20x INFOS – 40 Jahre Informatikunterricht

Werner Hartmann[1]

Abstract: Seit der Durchführung der ersten Tagung in Berlin 1984 waren Diskussionen zur Einführung eines Fachs Informatik, zu Lehrplänen und Bildungsstandards und zur Abgrenzung Informatik versus Medienbildung wiederkehrende Themen an den INFOS-Tagungen. Die Fachdidaktik Informatik bewegte sich dabei im Spannungsfeld zwischen den Ansprüchen an Forschung und Entwicklung seitens der Hochschulen, der Realität in der Schulpraxis und der raschen Entwicklung der Informatik selbst. Basierend auf den Erfahrungen der letzten 40 Jahre werden im Beitrag Überlegungen zur Zukunft des Informatikunterrichts vorgestellt verbunden mit einem Plädoyer für eine weniger fachsystematische, mehr auf die Menschen in einer digitalen Welt ausgerichtete Sicht.

Keywords: Informatikunterricht; Informatikdidaktik; Dagstuhl-Dreieck; Bildungsstandards; Programmieren; Künstliche Intelligenz; Quantum Computing

1 Der Blick zurück

1.1 Physik und Technik

In Deutschland fand die Physik erst im 19. Jahrhundert als Folge der industriellen Entwicklung breiten Eingang in die Lehrpläne. Zunächst in den Realschulen, allerdings nicht in der heutigen Form. So gab es zum Beispiel an Heckers Realschulen in Berlin Mechanikklassen, die sich mit Werkzeugen und Maschinen beschäftigten. Im Mittelpunkt der Ausbildung standen dabei – modern gesprochen – Anwendungskompetenzen bei der Nutzung von Maschinen. Nach und nach fand dann auch die Vermittlung physikalischer Grundkenntnisse zu Mechanik, Thermodynamik, Elektromagnetismus oder Optik Eingang in die Lehrpläne. Erst im Laufe der Zeit wurde die Industrialisierung zu einem übergreifenden Thema für die Gesellschaft und Berufswelt. Vielerorts herrschte jedoch weiterhin die Meinung vor, dass die Hauptverantwortung für das Thema “Industrialisierung” im Physikunterricht zu verankern sei. Heute herrscht Konsens darüber, dass Physik wichtige Grundlagen für das Verständnis einer von Technik geprägten Welt legt, die Auseinandersetzung mit Technik aber ein fächerübergreifendes Thema ist. Rückblickend stellt sich die Frage, ob es nicht besser gewesen wäre, anstelle von Physik ein Fach „Technik“ einzuführen.

[1] Pädagogische Hochschule Schwyz, Institut für Medien und Schule, werner.hartmann@phsz.ch

doi:10.18420/infos2023-001

1.2 Informatik und Digitalität

Die Digitalisierung hat weite Bereiche von Gesellschaft, Wirtschaft und Kultur verändert. Dabei zeigen sich viele Parallelen zur Industrialisierung. Wie das Fach Physik stieß die Einführung eines Faches Informatik[2] auf großen Widerstand unter anderem der etablierten Fächer. Die Informatik genoss in den Schulen allenfalls ein Nischendasein Eine Erhebung zur Computernutzung im schweizerischen Bildungswesen im Jahre 1989 ergab, dass 60-70% aller Geräte in einem Computerraum standen. Niederer / Frey [NF89] schreiben dazu:

> Dieser Befund gibt zu denken, wenn man ihn mit der Realität in Wirtschaft, Industrie, Verwaltung und Alltag vergleicht. Außerhalb der Schule sind PCs und Terminals in den Arbeitsplatz integriert. Sie stehen dort wo, wo man misst, rechnet, verwaltet, organisiert und spielt. In der Schule hat man sie in einem Spezialraum abgesondert [...]

Bereits hier wird eine ganzheitliche Sicht auf die Rolle des Computers eingenommen, wie sie heute im Dagstuhl-Dreieck mit einer anwendungsorientierten, technischen und soziokulturellen Perspektive allgemein anerkannt ist. Rückblickend stellt sich ebenfalls die Frage, ob es nicht besser gewesen wäre, anstelle von Informatik ein Fach „Information, Kommunikation, Kooperation, Automation“ einzuführen. Die Bewältigung der Herausforderungen im Zusammenhang mit der Digitalisierung erfordert einen ganzheitlichen Ansatz.

1.3 Lebenswelt- vor Wissenschaftsorientierung

Die Informations- und Kommunikationstechnologien haben im Bildungswesen verschiedene Rollen: Werkzeug, Medium und Unterrichtsgegenstand. Das Gleichsetzen von ICT-Anwendung mit Informatik hat wesentlich zum mittlerweile gescheiterten Modell der fächerintegrierten Informatik beigetragen, das um die Jahrtausendwende in der Bildungslandschaft weit verbreitet war. Um die Informatik als Fach zu legitimieren, grenzt(e) sich die Fachdidaktik Informatik von den Anwendungskompetenzen ab und orientiert(e) sich stark an der Fachsystematik der Ingenieurdisziplin Informatik. Dieser Ansatz trug wesentlich zur Akzeptanz der Fachdidaktik Informatik an Hochschulen bei. So fanden etwa Themen wie formale Sprachen und Automaten Eingang in die Curricula. Objektorientierte Modellierung und Java als Programmiersprache bildeten vielerorts den Kern des gymnasialen Informatikunterrichts. Pasternak und Engbring [PE21] beschreiben diese Entwicklung kritisch. In den fachdidaktischen Diskussionen und Analysen sei viel zu wenig berücksichtigt worden, dass Objektorientierung eher auf der Ebene der modularen Programmentwicklung anzusiedeln ist, die in größeren Programmierprojekten eingesetzt wird.

[2] Im Folgenden wird davon ausgegangen, dass Informatik inzwischen abgesehen von ein paar wenigen Ausnahmen in allen Schulen einen festen Platz erhalten hat.

Für die Schulen und die Bildungspolitik blieb aufgrund der fachwissenschaftlichen Ausrichtung des Informatikunterrichts der Nutzen einer fundierten informatischen Bildung im Rahmen der Allgemeinbildung unklar. Für die Schülerinnen und Schüler ging zudem der Bezug zur Lebenswelt weitgehend verloren. Die 2008 veröffentlichten Bildungsstandards Informatik für die Sekundarstufe I stellten dann eine Art Trendwende dar, indem neben den Inhaltsbereichen auch Prozessbereiche postuliert wurden: Modellieren und Implementieren, Begründen und Bewerten, Strukturieren und Vernetzen, Kommunizieren und Kooperieren, Darstellen und Interpretieren. Ein Blick in aktuelle Lehrmittel zeigt aber, dass ein Großteil der Unterrichtseinheiten sich immer noch an der Fachsystematik orientieren und nur bedingt von der Lebenswelt der Lernenden ausgehen.

1.4 Ohne Informatik keine Medienbildung (und umgekehrt)

Neben der Abgrenzung zu ICT-Anwendungen hat die Einführung eines Faches Informatik zu teilweise heftigen Diskussionen zwischen Vertretern der Medienbildung und der Informatik geführt. Ein offener Blick auf die Rolle digitaler Medien in unserer Welt wäre rückblickend sowohl für die Verankerung der Medienbildung als auch der Informatik in der Schule förderlicher gewesen. Wie die Informatik hatte auch die Medienbildung einen schweren Stand in der Schule. Möhler et al [Mo13] kommen in einer explorativen Studie zur schulischen Medienbildung zu folgendem Ergebnis:

> Als fächerübergreifende Bildungs- und Erziehungsaufgabe und kooperatives Arbeitsfeld aller Fächer hat sich die Medienbildung in der Unterrichtswirklichkeit an der Schule der befragten Schülerinnen und Schüler noch nicht durchgesetzt. Schulische Medienbildung wird aktuell noch häufig auf den Einsatz von Medien als Präsentationsmittel reduziert.

Aus Sicht der Schülerinnen und Schüler ist es gemäß Möhler et al wichtig, Medienkompetenz in enger Verbindung mit ihrer eigenen Lebenswelt aufzubauen. Schülerinnen und Schüler wollen Dinge ausprobieren, eigene Erfahrungen sammeln, sich nicht nur in einem geschützten Rahmen abseits der realen Welt bewegen und sich mit von den Lehrkräften vorgegebenen Fragestellungen beschäftigen. Datenschutzregulierungen tragen zudem dazu bei, dass viele Alltagswerkzeuge keine Berücksichtigung im Unterricht finden und so den Lebensweltbezug behindern. Übertriebene Datenschutzbestimmungen tragen genauso wenig zur Medienkompetenz der Jugendlichen bei wie Elterntaxis Kinder befähigen, sich angepasst und sicher im Verkehr zu bewegen. Will die Schule auf die Teilhabe an der Gesellschaft und die Berufswelt vorbereiten, muss sich die Informatik zusammen mit der Medienbildung aktuellen Themen stellen können.

1.5 Lernziele, Kompetenzen usw. – alter Wein in neuen Schläuchen

Zu Beginn des 21. Jahrhunderts wurde der etablierte Lernzielbegriff durch den Kompetenzbegriff abgelöst. Für einen Großteil der Lehrkräfte blieben die Gründe für diesen vermeintlich grundlegenden Paradigmenwechsel unklar. Auch namhafte Expertinnen und Experten standen diesen Reformbewegungen kritisch gegenüber. So äußerte sich Gabi Reinmann [Re12] mit deutlichen Worten:

> An sich gehen mir ja diese Kompetenzdiskussionen allmählich auf die Nerven. Ich frage mich, ob wir uns nicht auch verstehen würden, wenn wir von Wissen, Können und Haltungen sprechen würden, meinetwegen auch von Kenntnissen, Fähigkeiten, Fertigkeiten und Einstellungen.

Als neues Fach im Fächerkanon stand die Informatik unter einem besonderen Legitimationsdruck, der sich in umfangreichen Bildungsstandards, Kompetenzmodellen und Lehrplänen niederschlug. Wie bei Lehrplänen üblich, versuchen verschiedene Akteure ihre Anliegen einzubringen, was häufig zu einer Überfrachtung führt. Die Bildungsstandards Informatik haben zwar wesentlich zur Verankerung der Informatik in der Schule beigetragen, enthalten aber teilweise Themen, die eher einem Informatikstudium zuzuordnen sind. Diese Entwicklung zeigt sich auch beim Lehrplan 21 „Medien und Informatik“ in der Schweiz, dessen Wirkung im Schulalltag leider (noch) als eher gering bezeichnet werden muss. Mit Peter Denning’s Great Principles of Computing [De03] oder den fundamentalen Ideen [Sc93] stehen schon länger fundierte Leitlinien für Informatik-Curricula zur Verfügung, an denen sich man beim Erstellen von Lehrplänen orientieren könnte.

1.6 Didaktik ohne Praxis versus Praxis mit Didaktik

In den Ingenieurdisziplinen erwarten wir von der Forschung in aller Regel neue Erkenntnisse oder Entwicklungen, die konkrete Probleme lösen. In der Bildungsforschung steht oft die reine Erkenntnis ohne Anspruch auf unmittelbaren Nutzen im Vordergrund. Auch die Forschung in der Informatikdidaktik ist zunehmend geprägt durch Evaluationen von Unterrichtsmethoden oder Lernumgebungen in oft künstlichen „Laborumgebungen“, also ohne konkreten Bezug zur Schulpraxis. Ein Beispiel hierfür ist das Konzept der Notional Machines (NMs). Notional Machines sind seit den 70er-Jahren ein Thema in der Didaktik der Informatik. Das umfangreiche Cambridge Handbook of Computing Education Research [FR19] thematisiert NMs an vielen Stellen, bietet aber kaum konkrete Beispiele für den Unterricht und ist den wenigsten Informatiklehrkräften zugänglich. Die Nutzung von NMs wäre ein wertvolles didaktisches Prinzip für den Informatikunterricht. Im Artikel „Notional Machines in Computing Education: The Education of Attention“ [Fea20] wird in Analogie zu Parkplätzen das Konzept von Variablen veranschaulicht und welche Folgen die Wahl eines falschen Datentyps haben kann. Die NM „Parkplatz“ kann fast beliebig

erweitert werden, nummerierte Parkfelder als Analogie zu Arrays sind nur ein Beispiel. Die fachdidaktische Forschung ist gefordert, vermehrt Forschungsergebnisse mit konkreten Umsetzungsvorschlägen auch Lehrkräften zugänglich zu machen und sich weniger am Prinzip „Publish or Perish“ zu orientieren. Als Vorbild kann die Videoreihe „Informatikdidaktik kurz gefasst“ von Peer Stechert auf YouTube dienen, zum Beispiel das Video „Notional Machines als Erklärmodell beim Programmieren“.

2 Herausforderungen im Informatikunterricht

2.1 Wichtig auf Primarstufe: Lust an Informatik wecken!

In den letzten rund dreißig Jahren hat sich die Diskussion über den Informatikunterricht vor allem auf die Sekundarstufe und höher konzentriert, zudem meist auf den Wahlpflichtbereich, d.h. auf interessierte Schülerinnen und Schüler. Auch der Großteil der fachdidaktischen Forschung bezieht sich auf die höheren Schulstufen. Insbesondere für die Primarstufe liegen relativ wenige Erkenntnisse vor. Hier ist die Fachdidaktik Informatik gefordert: Was im Gymnasium funktioniert, muss nicht per se auch auf der Primarstufe funktionieren.

Waite, Curzon und Sentance zeigen in „Difficulties with design: The challenges of teaching design in K-5 programming“ [Wa20] auf, dass Schülerinnen und Schüler in der Primarstufe sich schwer tun mit dem Entwurf von Programmen. Sie wollen programmieren und empfinden das Planen und Entwerfen von Programmen als Zeitverschwendung. Dieses Problem wird durch den Zeitmangel im Unterricht und die fehlende Ausbildung der Lehrkräfte auf den unteren Schulstufen noch verschärft. Ein weiterer Unterschied betrifft die kognitiven Voraussetzungen auf den verschiedenen Schulstufen. Nach der kognitiven Theorie von Jean Piaget ist das formal-operationale Entwicklungsstadium erst etwa ab 12 Jahren durch die Fähigkeit zu abstraktem und systematischem Denken nach formal-logischen Regeln gekennzeichnet.

Für die unteren Schulstufen erscheint ein fachsystematischer Ansatz wenig geeignet. Als stufengerechten Zugang eignet sich beispielsweise das Modell PRIMM von Sentance und Waite [SW17]. Das Modell ist für das Programmieren in einer textbasierten Sprache konzipiert. Es umfasst die Stufen Predict, Run, Investigate, Modify, Make, also beginnend mit dem Lesen von Programmen und der Vorhersage ihrer Ausgabe bis hin zur eigenen Entwicklung von Programmen. Dieses Modell lässt sich auf fast alle Themen der Informatik anwenden. In eine ähnliche Richtung geht der Einstieg ins Programmieren über sogenannte Parsons-Probleme, also das Zusammensetzen von Programmen aus vorgegebenen Programmfragmenten. Gute Parsons-Umgebungen basieren dabei nicht nur auf dem Zusammensetzen einzelner Programmzeilen, sondern auf dem Zusammensetzen größerer Programmblöcke. Blockbasierte Sprachen wie Scratch würden hier ein breites Aufgabenspektrum ermöglichen, und Sammlungen solcher Parsons-Probleme könnten gerade für den Einstieg ins Programmieren eine ähnliche Rolle spielen wie Lernprogramme für Vokabeln und Grammatik im Fremdsprachenunterricht.

2.2 Breite der Informatik berücksichtigen

Die Disziplin Computer Science hat sich in den letzten 20 Jahren stark entwickelt. In den Bildungsstandards Informatik für die Sekundarstufe I (2008) und die Sekundarstufe II (2016) werden fünf Inhaltsbereiche festgelegt: Information und Daten, Algorithmen, Sprachen und Automaten, Informatiksysteme sowie Informatik, Mensch und Gesellschaft. Vergleichen wir diese fünf Bereiche mit dem neuen Vorschlag „Computer Science Curricula Knowledge Areas planned for CS2023“ von ACM und IEEE, so fällt auf, dass das heutige Verständnis von Informatik viel weiter gefasst ist:

> Algorithms and Complexity, Architecture and Organization, Artificial Intelligence, Data Management, Graphics and Interactive Techniques, Human-Computer Interaction, Mathematical and Statistical Foundations, Modeling, Networking and Communication, Operating Systems, Parallel and Distributed Computing, Programming Languages, Security, Society, Ethics and Professionalism, Software Development Fundamentals, Software Engineering, Specialized Platform Development, Systems Fundamentals

Die aktuellen Fassungen der Bildungsstandards sind stark von der um die Jahrtausendwende vorherrschenden Fachsystematik geprägt. Es ist erfreulich, dass die Bildungsstandards als wertvolle Orientierungshilfe für das Entwicklung von Lehrplänen und Lehrmitteln derzeit überarbeitet werden. Es ist zu hoffen, dass der Umfang nicht größer wird, sondern mehr Freiraum und Flexibilität für Anpassungen zulässt.

2.3 Programmieren: Ein alter Zopf? Oder neu denken?

Algorithmen, Datenstrukturen und Programme bilden die Grundlage von Informatiksystemen. Software Engineering, d.h. das Erstellen großer Systeme, ist jedoch eine anspruchsvolle Aufgabe und kann nicht Ziel der Allgemeinbildung sein. In der Vergangenheit ist der Informatikunterricht immer wieder der Versuchung erlegen, Software Engineering zum Thema zu machen. Prominentes Beispiel ist die Ende der 90er Jahre propagierte Objektorientierung und Modellierung. Dies obwohl Studien die weit verbreitete Ansicht nicht stützten, dass damit der Einstieg ins Programmieren erleichtert wird. So äußerten sich Robins und Rountree [RRR03]:

> The papers reviewed do not support this position. They show that identifying objects is not an easy process and that objects identified in the problem domain are not necessarily useful in the program domain, that the mapping between domains is not straightforward, and that novices need to construct a model of the procedural aspects of a solution in order to properly design objects/classes.

Gegenwärtig haben wir es mit einer umgekehrten Entwicklung zu tun. Unzählige Initiativen (z.B. Hour of Code) vermitteln den Eindruck, Programmieren lernen sei kinderleicht. Die blockbasierte Sprache Scratch weist beeindruckende Nutzerzahlen auf. Bei genauerem Hinsehen zeigt sich jedoch ein anderes Bild. Justin Reich [Re20] weist darauf hin, dass es sich bei den rund 60 Millionen Scratch-Projekten meist um Stories handelt, die abgesehen von Forever Loops nur aus Sequenzen bestehen. Die durchschnittliche Anzahl der von Scratch-Nutzern geschriebenen Programmen sei zudem kleiner als Eins. Scratch und andere blockbasierte Sprachen ermöglichen einen attraktiven ersten Kontakt mit dem Programmieren. Wir sollten aber nicht von „Programmieren lernen“ sprechen. Die große Herausforderung ist nicht mehr die Frage nach der geeigneten Programmiersprache, sondern ob überhaupt und in welchem Umfang Programmieren noch unterrichtet werden soll. Die Entwicklungen rund um die Künstliche Intelligenz erlauben es, kleine Programme auf Basis umgangssprachlicher Beschreibungen bereits mit hoher Zuverlässigkeit automatisch zu generieren. Das Deep Learning-Modell OpenAI Codex wurde auf rund 50 Millionen GitHub-Repositories für Python trainiert. Studien zeigen, dass damit Programmieraufgaben auf Hochschulniveau mit ähnlicher Qualität gelöst werden wie von Studierenden [FADea22]. Matt Welsh [We23] schreibt dazu im provokativen Artikel „The End of Programming: The end of classical computer science is coming, and most of us are dinosaurs waiting for the meteor to hit.":

> No doubt the earliest pioneers of computer science, emerging from the (relatively) primitive cave of electrical engineering, stridently believed that all future computer scientists would need to command a deep understanding of semiconductors, binary arithmetic, and microprocessor design to understand software. Fastforward to today, and I am willing to bet good money that 99% of people who are writing software have almost no clue how a CPU actually works, let alone the physics underlying transistor design. By extension, I believe the computer scientists of the future will be so far removed from the classic definitions of „software“ that they would be hard-pressed to reverse a linked list or implement Quicksort.

Die Deep Programming-Werkzeuge unterstützen heute Softwareentwickler beim Programmieren von Routineaufgaben, vergleichbar mit der Nutzung von Sprachübersetzungsprogrammen. Programmieren "from scratch"wird an Bedeutung verlieren. Wichtiger wird es dafür, Programmcode beurteilen zu können, eine anspruchsvolle Aufgabe. Die Entwicklung großer Informatiksysteme wird zudem auch in Zukunft nicht von diesen Werkzeugen übernommen werden. Software Engineering umfasst weit mehr als das Schreiben von Programmcode, angefangen bei der Erhebung der Kundenbedürfnisse über das Testen der entwickelten Lösung bis hin zur Wartung und Schulung. Daniel Yellin beschreibt diesen Sachverhalt treffend in seinem Artikel „The Premature Obituary of Programming - Why deep learning will not replace programming“ [Ye23].

In der Schule ist aber nicht das Programmieren im Großen das Ziel, sondern das Programmieren im Kleinen. Genau diese Aufgaben werden heute mit Deep Programming-Werkzeugen gelöst. Zukünftige Informatik-Curricula müssen sich daher die Frage stellen, wie viel man selbst programmiert haben muss, um die immer komplexeren Informatiksysteme zu verstehen. Solche Herausforderungen sind für die Schule nicht neu. In der Mathematik beispielsweise haben Hilfsmittel wie der Taschenrechner den Fokus vom reinen Rechnen immer mehr hin zu anspruchsvolleren Aufgaben verschoben. Damit einhergehend öffnete sich die Schere zwischen leistungsstarken und leistungsschwächeren Lernenden noch mehr.

2.4 Künstliche Intelligenz erklären ist nicht trivial

Künstliche Intelligenz ist für die Informatik kein neues Thema, aber erst die Verfügbarkeit großer Datenmengen in Verbindung mit hoher Rechenleistung hat ihr zum Durchbruch verholfen. Aufgabe der Allgemeinbildung ist es, die notwendigen Kompetenzen zu vermitteln, um den Einsatz, das Potenzial, aber auch die Risiken von KI-Anwendungen zu verstehen. Michaeli, Romeike und Seegerer [MRS22] weisen darauf hin, dass dabei der Blick nicht nur auf die technischen Aspekte gerichtet werden darf. Eine besondere Schwierigkeit liegt in der Komplexität von KI-Systemen. Zum Beispiel ähnelt die aktuelle Entwicklung bei großen Sprachmodellen in einer gewissen Weise der Geschichte der Medizin, in welcher man oft herausfand, welche Behandlungen gut funktionieren, ohne den Mechanismus dahinter wirklich zu verstehen. Für den Informatikunterricht sind komplexe Informatiksysteme aber kein Neuland. Mit vereinfachten Modellen wird versucht, die grundlegende Funktionsweise aufzuzeigen. KI-Verfahren können mit Modellen aus der Mathematik (z.B. Regression oder Clusteranalyse) veranschaulicht werden, vgl. zum Beispiel [KLR19]. Auf höheren Stufen können auch KI-Modelle beispielsweise auf der Plattform Hugging Face genutzt werden.

2.5 Offene Datenbestände und Schnittstellen nutzen

API-Schnittstellen ermöglichen heute den Zugriff auf externe und aktuelle Daten und Funktionen von Diensten. Diese Möglichkeiten können genutzt werden, um die Analyse und Visualisierung von Daten in realen Projekten anzuwenden. Gängige Beispiele sind die Analyse von Wetterdaten oder Trends in Social Media-Diensten, die Auswertung von Sportresultaten oder demografischen Entwicklungen. Data Science und die Nutzung von Open Data ist zunächst ein Thema für höhere Schulstufen. Die Plattform Kaggle etwa stellt Anwendungen im Bereich Big Data, maschinelles Lernen oder Data Mining in den Mittelpunkt und bietet eine an Python angelehnte Programmierumgebung zur Entwicklung eigener Analysen. Mit Hilfe von Tabellenkalkulationsprogrammen können aber auch in der Unterstufe größere Datenbestände analysiert werden. Allerdings müssen die Daten bereits in standardisierter Form vorliegen, damit sich die Lehrkräfte nicht aufwändig mit API-Schnittstellen auseinandersetzen müssen. Das Schweizer Portal opendata.swiss bietet hierzu z.B. rund 10’000 frei zugängliche Datensätze im CSV-Format.

2.6 Quantum Computing – abseits unserer Vorstellung?

Das Thema Künstliche Intelligenz zeigt einen Trend: Informatiksysteme werden immer komplexer und damit auch anspruchsvoller für die Behandlung im Unterricht. Sortieralgorithmen oder fehlerkorrigierende Codes sind vergleichsweise einfache Themen. Die Programme eines klassischen Computers können wir im Prinzip - wenn auch viel weniger effizient - von Hand nachvollziehen. Die physikalischen Gesetze der Quantenwelt hingegen haben keine Analogien zu unserer vertrauten Welt und lassen sich nur mathematisch beschreiben. Wären wir in einer Welt aufgewachsen, in der alles schwerelos ist, würden wir uns wundern, dass Dinge herunterfallen können. Ähnlich verhält es sich mit den Gesetzen der Quantenphysik. Bei der Programmierung eines Quantencomputers betreten wir Neuland. So ist aufgrund von Input/Output-Beschränkungen bei Quantencomputern heute zum Beispiel noch unklar, ob sich auch Big Data-Probleme in der Praxis für die Lösung mit Quantencomputern eignen. Aufgabe der Allgemeinbildung wird es sein, ein Verständnis für Quantum Computing zu vermitteln. Von der Didaktik der Informatik und der Physik ist Kreativität gefordert, wie Verständnis für eine Technologie vermittelt werden kann, die sich unserer Intuition entzieht. Ob sich einfache, didaktisch reduzierte Lernumgebungen für Quantum Computing finden lassen, ist derzeit noch offen. Generell gilt jedoch, dass solche Lernumgebungen in der Lehre mehr bewirken würden als empirische Forschung allein. Forschung und Entwicklung - Design Based Research - sollte daher in Zukunft einen höheren Stellenwert an den Hochschulen erhalten.

3 Fazit

Seit der ersten INFOS-Tagung 1984 in Berlin mit dem Titel „Informatik als Herausforderung an Schule und Ausbildung“ gab es - und gibt es zum Teil immer noch - kontroverse Diskussionen um die Legitimation eines Faches Informatik, seine Inhalte oder die Abgrenzung zur Medienbildung sowie zu reinen Informatikanwendungen. Ein Blick in die Tagungsbände der vergangenen INFOS-Konferenzen zeigt, dass die meisten der heute diskutierten Themen nicht neu sind: Von der Rolle der Informatik im Fächerkanon der verschiedenen Schulstufen über die Wahl der Programmierparadigmen und -sprachen bis hin zu Themen wie Künstliche Intelligenz. Dabei hat es sich gezeigt, dass anstelle einer zu einseitigen und fachsystematischen Ausrichtung auf einzelne Themenbereiche ein offener Blick auf die Breite der Informatik, auf neue Entwicklungen sowie der Einbezug der Lebenswelt der Schülerinnen und Schüler zu einem attraktiven und dennoch fundierten Informatikunterricht beiträgt. Informatikunterricht soll helfen, die Welt besser zu verstehen, aber auch Spaß machen.

Anmerkung: Der Autor dankt Michael Hielscher und Raimond Reichert für wertvolle Hinweise zu diesem Beitrag.

Literatur

[De03] Denning, P.J.: Great Principles of Computing. Commun. ACM, 46(11):15–20, nov 2003.

[FADea22] Finnie-Ansley, J.; Denny, P.; et al: The Robots Are Coming: Exploring the Implications of OpenAI Codex on Introductory Programming. In: Australasian Computing Education Conference. S. 10–19, 2022.

[Fea20] Fincher, S.; et al: Notional Machines in Computing Education: The Education of Attention. In: Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education. S. 21–50, 2020.

[FR19] Fincher, S. A.; Robins, A. V.: The Cambridge Handbook of Computing Education Research. Cambridge University Press, 2019.

[KLR19] Kersting, K.; Lampert, C.; Rothkopf, C.: Wie Maschinen lernen - Künstliche Intelligenz verständlich erklärt. Springer Wiesbaden, 2019.

[Mo13] Moehler, A.; Winkler, T.; Marquardt, N.; Knaus, T.: Schulische Medienbildung aus der Perspektive von Schülerinnen und Schülern - eine explorative Studie. kopaed, 2013.

[MRS22] Michaeli, T.; Romeike, R.; Seegerer, S.: What students can learn about artificial intelligence - recommendations for K-12 computing education. Proceedings of IFIP WCCE, 2022.

[NF89] Niederer, R.; Frey, K.: Informatik und Computernutzung im CH-Bildungswesen. EDK, 1989.

[PE21] Pasternak, A.; Engbring, D.: Den Blick zurück nach vorn. In (Humbert, Ludger, Hrsg.): INFOS 2021 – 19. GI-Fachtagung Informatik und Schule. Gesellschaft für Informatik, Bonn, S. 39–48, 2021.

[Re12] Reinmann, G.: Informationskompetenz und persönliches Wissensmanagement. In: W. Sühl-Strohmenger: Handbuch Informationskompetenz. De Gruyter Saur, Berlin, Boston, S. 85–92, 2012.

[Re20] Reich, J.: Failure to Disrupt: Why Technology Alone Can't Transform Education. Harvard University Press, 2020.

[RRR03] Robins, A.; Rountree, J.; Rountree, N.: Learning and Teaching Programming: A Review and Discussion. 2003.

[Sc93] Schwill, A.: Fundamentale Ideen der Informatik. Zentralblatt für Didaktik der Mathematik, S. 20–31, 1993.

[SW17] Sentance, S.; Waite, J.: PRIMM: Exploring Pedagogical Approaches for Teaching Text-Based Programming in School. In: Proceedings of the 12th Workshop on Primary and Secondary Computing Education. S. 113–114, 2017.

[Wa20] Waite, J.; Curzon, P.; Marsh, W.; Sentance, S.: Difficulties with Design: The Challenges of Teaching Design in K-5 Programming. 150(C):103838, 2020.

[We23] Welsh, M.: The End of Programming. Communications of the ACM, 66(1):34–35, 2023.

[Ye23] Yellin, D. M.: The Premature Obituary of Programming - Why deep learning will not replace programming. Communications of the ACM, 66(2):41–44, 2023.