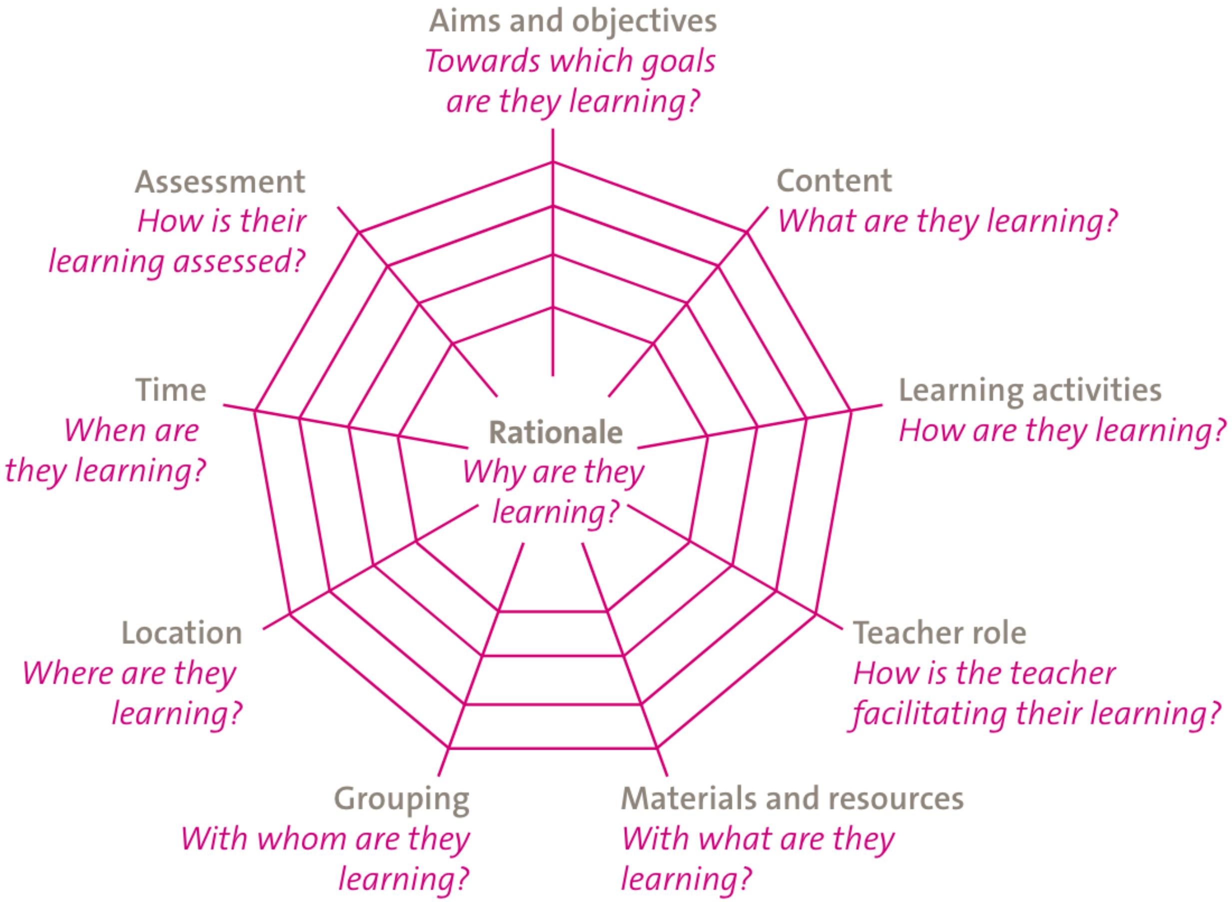
FAIBLE.nrw

Didaktik der Programmierung

UNIVERSITÄT PADERBORN

Notional Machine
Modellieren
Projektunterricht
Bildung
Pair Programming
PRIMM
Programmiersprachen
Software Life Cycle
4C/ID
Kompetenzen
Tools
Softwareentwicklung
Problemfelder
Lernroboter
Unterrichtsmethoden
Implementieren
Lernerfolg- und Leistungsbewertung
Worked Example
Inhalte
Programmverstehen
Lernziele
Digitale Welt
Debugging
Cognitive Load
STREAM
Block-basierte Sprachen
Computational Thinking
Programmieren
Use-Modify-Create
Programmierkonzepte
Computational Notebooks
Softwareprojekte
Lehrpläne

Kapitelübersicht



The curricular Spiderweb

(Van den Akker, 2013)

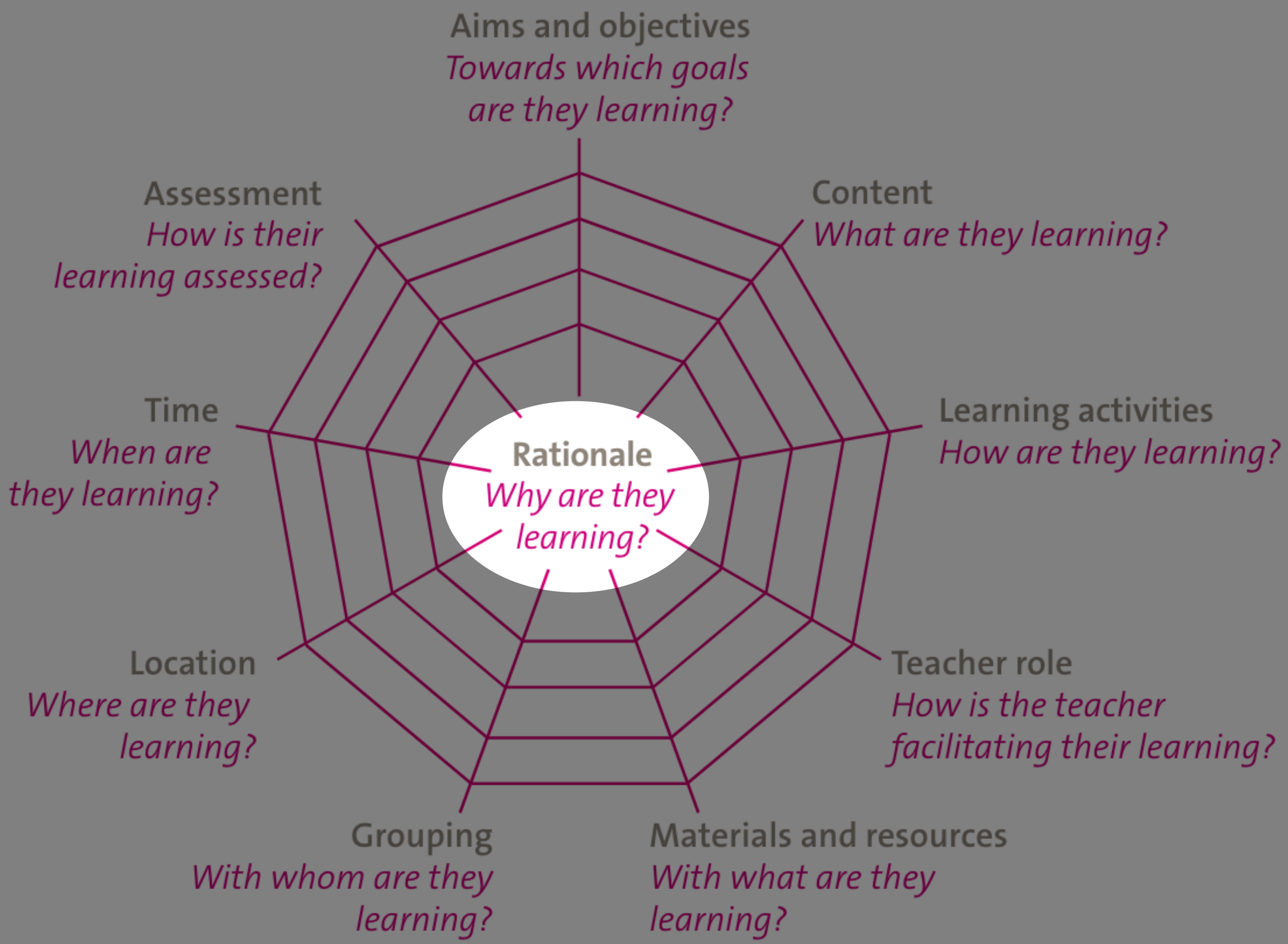
Van den Akker, J. (2013). Curricular development research as specimen of educational design research. Educational design research, 53–70.

1. Einführung

Die Rolle von Programmieren für das Fach Informatik

Warum sollen die Schüler:innen Programmieren lernen?

Welchen Beitrag kann Programmieren zur (Allgemein-)Bildung leisten?


Aims and objectives
Towards which goals are they learning?
Content
What are they learning?
Learning activities
How are they learning?
Teacher role
How is the teacher facilitating their learning?
Materials and resources
With what are they learning?
Grouping
With whom are they learning?
Location
Where are they learning?
Time
When are they learning?
Assessment
How is their learning assessed?
Rationale
Why are they learning?

- Stellen Sie sich vor, Sie sind Informatiklehrkraft und stehen vor Ihrer Klasse.
- Eine Schülerin kommt auf Sie zu und fragt, warum sie in der Schule Programmieren lernen muss.
- In Zeiten von KI-Assistenten wie CoPilot und ChatGPT: Müssen Schülerinnen und Schüler überhaupt noch Programmieren lernen?

Was sind Ihrer Meinung nach Gründe dafür Programmieren zu lernen?

Häufig genannte Gründe

- Berufliche Relevanz
  - Programmierkenntnisse sind in vielen Berufsfeldern nützlich oder erforderlich
- Computational Thinking (CT)
  - Erwerb einer allgemeinen Abstraktions- und Problemlösekompetenz
  - Fähigkeit zum algorithmischen Denken
- Digitalkompetenz und Technologieverständnis
  - Fortgeschrittene Form der Interaktion mit Informatiksystemen
  - Programmieren kann dabei helfen, die digitale Welt zu verstehen und zu formen
- Automatisierung
  - Mit Hilfe von Programmen können automatisierte Lösungen für wiederkehrende Probleme geschaffen werden (Alltag & Beruf)

- Kreativität
  - Programmieren als Ausdrucksmittel (z.B. Creative Coding)
  - Spieleprogrammierung
- Intrinsische Motivation
  - Programmieren macht Spaß
  - Selbstbestimmungs-Theorie *(Deci & Ryan, 2008)*
    - Kognitive Herausforderung
    - Soziale Eingebundenheit
      - Programm kann einen Nutzen / gesellschaftlichen Wert schaffen
      - Programmieren im Team / in Projekten
    - Autonomie
      - Programmieren als selbstgesteuerter Prozess
      - Ziele und Vorgehen können selbst bestimmt werden

Deci, E. L., & Ryan, R. M. (2008). Self-Determination Theory. [DOI]

Die Rolle der Programmierung

Welche Bedeutung hat Programmieren für das Fach Informatik?

Ein Blick in die Lehrpläne

1 Aufgaben und Ziele des Faches

Gegenstand der Fächer im mathematisch-naturwissenschaftlich-technischen Aufgabenfeld (III) sind die empirisch erfassbare, die in formalen Strukturen beschreibbare und die durch Technik gestaltbare Wirklichkeit sowie die Verfahrens- und Erkenntnisweisen, die ihrer Erschließung und Gestaltung dienen.

Innerhalb der von allen Fächern zu erfüllenden Querschnittsaufgaben tragen insbesondere auch die Fächer des mathematisch-naturwissenschaftlich-technischen Aufgabenfeldes im Rahmen der Entwicklung von Gestaltungskompetenz zur kritischen

…

Methoden zur Erforschung komplexer Phänomene und für die Entwicklung komplexer Systeme bereit, die zahlreiche andere Fachdisziplinen aufgreifen und adaptieren. Daher ist die Informatik in hohem Maße interdisziplinär ausgerichtet. Die Auseinandersetzung mit Themen und Methoden der Informatik in der Schule dient somit der Lebensvorbereitung und Orientierung in einer von der Informationstechnologie geprägten Welt.

Eine erkenntnistheoretische Perspektive

auf die Programmierung

„Programming is the essence of computing / informatics. Indeed, computing is much more than programming, but programming […] is essential to computing.“ *(Caspersen, 2023)*

„If we don't teach programming, we don't teach programming languages.
→ We are changing the way we understand and express our discipline.“ *(Martini, 2024)*

Caspersen, M. E. (2023). Principles of programming education. *Computer Science Education: Perspectives on Teaching and Learning in School*, 219.
Martini, S. (2024). Teaching programming in the age of generative AI. Proceedings of the 2024 on innovation and technology in computer science education V. 1, 1–2. https://doi.org/10.1145/3649217.3653527

Abstraktionsebenen im Laufe der Zeit

„As the level of abstraction in computing education [...] steadily arose, the credo among computing cognoscent / became that one needs to be familiar with at least one abstraction level below that at which one is working.“ *(Tedre et al., 2021)*

- 1950 zugrundeliegende Elektronik
- 1960 oktaler Maschinencode für Assembler
- 1970 Assembler für Hochsprachen

Tedre, M., Toivonen, T., Kahila, J., Vartiainen, H., Valtonen, T., Jormanainen, I., & Pears, A. (2021). Teaching Machine Learning in K–12 Classroom: Pedagogical and Technological Trajectories for Artificial Intelligence Education. IEEE Access, 9, 110558–110572. https://doi.org/10.1109/ACCESS.2021.3097962

Quelltext als Kommunikationsmedium einer Lösung oder eines Modells

„Programs are not text; they are hierarchical compositions of computational structures and should be edited, executed, and debugged in an environment that consistently acknowledges and reinforces this viewpoint.“ *(Teitelbaum, 1981)*

„Programs are meant to be read by humans and only incidentally for computers to execute.“ *(Knuth, 1997)*

Teitelbaum, T., & Reps, T. (1981). The Cornell program synthesizer. 24(9).
Knuth, D. E. (1997). The art of computer programming (Bd. 3). Pearson Education.

Aufgabe

Suchen Sie im Internet nach den **Lehrplänen** der Bundesländer für das Fach Informatik. Was wird dort über die **Rolle von Programmieren für das Fach Informatik** gesagt?

Einzel- oder Partnerarbeit

10 Minuten

Kernlehrplan Informatik, Saarland (2006)

„Das Programmieren ist eine **Primärerfahrung der Informatik**. Daraus erwachsen zentrale Begriffe wie Algorithmen und Datenstrukturen, Komplexität, formale Sprachen oder Compiler. Die Programmierung soll deshalb weiterhin eine zentrale Position in der Schulinformatik einnehmen. **Fehlinterpretationen** der vergangenen Jahre, die zu einem Informatikunterricht in Form eines **erweiterten Programmierkurses** führten, sollen jedoch vermieden werden. Im Vordergrund steht das **Problemlösen (Modellieren und Strukturieren)** unter Anwendung von Informatikprinzipien und Methoden. Die **Programmiersprache ist ein Mittel zum Zweck**."

Kernlehrplan Informatik, Sek II NRW (2014)

„Die **Umsetzung eines informatischen Modells in ein lauffähiges Informatiksystem** hat für Schülerinnen und Schüler nicht nur einen **hohen Motivationswert,** sondern ermöglicht ihnen auch die **eigenständige Überprüfung** der Angemessenheit und Wirkung des Modells im Rückbezug auf die Problemstellung. Im Unterricht lassen sich umfangreiche Informatiksysteme nur in arbeitsteiliger **projekt-orientierter Zusammenarbeit im Team** erstellen. Solche Projekte können nur gelingen, wenn die gemeinsame Arbeit strukturiert geplant und organisiert wird. Insgesamt leistet das Fach Informatik in der gymnasialen Oberstufe damit einen **wichtigen Beitrag zu einer erweiterten Allgemeinbildung** und **allgemeinen Studierfähigkeit** der Schülerinnen und Schüler.“

Bildungsplan Informatik Sek II Baden-Württemberg (2020)

„Programmieren als Realisierung von Ideen in Software als **schöpferischer und produktiver Prozess** ist ein **wesentlicher Bestandteil des Informatikunterrichts**. Die Schülerinnen und Schüler entwerfen **Problemlösungen**, die auf grundlegenden Programmierbausteinen basieren und erfahren so, dass die Lösung nicht in den Bausteinen selbst, sondern hauptsächlich in der Art und Weise ihrer Anordnung liegt. Grundsätzlich bieten Programmieraufgaben die Chance, dass Schülerinnen und Schüler die Arbeitsergebnisse anhand des Programmablaufs beziehungsweise Ergebnisses **selbstständig und unabhängig** von der Lehrkraft **überprüfen** können. Hier ist eine behutsame Heranführung durch die Lehrkraft erforderlich, damit die Schülerinnen und Schüler lernen, diese objektive Rückmeldung zur Weiterentwicklung ihrer Lösung zu nutzen.“

Diskussion

- Programmieren wird häufig **nicht** als unmittelbares Bildungsziel angesehen
  - Stattdessen: Abstraktions- und Problemlösefähigkeit (Computational Thinking), Organisations- und Teamfähigkeit (Softwareprojekte)
  - Programmieren dient lediglich als Mediator (Vermittler)
- Transfer-Problem *(Eberle, 1996)*
  - Die o.g. Fähigkeiten (allgemeine Problemlösefähigkeit) können auch ohne den Einsatz von Programmieren erworben werden (z.B. im Matheunterricht)
  - Worin liegt also der Mehrwert im Programmieren? Was macht Programmieren besonders?

Eberle, F. (1996). *Didaktik der Informatik bzw. Einer informations-und kommunikationstechnologischen Bildung auf der Sekundarstufe II*.

Exkurs

Computational Thinking

Computational Thinking

(Wing, 2008)

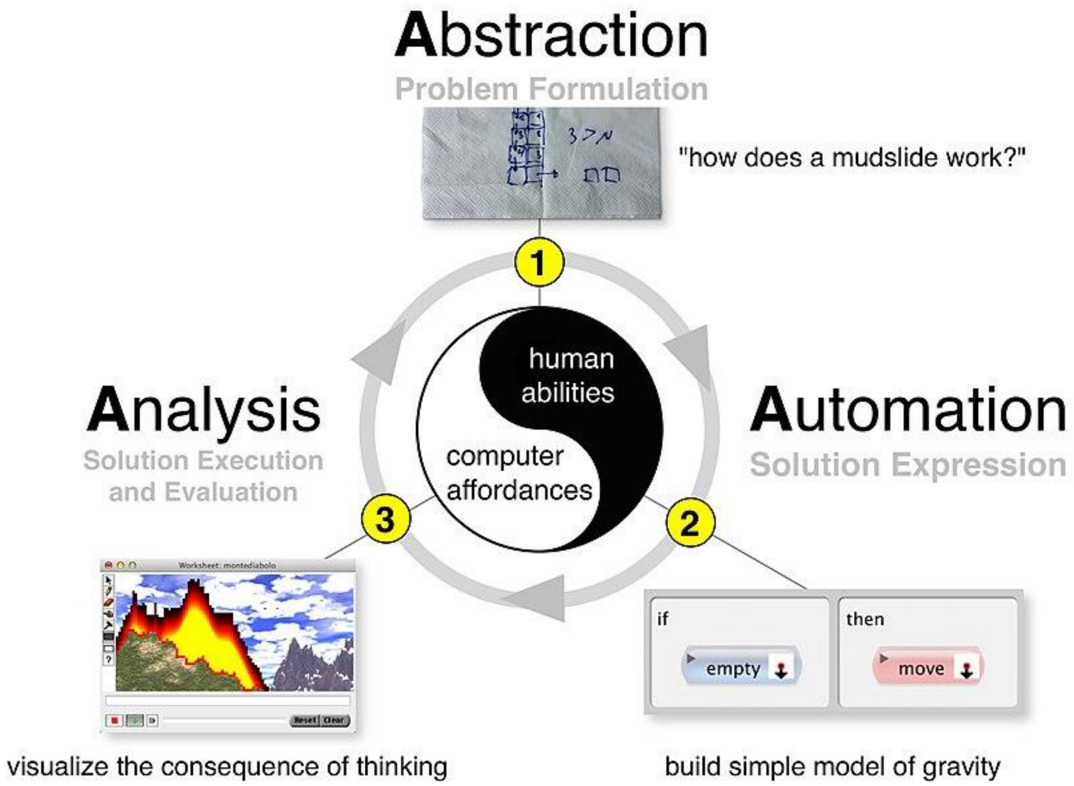
- Abstraktions- und Problemlösekompetenz
- Analytische Denkweise
  - Beinhaltet Aspekte des mathematischen, technischen und wissenschaftlichen Denkens
- Charakteristiken
  - Dekomposition (Zerlegung in Teilprobleme)
  - Mustererkennung (Pattern Recognition)
  - Datenstrukturen (Data Representation)
  - Abstraktion / Generalisierung (Generalization / Abstraction)
  - Algorithmen

Wing, J. M. (2008). Computational thinking and thinking about computing. [DOI]

Die drei As

(Repenning et al., 2016)

- Abstraktion
  - Formulierung des Problems
- Automatisierung
  - Lösung notieren (Programmieren)
- Analyse
  - Lösung ausführen und evaluieren



„The Computational Thinking Process“ von KaptainFire unter der Lizenz CC BY-SA 4.0 DEED via Wikipedia

Repenning, A., Basawapatna, A., & Escherle, N. (2016). Computational thinking tools. [DOI]

Programmieren und (Allgemein-)Bildung

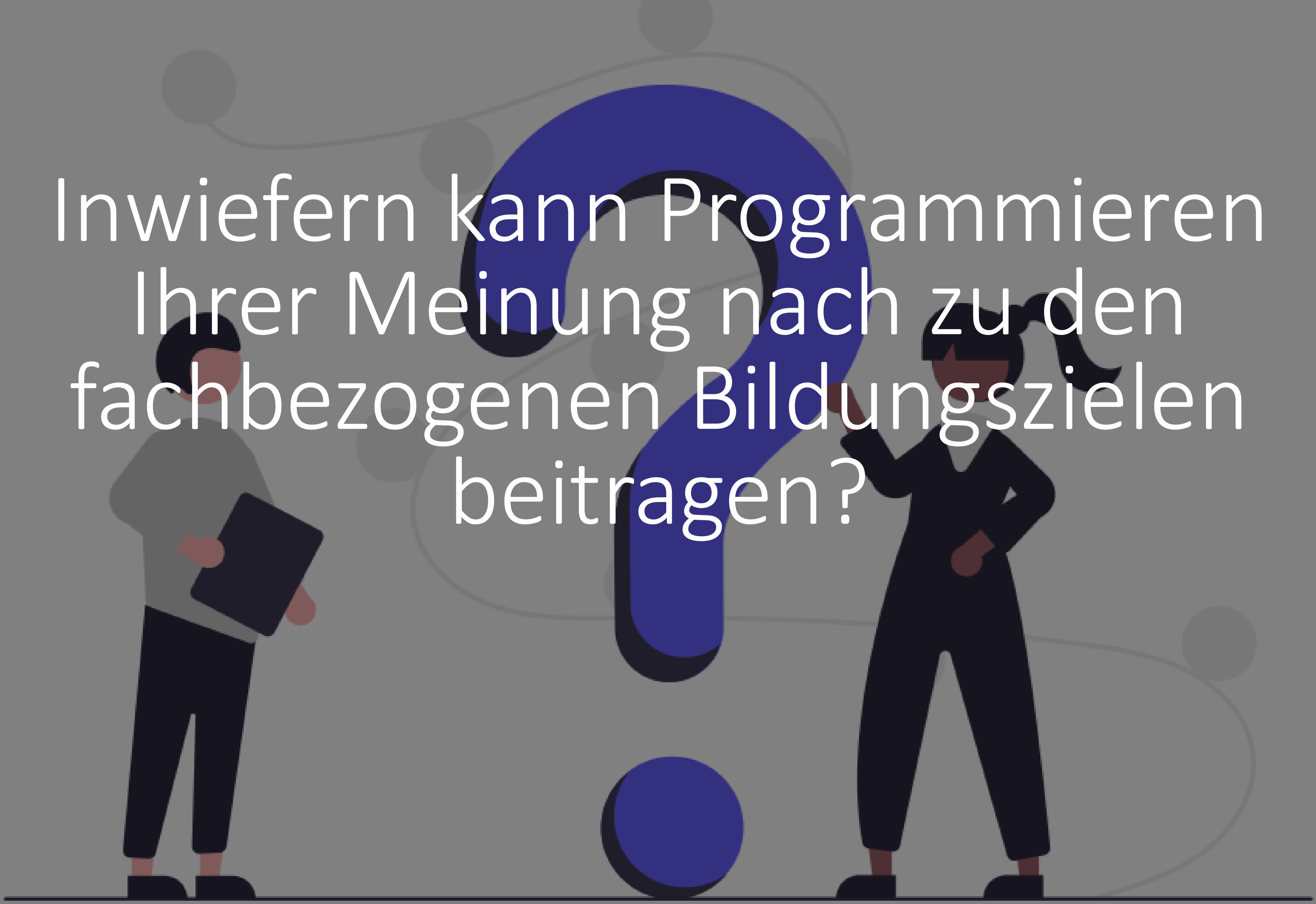
Welchen Bildungswert hat Programmieren?

Rahmen zur Beschreibung von fachbezogenen Bildungszielen

Gesellschaft für Fachdidaktik (GFD), 2009

- Identitätsbildung
    - Kompetenzen, die einen selbstbestimmten und reflektierten Zugang zu sich selbst und zur Welt eröffnen, so dass biografische Entwicklungsphasen erfolgreich bewältigt werden können
- Alltagsbewältigung
    - Kompetenzen, die im Alltag handlungsrelevant sind und über "Alltagswissen" oder "Allgemeinwissen" hinausgehen
- Ausbildungsreife
    - Kompetenzen für eine verantwortliche Berufstätigkeit auf der Grundlage basaler Kulturtechniken
- Partizipation
    - Kompetenzen, um am gesellschaftlichen Diskurs teilzuhaben und zusammen mit anderen – im Sinne sozialer Kohäsion – begründet zu handeln Rolle des Programmierens für Bildung

Inwiefern kann Programmieren Ihrer Meinung nach zu den fachbezogenen Bildungszielen beitragen?



Mögliche Aspekte

(vgl. Schulte, 2013)

- Programmieren als wichtige Fähigkeit im Beruf *(Ausbildungsreife)*
- Programmieren als „Tool Building“
  - Schaffung digitaler Artefakte als Beitrag zur (digitalen) Gesellschaft *(Partizipation)*
  - Werkzeuge zur Bewältigung von Alltagsproblemen *(Alltagsbewältigung)*
- Programmieren als Möglichkeit der Selbstverwirklichung / Ausdrucksmittel
  - Ein Programm ist Ausdruck der eigenen Ideen und Herangehensweisen *(Identitätsbildung)*
  - Programmcode wird von anderen Menschen gelesen und kommuniziert eine Lösung für ein Problem (Medium)
- Die (digitale) Welt verstehen und verändern
  - Programmieren kann dabei helfen die digitale Welt und dessen Einfluss auf die Gesellschaft zu verstehen und diese nach individuellen Bedürfnissen zu formen *(Partizipation & Alltagsbewältigung)*
  - „Shaping and at the same time being shaped“ *(Schulte & Budde, 2018)*
- Indirekte Faktoren
  - Organisations-, Kommunikations- und Teamfähigkeit (v.a. in Softwareprojekten)

Schulte, C. (2013). Reflections on the role of programming in primary and secondary computing education. [DOI]
Schulte, C., & Budde, L. (2018). *A Framework for Computing Education: Hybrid Interaction System.*

„Ich habe in Word programmiert, dass im Inhaltsverzeichnis automatisch alle Überschriften aufgelistet werden!“

„Ich habe über das Terminal eine SSH Verbindung zu einem Server aufgebaut und dort ein Programm installiert!“

Was ist eigentlich Programmieren?

Definition und Merkmale

„Wir programmieren in der Schule gar nicht richtig! Wir verwenden stattdessen Scratch.“

„Ich habe ChatGPT programmiert, mir immer zu widersprechen!“

„Ich habe mit WordPress eine Webseite erstellt!“

Zitate von Schüler:innen im IU

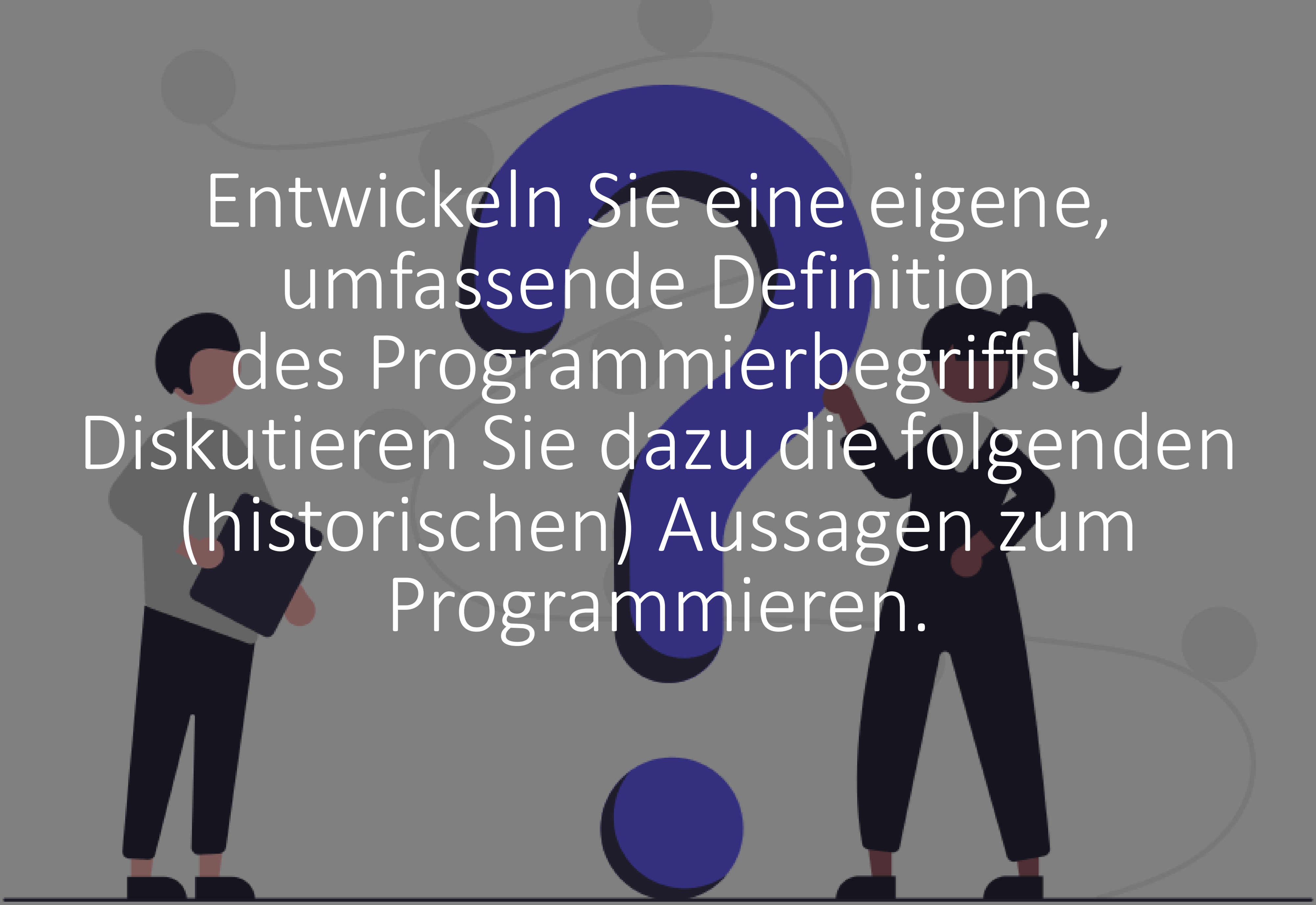
(Braune & Mühling, 2020)

Schüler:innen im Informatikunterricht

- „Scratch only, well, in the real world there are completely different programs that have no relation to Scratch.“
- „I would say that, for example, we have at our home – my father programmed something and as soon as the house reaches a certain temperature, the heating starts. And something like this is pretty useful but you cannot program such a thing with, for example, Scratch.“

Gibt es so etwas wie „echtes“ Programmieren?

Braune, G., & Mühling, A. (2020). Learning to program: The gap between school world and everyday world. [DOI]



Entwickeln Sie eine eigene, umfassende Definition des Programmierbegriffs!
Diskutieren Sie dazu die folgenden (historischen) Aussagen zum Programmieren.

Zitate zum Programmierbegriff I

(Blackwell, 2002)

- “The process of preparing a calculation for a machine can be broken down into two parts, 'programming' and 'coding'. Programming is the process of drawing up the schedule of the sequence of individual operations required to carry out the calculation” *(Hartree 1950, p. 111)*
- “The sequence of orders is known as the programme, and the machine performs it automatically without intervention from the user” *(Wilkes 1956, p. 2)*
- “Programming [...] is basically a process of translating from the language convenient to human beings to the language convenient to the computer” *(McCracken 1957)*
- “The process of organizing a calculation can be divided into two parts – the mathematical formulation and the actual programming [...] translating [...] into the language of the computing machine” *(Booth 1958)*

Blackwell, A. F. (2002). What is Programming? Annual Workshop of the Psychology of Programming Interest Group. [DOI]
Hartree, D. R. (1950): Calculating Instruments and Machines.
Wilkes, M. V. (1956). Automatic Digital Computers.
McCracken, D.D. (1957). Digital computer programming. Wiley.
Booth, K. H. V. (1958): Programming for an Automatic Digital Calculator.

Zitate zum Programmierbegriff II

(Blackwell, 2002)

- “[The] sequence [of basic operations] is called the program and the process of preparing it is called programming” *(Wrubel 1959, p. 4)*
- “[…] to write a sequence of coded instructions fed into a computer [...] to arrange data in a suitable form so that it can be processed by a computer [...] to feed a program into a computer” *(Collins Online Dictionary)*
- “Programming is a human activity that is a great challenge” *(Hoc, Green, Samurcay & Gilmore 1990, p. 3)*
- “Programming is an exceedingly diverse activity” *(Green 1990, p. 21)*
- “Programming is a complex cognitive and social task” *(Pennington & Grabowski 1990, p. 47)*

Blackwell, A. F. (2002). What is Programming? Annual Workshop of the Psychology of Programming Interest Group. [DOI]
Wrubel, M. H. (1959): A Primer of Programming for Digital Computers.
Hoc, J.-M.; Green, T. R. G.; Samurcay, R. & Gilmore, D. J., Psychology of Programming.
Green, T. R. G. (1990): The Nature of Programming. [DOI]
Pennington, N. & Grabowski, B. (1990): The Tasks of Programming. [DOI]

Verlust der direkten Manipulation

- Direkte Manipulation *(Shneidermann, 1983)*
  - Ständige Sichtbarkeit aller relevanten Objekte (z.B. Zellen in Excel)
  - Eingabe als physische Aktion (z.B. Auswahl mit der Maus) - anstelle von syntaktischen Befehlen
  - Schnelle und schrittweise Eingabeaktionen, umkehrbar
  - Direkte Sichtbarkeit von Effekten
- Charakteristiken von Programmieren *(Blackwell, 2002; Schulte, 2013)*
  - **Automatisierung** (spätere oder wiederholte Verwendung des Artefakts)
  - Besondere Art der Interaktion mit dem Computer, verbunden mit dem **Verlust der direkten Manipulation**
  - **Abstrakte Notation**
  - **Unmittelbare Rückmeldung fehlt** (oft)
  - **Bewältigung von Komplexität** in einer spezifischen Form (z.B. durch Aufteilung in Funktionen)
  - Ein Programm wendet sich nicht nur an den Computer als "Leser", sondern auch an einen menschlichen Leser (**Kommunikationsprozess**)
  - **Kreativer Prozess** des Suchens und Findens von Lösungen.
  - Programme können für den eigenen Gebrauch oder für die **Nutzung durch Andere** geschrieben werden.

Shneiderman, B. (1983). Direct manipulation: A step beyond programming languages. Computer, 16(08), 57–69. [DOI]
Blackwell, A. F. (2002). What is Programming? Annual Workshop of the Psychology of Programming Interest Group. [DOI]
Schulte, C. (2013). Reflections on the role of programming in primary and secondary computing education. [DOI]

Referenzen I

- Blackwell, A. F. (2002). What is Programming? Annual Workshop of the Psychology of Programming Interest Group. https://doi.org/10.5040/9781501325670.ch-001.
- Booth, K.H.V. (1958). Programming for an automatic digital computer. Butterworth.
- Braune, G., & Mühling, A. (2020). Learning to program: The gap between school world and everyday world. Proceedings of the 15th Workshop on Primary and Secondary Computing Education, 1–9. https://doi.org/10.1145/3421590.3421597.
- Deci, E. L., & Ryan, R. M. (2008). Self-Determination Theory: A Macrotheory of Human Motivation, Development, and Health. Canadian Psychology / Psychologie canadienne, 49(3), 182–185. https://doi.org/10.1037/a0012801.
- Eberle, F. (1996). Didaktik der Informatik bzw. Einer informations-und kommunikationstechnologischen Bildung auf der Sekundarstufe II. Verlag für Berufsbildung Sauerländer.
- Green, T. R. G. (1990). The Nature of Programming. In Psychology of Programming (S. 21–44). Elsevier. https://doi.org/10.1016/B978-0-12-350772-3.50007-0.
- Hartree, D. R. (2012). Calculating instruments and machines. Cambridge University Press.
- Hoc, J.-M., Green, T.R.G., Samurcay, R. and Gilmore, D.J. (Eds) (1990). Psychology of programming. Academic Press.
- Pennington, N., & Grabowski, B. (1990). The Tasks of Programming. In Psychology of Programming (S. 45–62). Elsevier. https://doi.org/10.1016/B978-0-12-350772-3.50008-2.

Referenzen II

- McCracken, D.D. (1957). Digital computer programming. Wiley.
- Repenning, A.; Basawapatna, A. & Escherle, N. (2016). Computational thinking tools. In IEEE Symposium on Visual Languages and Human Centric Computing (VL/HCC), S. 216 – 222. https://doi.org/10.1109/VLHCC.2016.7739688.
- Schulte, C. (2013). Reflection on the role of programming in primary and secondary computing education. In Caspersen, M. E.; Knobelsdorf, M. & Romeike, R., WiPSE'13: Proceedings of the 8th Workshop in Primary and Secondary Computing Education (S. 17 – 24). New York: ACM Press. https://doi.org/10.1145/2532748.2532754.
- Schulte, C., & Budde, L. (2018). A Framework for Computing Education: Hybrid Interaction System: The Need for a Bigger Picture in Computing Education. Proceedings of the 18th Koli Calling International Conference on Computing Education Research. https://doi.org/10.1145/3279720.3279733
- Shneidermann, B. (1983). Direct manipulation: A step beyond programming languages Computer. In Computer 16 (08) (S. 57 – 69). Washington: IEEE Computer Society Press. https://doi.org/10.1109/MC.1983.1654471.
- Van den Akker, J. (2013). Curricular development research as specimen of educational design research. Educational design research, 53–70.
- Wilkes, M. V. (1956). Automatic Digital Computers.
- Wrubel, M. H. (1959): A Primer of Programming for Digital Computers. New York: McGraw-Hill Book.
- Wing, J. M. (2008). Computational Thinking and thinking about computing. In Discussion Meeting Issue 'From computers to ubiquitous computing, by 2020' organized by Marta Kwiatkowska, Tom Rodden and Vladimiro Sassone 366 (80), S. 3717 - 3725. https://royalsocietypublishing.org/doi/10.1098/rsta.2008.0118.

Illustrationen

- Illustrationen von Limpitsouni, K. unter freier Lizenz via https://undraw.co

FAIBLE.nrw

Beteiligte Hochschulen:

- RWTH-Aachen
- Westfälische Wilhelms-Universität Münster
- Universität Duisburg-Essen
- Universität Bonn
- Universität Paderborn
- Technische Universität Dresden
- Carl von Ossietzky Universität Oldenburg

ORCA.nrw
Das Landesportal für Studium und Lehre.

Ein Kooperationsvorhaben empfohlen durch die:
DIGITALE HOCHSCHULE NRW
INNOVATION DURCH KOOPERATION

gefördert durch:
Ministerium für
Kultur und Wissenschaft
des Landes Nordrhein-Westfalen