KI & DIGITALE BILDUNG

Informatik in der Grundschule: So fördern Lehrkräfte ein korrektes Programmierverständnis

Kinder entwickeln oft falsche Vorstellungen von Konzepten der Informatik. Lehrkräfte sollten diese Fehlkonzepte frühzeitig identifizieren und korrigieren.

26. MAI 2025
KATRIN KUNZ

LESEZEIT: 4 MINUTEN

© AdobeStock 925817776 Junda

Digitale Technologien prägen unseren Alltag und unsere Arbeitswelt. Jüngste Studien zeigen jedoch, dass über 40 Prozent der Achtklässler:innen in Deutschland nur grundlegende digitale Fähigkeiten aufweisen. Ein früher Einstieg in die Informatik könnte dabei helfen, das digitale Verständnis zu verbessern. Doch wie lässt sich Informatik wirksam und kindgerecht in die Grundschule integrieren? Dazu forscht Katrin Kunz an der Universität Tübingen.

Digitale Kompetenzen sind heute wichtiger denn je. Laut der aktuellen ICILS-Studie (International Computer and Information Literacy Study) schneiden deutsche Achtklässlerinnen und Achtklässler im internationalen Vergleich zwar durchschnittlich ab, doch über 40 Prozent verfügen nur über grundlegende digitale Kompetenzen. Ein früher Einstieg in die Informatik könnte dazu beitragen, diese Defizite zu verringern.

Doch gerade junge Kinder entwickeln oft intuitive, aber fehlerhafte Vorstellungen von informatischen Konzepten, die ihr späteres Lernen erschweren können. Umso wichtiger ist es, dass Informatikunterricht und schulische Initiativen wie Medienunterricht oder Technik-AGs darauf abzielen, von Beginn an fundierte Inhalte zu vermitteln. Andernfalls könnten bestehende Defizite noch verstärkt werden. Die Frage bleibt also: Wie können Informatik-Kompetenzen im Grundschulalter nachhaltig gefördert werden? Und welche Rolle spielen Lehrkräfte dabei, um ein solides Verständnis für das Programmieren zu fördern?

Wie Kinder Programmieren lernen

Eine weit verbreitete Theorie darüber, wie wir das Programmieren erlernen, ist die der mentalen Modelle. Diese vereinfachten inneren Vorstellungen helfen uns, komplexe Zusammenhänge zu verstehen und vorherzusagen. Damit Schülerinnen

und Schüler tragfähige mentale Modelle entwickeln, ist es entscheidend, dass Lehrkräfte geeignete Erklärmodelle einsetzen. Insbesondere in der Grundschule, wo abstrakte Konzepte schwer greifbar sind, helfen anschauliche Analogien und visuelle Darstellungen, um zentrale Programmierprinzipien verständlich zu machen.

Ein bewährtes Beispiel ist die „Kochrezept“-Analogie für Algorithmen: So wie ein Rezept eine Abfolge von Schritten vorgibt, um ein Gericht zuzubereiten, besteht ein Algorithmus aus klar definierten Anweisungen, die ein Computer ausführt. Ebenso lassen sich Variablen mit Boxen vergleichen, in denen Werte gespeichert werden. Solche Erklärmodelle helfen Kindern, sich Programmierkonzepte nicht nur theoretisch anzueignen, sondern sie auch aktiv anzuwenden.

Falsche Vorstellungen bremsen den Lernfortschritt

Mentale Modelle bergen jedoch das Risiko von Fehlvorstellungen. Im Sachunterricht der Grundschule zeigen sich häufig Alltagsvorstellungen der Kinder, die nicht mit wissenschaftlichen Erkenntnissen übereinstimmen – etwa die Annahme, dass Schall durch die Luft „fliegt“. Ähnliche Fehlvorstellungen treten auch beim Erlernen des Programmierens auf. Lehrkräfte sollten sich dieser typischen Fehlvorstellungen bewusst sein, um sie frühzeitig zu erkennen und gezielt zu korrigieren. Denn einmal verinnerlichte falsche Modelle können spätere Lernfortschritte erschweren.

TYPISCHE FEHLVORSTELLUNGEN

1. Variablen können mehrere Werte fassen
   Wenn Kinder annehmen, dass eine Variable mehrere Werte gleichzeitig speichern kann, liegt das oft an ungeeigneten Erklärmodellen. Wird eine Variable etwa mit einer Box verglichen, könnte dies suggerieren, dass alle Werte, die einer Variablen nacheinander zugewiesen werden, in ihr erhalten bleiben, da ja meistens mehrere Dinge in eine Box passen.

2. Wiederholungen (Schleifen) werden einmal zu oft ausgeführt
   Ein weiteres häufiges Missverständnis betrifft das Verständnis von Schleifen. Wenn eine Schleife 10-mal “wiederholt” werden soll, kann es passieren, dass Kinder glauben, die Anweisung werde einmal ausgeführt und dann noch zehnmal wiederholt – also insgesamt elfmal. Solche Fehlvorstellungen können

dazu führen, dass Programme unerwartete Ergebnisse liefern und das Konzept von Schleifen falsch verstanden wird.

3. Der Dann-Zweig in bedingten Verzweigungen wird immer ausgeführt
   Eigentlich entscheidet eine Bedingung (‚Falls'), ob der darauffolgende Dann-Zweig oder der Sonst-Zweig im Programm ausgeführt wird. Arbeiten Kinder jedoch hauptsächlich mit wahren Bedingungen, bei denen immer der Dann-Zweig ausgeführt wird, könnten sie fälschlicherweise annehmen, dass eine Bedingung lediglich darüber entscheidet, ob der Sonst-Zweig entfällt (bei wahren Bedingungen) oder zusätzlich zum Dann-Zweig auch ausgeführt wird (bei falschen Bedingungen).

So können Lehrkräfte Fehlvorstellungen erkennen und korrigieren

Um nachhaltige Missverständnisse zu vermeiden, sollten Lehrkräfte bewusst darauf achten, wie ihre Schülerinnen und Schüler Programmierkonzepte verstehen und anwenden. Dabei helfen verschiedene Strategien, um Fehlvorstellungen frühzeitig zu identifizieren und gezielt zu korrigieren.

- Verständnis der Schülerinnen und Schüler erfassen
  Fehlvorstellungen zeigen sich oft in den Fehlern, die Kinder in ihren Programmen machen. Doch nicht jeder Fehler ist zufällig – häufig spiegeln sie ein tieferliegendes Missverständnis wider. Daher ist es hilfreich, nicht nur den geschriebenen Code zu betrachten, sondern die Schülerinnen und Schüler auch zu bitten, ihren Code laut zu erklären. Dies gibt Lehrkräften wertvolle Einblicke in die mentalen Modelle der Kinder und zeigt, wo möglicherweise Fehlvorstellungen bestehen.

- Gezielte Förderung durch Lesen und Nachvollziehen von Code
  Nicht nur das Schreiben von Programmen, sondern auch das Analysieren bestehender Codes kann helfen, Fehlvorstellungen zu korrigieren. Wenn Kinder vorgefertigte Programme untersuchen und vorhersagen, was passieren wird, können Lehrkräfte gezielt Materialien einsetzen, die typische Denkfehler aufgreifen und richtigstellen. Durch das Vergleichen der eigenen Erwartung mit der tatsächlichen Programmausgabe erkennen die Schülerinnen und Schüler selbstständig, wo ihr mentales Modell angepasst werden muss. Auch Programmablaufpläne – Diagramme, die den Ablauf von Anweisungen veranschaulichen – können ebenfalls hilfreich sein. Sie zeigen die Reihenfolge der Schritte und machen zum Beispiel Verzweigungen durch

Pfeile sichtbar, was die Logik des Programms greifbarer macht.

- Positive Fehlerkultur etablieren
  Fehlvorstellungen entstehen oft, weil Kinder nur mit einer begrenzten Anzahl von Beispielen arbeiten und dadurch falsche Annahmen treffen. Eine gezielte Strategie zur Korrektur ist die bewusste Konfrontation mit Gegenbeispielen. Werden Schleifen zu oft ausgeführt, kann es hilfreich sein, die Kinder zunächst abstimmen zu lassen: Passiert es zehn- oder elfmal? Anschließend wird der Code ausgeführt, sodass sie das Ergebnis direkt sehen. Dieser praktische Ansatz schärft das Verständnis für Schleifenlogik und hilft, Fehlvorstellungen zu vermeiden. Dieser sogenannte „conceptual change“ hilft den Schülerinnen und Schülern, ihr Verständnis schrittweise anzupassen, ohne dass sie sich entmutigt fühlen.

- Erklärmodelle bewusst wählen
  Da Erklärmodelle eine zentrale Rolle bei der Vermittlung von Programmierkonzepten spielen, sollten sie gezielt an mögliche Fehlvorstellungen angepasst werden. Wird beispielsweise eine Box als Analogie für Variablen genutzt, kann dies zu der falschen Vorstellung führen, dass eine Variable mehrere Werte gleichzeitig speichern kann. Um diesem Missverständnis entgegenzuwirken, kann die Box als besonders klein beschrieben werden, sodass immer nur ein Wert hineinpasst. Alte Werte müssen erst gelöscht – also aus der Box entfernt – werden, bevor ein neuer Wert eingefügt werden kann.

Die frühe Förderung informatischer Kompetenzen in der Grundschule bietet großes Potenzial – doch sie erfordert mehr als nur den Zugang zu Programmierumgebungen. Die bisherigen Erkenntnisse zeigen, dass eine gezielte Auseinandersetzung mit Fehlvorstellungen von Kindern hilft, ein tragfähiges Verständnis für Programmierkonzepte zu entwickeln. Lehrkräfte können mit durchdachten Erklärmodellen und gezielten Code-Analyseaufgaben frühzeitig ansetzen, um Missverständnisse aufzulösen – noch bevor diese sich verfestigen. Denn wenn Kinder frühzeitig fundierte mentale Modelle entwickeln, wird Informatik zu einem Werkzeug, mit dem sie die digitale Welt aktiv gestalten können.

Zur Person

Katrin Kunz hat Kognitionswissenschaft und gymnasiales Lehramt für die Fächer Mathematik und Informatik studiert. Sie arbeitet am Hector-Institut für empirische Bildungsforschung an der Entwicklung von Informatikkursen für die Hector Kinderakademien und schreibt ihre Doktorarbeit über die Entwicklung mentaler Modelle von Programmierkonzepten in begabten Grundschülerinnen und Grundschülern.

Weiterführende Literatur

- Übersicht über Informatik als Schulfach in den Ländern
- Übersicht über mögliche Erklärmodelle
- Übersicht über häufige Fehlvorstellungen
- Eickelmann, B., Fröhlich, N., Bos, W., Gerick, J., Goldhammer, F., Schaumburg, H., ... & Vahrenhold, J. (2024). ICILS 2023# Deutschland. Computer-und informationsbezogene Kompetenzen und Kompetenzen im Bereich Computational Thinking von Schüler* innen im internationalen Vergleich.
- Humbert, L., Herper, H., Best, A., Borowski, C., Freudenberg, R., Fricke, M., ... & Thomas, M. (2019). Empfehlungen der GI–Kompetenzen für informatische Bildung im Primarbereich.
- Kunz, K. (2024). The Association between Computational Thinking and Mental Models in Talented Children. In Proceedings of the 24th Koli Calling International Conference on Computing Education Research, 1-2.
- Nitz, S., & Fechner, S. (2018). Mentale Modelle. Theorien in der naturwissenschaftsdidaktischen Forschung, 69-86.
- Sorva, J. (2018). Misconceptions and the beginner programmer. Computer science education: Perspectives on teaching and learning in school, 171.
- Tsarava, K., Kunz, K., & Trautwein, U. (2024). Das Informatikcurriculum der

Hector Kinderakademien. Informatische Bildung in Schulen.