahezu überall präsent und notwendig. Man kann auch formulieren, sie sind derart essenziell und können als Grundlagenwissen der Programmierung aufgefasst werden.

Eine Analogie zu einem anderen Fachgebiet soll diese These stützen. In der Mathematik werden grundlegende Methoden zur Berechnung von Nullstellen, zum Beispiel auf der Basis von Näherungsverfahren wie das Verfahren von Newton, vermittelt. Die Anwendung dieser Methoden ist in der Praxis nahezu nie notwendig. Zwar werden Nullstellen von komplexen funktionalen Zusammenhängen an vielen Stellen in der Praxis benötigt, jedoch wird niemand ernsthaft auf die Idee kommen, dazu eine manuelle Rechnung mit einem Näherungsverfahren einzusetzen. Einfachste Rechentechnik – zum Beispiel ein moderner Taschenrechner – hat die „manuelle“ Berechnung schon lange überflüssig gemacht. Dennoch ist es für das Gesamtverständnis notwendig, dass man dieses Wissen erlernt und geübt hat. Nur mit einem soliden Basiswissen besteht die Möglichkeit, die gesamte

Breite eines Themas zu erfassen, das Problem zu durchdringen und die Voraussetzungen für ein tiefergehendes Verständnis zu schaffen.

Übertragen wir dieses in den Bereich der Programmierung so sind die Basiskonzepte nicht nur weitgehend zeitlos, sondern sie fördern auch das logische Denken und das Verständnis der Gesamtzusammenhänge. Die Lernenden werden in die Lage versetzt, Lösungen nach einem algorithmischen Vorgehen eigenständig zu ermitteln. Man muss dabei sorgfältig schauen, welches Wissen aus dem Themenfeld der Programmierung man einem solchen Basiswissen zuordnen kann. Nach Ansicht des Autors kann die oben aufgeführte Liste der Begriffe beispielsweise noch um das Konzept der objektorientierten Programmierung erweitert werden. Die objektorientierte Programmierung gilt in sehr vielen Bereichen der Programmierung schon lange als Standard um Anwendungen zu erstellen. Die Modellierung von Problemstellungen als ein System von Klassen und Objekten zwingt zur Systematisierung. Dieses Paradigma der Programmierung wird von einer sehr großen Zahl von Programmiersprachen unterstützt und kann ebenfalls als Basic angesehen werden.

Basierend auf dieser Argumentation gibt es genügend Basisthemen der Programmierung, welche nicht oder nur minimal vom „Zeitgeist“ abhängig sind. Sie könnten das Fundament des Themas Programmierung bilden, wenn man diese Fähigkeit in die Lehrpläne eines informationstechnischen Unterrichts aufnehmen möchte.

Zum Aspekt der Motivation

Gerade im Informatikunterricht besteht oft der Wunsch der Schülerinnen und Schüler danach, aktiv am Computer etwas zu „machen“. Nur „trockener“ Theorieunterricht dürfte zu wenig Motivation führen. Findet informationstechnischer Unterricht in der Schule statt und gibt es auch entsprechende Hardware, dann sollte auch während der Programmierstunden aktiv am Computer gearbeitet werden. Moderne Werkzeuge der Softwareentwicklung erlauben es, dass man erste Programme nach wenigen Stunden eigenständig erstellen kann. Das erzeugt Motivation und die Schülerinnen und Schüler haben ein eigenes Programm erstellt. Dabei kann man bewusst eine Programmierumgebung wählen, in welcher eine Vielzahl von Aufgaben der Programmierung durch grafische Assistenten und auf dem Wege der Konfiguration erledigt werden können. Ein typisches Beispiel ist die Erstellung der grafischen Oberfläche mit Hilfe eines visuellen Designers. Dieses Vorgehen kann in wenigen Stunden eingeübt werden und kann dabei zu ersten Erfolgen führen. Die Ergebnisse werden zwar stets experimentellen Charakter haben, ein eintretender Erfolg wird aber wahrscheinlich häufig zu einer Steigerung der Motivation führen.

Zum Aspekt der Aktualität der Lehrinhalte

Ein wichtiger Trend in der Entwicklung von Programmen geht dahin, dass man so genannte RAD- und Low Code-Tools verwendet, um Programme auf sehr einfache Art zu erstellen. Hier werden Routineaufgaben auf dem Wege der grafischen Gestaltung – vergleichbar mit einem Zeichenprogramm – erledigt. Das erleichtert den Einstieg, Ergebnisse und Erfolge werden schneller erreicht. Darüber hinaus ist das Vorgehen keinesfalls unprofessionell, sondern es ist durchaus auch in der Praxis üblich hoch integrierte Tools zum Erstellen von Programmen zu nutzen. Im Informatikunterricht kann man sich dann wiederum darauf konzentrieren mit den vermittelten Basiskenntnissen und dem Grundlagenwissen erste Programme zu schreiben, d.h. zum Beispiel am Algorithmus zu arbeiten und nicht an der Gestaltung der Oberfläche zu verzweifeln.

Ein Fokus der Lehrdidaktik sollte zum Beispiel darauf liegen, für Schülerinnen und Schüler interessante und motivierende Beispiele zu erstellen. Beispiele aus dem Bereich der Erwachsenenbildung sind hier sicherlich nicht gut geeignet. Aber kleinere spielerische Anwendungen, können nebenbei für Spaß und Freude bei der Beschäftigung mit der Materie sorgen und gleichwohl wertvolles Wissen vermitteln.

Ein Vorschlag für eine Umsetzung des Themas Programmierung in der Schule

Im Folgenden soll basierend auf Praxiserfahrungen ein möglicher Rahmen zur Etablierung des Themas „Programmieren lernen“ im informationstechnischen Unterricht vorgestellt werden. Dabei stellen sich beispielsweise die folgenden Fragen:

- In welchem Alter und mit welchen Vorkenntnissen sollte dieses Thema angegangen werden?
- Welche Programmiersprache sollte man auswählen?
- Wie könnte eine Arbeitsumgebung ausgestattet sein, mit welcher die Schülerinnen und Schüler lernen können?
- Können weitere moderne mediale Hilfen als Lernquellen genutzt werden?

Die Frage, ab welchem Alter man mit dem „Programmieren lernen“ beginnen kann, richtet sich nach den Lehrzielen und den Methoden. Erste Schritte im Bereich der Programmierung können Kinder bereits im Grundschulalter erlernen. Systeme wie die visuelle Programmiersprache Scratch ermöglichen ein solches Vorgehen. Mit Scratch können sehr einfach digitale Geschichten, Spiele und Animationen erstellt werden. Die Scratch Foundation ist eine Non-Profit-Organisation, die die Sprache designt, entwickelt und moderiert. [SC23]

Dieses Papier fokussiert jedoch auf Schülerinnen und Schüler in höheren Klassenstufen. Um programmieren zu lernen, sind bestimmte Kenntnisse in Mathematik und Englisch notwendig. Mathematik ist wichtig, um ein Verständnis für Basiselemente des Programmierens, wie Variablen, Rechenoperationen usw., zu vermitteln. Sprachkenntnisse in Englisch sind eine Voraussetzung, da die Schlüsselwörter der Programmiersprachen nahezu alle aus der englischen Sprache stammen.

Welche Programmiersprache sollte angewendet werden? Auch dazu gibt es unterschiedliche Ideen und Meinungen. Das Landesmedienzentrum Baden-Württemberg führt beispielsweise Hinweise und Kurse zu den Programmiersprachen Scratch, Python und den Sprachen des Web, d.h. JavaScript, HTML und CSS auf. [PR23] Eine Nutzung dieser Programmier- und Auszeichnungssprachen wird also für möglich gehalten. Seit vielen Jahren wird von Praktikern die Nutzung der Programmiersprache Pascal für den Einstieg empfohlen und auch aktiv beim Lernen von Programmierfähigkeiten verwendet. Pascal wurde von Niklaus Wirth an der ETH Zürich als Lehrsprache eingeführt, um die strukturierte Programmierung zu lehren. Auch Herper nennt in seinem Vortrag beispielsweise Pascal als geeignete Programmiersprache für den Einstieg. [HE19] Für diese Programmiersprache spricht nicht nur ihre einfache und klare Syntax. Ein Algorithmus, welcher in Pascal formuliert ist, den verstehen auch Anfänger i.d.R. in kürzester Zeit. Für die Nutzung von Pascal sprechen auch beispielsweise die folgenden Argumente. Die Sprache wurde zu Object Pascal weiterentwickelt. Dabei handelt es sich nicht um eine neue Programmiersprache, sondern um mehrere teilweise miteinander kompatible Programmiersprachenderivate, die Pascal um objektorientierte Sprachmerkmale erweitern. Weitere Argumente, welche für den Einsatz von Pascal sprechen, ist die Verfügbarkeit von sehr guten Werkzeugen in Form von integrierten Entwicklungsumgebungen. In diesen Entwicklungsumgebungen können die Programme vollständig erstellt und ausgeführt werden. Beispiele für solche integrierten Entwicklungsumgebungen, welche Pascal bzw. Object Pascal verwenden, sind Lazarus und Delphi. Die integrierte Entwicklungsumgebung Lazarus wird unter einer Open Source-Lizenz bereitgestellt und steht für alle relevanten Zielsysteme, d.h. Windows, macOS und Linux zur Verfügung. [LA23]. Die Entwicklungsumgebung Delphi ist dagegen ein kommerzielles Produkt. Verfügbar ist jedoch eine kostenfreie Community Edition, welche gerade für Lehr und Lernzwecke ohne Einschränkungen eingesetzt werden kann. [DE23] Der Vorteil eines solchen proprietären Werkzeuges ist, dass man von allen Vorzügen einer professionellen integrierten Entwicklungsumgebung profitiert. Beispielsweise wird diese stetig weiterentwickelt und die Schülerinnen und Schüler können direkt mit den Werkzeugen der Profis arbeiten.

Beim Rückgriff auf kommerzielle Produkte könnte man der Meinung sein, dass man sich in eine zu große Abhängigkeit eines Herstellers begibt. Diesem Argument kann nur bedingt gefolgt werden. Letztendlich muss i.d.R. der Aufwand zur Entwicklung an Open Source-Projekten auch finanziell abgegolten werden, d.h. über Umwege sind auch an diesen Projekten meist kommerzielle Anbieter

beteiligt. Kommerzielle Anbieter bieten sehr oft auch eine weitergehende Unterstützung für Lehr- und Lernzwecke für ihre Produkte. Für die genannte Entwicklungsumgebung Delphi gibt es beispielsweise neben der kostenfreien Community Edition auch kostenfreie Klassenraumlizenzen.

So genannte akademische Programme von Softwareherstellern richten sich an die Lehrenden und Lernende und bieten eine weitere Unterstützung bei der Ausbildung. Das sind beispielsweise aufbereitete Programmierbeispiele, vorbereitete Foliensätze, E-Books usw. Diese erleichtern nicht nur den Einstieg, sondern können auch die stets knapp bemessene Zeit zur Unterrichtsvorbereitung deutlich verkürzen.

Ein weiteres Argument spricht dafür, dass man sich für Programmiersprache, Werkzeuge usw. entscheidet, welche auch im professionellen Umfeld verwendet werden. Das Thema „Programmieren lernen“ bietet auch das Potenzial zu einer echten Motivation und einem wirklichen Interesse. Nutzt man sofort die passenden Tools, dann schränkt man die Möglichkeiten nicht ein, wenn man die Interessen bei einigen Schülerinnen und Schülern geweckt hat. Mit anderen Worten: Das nahezu unendliche Themenfeld der Programmierung steht den Interessierten dann jederzeit offen und wird nicht durch Einschränkungen in den Arbeitsmitteln begrenzt.

Der letzte Punkt in dieser Abhandlung wirft noch einen weiteren Blick auf die Art und Weise der Wissensvermittlung. Die Schülerinnen und Schüler wachsen heute mit einer Omnipräsenz der digitalen Medien auf. Statt Lehrbücher ist der Konsum von Videos in den einschlägigen Plattformen, wie YouTube zur gängigen Lernmethode geworden. Sucht man beispielsweise nach dem „Satz des Pythagoras“ in Google und den Einträgen unterhalb der Sektion Videos, dann bekommt man über 32.500 Ergebnisse präsentiert. Die Schülerinnen und Schüler nutzen diese Angebote. Gerade im Themenfeld der Programmierung kann man auf diese Quelle umfassend setzen. Unzählige Lernquellen können genutzt werden. Das Spektrum reicht von didaktisch aufbereiteten Lernvideos, welche die Schülerinnen und Schüler besser erreichen dürften als beispielsweise klassische Lehrbücher. Ein Beispiel für das Erlernen von Object Pascal sind die Lernvideos „Delphi Lernen mit Leon“. [DE23b] Weitere online verfügbare Quellen sind Communities, zum Beispiel in Form von Diskussionsforen bis hin zu Dokumentationen. „Programmieren lernen“ ist ein Themengebiet, wo diese moderne Form der Wissenssuche und des Lernens guten Gewissens angewendet werden kann. Im Bereich der IT sind die Quellen im Netz denen der gedruckten Presse meist in Vielfalt, Aktualität und Spezifik überlegen.

FAZIT

Die Diskussion um die Integration des „Programmieren Lernens“ in den Schulunterricht hält schon zu lange an. Der ehemalige Wirtschaftsminister Siegmar Gabriel ließ bereits 2014 verlauten:

„Programmiersprachen gehören zu den Sprachen des 21. Jahrhunderts. Es gibt viele Wege, wie wir Kinder und Jugendliche für das Programmieren begeistern können – der Schulunterricht ist nur einer davon. Für mich wäre eine der Möglichkeiten, Programmiersprachen als zweite Fremdsprache in Schulen anzubieten.“ [GA14] Fast eine Dekade später erscheint die Diskussion noch immer nicht die Reife zu haben, dass man den Informatikunterricht im Allgemeinen und das „Programmieren lernen“ im Speziellen in die Lehrpläne der Schulen flächendeckend integriert hat. Dabei kann das vermeintliche Dilemma aus der Aktualität und der Zeitlosigkeit der Lehrinhalte gelöst werden. Basiskonzepte der Programmierung sind über viele Jahre etabliert und können durchaus als Grundlagenwissen aufgefasst werden. Trotz dieser Fokussierung auf diese Grundlagen, können mit diesen Mitteln bereits ansprechende Ergebnisse erreicht werden.

Literaturverzeichnis

[BA05] Balzert, Helmut: Lehrbuch Grundlagen der Informatik, Spektrum Akademischer Verlag, 2005

[CO21] Cook, Tim: Programmieren ist für Kinder wichtiger, als eine zweite Sprache zu lernen, Zeitschrift Capital, https://www.capital.de/wirtschaft-politik/programmieren-ist-fuer-kinder-wichtiger-als-eine-zweite-sprache-zu-lernen (13.02.2023)

[GA14] Gabriel, Siegmar: Computersprache soll Schulfach werden, in Rheinische Post, Ausgabe vom 25.09.2014, https://rp-online.de/politik/deutschland/sigmar-gabriel-computersprache-soll-schulfach-werden_aid-9190833

[DE23] Homepage der integrierten Entwicklungsumgebung Delphi: https://www.embarcadero.com/de/products/delphi (09.02.2023)

[DE23a] Homepage des akademischen Programms von Embarcadero, https://www.embarcadero.com/de/development-tools-for-education-dach (02.02.2023)

[DE23b] Delphi lernen mit Leon, YouTube, https://www.youtube.com/watch?v=yY9-sc38RA4 (14.02.2023)

[HE19] Herper, Henry: Didaktik der Informatik. Programmiersprachen im Informatikunterricht, https://mttcs.ovgu.de/mttcs_media/Lehre/Didaktik+des+Informatikunterrichts+II/DDI_2+_+Downloadbereich/Thema_DDI_2_5_Programmiersprachen.pdf (05.02.2023)

[LA23] Homepage der integrierten Entwicklungsumgebung, https://www.lazarus-ide.org/ (08.02.2023)

[LO17] Lobo, Sascha: Programmieren lernen hilft nicht. Digitalverständnis von Schülern, in Spiegel Netzwelt, https://www.spiegel.de/netzwelt/web/programmieren-in-der-schule-sollen-kinder-programmieren-lernen-kolumne-a-1140928.html (13.02.2023)

[PR23] Programmieren. Handreichungen des Landesmedienzentrums Baden-Württemberg, https://www.lmz-bw.de/medienbildung/medienbildung-an/weiterfuehrende-schulen/mint-portal/informatik/3x3-mediendossiers/programmieren

[SC23] Homepage des Tools Scratch, https://scratch.mit.edu/ (10.02.2023)