Daten- und
Methodenbericht
Juni 2022

Natalie Kiesler

Rekursive Problemlösung in der Online Lern-umgebung CodingBat durch Informatik-Studierende

Daten- und Methodenbericht zum qualitativen Datenpaket des Projekts Kompetenzförderung in der Programmierausbildung durch Modellierung von Kompetenzen und informativem Feedback

Projektleitung
Dr. Natalie Kiesler
Telefon: +49 69 24708-543
E-Mail: kiesler@dipf.de

Der vorliegende Daten- und Methodenbericht soll folgendermaßen zitiert werden:
Kiesler, N. (2022). Rekursive Problemlösung in der Online Lernumgebung CodingBat durch Informatik-Studierende. Daten- und Methodenbericht zum qualitativen Datenpaket des Projekts Kompetenzförderung in der Programmierausbildung durch Modellierung von Kompetenzen und informativem Feedback. Version 1.0.0.

Juni 2022

Inhaltsverzeichnis

Tabellen-/Abbildungsverzeichnis

Einleitung

Bestehende Online (Selbst-)Lernangebote wie CodingBat, CodeRunner, Alice oder auf Scratch basierende Tools bieten zahlreiche Potenziale für die Programmierausbildung, indem Lernende selbstgesteuert Programmieren üben können. Viele dieser Tools und Methoden zielen auf die Visualisierung von Ausführungsschritten in Programmen oder die Darstellung deren Ergebnissen ab, um deren Abfolge und das Zusammenwirken von Codebausteinen zu verdeutlichen. Insbesondere unerfahrene Lernende, so die Hoffnung, sollen von derartigen Angeboten profitieren.

Informatives Feedback stellt insgesamt eine bedeutende Größe in Lernprozessen dar (vgl. Hattie 2009; Hattie und Timperley 2007). Seit Skinners (1968) Ansätzen zur programmierten Instruktion sowie den frühen Experimenten von Thorndike (1913) und Skinner (1938) ist die Bedeutung von Feedback unstrittig. Dennoch konnte Feedback als Forschungsgegenstand bisher noch nicht vollständig durchdrungen werden, da teilweise sehr unterschiedliche Wirkungen von Feedback dokumentiert sind. Zunächst lassen sich zahlreiche Formen und Typen von Feedback unterscheiden, z. B. in Anlehnung daran, welche Informationen enthalten sind. Narciss (2006) unterscheidet diesbezüglich acht grundlegende, inhaltsbezogene Formen informativen Feedbacks. Das Ziel von informativem Feedback ist es, Lernenden Informationen über ihre eigene Arbeit zu vermitteln, um Gemeinsamkeiten bzw. Unterschiede zu einem erwarteten Zielzustand feststellen zu können. Dadurch sollen Lernende befähigt werden, ihr Verhalten zu ändern, sprich zu lernen. Die Überprüfung und Übertragung der bestehenden Klassifizierung von Narciss (2006) auf die Informatik, respektive die grundlegende Programmierausbildung stellt einen bisher noch wenig untersuchten Bereich dar, wenngleich Keuning, Jeuring und Heeren (2016) sowie Le (2016) erste Typisierungen von Feedback für Programmier-Übungsaufgaben vorschlagen.

Eine große Herausforderung während der selbstgesteuerten Anwendung von Online-Selbstlerntools im Kontext der Programmierung stellt der nach wie vor begrenzte Einsatz von Feedback-Optionen dar. Feedback sollte sich nicht auf die binäre Angabe zur Richtigkeit der Eingabe oder einzelne Testfälle beschränken, wie verschiedene Studien aufzeigen (vgl. Kohl 2009; Malan und Leitner 2007; Zhang, Surisetty und Scaffidi 2013). Anhand der Literatur wird nebst zahlreichen Inkonsistenzen in Befunden und widersprüchlichen Ergebnissen die enorme Bedeutung von individuellen und situativen Faktoren deutlich (vgl. Phye und Sanders 1994; Smith und Ragan 1993).

Die hier vorgestellten und qualitativ-empirisch erhobenen Forschungsdaten bestehen aus der Dokumentation durchgeführter Lautes Denken-Experimente im Rahmen einer Usability-Studie zur Modellierung von informativem Feedback für die grundlegende Programmierausbildung. Der Fokus liegt dabei auf den studentischen Arbeitsschritten während der rekursiven Problemlösung von Standardproblemen in der Informatik: der Berechnung der Fakultät natürlicher Zahlen, sowie der Berechnung der Fibonacci-Zahlenfolge. Als Grundlage wird das frei verfügbare Online Tool CodingBat genutzt. Anhand der Daten können die studentischen Schritte im Problemlöseprozess qualitativ nachvollzogen werden, indem die Bearbeitungszeit, Interaktionen, Feedback-Bedarfe und -Nutzung aus den bereitgestellten Daten hervorgehen. So können z.B. mentale Modelle, oder generell das studentische Vorgehen transparent und nachvollziehbar werden.

1 Inhalt und Anlage der Studie

1.1 Administrative und organisatorische Angaben

- **Titel des Forschungsprojekts:** Kompetenzförderung in der Programmierausbildung durch Modellierung von Kompetenzen und informativem Feedback
- **Verortung:** Dissertation, Goethe-Universität Frankfurt am Main
- **Projektleitung:** Dr. Natalie Kiesler, Leitende wissenschaftliche Mitarbeiterin, DIPF | Leibniz-Institut für Bildungsforschung und Bildungsinformation (ORCID-ID: https://orcid.org/0000-0002-6843-2729)
- **Projektzeitraum:** 15.6.2015 bis 11.1.2022, abgeschlossen

1.2 Inhalt und Aufbau des Forschungsprojekts

Die hier beschriebenen Daten entstammen aus dem zweiten Teilstrang der Dissertation der Autorin. Dieser hatte den Schwerpunkt informatives Feedback, dessen Gestaltung und Umsetzung im Kontext der grundlegenden Programmierausbildung. Demnach werden hier lediglich die Ziele und Forschungsfragen, etc. für diesen Teil der Arbeit dargelegt. Konkret handelt es sich dabei um die Forschungsfrage nach den Wirkungen informativen Feedbacks, welches von (Online)-Selbstlerntools angeboten wird.

- Forschungszweck
  - **Ziele:** Erkenntniszugewinn zu den Wirkungen und Anwendungsszenarien von Feedback (-Typen) für den Kontext der grundlegenden Programmierausbildung, anhand derer Empfehlungen der Förderung von Programmierkompetenzen abgeleitet werden können.
  - **Forschungsfragen:** Wie wirkt sich informatives Feedback während der Nutzung von (Online)-Selbstlerntools mit Programmier-Übungsaufgaben auf Studierende aus?
- **Forschungskontext:** Informatik, Hochschulbildung, grundlegende Programmierausbildung
- **Inhaltlicher Bezugsrahmen und Forschungsdesign**: Anhand von Lautes Denken-Experimenten sollen die Auswirkung der eigens entwickelten Umsetzungen informativen Feedbacks auf Lernende exploriert werden, um weitere Anhaltspunkte für die lernförderliche Gestaltung von Feedback in der grundlegenden Programmierausbildung zu erhalten. Besonders Übungstools mit automatischer Bewertung können zu Frustrationen bei Lernenden führen, wenn letztere minimale Rückmeldung erhalten oder Aufgaben zu komplex sind (vgl. Battestilli et al. 2019). Daher erscheint es sinnvoll, die Wirkungen einzelner Feedback-Typen näher zu untersuchen, um zu einer Einschätzung bezüglich dessen Angemessenheit im Kontext gelangen zu können. Die Determinanten informativen Feedbacks lassen aktuell den Kontext und konkrete, geeignete didaktischen Szenarien außen vor. In der grundlegenden Programmierausbildung schlagen neben Narciss (2006) sowohl Keuning, Jeuring und Heeren (2016) als auch Le (2016) eine Auswahl scheinbar geeigneter Feedback-Typen vor. Dennoch erscheinen diese teilweise noch zu unspezifisch, da klare Empfehlungen für den Einsatz der jeweiligen Typen in der grundlegenden Programmierausbildung und der damit verbundenen Zielgruppe fehlen. Darüber hinaus werden verschiedene Anforderungen der Aufgabe und angestrebte (Kategorien von) Kompetenzen nicht berücksichtigt. Daher sollen aus dieser Arbeit Implikationen zur Angemessenheit der jeweiligen Feedback-Typen abgeleitet werden.

1.3 Theoretische Vorannahmen

Bezogen auf den zweiten inhaltlichen Schwerpunkt der vorliegenden Dissertation wird Feedback als Chance definiert, um Lernende auf mögliche Diskrepanzen zwischen Ist- und Soll-Zustand hinzuweisen, und damit ggf. notwendige Verhaltensänderungen bei diesen bewirken zu können. Mit Blick auf formatives, informatives Feedback bestehen eine Reihe von Potenzialen, die weit über behavioristische Ansätze aus der frühen pädagogischen Psychologie hinausgehen. Aus der Literatur gehen vor allem die unter Umständen überwiegend positiven Auswirkungen von computergestütztem, formativen Feedback hervor. So kann Feedback z. B. Unsicherheiten von Lernenden als aversiven Zustand reduzieren und Fehlkonzepte aufklären (vgl. Ashford, Blatt und VandeWalle 2003; Narciss 2020). Außerdem kann Feedback in Form von hilfreichen Informationen zum Problemlöseprozess die kognitive Belastung von Lernenden verringern (vgl. Spohrer und Soloway 1986; Paas, Renkl und Sweller 2003), und z. B. zur Korrektur bzw. Verbesserung von Problemlösestrategien und Methoden führen (vgl. Baron 1988). Nicht zuletzt bestehen Auswirkungen auf die Volition, Motivation und Emotionen von Lernenden. Eine isolierte Betrachtung ohne Berücksichtigung letzter Aspekte erscheint kaum möglich (vgl. Goldstein, Emanuel und Howell 1968; Narciss und Huth 2004; Narciss 2020), weil die motivierende Wirkung des Feedbacks durch den entstehenden Leistungsanreiz zu begründen ist. In der Literatur lassen sich noch weitere, interessante Befunde zur Wirkung von Feedback wiederfinden. Diese weisen insbesondere eine Reihe von Inkonsistenzen bzgl. des Zeitpunkts von Feedback und des spezifischen Charakters des Feedbacks auf (siehe Abschnitt 2.5.4). So kann zeitnah bereitgestelltes Feedback Lernende zwar im Lernprozess unterstützen, dies trifft jedoch nicht in jedem didaktischen Szenario und Lernkontext zu (vgl. Bangert-Drowns et al. 1991; Butler und Winne 1995; Kulhavy 1977; Kulik und Kulik 1988). Kulhavy (1977) dagegen verweist auf die sogenannte Interference-Persevation Theory, wonach sich unmittelbares Feedback negativ auf den Lernprozess auswirken kann. Die widersprüchlichen Forschungsergebnisse implizieren daher sowohl positive als auch negative Effekte von unmittelbarem Feedback auf den Lernprozess. Daher weist z. B. Shute (2008) auf den Einfluss weiterer Variablen auf formatives Feedback und dessen Wirkung in Lernprozessen hin. Im Kontext der Programmierausbildung erscheint es naheliegend, direktes Feedback zu wählen. Informatik-Studierende erhalten aktuell beim Programmieren bereits zahlreiche Rückmeldungen von Compiler und Interpreter. Daher sind sie zeitnahes Feedback gewohnt und eine Verzögerung könnte sich als zusätzliche Frustrationsquelle erweisen.

Darüber hinaus sind die divergierenden Studienergebnisse den teils erheblich voneinander abweichenden theoretischen Grundannahmen und angestrebten Feedback-Funktionen geschuldet. Narciss (2006) fasst nichtsdestotrotz die möglichen Wirkungen von Feedback als Konsens zusammen. Dazu gehören kognitive, motivationale und meta-kognitive Auswirkungen von Feedback. Die Bedeutung von Feedback bezogen auf die Wirkung wird daher mehr als deutlich in der Literatur dargestellt. In der Dissertation der Autorin werden vor allem formatives, informatives Feedback und die verschiedenen elaborierten Feedback-Typen nach Narciss (2006) sowie Keuning, Jeuring und Heeren (2016) als Ausgangspunkt für die Untersuchung von deren Wirkung im Kontext der Programmierung genutzt.

Aktuell sind zahlreiche Ansätze, Prototypen und Versionen von Übungstools, zum Teil unter Nutzung visueller Programmiersprachen frei zugänglich im Internet verfügbar. Einige dieser Tools, wie etwa Alice und Scratch nutzen u. a. visuelles Feedback, indem sie die Ausführung eines konstruierten Programms zeigen. Dadurch wird auch das Feedback unter Umständen überaus individuell, passend zur Eingabe konstruiert und wortwörtlich sichtbar. Einzelne Abstufungen von derartigem Feedback bzw. von informativem Feedback sind bisher allerdings nur selten in der Praxis zu finden, obwohl sich enorme Potenziale von möglichst individuellem Feedback auf den Lernprozess vermuten lassen. Daraus resultiert der Bedarf zur zumindest prototypischen Exploration und Untersuchung von weiteren, spezifischeren Feedback-Formen im Kontext der grundlegenden Programmierausbildung, die gleichzeitig an die aktuellen und kreativen Möglichkeiten für die Feedback-Vergabe angelehnt sind.

2 Sampling, Sample und Untersuchungssituation

- **Gegenstand der Primär-Untersuchung:** Feedback-Wirkung auf Lernende in der grundlegenden Programmierausbildung an Hochschulen.

- **Auswahl der Testpersonen:** Die ausgewählten Testpersonen haben zum Zeitpunkt der Tests mindestens die Lehrveranstaltung „Programmieren 1“ an der Hochschule Fulda belegt und abgeschlossen. Aus diesem Grund liegt der Zeitpunkt der Testreihe am Ende des Wintersemesters und nach Ende der Klausurenphase. In dem Einführungskurs werden Grundlagen der Java-Programmierung gelehrt, darunter auch das Thema Rekursion. Weitere spezielle Präferenzen bezüglich eines Studiengangs oder einer bestimmten Fachsemesterzahl bestanden nicht, da alle Studierenden am Fachbereich Angewandte Informatik im Wintersemester 2016/2017 ein und dieselbe Lehrveranstaltung zur Einführung in die Programmierung besucht haben. Diesbezüglich ist zumindest von einer gemeinsamen Basis an Kenntnissen auszugehen, die im Rahmen der Vorlesung gehört wurden. Außerdem soll dadurch die Suche von Probandinnen und Probanden vereinfacht werden. Insofern wurden alle Studierenden als Testpersonen berücksichtigt, die zumindest den ersten Programmier-Kurs im Umfang von 5 ECTS und 4 Semesterwochenstunden (SWS) als gemeinsame Basis belegt haben.

  So sollte sichergestellt werden, dass zumindest alle Studierenden die relevanten Inhalte für die Aufgaben als Teil ihres Studiums kennen. Vorkenntnisse und Programmiererfahrungen sind bei Informatik-Studierenden insgesamt mehr als divers, wie in verschiedenen Studien dargelegt wird. Daran ändert sich auch nach dem ersten Semester relativ wenig. Ein Ausschluss spezieller Studiengruppen, Semesterzahl oder Ähnliches stellt vor dem Hintergrund der extremen Heterogenität an Hochschulen und der überaus individuellen Vorbildung daher keine sinnvolle Option dar. Es sollten außerdem nicht Individuen untersucht werden, sondern die Auswirkungen von Feedback auf kognitiver, meta-kognitiver und motivationaler Ebene. Daher werden Selbsteinschätzungen zu Vorwissen und der subjektiven Schwierigkeit einzelner Themenbereiche mittels Eingangsfragebogen erhoben. Darüber hinaus wurde z. B. der bisherige Umgang mit Selbstlerntools bzw. bekannten Online-Übungstools erfragt.

- Kontaktaufnahme zu Testpersonen: Studierende des Fachbereiches Angewandte Informatik galten als potenzielle Testpersonen, insofern sie die Grundlagenveranstaltung zur Programmierung belegt und abgeschlossen haben. Diese Zielgruppe wurde in zwei Stufen und über zwei Kanäle kontaktiert. Zuerst wurde das Pendant eines Schwarzen Brettes auf einer Lernplattform (Moodle) genutzt, um Probandinnen und Probanden zu kontaktieren, die sich daraufhin per E-Mail zurückmeldeten. Aus diesem Versuch der Kontaktaufnahme resultierten die ersten sechs Probandinnen und Probanden für die erste Testreihe. Dafür wurden in Zusammenarbeit mit dem verantwortlichen Labor-Ingenieur des Usability Labors zwei Tage im Labor organisiert. Die Studierenden wurden je nach zeitlicher Präferenz auf verschiedene Zeitfenster an diesen zwei Tagen verteilt.

  Zur Vergrößerung der Stichprobe wurden nach dieser ersten Testreihe gezielte E-Mails an die Studierenden des letzten, abgeschlossenen Kurses „Programmieren 1“ versendet mit der Einladung zur Teilnahme am Lauten Denken. Daraufhin meldeten sich fünf Studierende zurück, welche erneut auf jeweilig passende Zeitfenster an zwei Tagen aufgeteilt wurden. Da beide Testreihen während der Semesterferien stattfanden, konnten vor Ort am Campus keine Studierenden

ad hoc für Tests eingeladen werden. Dennoch erscheint der Rücklauf gemessen am Modus der Einladung erwartungsgemäß. Für beide Anschreiben wurden jeweils die gleichen Texte verwendet, in denen der grobe Ablauf, der zeitliche Rahmen und die Ziele der Untersuchung kurz erläutert wurden. Die Teilnahme an den Tests basiert vollständig auf Freiwilligkeit und Anonymität. Finanzielle Anreize wurden nicht gegeben, lediglich auf Erfrischungen und eine kleine Verpflegung während der Testreihe wurde hingewiesen. Darüber hinaus wurden die Probandinnen und Probanden schon in der Ankündigung der Tests über die geplante Aufzeichnung unter Labor-Bedingungen informiert.

- Erhebungsort(e)*:* In Präsenz im Usability Labor der Hochschule Fulda, Fachbereich Angewandte Informatik.

- Untersuchungssituation: Mit dem Usability Labor der Hochschule Fulda stand ein großer und flexibel einzurichtender Raum für die Durchführung der Lauten Denken-Experimente zur Verfügung. Der Testraum wird durch einen Beobachtungsraum ergänzt. Nachfolgende Abbildung zeigt den exemplarischen Blick aus dem Beobachtungsraum hin zu einem Probanden und der Testbetreuung im Raum. Durch den großen Einwegspiegel konnte die Testleitung, in diesem Fall die Autorin, jederzeit alle Aktionen im Testraum beobachten, ohne von den Probandinnen und Probanden wahrgenommen zu werden.

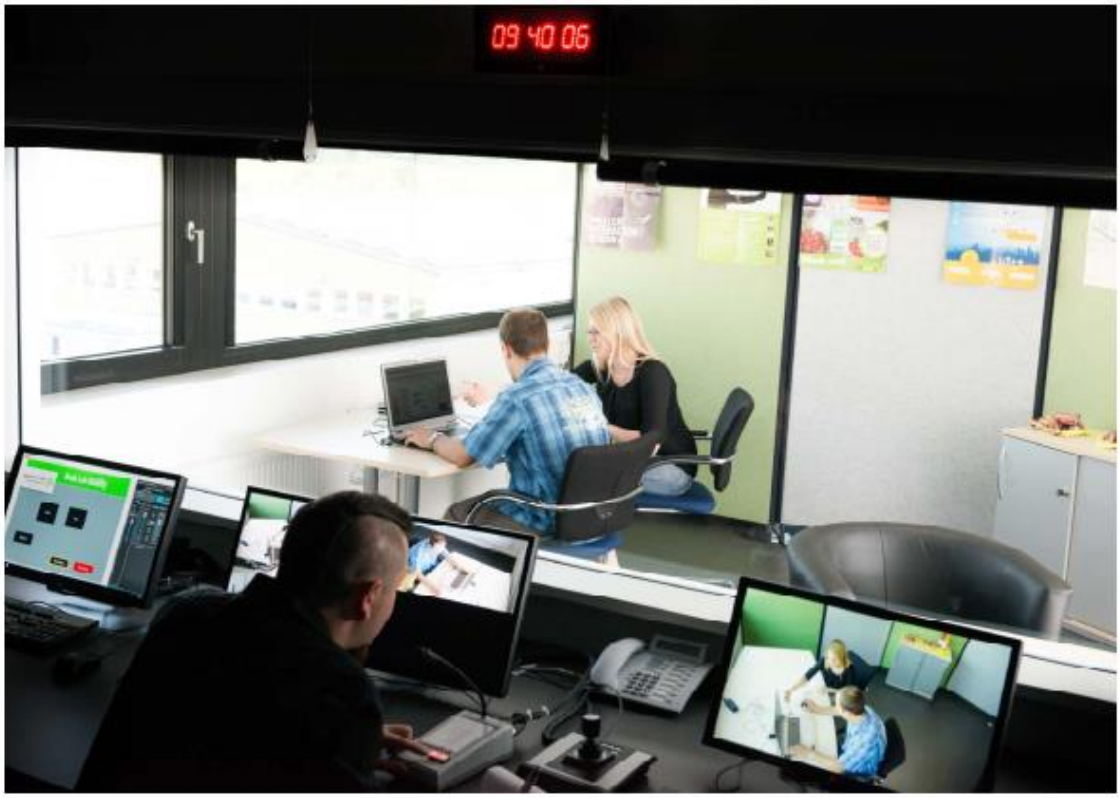

Abbildung 1: Aufbau im Usbility Labor aus der Perspektive des Beobachtungsraums

Als Testbetreuung wurde eine studentische Hilfskraft eingesetzt, die im Vorfeld bzgl. der Methode Lautes Denken geschult wurde. Dadurch konnte die adäquate Begleitung der Probandinnen und Probanden während der Experimente auf Augenhöhe sichergestellt werden. Die Testleitung verbrachte daher und um eine mögliche Beeinflussung der Probandinnen und Probanden zu verhindern, die meiste Zeit während der Durchführung des Lauten Denkens im Beobachtungsraum. Das Setting des Labors wurde dazu leicht auf die Bedarfe der Tests hin angepasst. Aufgrund der Raumgröße und hoher Decken wurde der Raum z. B. durch mobile Trennwände optisch verkleinert, um u. a. eine angenehmere Arbeitsatmosphäre zu schaffen. Zusätzlich wurden helle, farbige Poster an den Pinnwänden platziert, um einen studentischen

Arbeitsplatz zu simulieren, wie er etwa im Selbstlernzentrum der Hochschule Fulda zu finden ist. Nach mehreren Tests mit den Lichtverhältnissen wurde der Arbeitsplatz außerdem auf den Sitzplatz vor der Fensterfront festgelegt. Die Außenjalousien wurden dazu leicht heruntergefahren, um starkes Gegenlicht zu verhindern. Der Tisch bietet zudem genügend Platz für Testperson und Testbetreuung, sodass dort die Arbeit im Browser durchgeführt werden konnte. Die genutzten Anwendungen werden dazu jeweils in den Browser-Tabs vorgeladen. Für den Beginn und den Abschluss der Tests wurde außerdem eine weitere Sitzgelegenheit mit Sesseln und einem kleineren Tisch arrangiert, wie in der Abbildung vorne rechts teilweise zu sehen ist. In diesem Bereich wurde z. B. die Einführung in den Ablauf, Klärung von Formalia wie der Einverständniserklärung, und das Ausfüllen des Eingangsfragebogens durchgeführt. Nach der Arbeit am Laptop wurde der Ausstieg mitsamt Abschlussfragebogen erneut in dieser Sitzecke verlagert. Hinten rechts in der Abbildung des Testraums wird außerdem der kleine Schrank mit Getränken, etc. ersichtlich. Durch den kalkulierten Weg vorbei am Schrank erhalten die Probandinnen und Probanden zu Beginn und Ende des jeweiligen Durchlaufs die Gelegenheit, sich zu bedienen.

Die hochauflösenden Kameras wurden so platziert, dass die Probandinnen und Probanden, und Testbetreuung im Raum jeweils von vorne und hinten oben aufgezeichnet werden. Diese Entscheidung ist darin begründet, dass die Probandinnen und Probanden sich möglichst wenig beobachtet fühlen sollen. Tiefer platzierte Kameras wären kontraproduktiv, so die Annahme. Zudem wurde die Videoaufzeichnung der Bildschirmaktivitäten und über die Webcam vom Beobachtungsraum aus gesteuert. Die insgesamt vier Videodateien boten damit eine umfassende Basis für die spätere Analyse. Parallel dazu wurden die verbal ausgetauschten Informationen ab einem bestimmten Zeitpunkt und nach schriftlicher Zustimmung der Probandinnen und Probanden aufgezeichnet. Dazu wurde jede Person im Testraum zum gegebenen Zeitpunkt mit einem Ansteckmikrofon ausgestattet. Eine weitere, nützliche Funktion des Labors umfasst die Möglichkeit, der Testbetreuung per Tischmikrofon vom Beobachtungsraum aus Anweisungen über ein In-Ear System geben zu können. Dadurch wurden gelegentliche Interventionen z. B. zu weiteren Nachfragen ermöglicht. Darüber hinaus konnten vom Beobachtungsraum aus alle Bildschirmaktivitäten detailliert nachverfolgt werden. Erste Kodierungen parallel zur Durchführung wurden so mit Hilfe der Software Interact ermöglicht.

3 Daten- und Materialerhebung

- **Auswahl der zu testenden Tools:** Für die Untersuchung der Auswirkungen informativen Feedbacks werden zwei verfügbare Selbstlernangebote mit verschiedenen Feedback-Komponenten ausgewählt. Die Untersuchung von zwei Tools und Aufgaben ist vor allem der unterschiedlichen Umsetzung und variierender Feedback-Typen geschuldet. Das erste Tool wird durch den selbst entwickelten Feedback-Prototyp und dessen Aufgabe zur Rekursion dargestellt (vgl. Kiesler 2016). Die zweite Anwendung, die im Rahmen der Lautes Denken-Experimente von den Testpersonen genutzt wird, ist CodingBat. Vor allem die Übungsaufgaben in Java sind von Interesse für die Versuche, da diese Programmiersprache an der Hochschule Fulda in den ersten Semestern vorwiegend gelehrt wird. Daher wird die Webseite auch durchaus an verschiedenen deutschen Hochschulen von Studierenden genutzt, selbst wenn die Seite nur auf Englisch verfügbar ist. Für die Lautes Denken-Experimente wird ganz konkret die Aufgabe zur Berechnung der Fakultät natürlicher Zahlen[1] sowie die Aufgabe zur Berechnung der Fibonacci-Zahlenfolge[2] in Java ausgewählt. Nick Parlante als Informatik-Lehrender der Universität Stanford entwickelte das Tool basierend auf seinen Lehrerfahrungen. Dabei zielt er auf Programmierübung ohne großen Vorbereitungsaufwand ab. Es bedarf keiner Vorabinstallation, lediglich kurze Probleme werden beschrieben. Lernende können direkt programmieren und erhalten darauf direktes Feedback im Browser. Die Aufgaben sind jeweils so gestaltet, dass sie sehr kleine Programmierprobleme adressieren, die wiederum sehr häufig in größeren Projekten vorkommen. Dazu gehören z. B. Aufgaben zu Listen, Strings, Arrays, Schleifen, Logik, Rekursion, und viele weitere. Daraus resultiert das einfach aufgebaute, schlanke Webangebot mit simpel und relativ eindeutig formulierten Aufgabenstellungen, in denen Studierende direkt programmiersprachliche Lösungen in Java oder Python für kleinere Probleme entwickeln und ausprobieren können. Zusätzlich wird mit Hilfe von Unit-Tests Feedback zu den studentischen Eingaben generiert, um die Lernenden zu motivieren. In der Regel handelt es sich dabei um farbliche Hervorhebungen oder etwa einen grünen Haken, insofern die Aufgabe vollständig korrekt gelöst wurde. Hinweise und Musterlösungen sind dagegen nicht durchgängig für alle Aufgaben verfügbar. Zusätzlich können Lernende Sternchen für die Aufgabenbearbeitung sammeln.

[1] Online verfügbar: https://codingbat.com/prob/p154669
[2] Online verfügbar: https://codingbat.com/prob/p120015

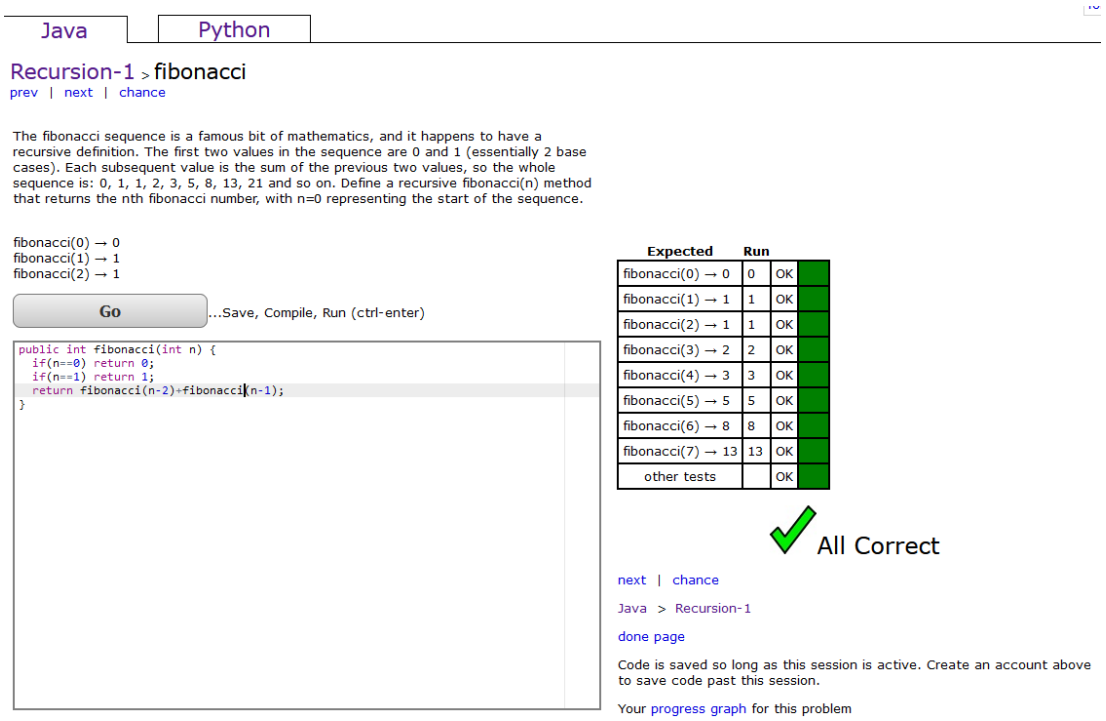

Abbildung 2: Screenshot einer CodingBat Übungsaufgabe und dessen Rückmeldung

Neben der Berechnung der Fakultät sind 29 weitere Aufgaben in dem Bereich Rekursion zu finden. In der zumeist knapp formulierten Aufgabenstellung sind jeweils alle notwendigen Informationen zur Lösung enthalten. Zusätzlich werden drei Beispiele zur Berechnung angezeigt, so ist z. B. die Fakultät von eins gleich eins, die Fakultät von zwei ist zwei und die Fakultät von drei ist gleich sechs. In der exemplarischen Aufgabenstellung zur Berechnung der Fakultät natürlicher Zahlen sind jeweils ein unabhängig von der Eingabe gleichbleibender Hinweis und eine Musterlösung verfügbar. Dies trifft jedoch nicht auf alle verfügbaren Aufgaben zu. Zur Bearbeitung der Aufgaben ist es jeweils erforderlich, Programmcode (hier in Java) in das Eingabefenster zu schreiben. Bei allen Aufgaben ist der Methodenkopf vorgegeben, sodass kein weißes, leeres Fenster vorliegt. Nach der Eingabe des Codes kann dieser über den Go-Button ausgeführt werden. Danach wird eine Tabelle erzeugt, in der die einzelnen Ergebnisse zu der Eingabe mit den angestrebten, korrekten Ergebnissen verglichen werden. Über ein grünes oder rotes Farbsignal in der letzten Spalte der Tabelle wird das richtige bzw. falsche Ergebnis der einzelnen Unit-Tests ersichtlich. Sind alle Tests erfolgreich, erscheint ein grüner Haken mit dem Hinweis „All Correct“ und im Fall der Aufgabe zur Fakultät der Button zu einer vorbereiteten Musterlösung. Enthält die Eingabe Syntaxfehler, wird der entsprechende Compilerfehler im rechten Bereich an Stelle der Tabelle angezeigt. Zusätzlich wird in diesem Fall auf eine Seite mit exemplarischen Compiler-Fehlermeldungen und deren Überarbeitung verlinkt. Einrückungen oder Programmierkonventionen werden bei der Auswertung der Eingaben grundsätzlich nicht berücksichtigt und daher auch nicht mit Rückmeldung versehen.

- **Entwicklung von Erhebungsmaterialien**: Die Lautes Denken-Experimenten wurden jeweils durch einen Fragebogen vorab, ein nachgeschaltetes vorstrukturiertes Leitfaden-Interview sowie einen weiteren kurzen Fragebogen begleitet. Während die Probandinnen und Probanden laut Denken und die Aufgaben bearbeiten, wurden nach einer jeden Aufgabe jeweils kurze Zwischenfragen gestellt. Dafür wurde jeweils der Durchführungsbogen entwickelt. Anhand von Durchführungsbögen wurde der Ablauf koordiniert. Dieser Ablauf wurde im Vorfeld mit Studierenden getestet, u.a. um die zeitliche Dauer abschätzen und die räumliche Aufteilung gestalten zu können.

- **Ablauf der beiden Testreihen**: Konkret wurden zwei Testreihen mit je fünf (Testreihe B) bzw. sechs (Testreihe A) Probandinnen und Probanden im Abstand von etwa zwei Wochen durchgeführt. Dafür wurden jeweils zwei Tage im Labor reserviert. Der Ablauf beider Reihen bzgl. der Einführung, Fragebögen, und des kurzen Abschluss-Interviews wurde grundsätzlich analog zuei-

nander konzipiert, mit Ausnahme der zu bearbeitenden Selbstlernangebote und einzelner Aufgaben, wie die folgende Liste zeigt:

- Testreihe A:
  - Aufgabe des Feedback-Prototyps zur Rekursion (im Datenpaket nicht enthalten)
  - CodingBat Aufgabe zur Berechnung der Fakultät natürlicher Zahlen
- Testreihe B:
  - CodingBat Aufgabe zur Berechnung der Fakultät natürlicher Zahlen
  - CodingBat Aufgabe zur Berechnung der Fibonacci-Zahlen

Pro Testdurchlauf wurden 90 Minuten sowie weitere 30 Minuten als Zeitpuffer für die Vor- und Nachbereitung eingeplant, sodass jeder Durchlauf kurz nachbesprochen und entsprechende Notizen angefertigt werden konnten.

4 Aufbereitung von Daten und Materialien

- **Bearbeitung/Aufbereitung der Daten und Analyse**: Da die Experimente in einem professionellen Usability Labor durchgeführt wurden, konnte sowohl die Aufzeichnung von Audio als auch von Video-Daten erfolgen. Insbesondere die Audioaufzeichnung stellte die Basis für die spätere Transkription der verbalen Daten dar. Als Ergänzung der Transkripte und bei Unsicherheiten wurden die Videoaufzeichnungen zusätzlich zu Rate gezogen, um non-verbale Reaktionen weitestgehend dokumentieren zu können. Bei den Videoaufzeichnungen handelte es sich um gleichzeitige Aufnahmen aus vier verschiedenen Perspektiven. Zwei Videos zeigen die Probandinnen und Probanden im Testraum. Eine Ansicht zeigt dabei jeweils die Testperson mit der Testbetreuung von vorne und von hinten. Die beiden Kameras dafür befanden sich an der Decke des Raums, sodass eine Perspektive von oben auf die Subjekte resultiert. Zudem waren die Bildschirmaktivitäten insgesamt als Videodatei verfügbar. Eine vierte Videospur erfasste die Probandinnen und Probanden von vorne über die Webcam des Laptops, an dem die Aufgaben bearbeitet werden. Mit Hilfe der Software Interact wurde die computergestützte Erfassung und Auswertung des Verhaltens während und nach der Beobachtung unterstützt. So konnten neben den verschiedenen Videospuren z. B. auch die Interaktionen noch während der Lautes Denken-Experimente aufgezeichnet werden. Darüber hinaus werden alle Mausbewegungen, inklusive der Cursor-Koordinaten gespeichert, wobei der Cursor auch als Teil der Bildschirmaktivitäten in Videoform aufgezeichnet wird. Vor allem die Quantifizierung der genutzten Hinweise und Feedback-Typen wurde durch die Video-Analyse ermöglicht. So konnte die Häufigkeit der Nutzung der jeweiligen Feedback-Typen, Buttons und anderer Funktionen der beiden Selbstlerntools erfasst werden.

  Durch die Software Interact wurde zudem das gleichzeitige Anschauen der verschiedenen Audio- und Video-Daten maßgeblich vereinfacht. Annotationen und Kodierungen, z. B. für gedrückte Buttons, konnten außerdem direkt im Programm erfolgen. Es erfolgte insofern eine weitere Überprüfung durch Ansehen der Video-Aufzeichnungen und Abgleich mit den Transkripten.

  Die Lautes Denken-Experimenten wurden jeweils durch einen Fragebogen, ein Leitfaden-Interview sowie einen weiteren kurzen Fragebogen begleitet. Ziel dieser ergänzenden Erhebungsmethoden war es, neben den beobachtbaren auch die berichtbaren Indikatoren für die verschiedenen Feedback-Wirkungen erfassen zu können. Dazu gehören z. B. die Antwortsicherheit, Einschätzung zur Schwierigkeit der vorgegebenen Aufgabe oder Einschätzungen zur Nützlichkeit der angebotenen Feedback-Typen. Weitere, berichtbare Wirkungen umfassen den Spaß und die Motivation bei der Aufgabenbearbeitung sowie die Selbsteinschätzung bzgl. der eigenen Fähigkeiten und Leistung. Durch die Audio- und Videoaufzeichnungen konnten auch diese Komponenten festgehalten werden.

- **Transkription der verbalen Daten**: Die Transkription der Protokolle lauten Denkens erfolgte unter Anwendung eines Transkriptionssystem mit leichter sprachlicher Glättung. Es wurde wörtlich transkribiert, sodass Füllworte und Wortdopplungen erhalten bleiben und z. B. Pausen rekonstruiert werden können. So sollen die Gedankengänge leichter nachvollziehbar bleiben, da der Fokus auf den kognitiven Prozessen liegt. Darüber hinaus wurden die Bezeichnungen der einzelnen Code-Blöcke als Teil der prototypischen Anwendung in Großbuchstaben notiert.

Für die Protokolle des Lauten Denkens wurde die Software f4transkript zusammen mit einem Fußpedal verwendet. Die Transkripte der Lautes Denken-Experimente dienten gemeinsam mit den Videoaufzeichnungen und Fragebögen als Basis für die weiterführende Analyse und Zusammenfassung der Wirkungen informativen Feedbacks als Teil der beiden getesteten Anwendungen.

- Datenschutz: Die umfassenden Transkripte (ca. 130 Seiten) der Lautes Denken-Experimente werden aus Gründen der Datensparsamkeit nicht bereitgestellt, sondern nur die Transkriptionen der Bildschirmaktivitäten der Probandinnen und Probanden. Insofern sind im bereitgestellten Datenpaket weder personenbezogene, noch sensible Informationen enthalten.

- Aufbereitung der Bildschirmaktivitäten: Zur gezielten Erfassung der Bildschirmaktivität wurden jeweils die entsprechenden Videoaufzeichnungen (Screencasts) genutzt. Darin wurden die Aktivitäten der Probandinnen und Probanden Schritt für Schritt dokumentiert.

- Datenpaket zur Archivierung und Nachnutzung: Das bereitgestellte Datenpaket umfasst die aufbereiteten Daten aus der Aufzeichnung der Bildschirmaktivitäten.

- Weitere Daten und Materialien:
  Kiesler, Natalie (2022). *Kompetenzförderung in der Programmierausbildung durch Modellierung von Kompetenzen und informativem Feedback*. Dissertation, Goethe-Universität Frankfurt am Main.

5 Datenpaket

Zur gezielten Erfassung der Bildschirmaktivität wurden jeweils die entsprechenden Videoaufzeichnungen genutzt. Pro Aufgabenstellung wurde eine Excel-Tabelle angelegt. Darin wurde für jede Probandin und jeden Probanden ein eigenes Tabellenblatt angelegt. Daraus geht zunächst die bearbeitete Aufgabenstellung (Fakultät der natürlichen Zahlen, bzw. Fibonacci-Zahlenfolge) hervor. Die Nummerierung der Probandinnen und Probanden erfolgt in Anlehnung an die Testreihe. D.h. die Fakultät der natürlichen Zahlen wurde in Testreihe A und B rekursiv berechnet, wobei die Probandinnen und Probanden von A01 bis A06, und von B01 bis B05 durchnummeriert wurden. Die Aufgabe zur rekursiven Berechnung der Fibonacci-Zahlenfolge wurde lediglich in Testreihe B gefordert. Daher sind in der Datei zur Aufgabenstellung Fibonacci lediglich 5 Probandinnen und Probanden (B01 bis B05) enthalten. Zu ergänzen ist außerdem, dass es sich bei den Probandinnen und Probanden B01 bis B05 der Fakultät-Aufgabe um die gleichen Personen B01 bis B05 handelt, die außerdem die Fibonacci-Aufgabe bearbeitet haben.

In den Excel-Dateien sind pro Probandin bzw. Proband und Aufgabe folgende Informationen enthalten:

- Spalte A: Zeitstempel gemäß der Bildschirmaufzeichnung, sodass die Dauer der jeweiligen Arbeitsschritte und Interaktionen nachvollziehbar bleibt.
- Spalte B: Der von den Probandinnen und Probanden eingegebene Programmcode in Java. Hierbei ist anzumerken, dass der Methodenkopf in CodingBat vorgegeben wird, sodass Lernende nicht mit einem „weißen Blatt“ beginnen müssen.
- Spalte C: Ein qualitativer Code zur Kurzbezeichnung der studentischen Handlung/Aktivität (z.B.: Schlüsselwort wurde hinzugefügt, engl. „keyword added (if)“).
- Spalte D: Insofern die Studierenden Interaktionen vorgenommen haben, werden diese hier vermerkt. Bei CodingBat handelt es sich vor allem um die Nutzung des „Go-Buttons“ zur Ausführung des Codes, den „Help Button“, oder den „Show Solution Button“.
- Spalte E: Hier wird das Feedback des Systems (CodingBat) ergänzt. Je nach Eingabe erscheinen Meldungen wie „Compile Problems“ oder „Bad Code“ mit Anweisungen zur Verbesserung. Ist der Code ausführbar, werden die Ergebnisse von Unit-Tests angezeigt. Deren Ergebnisse sind jeweils dokumentiert.

Nicht alle Aufgaben wurden von den Probandinnen und Probanden erfolgreich gelöst. Diese Information ist jeweils der letzten Tabellenzeile zu entnehmen. Im Erfolgsfall zeigen die Uni-Tests am Ende in Spalte E das Ergebnis „All Correct“. Im Falle eines Misserfolgs wird der Abbruch anhand des Codes in Spalte C ersichtlich („problem solving aborted“). Die in dieser Form aufgezeichneten Bildschirmaktivitäten bilden den Kern der hier bereitgestellten Daten.

Die Aufzeichnung der Bearbeitungsschritte, Interaktionen und Feedback-Meldungen erfolgt in Anlehnung an die Systemvorgaben von CodingBat in englischer Sprache.

6 Nachnutzungspotenzial

Die Nachnutzung der Daten ist im Zusammenhang mit verschiedenen Forschungsfragen und -ansätzen, vor allem in der Informatik und dem Bereich Educational Technologies denkbar. Zum einen kann der Datensatz mit studentischen Arbeitsschritten genutzt werden, um Online Lernumgebungen mit Programmierübungsaufgaben, zu denen Feedback angeboten wird, zu evaluieren und zu vergleichen (insofern die Lernumgebungen identische Aufgaben anbieten).

So könnte dem Systemfeedback zum Beispiel Feedback von Expertinnen und Experten gegenübergestellt werden. Zum anderen können die studentischen Arbeitsschritte genutzt werden, um mentale Modelle der Rekursion nachzuvollziehen bzw. zu rekonstruieren. Die Erkenntnisse können wiederum in die Hochschullehre einfließen, sowie für die (Weiter-)Entwicklung von Lernsystemen genutzt werden.

7 Literatur

Ashford, Susan, Ruth Blatt und Don VandeWalle (2003). „Reflections on the looking glass: A review of research on feedback-seeking behavior in organizations“. In: *Journal of management* 29.6, S. 773–799.

Bangert-Drowns, Robert L. et al. (1991). „The instructional effect of feedback in test-like events“. In: *Review of educational research* 61.2, S. 213–238.

Baron, Robert A. (1988). „Negative effects of destructive criticism: Impact on conflict, self-efficacy, and task performance“. In: *Journal of Applied Psychology* 73.2, S. 199–207.

Battestilli, Lina et al. (2019). „Using Bloom's Taxonomy to Write Effective Programming Questions for Autograding Tools“. In: *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*. SIGCSE '19. Minneapolis: ACM, S. 1260. doi: 10.1145/3287324.3293858.

Butler, Deborah L. und Philip H. Winne (1995). „Feedback and self-regulated learning: A theoretical synthesis“. In: *Review of Educational Research* 65.3, S. 245–281. doi: 10.2307/1170684.

Goldstein, Irwin L., Joseph T. Emanuel und William C. Howell (1968). *Effect of Percentage and Specifity of Feedback on Choice Behavior in a Probabilistic Information-Processing Task*, S. 163–168.

Hattie, John (2009). *Visible Learning: A synthesis of over 800 meta-analyses relating to achievement*. Abingdon: Routledge.

Hattie, John und Helen Timperley (2007). „The power of feedback“. In: *Review of educational research* 77.1, S. 81–112.

Keuning, Hieke, Johan Jeuring und Bastiaan Heeren (2016). „Towards a systematic review of automated feedback generation for programming exercises“. In: ITiCSE '16. Association for Computing Machinery, S. 41–46. doi: 10.1145/2899415.2899422.

Kiesler, Natalie (2016). „Ein Bild sagt mehr als tausend Worte - interaktive Visualisierungen in webbasierten Programmieraufgaben“. In: Lucke, U., Schwill, A. & Zender, R. (Hrsg.), DeLFI 2016 -- Die 14. E-Learning Fachtagung Informatik. Bonn: Gesellschaft für Informatik e.V., S. 335-337.

Kiesler, Natalie (2022). *Kompetenzförderung in der Programmierausbildung durch Modellierung von Kompetenzen und informativem Feedback*. Dissertation, Goethe-Universität Frankfurt am Main.

Kohl, Lutz (2009). „Kompetenzorientierter Informatikunterricht in der Sekundarstufe I unter Verwendung der visuellen Programmiersprache Puck“. Dissertation. Friedrich-Schiller-Universität Jena.

Kulhavy, Raymond W. (1977). „Feedback in written instruction“. In: *Review of educational research* 47.2, S. 211–232.

Kulik, James A. und Chen-Lin C. Kulik (1988). *Timing of Feedback and Verbal Learning*, S. 79–97.

Le, Nguyen-Thinh (2016). „A Classification of Adaptive Feedback in Educational Systems for Programming“. In: *Systems* 4 (2), S. 22. doi: 10.3390/systems4020022.

Malan, David J. und Henry H. Leitner (2007). „Scratch for Budding Computer Scientists“. In: *Proceedings of the 38th SIGCSE Technical Symposium on Computer Science Education*. SIGCSE ’07. Covington, USA: Association for Computing Machinery, S. 223-227. doi: 10.1145/1227310.1227388.

Narciss, Susanne (2006). *Informatives Tutorielles Feedback: Entwicklungs- und Evaluationsprinzipien auf der Basis instruktionspsychologischer Erkenntnisse*. Münster: Waxmann Verlag.

Narciss, Susanne (2020). „Feedbackstrategien für interaktive Lernaufgaben“. In: Springer Berlin Heidelberg, S. 369–392. doi: 10.1007/978-3-662-54368-9_35.

Narciss, Susanne und Katja Huth (2004). „How to design informative tutoring feedback for multimedia learning“. In: Waxmann Verlag, S. 181–195.

Paas, Fred, Alexander Renkl und John Sweller (2003). „Cognitive load theory and instructional design: Recent developments“. In: Bd. 38. Lawrence Erlbaum Associates Inc., S. 1–4. doi: 10.1207/S15326985EP3801_1

Phye, Gary D. und Cheryl E. Sanders (1994). „Advice and Feedback: Elements of Practice for Problem Solving“. In: *Contemporary Educational Psychology* 19, S. 286–301.

Shute, Valerie J. (2008). „Focus on formative feedback“. In: *Review of Educational Research* 78 (1), S. 153–189. doi: 10.3102/0034654307313795.

Skinner, Burrhus Frederic (1938). *The Behavior of organisms: An experimental analysis*. New York: Appleton-Century.

Skinner, Burrhus Frederic (1968). *The technology of teaching*. New York: Meredith.

Smith, Patricia L. und Tillman J. Ragan (1993). „Designing instructional feedback for different learning outcomes“. In: *Interactive instruction and feedback*. Hrsg. von John V. Dempsey und Gregory C. Sales. Englewood Cliffs, NJ: Educational Technology, S. 75–103.

Spohrer, James C. und Elliot Soloway (1986). „Novice mistakes: Are the folk wisdoms correct?“ In: *Communications of the ACM* 29.7, S. 624–632.

Thorndike, Edward L. (1913). *Educational psychology, Vol 2: The original nature of men*. New York: Teachers College Press.

Zhang, Yan, Sheela Surisetty und Christopher Scaffidi (2013). „Assisting comprehension of animation programs through interactive code visualization“. In: *Journal of Visual Languages & Computing* 24.5, S. 313–326